# XroadMedia Ncanto

## Public API Specification

# Contents

# 1 History

| Version | Date | Author | Comments/changes |
|---|---|---|---|
| 1.0 | 2012-07-23 | Andras Kalmar | First released version for Recommender R3.0, aligned with Feature List 1.1 and Roadmap 1.1 |
| 1.01 | 2012-07-31 | Andras Kalmar | • added "position" to the interaction log<br>• added "groupId" to the list of recommendations<br>• storeId only supports lowercase letters and digits<br>• annotation attributes extended to all @asset attributes<br>• renamed uiGenreClassification to uiGenreType, sourceNameClassification to sourceNameType and added synopsisType to @annotation, changed all of them to list of strings.<br>• added stereo (boolean) to @asset attributes<br>• limited rating value range to [-1...+1]<br>• extended reserved interaction types |
| 1.02 | 2012-08-06 | Andras Kalmar | • added productionCountry attribute to @asset<br>• aligned allowed filter constraint types with future search API requirements |
| 1.03 | 2012-08-09 | Andras Kalmar | • added rating attribute to @recommendation<br>• removed age and ageCountry constraint types from @filter in R3.0 |
| 1.04 | 2012-09-06 | Andras Kalmar | • changed language from data type string to type langCode<br>• replaced the reference to a subscriber ID in the parameter section of a recommendation request by the subscriber ID itself<br>• added sourceLogo attribute to @asset<br>• 409 error added to "delete store" request for the case that there is an ongoing ingest into the store |
| 1.05 | 2012-09-11 | Andras Kalmar | • renamed @asset attribute sourceLogo to sourceLogoUrl<br>• extended @topic with topicName and topicUrl |
| 1.06 | 2012-09-19 | Andras Kalmar | • changed startBin and durationBin format from string to integer<br>• removed customProperties from the list of allowed conjuncts |

| | | | |
|---|---|---|---|
| | | | • renamed all attributes referencing an asset ID consistently to "assetId"<br>• renamed REST object for rating and scoring consistently to /profile/<profileId>/scorer |
| 1.07 | 2012-10-04 | Andras Kalmar | • added sourceLogoUrl to annotation template and to the Annex: Metadata attributes<br>• changed priceBin format from string to integer |
| 1.08 | 2012-10-10 | Andras Kalmar | • explained entitlement processing in detail<br>• corrected error in syntax of recommendation request response |
| 1.09 | 2012-10-16 | Andras Kalmar | • changed store attribute "import": <string> to "importing": <boolean> and added new store attribute "importStatus" |
| 1.10 | 2012-10-31 | Andras Kalmar | • Timestamp of the interaction log request changed from required to optional. If not present, the Recommender will automatically generate the timestamp. |
| 1.11 | 2012-11-12 | Andras Kalmar | • Added filtering of social recommendations to owners / topics (R3.1 feature)<br>• specified XSD time format for time of day conditional context rules<br>• explained the definition of filter conjuncts more in detail<br>• renamed createdTime to created_time in @topic<br>• corrected uiGenre to be a list of genres per type |
| 1.12 | 2012-11-27 | Andras Kalmar | • added businessScore to the response of recommendation requests<br>• added search functionality<br>• renamed REST resource for moderated list management from /list to /list/moderated<br>• reworked the overview section to cover R3.1 functionalities |
| 1.13 | 2012-12-11 | Andras Kalmar | • deleted query from @searchEngProps, the query is a parameter of the recommendation request and cannot be put into the context<br>• updated @predicate syntax for conditional rules of R3.1<br>• added optional attribute clientTimeZone to context.global |
| 1.14 | 2012-12-13 | Andras Kalmar | • renamed clientTimeZone to timeZone<br>• improved @predicate syntax for better readability |

| | | | |
|---|---|---|---|
| 1.15 | 2012-12-14 | Andras Kalmar | • added uiGenre filtering of recommendations |
| 1.16 | 2012-12-17 | Andras Kalmar | • turned uiGenre filtering into generic filtering |
| 1.17 | 2013-01-02 | Andras Kalmar | • changed timezone and time format (from R3.1, backward compatible to R3.0)<br>• timestamp in update blacklist request made optional |
| 1.18 | 2013-01-08 | Andras Kalmar | The following R3.2 features have been added:<br>• prefilter of the preference engine<br>• cloning of subscribers<br>• playlists<br>• added optional customProperties and optional timestamp to moderated lists and playlists |
| 1.19 | 2013-01-09 | Andras Kalmar | • "annotation" added to @assetList object<br>• from R3.2 retrieve list(s) calls support the annotation query parameter |
| 1.20 | 2013-01-11 | Andras Kalmar | • renamed "annotation" attribute of @scoredAsset to "asset"<br>• added collaborative similarity engines |
| 1.21 | 2013-01-21 | Andras Kalmar | • added dateTime condition for context rules<br>• added errorRatio to @profile<br>• added calibrateProfile to interaction log<br>• renamed request_recommendations interaction type to requestRecommendations<br>• defined targeted advertising features for R3.2<br>• moved the adFrequency attribute from the global section of the context to the adEngine section |
| 1.22 | 2013-02-20 | Andras Kalmar | • removed requirement that lists have to have at least one element<br>• simplified ad targeting configuration (context management)<br>• defined @scoredAssetId<br>• updated block diagram and the description of targeted advertising in Section 2.2 |
| 1.23 | 2013-02-26 | Andras Kalmar | • some typos corrected<br>• changed REST resource of separate ad interleaving request from /adInterleaver to /recommendations/adInterleaver |
| 1.24 | 2013-03-01 | Andras Kalmar | • simplified adInterleaving configuration |

| | | | |
|---|---|---|---|
| 1.25 | 2013-03-11 | Andras Kalmar | • added "excludeSeed" parameter to similarity engine and collaborative similarity engine properties |
| 1.26 | 2013-03-26 | Andras Kalmar | • limited range of editorial scores in moderated lists to [0,1]<br>• renamed adPool consistently to adpool |
| 1.27 | 2013-04-07 | Andras Kalmar | • deleted adFrequency attribute from context examples, was not corrected when advertising parameters were moved to their own context section in v1.21 of this specification<br>• extended @asset by new optional attributes copyControl and contentUrl, supported from R3.2.5<br>• added copyControl to the list of supported filter constraints<br>• extended @profile by "name" attribute, only used in explanations. Supported from R3.2.5<br>• added expanded information retrieval for single subscribers, supported from R3.2.5<br>• added remark that the subscriber ID shall be provided with all recommendation requests to allow for internal quality monitoring<br>• added recordingSuggestion parameter to @context and to the response of recommendation requests |
| 1.28 | 2013-04-09 | Andras Kalmar | • added GET analytics request (R3.2.5)<br>• simplified @elements |
| 1.29 | 2013-04-23 | Andras Kalmar | • added GET all context IDs request (R3.2.5)<br>• added separate GET/PUT requests for customProperties of subscribers and profiles (R3.2.5)<br>• added "rating" attribute to @scoredAsset (R3.2.5, used by playlists only)<br>• made contribution argument of engines optional (R3.2.5)<br>• added recordingSuggestion attribute to blending preference engine properties (R3.2.5)<br>• added engine specific age limit (R3.2.5) |
| 1.30 | 2013-04-25 | Andras Kalmar | • added separate annotation for advertising (R3.2.5)<br>• optional error tolerance for missing asset IDs added to scoring, rating, and ad interleaving requests (R3.2.5) |

| | | | |
|---|---|---|---|
| 1.31 | 2013-05-14 | Andras Kalmar | • generic recommendation filter extended from single criterion to multiple criteria with AND/OR operators. Supported from R3.2.6<br>• aligned parameters of interleave ads request with those of a recommendations request: separate annotations for ads and recommendations supported (R3.2.5)<br>• added implicit playlist creation<br>• removed all references to releases up to R3.2.5. This version of the API specification shall only be used with R3.2.5 and above.<br>• Collaborative score request added for R3.3<br>• renamed context attribute specifying the annotation for ads consistently to adAnnotation<br>• removed errorTolerant parameter of interleave ads request. That request is error tolerant even without the parameter. |
| 1.32 | 2013-06-05 | Andras Kalmar | • corrected typo: "timestamp" attribute of @scoredAsset shall read "time"<br>• added recordingSuggestion request parameter to rate/score assets in single profile, score assets in multiple profiles, score assets in all profiles of subscriber requests. Supported from R3.2.8.<br>• retrieve assets from active store added for R3.2.9 |
| 1.33 | 2013-06-013 | Andras Kalmar | • "rating" attribute removed from @scoredAsset and added to @scoredAssetId (was a typo in the specification, implementation is correct)<br>• extended @genericFilter to support arbitrary operators starting from R3.3<br>• retrieve assets by filter criteria added to R3.3<br>• retrieve asset attribute values added to R3.3 |
| 1.34 | 2013-06-24 | Andras Kalmar | • rating of features added to R3.3<br>• profile.ratedFeatures.likeDegree renamed to rating (R3.3)<br>• added optional "name" attribute to create profile from seed request (analogous to create/update profile) R3.2.10<br>• explained that recommendation requests don't throw an exception if a non-existing ad pool ID is specified but return recommendations without ads. |

| | | | |
|---|---|---|---|
| | | | • weight function "priceBin" added to R3.3<br>• Global context attribute maxExplanationLevel added to R3.3 |
| 1.35 | 2013-07-09 | Andras Kalmar | • list of supported features added (R3.3)<br>• separate request to change profile name added (R3.2.11)<br>• query builder API added (R3.3) |
| 1.36 | 2013-07-10 | Andras Kalmar | • changed body/response format of change profile name request (R3.2.12) |
| 1.37 | 2013-07-18 | Andras Kalmar | • Twitter based recommendations (engine type "postingBased" added for R3.3<br>• sorting by community rating added for R3.3<br>• described complex search queries (R3.3)<br>• defined allowed data type/operator combinations in @genericFilter (R3.3) |
| 1.38 | 2013-07-31 | Andras Kalmar | • conversion of availabilityStartTime and availabilityEndTime returned by recommendation requests to local time of the subscriber added for R3.3 |
| 1.39 | 2013-08-12 | Andras Kalmar | • support for YouTube API v3.0 added (R3.3)<br>• mood attributes of @asset added<br>• clarified which interaction log entries are interpreted as ratings |
| 1.40 | 2013-08-23 | Andras Kalmar | • excludeOwnRatedItems attribute added to collaborative engine properties (R3.3)<br>• excludeSeed enum of similarity engine and collaborative similarity engie properties extended by value "none" (R3.3)<br>• new statistical list "latestArrivals" (R3.3)<br>• a rating value of 0 overwrites any previous rating of the same asset (R3.3) |
| 1.41 | 2013-08-29 | Andras Kalmar | • subscriber added as query parameter to retrieve assets by filter criteria request (R3.3)<br>• changed response object of retrieve assets by filter criteria from [@asset] to [@augmentedAsset]<br>• update individual profile/subscriber customProperties attributes added for R3.3 |
| 1.42 | 2013-09-11 | Andras Kalmar | • personalization of search engine results added for R3.3.2<br>• described special role of uiGenre types "broad", "medium", and "fine" for preference engine explanations |

| | | | |
|---|---|---|---|
| | | | • new optional asset attribute uiProgramType (from R3.3.2)<br>• retrieval of active asset store info without specifying the storeId (R3.3.2) |
| 1.43 | 2013-09-24 | Andras Kalmar | • topic based YouTube query builder added (R3.3.2)<br>• renamed product from Recommender to Ncanto |
| 1.44 | 2013-10-11 | Andras Kalmar | • new optional asset attribute subtitleLanguage from R4.0<br>• from R4.0: metadata of @ratedAsset (in PUT profile body) is now optional and will be loaded from the asset store if missing<br>• new profile filter constraint "priceBin" supported from R3.3.3, removed price constraint<br>• changed recommended priceBin mapping: 0=free, 1=cheap, ... |
| 1.45 | 2013-10-22 | Andras Kalmar | • file based Axel Springer ingester added (from R3.3.4)<br>• new sorting criteria & multi-criterion sorting added for R4.0 |
| 1.46 | 2013-11-14 | Andras Kalmar | • assetType filtering added to context management for R4.0<br>• support for multiple active asset stores added for R4.0<br>• support for multiple collaborative matrices added for R4.0<br>• sourceLineup added for R4.0<br>• changed entitlement behavior (R4.0)<br>• per asset type annotation added for R4.0<br>• dot notation support in annotations added for R4.0 |
| 1.47 | 2013-12-12 | Andras Kalmar | • multiSorting feature added to retrieve assets by filter request (R4.0)<br>• sourceLineupName and sourceLineup added to retrieve assets by filter request (R4.0)<br>• default references "<engineName>.seed" and "<engineName>.profile" in context prefEngProps, simEngProps and collSimEngProps (R4.0)<br>• hottest statistical lists aliased as trending (R4.0)<br>• mostPopular statistical lists (R4.0)<br>• recommendation pagination (R4.0)<br>• relative window of availability filtering (R4.0) |

| | | | |
|---|---|---|---|
| | | | • episode constraint type supported in profile filters (R4.0)<br>• new automatic decision logic about rating type (R4.0) |
| 1.48 | 2014-02-06 | Andras Kalmar | • corrected groupId type from int to string<br>• removed unnecessary sorting criteria from retrieve assets by filter request<br>• clarified behavior of moderated and statistical lists in case of missing assets<br>• valuator features added (R4.0) |
| 1.49 | 2014-04-22 | Andras Kalmar | • number of asset stores and total number of assets added to analytics response (R4.0)<br>• skipProfileDislikeThreshold to ignore inactive profiles added to preference engine and blendingPreferenceEngine context (R4.0)<br>• in, exists, notExists generic filter operators added (R4.1)<br>• entitlementSources and entitlementSourceGroups attributes added to context and to subscriber (R4.1)<br>• automatic rating assignment to matching profile added (R4.1)<br>• new asset attributes media, structTip, structEditorialRating (R4.1) |
| 1.50 | 2014-04-24 | Andras Kalmar | • removed some restrictions from structTip, structEditorialRating, media asset attributes (R4.1) |
| 1.51 | 2014-05-07 | Andras Kalmar | • added http response codes 403, 415, 429<br>• deprecated @generic filter operators contains and notContains<br>• added dump feature (R4.0) |
| 1.52 | 2014-07-29 | Andras Kalmar | • new asset attributes of R4.1 added<br>• sorting by any asset attribute (R4.1)<br>• search term highlighting added (R4.1)<br>• CORS support added (R4.0)<br>• dot and underscore support in store names (R4.1)<br>• window of availability type (R4.1)<br>• beforeAfter (what's on now) (R4.1)<br>• asset types to be processed by IPDS ingester are configurable (R4.1)<br>• pagination for retrieve assets by filter (R4.1)<br>• assetIds attribute of retrieve assets by filter (R4.1) |

| | | | |
|---|---|---|---|
| | | | • direct annotation (R4.1)<br>• flexible activation/deactivation of stores (R4.1) |
| 1.53 | 2014-08-20 | Andras Kalmar | • rate an asset in a profile identified by subscriberId and profileName (R4.1)<br>• profile used by preference enginea addressed by subscriberId and profileName instead of profileId (R4.1) |
| 1.54 | 2014-09-23 | Andras Kalmar | • retrieve all dump object IDs (R4.1)<br>• subscriber expiry (R4.1)<br>• activity aware blending preference engine (R4.1)<br>• preference serendipity engine (R4.1)<br>• collaborative serendipity engine (R4.2)<br>• overruling of active stores for program guide and recommendation requests (R4.1)<br>• assetIds global context parameter (R4.1)<br>• rate an asset in multiple matching profiles (R4.1)<br>• weight function trigger "communityRating" (R4.1)<br>• skip profiles attribute of blending preference engine (R4.1)<br>• http compression (R4.1) |
| 1.55 | 2014-10-02 | Andras Kalmar | • corrected typo in serendipity engine context |
| 1.56 | 2014-10-13 | Andras Kalmar | • rating in multiple profiles (R4.2)<br>• corrected incomplete conditional annotation specification |
| 1.57 | 2014-11-28 | Andras Kalmar | • replaced Axel Springer by Funke throughout the document (because of acquisition of the relevant parts of Axel Springer by Funke Mediengruppe)<br>• clarified that Funke ingest from files does not support type filtering<br>• new rating significance algorithm "prefilter" (R4.2)<br>• customProperties attribute of asset stores (R4.2) |
| 1.58 | 2014-12-15 | Andras Kalmar | • viewing setting support (R4.2) |
| 1.59 | 2015-01-11 | Andras Kalmar | • window of availability per source and window of availability priority added (R4.2)<br>• blacklist length limiting added (R4.2)<br>• shared profile preference engine (R4.2) |
| 1.60 | 2015-03-06 | Andras Kalmar | • corrected typo in windowOfAvailabilityPerSource spec<br>• search phrase suggestions (R4.2) |

| | | | |
|---|---|---|---|
| | | | • configurable search weights (R4.2) |
| 1.61 | 2015-06-30 | Andras Kalmar | • language support added to query builder API (R4.3)<br>• configurable search phrase suggestions (R4.4)<br>• playlist asset support for all engines (R4.3)<br>• sourceGroup support for entitlement templates (R4.3)<br>• window type "strictEnd" (R4.3)<br>• conditional attribute value search (R4.3) |
| 1.62 | 2015-10-12 | Andras Kalmar | • interaction type "share" (R4.4)<br>• mostShared statistical list (R4.4)<br>• new automatically calculated asset attributes communityViewCount, communityShareCount, communityTrendScore (R4.4)<br>• customProperties of list items (@scoredAsset, @scoredAssetId) (R4.4) |
| 1.63 | 2015-12-04 | Andras Kalmar | • new asset attribute sourceGroupNames (R4.4)<br>• excludeOwnRatedItems for preference and blending preference engines (R4.4)<br>• notIn operator of @genericFilter (R4.4)<br>• genericPrefilter support for profiles (R4.4)<br>• per engine generic filter support in contexts (R4.4)<br>• arbitrary query parameter based if conditions in context (R4.5)<br>• delete assets by filter (R4.4) |
| 1.64 | 2016-01-20 | Andras Kalmar | • filterFacets added to context.global and to EPG context (R4.4) |
| 1.65 | 2016-02-26 | Andras Kalmar | • prefilter and genericPrefilter support for create profile from seed request (R4.5)<br>• optional profile filters (R4.5)<br>• clientCustomProperties (R4.5)<br>• interaction type support in rating and create profile from seed requests (R4.5)<br>• scoring requests support feature level explanation (R4.5) |
| 1.66 | 2016-03-18 | Andras Kalmar | • personalized statistical, moderated, and similarity engines (R4.5)<br>• defaultLanguage asset attribute (R4.5)<br>• extended language filtering: languages and fallbackLanguages context attributes (R4.5) |
| 1.67 | 2016-04-28 | Andras Kalmar | • new profile filter constraint types (R4.5)<br>• new numeric weight functions of R4.5<br>• discrete weight functions added (R4.5) |

| | | | |
|---|---|---|---|
| | | | • profile strength added (R4.5)<br>• profile like degree features added (R4.5)<br>• sorting sequence added (R4.5) |
| 1.68 | 2016-05-30 | Andras Kalmar | • defaultSubscriber in recommendation requests (R4.5)<br>• new blending preference engine profileWeights values (R4.5) |
| 1.69 | 2016-06-10 | Andras Kalmar | • profile selection options for scoring in multiple / all profiles (R4.5) |
| 1.70 | 2016-06-24 | Andras Kalmar | • Profile strength based contribution adjustment of preference, blending preference, shared profile preference, and serendipity preference engines (R4.6) |
| 1.71 | 2016-07-25 | Andras Kalmar | • Subscriber merging (R4.6)<br>• Dot characters are not allowed in asset attribute names (R4.6, Elasticsearch 2 limitation) |
| 1.72 | 2016-08-29 | Andras Kalmar | • Subscriber customProperties merging (R4.6)<br>• Configurable mapping of asset customProperties fields (R4.6)<br>• Generic entitlements (entitlementFilter), R4.6<br>• Generic grouping (R4.6)<br>• nestedAnd generic filter operator (R4.6)<br>• multiple windows of availability (R4.6)<br>• sorting condition (R4.6) |
| 1.73 | 2016-09-19 | Andras Kalmar | • explanationConfig (R4.6) |
| 1.74 | 2016-10-27 | Andras Kalmar | • lineup filtering (R4.7)<br>• direct subscriber lineup manipulation (R4.7) |
| 1.75 | 2016-11-28 | Andras Kalmar | • deprecated additive weight functions (R4.7)<br>• feature rating w/o rating value removes previously active rating (R4.7) |
| 1.76 | 2017-01-09 | Andras Kalmar | • randomization for preference, blending preference, shared profile preference and serendipity preference engines (R4.7)<br>• CRUD subscriber entitlement, entitlementSources, entitlementSourceGroups, entitlementFilter requests (R4.7)<br>• Top subscriber level like degrees (R4.7)<br>• Individual subscriber level like degrees (R4.8)<br>• Clarified that sorting byEngine sorts by engine name<br>• assetGroups asset attribute (R4.7) |

| | | | |
|---|---|---|---|
| | | | • per asset group search phrase suggestions (R4.7)<br>• episode grouping priority (R4.7) |
| 1.77 | 2017-01-19 | Andras Kalmar | • language filtering for search phrase suggestions (R4.8)<br>• seasonTitles & seasonOriginalTitle asset attributes (R4.8)<br>• filterOwnSeries context parameter (R4.8)<br>• delete profile by subscriber ID and profile name (R4.8)<br>• deprecated multiple collaborative matrices support (collaborativeMatrices parameter) in preparation of new collaborative engine in R4.7.1 |
| 1.78 | 2017-04-27 | Andras Kalmar | • introduced data type string512 and changed all IDs to be limited in length. This is not validated in R4.8 yet but the usage of IDs longer than 512 bytes is deprecated.<br>• Distinction parameter of collaborative engine context (R4.8)<br>• Calculate collaborative scores and collaborative valuation features removed from R4.8<br>• Inputs filter support for retrieve assets by filter requests (R4.8)<br>• Random sorting (R4.8)<br>• New contains/notContains logic of genericFilter (R4.8) |
| 1.79 | 2017-08-28 | Andras Kalmar | • Multiple language filtering for search phrase suggestions (R5.0) |
| 1.80 | 2017-10-09 | Andras Kalmar | • track & trackSearchTerm parameters added to recommendation and filter requests (R4.8.6)<br>• Frequently used search terms (R4.8.6) |
| 1.81 | 2017-10-10 | Andras Kalmar | • Count parameter for POST /like-degrees calls (R5.0)<br>• Viewing setting support for POST /like-degrees calls (R5.0)<br>• Multi-seed similarity engine (R5.0)<br>• assetStores & playlists context attributes supported at engine level (R5.0)<br>• removed deprecated collaborativeMatrices context parameter<br>• collaborativeLearningLevel context parameter (R5.0)<br>• new asset attribute metadataQuality (R5.1) |

| | | | |
|---|---|---|---|
| 1.82 | 2017-11-27 | Andras Kalmar | • allow negative zeroContributionProfileStrengthLimit (R5.0)<br>• collaborative engine excludeOwnRatedItems supports "program" and "series" values (R5.0)<br>• multiple moderated lists support for moderated engines (R5.0)<br>• window of availability filtering at engine level (R5.0)<br>• deprecated retrieve all statistical lists (R5.0)<br>• variables support in genericFilter (R5.0) |
| 1.83 | 2017-12-31 | Andras Kalmar | • custom explanations support in context (R5.0)<br>• custom explanations of moderated list items (R5.0)<br>• contribution adjustment for all engines (R5.0) |
| 1.84 | 2018-03-05 | Andras Kalmar | • aligned rating significance body style to rest of the API: assetId instead of asset is used. Asset is deprecated.<br>• Corrected typo in collaborativeLearningLevel: allowed values are asset/program/series instead of assetId/programId/seriesId<br>• Fixed incorrect examples for feature ratings of keywords, persons, and customProperties<br>• New origin value of relativeWindowOfAvailability: startOfWeek (R5.0) |
| 1.85 | 2018-04-05 | Andras Kalmar | • Randomization factor for statistical and moderated engines (R5.0)<br>• Configurable ratingAggregation for all rating requests (R5.1)<br>• Per engine minBusinessScore parameter (R5.1)<br>• relativeMinScore attribute for search engines (R5.1)<br>• bookmark lists added (R5.1)<br>• new context attribute bookmarklists (R5.1)<br>• underscores supported in langCode type (R5.0)<br>• configurable availability windows (R5.1)<br>• query parameter variables (R5.1)<br>• variables applied to context via patch syntax (R5.1) |

| | | | |
|---|---|---|---|
| | | | • minimumLikeDegree threshold for likeDegree variables (R5.1) |
| 1.86 | 2018-05-09 | Andras Kalmar | • contextId attribute of rating and of log interaction requests (R5.2)<br>• groupingPriority support for filter request (R5.2)<br>• new groupingPriority value episodeForced (R5.2) |
| 1.87 | 2018-05-22 | Andras Kalmar | • awards added to @asset (R5.2)<br>• deprecated "tip" and "editorialRating" asset attributes (R5.2)<br>• added missing "parameterName" attribute to context.variables (R5.1)<br>• variables support to customExplanations (R5.1)<br>• personal recent search terms (R5.2)<br>• blendingFilterPreferenceEngine (R5.2) |
| 1.88 | 2018-07-06 | Andras Kalmar | • new value "none" supported by filterOwnSeries (R5.3)<br>• added missing startBin mapping to Table 4<br>• timestamp variables (R5.3)<br>• groupSize information in retrieve assets by filter and recommendation responses (R5.3) |
| 1.89 | 2018-07-16 | Andras Kalmar | • multiChannelAudio boolean attribute added to @asset (R5.3) |
| 1.90 | 2018-07-27 | Andras Kalmar | • search engine property: autocorrection (R5.3)<br>• contextNames (R5.3)<br>• generalized statistical lists (R5.3)<br>• delete statistical list from cache (R5.3)<br>• stacked groups (R5.3)<br>• panel management (R5.3)<br>• multi-context (panel) recommendations (R5.3) |
| 1.91 | 2018-07-28 | Andras Kalmar | • corrected description of clear statistical list cache command (clear and recalculate)<br>• fixed typo: productionYearBin should read yearBin<br>• validation of string512 type will be activated from R5.4 |
| 1.92 | 2018-10-22 | Andras Kalmar | • new asset attribute: assetFamilies (R5.4)<br>• assetFamilies query parameter for GET /list/statistical requests (R5.4)<br>• program and series level statistical lists (R5.4)<br>• dynamic subscriber groups (R5.4) |

| | | | |
|---|---|---|---|
| 1.93 | 2018-10-25 | Andras Kalmar | • "default" value for variables which cannot be resolved (R5.4)<br>• Additional response code 429 for update asset requests (R5.4)<br>• Changed dynamicSubscriberGroups syntax (R5.4) |
| 1.94 | 2018-11-05 | Andras Kalmar | • Corrected some typos in dynamicSubscriberGroups (R5.4) |
| 1.95 | 2018-11-14 | Andras Kalmar | • assetFamilies attribute of context.statisticalEngineProperties (R5.4)<br>• statisticalListLevel attribute of context.statisticalEngineProperties (R5.4) |
| 1.96 | 2018-11-26 | Andras Kalmar | • specified binning of awards for like degree requests<br>• frequency capping (R5.4) |
| 1.97 | 2019-02-12 | Andras Kalmar | • changed json data type to noDotsJson for all custom properties<br>• added revenue+currency to log interaction request (R5.6)<br>• added revenue+currency to rating requests (R5.6)<br>• subscriberProperty variables (R5.6)<br>• subscriber conditions for context rules (R5.6)<br>• presentationProperties (R5.6)<br>• per subscriberGroup statistical lists (R5.6)<br>• subscriberGroupFilter for statistical engines (R5.6)<br>• autoclone subscriber feature for recommendation requests (R5.6) |
| 1.98 | 2019-03-31 | Andras Kalmar | • simplified the specification of context variables |
| 1.99 | 2019-05-27 | Andras Kalmar | • seedProperty context variables (R5.6)<br>• customScore engine (R5.6)<br>• multi-seed collaborative similarity engine (R5.6) |
| 1.100 | 2019-06-24 | Andras Kalmar | • subscriber bookmark lists (R5.6)<br>• bookmarkListFilter in global context (R5.6)<br>• grouping priority support in global context (R5.6)<br>• subscriberPlaylists support by individual engines (R5.6)<br>• score asset in single/multiple profiles support viewing settings (R5.6)<br>• windowOfAvailabilityPerSourceGroup (R5.6) |

| | | | |
|---|---|---|---|
| 1.101 | 2019-07-10 | Andras Kalmar | • new groupingPriority values firstUnwatchedEpisode, firstUnwatchedEpisodeForced, firstEpisode, lastEpisode, latestArrival, earlierstArrival (R5.6)<br>• deprecated groupingPriority values episode and episodeForced |
| 1.102 | 2019-07-15 | Andras Kalmar | • ad-hoc preference engine (R5.6) |
| 1.103 | 2019-07-16 | Andras Kalmar | • autoTimeOfDayBin, autoTimeOfWeekBin (R5.6) |

# 2 Overview

The purpose of Ncanto is to provide video recommendations to consumers. This document describes the public Application Programming Interface (API) of release 3 of Ncanto. Section 2 gives a generic and by no means exhaustive overview of the features. Authoritative and subrelease specific information is contained in Section 3.

## *2.1 Usage Model*

Ncanto implements a server-server usage model depicted in Figure 1. Ncanto is connected to a customer server, which is passing requests on behalf of distributed clients of Subscribers. Clients are not directly connected to Ncanto in that case.


**Figure 1: Server-server usage model of Ncanto**

In addition to the server-server model, the client-server usage model shown in Figure 2, in which distributed clients directly connect to the API of Ncanto, is supported. In this case, Ncanto has to provide additional security features like client authorization and DoS attack prevention. In addition, CORS (Cross-origin resource sharing) is supported to bypass the same origin security policy of browser based clients in client-server usage scenarios.

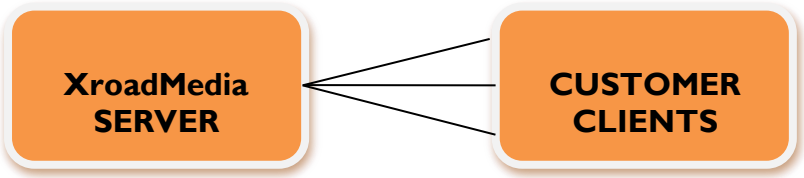


**Figure 2: Client-server usage model of Ncanto**

## *2.2 Privacy and Data Protection*

Ncanto's recommendation algorithms do not require and do not use any personal information. For privacy reasons no personally identifiable information (PII) may be stored in Ncanto. In particular, all IDs shall be anonymous and the subscriber object shall not contain any PII like e-mail address, gender, age, etc.
For the same reason Ncanto's logs do not contain the IP address of the client.

## *2.3 Functional and Architectural overview*

Figure 3 depicts the functional block diagram of Ncanto. The system is recommending video assets, metadata about which is available in the Asset Store, to clients of the system. Recommendations are generated by (one or more) recommendation Engines implementing various recommendation algorithms. The Mixer combines intermediate results of the recommendation Engines into one list of recommendations (Reclist), which is then sorted and filtered by the Global Postprocessor.

Finally, the Ad Interleaver applies targeted advertising to the Reclist. Ad targeting is done by (one or more) Ad Engines employing different algorithms to find and personalize Ad assets.

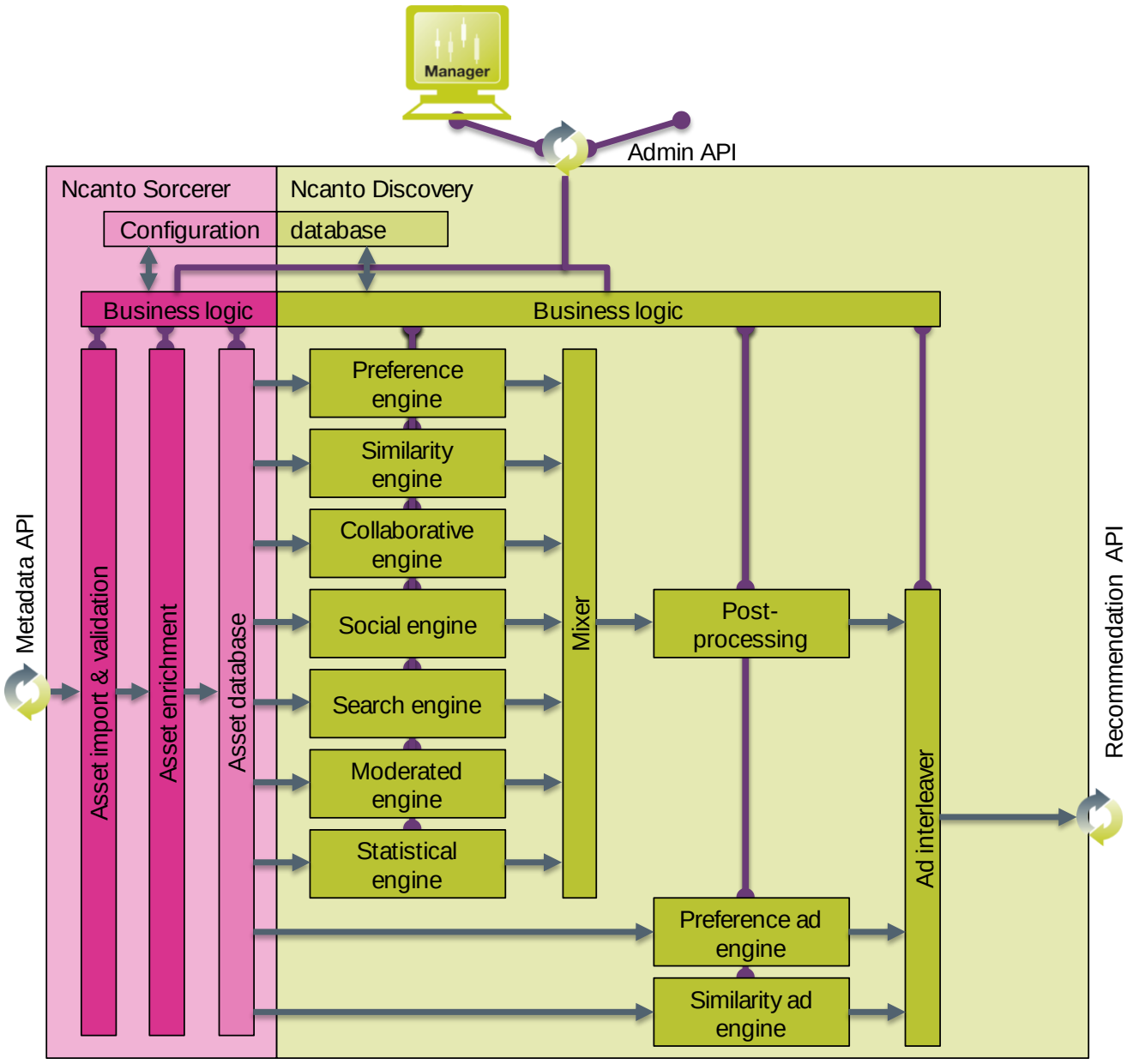


**Figure 3: Functional block diagram of Ncanto**

### 2.3.1 Asset Database

The Asset Database stores metadata about all the assets known to Ncanto. The asset database may contain an arbitrary number of asset stores (tables). Asset stores may be active or inactive, only assets in active stores will be considered by Ncanto.
Ncanto supports arbitrary types of assets, e.g. Linear Broadcast, Video on Demand, Over The Top, and Advertising.

Assets in the asset database are identified by their unique Asset ID. Ncanto Assets are organized into Sources, one Source might correspond to one TV channel or a VoD library, or a set of advertising clips. Each source is identified by its unique Source ID. Multiple Sources might be combined into one Source Group. Note that Asset IDs have to be unique across the Asset Store even for assets with

different source IDs in different Source Groups. Moreover asset IDs have to be unique across all active asset stores.

Customer specific asset attributes which do not fit into the predefined Ncanto attributes may be added to the open (schema-less) attribute customProperties.

Asset metadata is provided by one or more metadata providers of the customer. The Asset Store provides an interface (cf. Section 3.6) for metadata providers to ingest assets into the system.

**Metadata Enrichment**
Ncanto is prepared to enrich available metadata by automatically generating additional attributes. Currently the mood attributes are automatically generated by the "ingest assets" request.

### 2.3.2 Configuration Store

The Configuration Store is the main database of Ncanto containing information about the system's Subscribers and configuration:

- List of Subscribers and the subscribers' properties
  - Entitlement: defines which Sources and Source Groups a Subscriber has access to
  - Blacklist: contains items (assets, programs, series) which should not be recommended
  - Profiles: preference profiles associated with the Subscriber
  - Topics: preferences from the Subscriber's social network profile, which are relevant to Ncanto
- the preference Profiles comprising
  - a filter defining which assets do roughly match the profile
  - the list of rated assets, e.g. "likes the Simpsons" but "dislikes Two and a half men", defining the implicit part of the profile,
  - rated features, e.g. "likes action movies", "likes Clint Eastwood as actor", defining the explicit part of the profile,
  - blacklist: containing items (assets, programs, series) which should not be recommended
- Annotation templates defining the set of metadata to be returned together with recommendations
- Moderated Lists
- Statistical Lists
- the Collaborative Filtering Matrix
- etc.

### 2.3.3 Engines

A recommendation Engine is identified by its unique Engine Name and is primarily characterized by its Engine Type. The Engine Type defines which recommendation algorithms and which processing steps (e.g. filtering) are used.
Ncanto supports the following Engine Types:

- Similarity: a Similarity Engine recommends Assets which are semantically similar to a seed asset provided. Optionally the preference profiles of the user are taken into account too to provide personalized similarity recommendations.
- Preference: the Preference Engine recommends assets matching a preference profile, which is derived from user ratings. Multiple profiles per subscriber are supported by Ncanto.
- Social Engine: recommends assets based on the subscriber's social network profile (e.g. Facebook profile)

- Collaborative Engine: recommends assets based on collaborative filtering (Subscribers who watched this, also watched that…)
- Statistical Engine: recommends statistically relevant assets, e.g. most watched or best rated assets. Optionally the preference profiles of the user are taken into account too to provide personalized statistical recommendations.
- Moderated Engine: recommends assets which have been composed by an editor of the customer. Optionally the preference profiles of the user are taken into account too to provide personalized editorial recommendations.
- Search Engine: offers free text and structured search in the Asset Store. Optionally the preference profiles of the user are taken into account too to provide personalized search results.
- Posting Engine: recommends assets based on the subscribers postings to social networks (e.g. Tweets).
- Serendipity Engines: the serendipity flavor of the preference engine provide "surprising" recommendations, i.e. recommendations which to not perfectly match the user's expectations but which are expected to enrich the user experience.
- Custom Score Engine: uses a customer defined score formula to select the assets it recommends

Multiple Engines of the same Engine Type are supported.

A recommendation Engine operates on a set of candidate assets defined by asset type and the Engine's input. By default all engines use all asset types with the exception of "ad" (advertising), this default setting can be overruled on a global or per engine basis.The Input corresponds to one or more Source Groups (cf. Section 2.3.1). Any Engine will only recommend assets which match the specified asset types and which are part of its input.

Other configuration parameters of an Engine comprise e.g. the subscriber or profile IDs to be used, the seed asset for similarity calculation, or filtering criteria like parental control.

The contribution attribute of an Engine defines the relative contribution of the engine to the final Reclist. If the contribution attribute is missing, a default value of 1 is applied to all engines (resulting in all engines contributing to the final Reclist in a uniform way).

**Similarity Engine**

A Similarity Engine uses the ID of an asset, called the seed asset, as its main input. It searches for other assets with similar characteristics and provides a score indicating how similar the recommended asset is to the seed asset. A list of seed assets is supported too.
Similarity recommendations do not require to store or learn user preference, they are available immediately and, since there is no preference data stored, need little resources.

**Preference Engine**

Preference engines recommend assets based on the Subscriber's preference, which is learned from explicit or implicit feedback and may be complemented by direct subscriber input ("I like action movies"). Subscriber preference is stored in Profiles.

Ncanto supports feedback communicated to Ncanto as Ratings. A Rating corresponds to hitting a like or dislike button on the user interface of Ncanto's client (explicit asset rating) or is derived from subscriber 0, and +1 are intended to represent "like", "undo", and "dislike" and overwrite any previous

rating of the same asset, rating values in the interval ]-1,0[ U ]0,+1[ are intended for implicit ratings and are accumulated.
In addition to asset ratings the subscriber may add feature ratings of the type "I like actor XY" or "I don't like genre horror". Asset and feature ratings may be combined in the same profle.
In order to increase the learning speed of Ncanto, multiple Profiles per Subscriber, corresponding e.g. to specific segments of the Subscriber's interests are supported.

Profiles can be created from a seed asset or by defining the Profile's pre-filter and filter explicitly. The Profile's pre-filter is used to define criteria all recommendations for the given profile have to fulfill. In contrast the profile's filter is used to preselect assets which roughly match the Subscriber's preference. This preselection is further narrowed down by scoring according to the preference profile. The more Ratings the Subscriber submits to a specific Profile, the more accurate recommendations will be. Right after the creation of a Profile, there are no preferences learned yet, and a Preference Engine will recommend assets based on its filter only. This phase is characterized by all scores of not rated assets being 0.5. The preference profile kicks in and starts to personalize assets passing the filter by individual scoring as soon as the sum of positive ratings or the sum of negative ratings reaches 1.0 (i.e. in case of "like"/"disilke" ratings after the first positive or first negative rating).

The Profile Strength value is an indicator for the quality and completeness of a preference profile. In general, increasing the number of ratings will improve the profile strength.
Ncanto is also providing "like degree" information about individual feature value pairs in the preference profile, e.g. how much a specific genre or actor is liked by the subscriber. Alternatively, the like degrees of all feature value pairs in the profile or a list of feature value pairs with the highest like degrees can be requested too.

The score provided by a Preference Engine indicates how well the Asset matches the Profile used.
To provide recommendations, a Preference Engine takes the ID of one Profile as a reference. Multiple Preference Engines can be configured in parallel to retrieve recommendations mixed from blended Profiles of a Subscriber.

The preference engine takes into account the environment in which feedback is given and content is consumed. The environmental parameters are called viewing setting. The viewing setting considered by Ncanto comprises the variables time of day, time of week, device, mood, location, and activity.
The setting parameters of each asset rating are stored in the preference profile. If the recommendation request contains the setting parameters, Ncanto automatically selects the matching part of the profile.
This feature allows to distinguish different consumption behavior depending on device, time, mood, or location, within the same profile.

The Blending Preference Engine uses all Profiles belonging to a Subscriber and blending recommendations from all of them. If multiple profiles per subscriber are present (e.g. movie, series, news profiles), the blendingPreferenceEngine is used to combine them. The blendingPreferenceEngine ensures that all profiles get their fair share of the recommendations and that profile weights are set smartly depending on how important the profile is for the subscriber. There are various algorithms to calculate the weights of the individual profiles. Besides the default uniform weighting the amount of ratings (positive or all) in the profile, can be taken into account. Excluding inactive profiles completely is supported too.

In contrast, the Bleding Filter Preference Engine is used in single profile per subscriber scenarios. To deliver a broad variety of recommendations from a single profile, the Blending Filter Preference

Engine supports multiple segment filters (e.g. movie series, news,...) and ensures that all segments get their fair share of the recommendations and that segment weights are set smartly depending on how important the segment is for the subscriber. There are various algorithms to calculate the weights of the individual segments. Besides the default uniform weighting the amount of ratings (positive or all) matching the segment, can be taken into account. Excluding inactive segments completely is supported too.

The shared profile preference engine blends recommendations from the subscriber's own profile and from profiles of other subscribers who are sharing their profiles with the subscriber. The amount of recommendations from the subscriber's own profile w.r.t. the shared profiles is configurable. The relative distribution of recommendations among the shared profiles is determined automatically by Ncanto by calculating the similarity between the subscriber's own profile and the individual shared profile.

The ad-hoc preference engine does not require the client to send all user interactions to Ncanto to build the profile. A preference profile is created on the fly based on the recently watched items of the subscriber at the time recommendations are requested.

The contribution of the preference engines (preference, serendipity preference, blending preference, shared profile preference, blending filter preference) is automatically adjusted depending on the Profile Strength if the contribution adjustment feature is activated. Contribution adjustment allows to reduce the contribution of a preference engine in case the profile strength is low.

All flavors of the preference engine support randomization, i.e. instead of recommending the content with the highest preference score, a random subset of a larger candidate group of content is recommended.

### Social Engine

The Social Engine uses the Subscriber's and the Subscriber's friends' profiles in social networks to generate recommendations. Facebook profiles are supported, other social networks will be added in later releases.
Since Ncanto does not store any personally identifiable information about Subscribers and in order to comply with the terms of use of most social networks, it is the task of the client to retrieve the Subscriber's (and, optionally, also the Subscriber's friends') social profile and to send it to Ncanto. A reference implementation of the social network connector retrieving the profiles and forwarding them to Ncanto is provided.
From the social profiles submitted, Ncanto automatically selects Topics which are relevant to Ncanto. The list of Topics found relevant is returned and stored as a property of the Subscriber.
The Social Engine uses the Subscriber's ID as a reference to provide recommendations matching one of the Subscriber's Topics. A score indicating how well the recommended Asset fits to the Subscriber's social profiles is calculated and returned too. In order to identify the social network profile a recommendation originates from , an optional attribute "owner" is supported. Clients may use the "owner" attribute to store e.g. the first names of the Subscriber's friends.

### Posting based Engine

This is another type of a social engine, instead of the user's profile in social networks, messages sent by the user to a social network or received from the user's friends (postings) are used to derive recommendations. Ncanto supports Twitter postings (i.e. tweets) as an input, analyzes the postings, identifies relevant keywords and topics, and returns recommendations matching them. In contrast to

the topics identified by the social engine, postings are highly dynamic and short-lived, they are not stored in Ncanto's database.

### Collaborative Engine

A Collaborative Engine learns from the behavior of all the Subscribers of an Ncanto instance. It takes either the ID of a Subscriber or the ID of an Asset as reference and recommends Assets which other Subscribers with similar taste liked before. The Collaborative Engine is tolerant toward poor asset metadata but requires a significant amount of historical data to provide meaningful recommendations. The collaborative engine of Ncanto supports non-video assets (e.g. books, apps, games, etc.) too. By default, all assets (video and non-video) are put into one common collaborative matrix (database). To support large catalogs, Ncanto supports separate collaborative matrices per asset type. If video and non-video assets are used together in one Ncanto instance, the collaborative matrices have to be designed on a case by case basis by XroadMedia personnel, taking into account catalog sizes and cross-domain recommendation requirements.
The multiple matrices support is deprecated from R4.7.1.

### Statistical Engine

Statistical Engines recommend Assets from
Ncanto supports the following statistical lists:

- Best rated Assets
  This list is created based on all Ratings of the system's Subscribers, taking into account the absolute number of Ratings as well as the ratio between positive and negative Ratings. Note that because the number of ratings is taken into account too, the score of the statistical engine does not equal the average rating (community rating) of an asset.
- Most watched Assets
  This list is created from entries in the interaction log. Ncanto needs such information from the client to build the most watched list.
- Trending Assets
  To create the trending assets list, Ncanto calculates the increase in the number of interactions per time unit and takes the assets with the highest value.
- Latest arrivals
  The latest arrivals lists contain new assets which have recently become available.
- Most shared Assets
  This list is created from "share" entries in the interaction log.

The time window of each statistical list is selectable via the context.

The Statistical Engine takes one of the statistical lists as input and recommends assets on the list to the Subscriber. A score is provided indicating the relevance of the asset in the statistical list.

Each statistical list (defined by the type/timeWindow/assetTypes/programTypes parameters) is created on the fly when it is first used. It is then cached for a short period of time depending on the length of the time window used. The default caching durations are:

- 1 minute up to a time window of 10 minutes
- 10% of the length of the time window for time windows between 15 minutes and 40 hours
- 4 hours for time windows longer than 40 hours

### Moderated Lists

Moderated Engines use lists of editorial recommendations as input and recommend assets from them.

## Search Engine

The Search Engine offers intelligent free text search capabilities among the assets in the asset store. Auto-completion and auto-correction are supported. By default search is performed in the following asset attributes: titles, uiGenres, genreBroad, genreMedium, genreFine, originalTitle, keywords, categories, persons, subtitles, synopses, sourceNames, uiProgramType, uiFunctionNames. The attributes used and the weights per attribute are configurable.

In addition. Structured search based on Lucene syntax is supported:

| | |
|---|---|
| search for an exact word or phrase | Use double quotes "<phrase>" to search for an exact word or set of words. |
| include a word | Add a plus before the word (immediately i.e. w/o blank between the operator and the operand) to indicate that the word is mandatory. If the plus sign should not be interpreted as "include" but as part of the query string, escape it with a backslash \. |
| exclude a word | Add a dash – before the word (immediately i.e. w/o blank between the operator and the operand) to exclude all assets containing the word. If the dash should not be interpreted as "exclude" but as part of the query string, escape it with a backslash \. |
| proximity search | Use the tilde ~ operator at the end of a phrase to search for the words of the phrase within a limited distance from each other, e.g. "rain man"~2 |
| wildcards | * and ? wildcards represent arbitrary terms or characters, respectively. If the * or ? characters should not be interpreted as wildcards but as part of the query string, escape them with a backslash \. |
| logical operators | AND, OR, NOT, note that logical operators have to be in CAPITAL letters |
| parentheses | ( and ) may be used to control the sequence of evaluation |
| search within specific fields | Use the asset attribute name followed by a colon and the desired asset attribute value to search in specific fields only. In case of hierarchical asset attributes the dot notation is supported (e.g. persons.actor:eastwood) |

Note that the following characters are interpreted as operators by Lucene and have to be escaped by a backslash \ if they should be part of the search string: + - && || ! ( ) { } [ ] ^ " ~ * ? : \ /

The search terms are highlighted in the response and the names of the asset attributes the search term was found in are returned too.

Ncanto is able to personalize search results. If a subscriber ID is known and personalization is requested, the search algorithm is used to obtain candidate assets, then the scoring mechanism of the preference engine (scoring in all profiles of the subscriber) is applied to determine the result's scores.

In addition to the search engine, which returns assets matching the search query, Ncanto also supports search phrase suggestions. Based on an (incomplete) search term Ncanto suggests complete search phrases together with the asset attribute name the phrase was found in. Note that for best real time performance the database to provide search phrase suggestions is created at ingest time. Hence it is not possible to appiy the usual filters (generic filter, window of availability filter, etc.) to search phrase suggestions. Limited filtering of the suggestions is available via the assetGroups parameter. If an asset carries the assetGroups attribute, its phrases are stored in the search phrase databases of each given asset group. The search phrase suggestion request supports the same assetGroups parameter to select the databases the search phrase suggestions should come from. This allows for basic filtering of the search phrase suggestions, e.g. to separate suggestions from adult and non-adult content. The asset group aware search phrase suggestion feature is available from release 4.7.

Another search related feature of Ncanto is the tracking of search terms used and the maintaining of a "popular search terms" list. Popular search terms can be retrieved for any given time period. Blacklisting and whitelisting are supported to avoid offending search terms (e.g. blacklist "sex" but whitelist "Sex and the City".
Popular search terms can also be filtered to a single subscriberId to get the recently used search terms of one subscriber only.

**Custom Score Engine**
The custom score engine uses a number of custom score functions defined in the context to calculate the score. The contributions of the individual custom score functions are added up and the result is normalized to the value range 0...1

### 2.3.4 Mixer

The output of the recommendation Engines is combined into a single Reclist by the Mixer. In addition, the Mixer can group recommended assets similar to each other to indicate that they belong together. E.g. if grouping by programId is configured, assets representing the same program, e.g. multiple airings of the same movie on TV, or different versions of the same VoD item, are grouped together. The maximum number of assets per group can be limited to receive e.g. only one of such assets. If grouping at series level is selected ("grouping":"series" until R4.6, "groupingAttributes":["seriesId","programId"] in later releases), all episodes of the same series are grouped together in addition. Starting from R4.6 Ncanto supports generic grouping by any asset attribute. A list of asset attributes is supported, the first attribute present in the asset to be grouped defines which group it will belong to. If series ID is among the grouping attributes, the grouping priority parameter may be used to select which assets shall make it into the groups. By default the assets with highest business score are selected first. Grouping priority based on season and episode numbers selectes the first not yet watched new episodes of each series.

Starting from R5.3 the mixer can be configured to stack groups. Instead of returning a flat list of recommendations (with the groups being indicated by the same groupId only), all recommendations of a group may be stacked into one object. The first recommendation of the group will constitute the main recommendation, all other recommendations belonging to the same group will be added as stacked recommendations into the main recommendation.

There is one Mixer per Ncanto instance.

### 2.3.5 Ad Generator and Ad Interleaver

These components offer targeted advertising features integrated into Ncanto. The Ad Generator selects the best matching Ad assets from their for every asset of the Reclist. By default all assets with asset type "ad" are considered as advertising. This may be extended to other asset types via the context. Similarity and user preference based algorithms are supported. The Ad Generator is using a subset of ad assets called an Ad Pool. Ad Pools represent all ads which are applicable at a given point in time or in a given context and are typically managed by the customer's campaign management system.
The task of the Ad Interleaver is to interleave the candidate Ads provided by the Ad Generator into the Reclist. The Ad Interleaver supports the Ad Frequency property: a floating point number between 0 and 1 determining the rate of Ads in the Reclist. E.g. an Ad Frequency of 0.2 means that on average one Ad will be placed for every 5$^{th}$ asset of the Reclist.
There is one Ad Interleaver per Ncanto instance.

### 2.3.6 Global Postprocessing

Assets can be filtered and sorted both by the Recommendation Engines themselves and also by the Global Postprocessing block. Some of the filtering and sorting rules can be applied at Engine and at Global level. The following filtering options are supported:

- Window of availability
  Only Assets which are available during the specified time window are recommended. Time windows may be specified via absolute timestamps (2013-01-01 12:00:00) or relative to the current time (current time + 2h). The following window types are supported: "overlap": any asset the availability of which overlaps with the specified window of availability will be returned. "strictStart": only assets which become available during the specified window are returned. "strictEnd": only assets the availability of which ends during the specified window are returned. "strict": only assets the availability of which is completely inside the specified window (i.e. start and end inside the specified window) are returned.
  In addition to a default window of availability applicable to all sources, individual windows per source may be specified to account for different catch-up durations.
  Until R4.5 window of availability filtering looks at the asset attributes availabilityStartTime and availabilityEndTime. From R4.6 multiple availability windows may be specified (e.g. the broadcast window and the catch-up window of a TV asset) and will be taken into account by the filter. To this end the additional availabilities have to be provided in the asset's custom properties and the corresponding attributes have to be configured as availability window attributes. Please ask XroadMedia personnel to configure your instance accordingly. From R5.1 the window defined by originalStartTime and originalEndTime can be used, too, and the configuration of the availability windows to be used is exposed via the context.
- Minimum score
  Only Assets, the score of which reaches the given threshold are recommended. Threshold are defined on a per engine basis. Preference engines ignore the minimum score requirement until the profile becomes active.
- Minimum Age
  Only Assets with a minimum age requirement lower than or equal to the given value are recommended. Ncanto supports parental control per country, hence the country of the viewer is taken into account by the minimum age filter. Assets without minimum age

specification are considered to be suitable for any age. Age limits may be set globally or on a per engine basis.

- viewing Device
  Ncanto supports filtering recommendations according to suitability for a specific viewing device. The list of suitable viewing devices is an attribute of every Asset in the sytem. By specifying the Device in a recommendation request, Assets not suitable for the device can be avoided.
- Generic asset filter
  Additional filter criteria based on asset attributes may be defined e.g. to filter recommendations to selected program types or genres. The generic filter syntax supports all asset attributes and the operators exists, equals, in, notEquals, notIn as well as smaller/larger/atLeast/atMost for numberical attributes. Multiple criteria can be defined using the and / or operators. An additional operator, nestedAnd, is supported if multiple criteria have to be matched by the same element of list.
  In addition to providing exact values to the generic filter (e.g. genreBroad == thriller), variables are supported, too. In that case Ncanto will resolve e.g. the favorite genreBroad value from the subscriber's profiles and substitute the variable name with the favorite genre automatically.
- Lineup filter
  Results are filtered to the personal lineup of the subscriber. Multiple lineups per subscriber are supported and an index based subset of a lineup may be used for filtering (e.g. return assets from the first 8 sources of lineup "default" of the subscriber).

Ncanto supports the following sorting options:

- sorting by score, default direction: descending
  The Assets in the final Reclist will be sorted by Score, starting with the highest. This is the default sorting.
- sorting by availability start time, default direction: ascending
  The Assets in the final Reclist will be sorted by the start of the window of availability, starting with the earliest.
- sorting by community rating, default direction: descending
  Assets with highest community rating are shown first.
- sorting by source lineup:
  Results are sorted according to the sequence of sourceIDs in the source lineup, which shall correspond e.g. to the subscriber's lineup of TV channels
- sorting by season/episode, default direction: ascending
  Results are sorted by ascending season and ascending episode number (episodes of lowest season in ascending order, then episodes of the later seasons). Note that sorting by episode does not take into account the series ID, i.e. it only makes sense together with sorting by groups.
- sorting by engine:
  Results are sorted by engine name.
- sorting by group maximum score, default direction: descending:
  Results are sorted by groups, starting with the group containing the asset with the highest score, then the group containing the next lower maximum score.
- sorting by recording suggestion, default direction: descending:
  Results with recording suggestion = true are returned first
- sorting by business score, default direction: descending
  The asset with the highest business score (weight function of score) is returned first.

- random sorting,
  The assets are sorted randomly, direction is ignored.
- sorting by any asset attribute, default direction: ascending:
  Results are sorted lexicographically by the asset attribute given in dot notation. Assets with non-existing or empty sorting criterion are appended at the end of the list. If the criterion is not a leaf of the asset but a branch node, the whole node is serialized and used for lexicographic sorting (objects are sorted by keys first). Lists are compared element by element, in case of different lengths, the longer list comes first. If an attribute name contains a dot (e.g. in customProperties), it has to be "escaped" by putting the attribute name in single quotes, e.g. "customProperties.'attribute.1'". Please note that in contrast to sorting byTime, sorting by asset attribute availabilityStartTime treats the value as string and hence does not correctly deal with different time zones.
- sorting by multiple criteria:
  An ordered list of the above criteria is specified, the criteria are applied in the order of their appearance in the list. I.e. the result will start with the first (lowest or highest) value of the first criterion. Within the group with the same value of the first criterion assets are sorted by the second criterion.
  The default sorting direction can be overruled for all criteria.
- sorting sequence:
  A sorting sequence exactly defines the desired order for the given list of attribute values. Values in the sorting sequence are always sorted to the beginning of the result list, in the order of their appearance in the sequence (even if the sorting direction is descending). The remaining values are sorted according to the specified direction.

Starting from R4.4, filter facets may be optionally added to the response. Filter facets contain all available values of a given asset attribute and the corresponding number of assets having that attribute value.

There is one Global Postprocessor block per Ncanto instance.

### 2.3.7 Blacklisting

Subscribers can put assets they don't want to show up in recommendation lists on a blacklist. Ncanto supports blacklisting at Subscriber level (blacklisted items will not be recommended to the Subscriber, independent of the Engine used) and at Profile level (allowing to create individual blacklists per Profile).
A blacklist comprises three sublists:

- List of unwanted asset IDs to blacklist individual assets
- List of unwanted program IDs to blacklist all assets representing the same program (e.g. multiple airings of the same TV show would be indicated by the same program ID)
- List of unwanted series IDs to blacklist all episodes of a series

The amount of items on the blacklist can be limited automatically by Ncanto. Limits have to be set by XroadMedia personnel during provisioning of a system. If a limit is set, the number of assets, programs, and series is independently limited to the given threshold by applying the FIFO principle.

### 2.3.8 Business Scoring

Independent of the Engine used, Ncanto provides a recommendation score for every recommended asset, indicating the quality of recommendation.

This recommendation score represents Ncanto's estimate, how well the asset matches the Subscriber's taste, how similar the asset is to the reference asset, etc.

Business Scores are calculated from the Recommendation Scores according to Weight Functions and represent a combined value of how well the asset matches the subscriber's taste and how well recommending the asset serves the service provider's business goals.
With the help of Weight Functions, a service provider can fine tune which assets are recommended, by e.g. pushing assets which are particularly profitable or which are expiring soon.

Business scores are calculated as recommendation score * weight. Note that additive weight functions (business score = recommendation score + weight) are deprecated and should not be used. Additive weights are applied before normalizing the recommendation score, hence the actually applied weights may be smaller than specified for similarity, search, social, and posting engines.

Ncanto supports the following numerical triggers to weight Recommendation Scores:

- remaining availability (based on availabilityEndTime, even if multiple availability windows are used)
- elapsed availability (based on availabilityStartTime, even if multiple availability windows are used)
- profitability
- price bin
- community rating
- community share count
- community trendscore
- community viewcount
- production year
- lease duration
- lease viewcount
- duration
- lightness & pace

The Weight is characterized as a linear spline (piecewise linear approximation) of the desired weight curve and is specified by its nodes. The concept is illustrated in Figure 4. This weight function example is intended to push VoD assets shortly before they expire, i.e. in the last days of availability. The service provider decided to use a multiplicative weight function to start pushing assets 10 days before expiration. The assets are moderately pushed between 5 and 2 days of remaining availability, and finally strongly pushed between 2 days and one day remaining availability. The weight remains at its constant maximum during the last 24 hours.

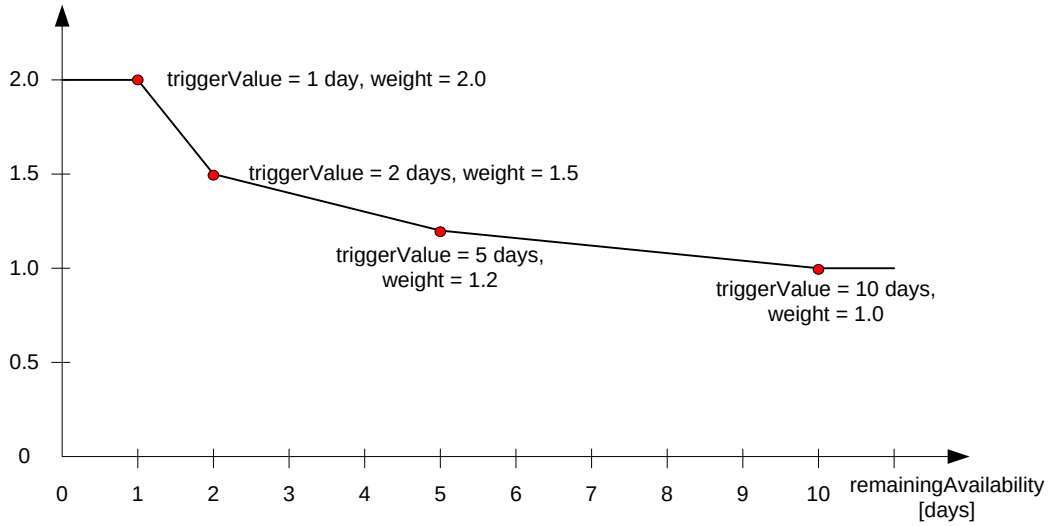


**Figure 4: Weight function example**

This results in the following nodes of the weight function:

1. triggerValue = 10 days: weight =1.0. Since this is the node with the biggest triggerValue, the corresponding weight is applied constantly for remainingAvailability > 10 days.
2. triggerValue = 5 days: weight = 1.2
3. triggerValue = 2 days: weight = 1.5
4. triggerValue = 1 day: weight = 2.0. Since this is the node with the smallest triggerValue, the corresponding weight is applied constantly for remainingAvailability < 1 day.

In addition to the numeric triggers, Ncanto supports discrete weight functions too. Any asset attribute/attribute value pair may be defined to boost the business score either additively or multiplicatively.
Discrete weight functions allow e.g. to boost special program types or genres in the results. In case of non-numeric triggers the weight function is not interpreted as a linear spline, the nodes represent independent trigger values which result in a boost. For trigger values not included in the weight function the business score will not be changed.

### 2.3.9 Context Management

All configuration parameters and weight functions used to provide recommendations are called a Context. Every recommendation request is based on a Context. Contexts can be predefined and referenced in the recommendation request by a context ID, or can be provided as body of the recommendation request, called an Ad-hoc Context. A combination of predefined and ad-hoc Contexts is supported, too.

Context Management allows creating, retrieving, updating and deleting pre-defined Contexts. Each Context is identified by its unique Context ID and comprises one or more Rules of the type

```
IF <predicate> THEN <properties> ELSE <properties>,
```

i.e. if the specified condition is met the properties are set according to the THEN clause. If the condition is not met, the properties are set according the ELSE clause.

The predicate may include time (time of day or absolute time) based conditions, subscriber (e.g. subscriber group) based conditions, or conditions dependent on arbitrary query parameters of the recommendation request (e.g. device based). When evaluating tubscriber group based conditions, Ncanto takes into account the static subscriber groups ("subscriberGroups" property of the subscriber) as well as the dynamic subscriber groups ("dynamicSubscriberGroups" property of the subscriber, based on the dynamic group definitions described in Section 0.

Time or time of day based conditions are evaluated at the beginning of the time window, for which the recommendations are requestred ("`windowOfAvailabilityStart`" or "relative`WindowOfAvailabilityStart`" attribute of the context). If that is not available, the current time (converted into the time zone of the client, if that info is available) is used.

A Context comprises an ordered list of globally valid and Engine specific rules. The Rules of a Context are evaluated at runtime and result in a set of configuration parameters used by Ncanto. The unconditional Rules of the pre-defined Context are evaluated first, in the order of their appearance. The unconditional Rules of the ad-hoc Context of the request are evaluated next, again in the order of appearance. After that step the rule conditions are evaluated and the remaining Rules (the ones the conditions of which are met) are evaluated, again starting with the pre-defined Context.

Because Rules are evaluated sequentially, a Rule can overwrite properties set by a previous Rule.
If an attribute is defined multiple times in the context, the attribute value of the latest occurrence is kept. Weight functions, global filters, multiSorting criteria, and relativeWindowOfAvailability criteria are replaced as a whole, individual nodes or criteria are not merged.

If the Business Rules result in an incomplete set of parameters for any component of the system (i.e. if mandatory properties like the Engine Type of one Engine are missing), the corresponding component will not be used.

The minimum set of parameters is defined as follows:

Global section:
- numberOfItems
- if typeOfItems == groups, the grouping and maxGroupSize parameters have to be present too

Preference Engines:
- name
- reference profile ID

Social Engines:
- name
- subscriber ID

Similarity Engines:
- name
- reference asset ID

Collaborative Engines:
- name
- subscriber ID

Statistical Engine
- name
- statisticalList ID

Moderated Lists

- name
- moderatedList ID

Search Engine

- name
- search query
- subscriber ID if personalization==subscriber

Blending Preference Engines:

- name
- subscriber ID

Collaborative Similarity Engines:

- name
- reference asset ID

Ad Targeting:

- algorithm
- adFrequency
- adPools

The concept is illustrated by an example below. In this case two Preference Engines, one Social Engine, one Moderated Engine, and one Statistical Engine are used. There are Rules to be applied at global level, as well as Engine specific rules. Business Scoring is applied to the Preference Engine recommending VoD assets to push assets expring soon.

```
{
    "contextId": "default",
    "rules": [
        {
            "title": "basic engine config",
            "then": {
                "global": {
                    "grouping": "none",
                    "numberOfItems": 20,
                    "typeOfItems": "assets",
                    "windowOfAvailabilityStart": "2013-01-01T00:00:00Z",
                    "windowOfAvailabilityEnd": "2013-01-02T00:00:00Z ",
                    "sorting": "byScore",
                    "country": "deu",
                    "minAge": 16,
                    "subscriberBlacklisting": "true",
                    "annotation": "smartphone_short",
                    "entitlement": [
                        "BBC_TV",
                        "VoD_library_17"
                    ],
                    "device": [
                        "smartphone"
                    ]
                },
                "preferenceEngines": [
                    {
                        "name": "prefTv",
                        "contribution": 10,
                        "inputs": [
                            "SrcGrp_TV1",
                            "SrcGrp_TV2"
```

```
            ],
            "profileBlacklisting": "true",
            "profile": "prof1"
        },
        {
            "name": "prefVod",
            "contribution": 10,
            "inputs": [
                "SrcGrp_VoD"
            ],
            "profileBlacklisting": "true",
            "profile": "prof2",
            "weightFunctions": [
                {
                    "trigger": "remainingAvailability",
                    "type": "multiplicative",
                    "nodes": [
                        {
                            "triggerValue": "86400",
                            "weight": "2.0"
                        },
                        {
                            "triggerValue": "172800",
                            "weight": 1.5
                        },
                        {
                            "triggerValue": "432000",
                            "weight": 1.2
                        },
                        {
                            "triggerValue": "864000",
                            "weight": 1
                        }
                    ]
                }
            ]
        }
    ],
    "socialEngines": [
        {
            "name": "social",
            "contribution": 10,
            "inputs": [
                "SrcGrp_TV1",
                "SrcGrp_TV2",
                "SrcGrp_VoD"
            ]
        }
    ],
    "moderatedEngines": [
        {
            "name": "editorial",
            "contribution": 10,
            "inputs": [
                "srcGrp_VoD"
            ],
            "moderatedList": "moderatedListId17"
        }
    ],
```

```
                "statisticalEngines": [
                    {
                        "name": "hottest",
                        "contribution": 10,
                        "inputs": [
                            "srcGrp_TV1",
                            "srcGrp_TV2",
                            "srcGrp_VoD"
                        ],
                        "statisticalList": "hottest4h"
                    }
                ]
            }
        }
    ]
}
```

In addition to the ad-hoc context which allows to extend or modify a pre-defined context at runtime, context variables are supported, too. Context variables are a convenient solution to change a few individual context parameters only. The following variable types are supported:

- like degree variables automatically resolved to a feature value from the subscriber's profiles, e.g. to the subscriber's favorite genre or series.
- query parameter variables the value of which is set by a query parameter of the recommendation request
- timestamp variables resolving a relative time specification (e.g. current time + offset) into an absolute timestamp

See Section 3.16 for a detailed specification of the Context management API.

### 2.3.10 **Moderated List Management**

The Moderated List Management API provides create/retrieve/update/delete methods for Moderated Lists. Ncanto supports an arbitrary number of Moderated Lists identified by unique Moderated List Identifiers.
Each Moderated List consists of an ordered sequence of Assets. When a Moderated List is created or updated, Ncanto does not check if the referenced Assets are available in the Asset Store.
However, Assets not available in the Asset Store at the time of recommending them are ignored.

### 2.3.11 **Playlists and Bookmark Lists**

Playlists and Bookmark lists both contain references to assets. Although the lists are inherently independent of subscribers, they may be linked to subscribers if required. There are two differences between the list types:

1. The metadata of assets added to a playlist is persisted by Ncanto independent of the availability of the asset in the active asset store. Playlists are intended e.g. for recording applications where a client side recording may exist much longer than the lifetime of the asset in the active store. Keeping the metadata of playlist assets means that such recorded assets can be rated and scored even if they are not available in the store anymore.
2. In contrast Bookmark Lists can only refer to assets present in the active store. Assets removed from the active store are automatically purged from bookmark lists, too. In addition a time to live can be defined for each bookmarked asset, after which it will be removed from the list automatically.

### 2.3.12 Recommendation API

The core functionality of the Ncanto is to provide recommendations.
Recommendations are requested by specifying the Context (mandatory) and the subscriber, profile, and seed asset IDs to be used.

The response consist of a list of Assets identified by their unique Asset ID and accompanied by metadata for displaying purposes (Annotation), by an Explanation, why the Asset has been recommended, by the Recommendation Score indicating the relevance of the Asset, and by targeted advertising matching the asset.

For consistency reasons, the search functionality is covered by the Recommendation API too.

Pagination, i.e. defining a subset of the recommendation response which should be returned in one step, is supported. Note, however, that Ncanto does not support sessions, each recommendation request is served independently, and the result pages are in fact taken from different recommendation results.

**Annotation**

Recommended Assets and targeted advertising are annotated by metadata allowing client devices to display basic information about the Assets. The set of metadata attributes to be returned is configured via Annotation Templates. Annotations may be specified globaly for all asset types or individually on a per asset type basis.

For all asset attributes which are available in multiple languages one or a list of desired languages may be specified. All other languages are filtered out from the response. If none of the preferred languages is available, the languages from a configurable list of fallback languages are used. If none of the fallback languages is available either, the default language of the asset is used.

**Explanations**

For transparency reasons Ncanto provides explanations, why a specific asset is recommended. The Explanation comprises a list of strings explaining the cause of recommendation. The importance of the explanation is indicated by its level, the primary explanation having level = 0 contains the Engine Type used to derive the recommendation. More detailed explanations are e.g. sorting and filtering rules or the list of most significant features of a profile which led to the recommendation. The desired depth of explanations is configurable. Starting from R4.6 the language of the explanation itself and the languages of asset attributes used in the explanation are configurable independently. Moreover the maximum level based configuration has been replaced by an exact specification of which explanations are desired and which are not.

**Recommendation Score**

In addition to the explanation, the relevance of the recommendation is provided. The relevance of a recommendation is expressed as a Score, a decimal number between 0.0 (indicating “not relevant at all”) and 1.0 (indicating “perfect match”).

**Recording suggestion**

Ncanto might be used by digital video recorders to find assets which are automatically recorded for the user. Ncanto can tag the best recommendationns with a separate flag indicating such a recording suggestion.
Recording suggestion is currently supported for preference engines only.

**Frequency Capping**
In order to avoid the same recommendation served to the same subscriber several times, Ncanto supports frequency capping. The maximum frequency of the same asset in successive recommendation requests can be limited by specifying how often the same asset may be recommended in a given time period to the same subscriber. For performance reasons the time period is limited to 72 hours and a maximum of 1000 recommendation impressions are tracked per subscriber.

In a typical scenario not all the recommendations served by Ncanto are shown to the user (e.g. because a recommendation container in the UI is not completely visible and the user does not scroll to its end). Therefore only the recommendation impressions which are logged by the client via the Log Subscriber Interaction API are considered by frequency capping.

### 2.3.13 Panel Recommendations

Starting from R5.3 Ncanto is able to combine recommendations from mutiple contexts into one single response object. The aim of this feature is to serve all recommendations required to populate a page in the user interface of the client, comprising multiple recommendation containers, in one single request.
Panels consist of multiple containers, each of them associated with one context. Containers may be split into multiple slots. For each slot the recommendation engine which is expected to populate the slot can be defined. If that preferred engine does not deliver enough results, fallback engines defined for the slot will be used.
Ncanto automatically ensures that there are no duplicates in panel recommendations. Assets present in a container are automatically suppressed in subsequent containers of the same panel.

### 2.3.14 Related online content
Ncanto is able to generate queries to retrieve related content from public online video sources like YouTube etc. Queries are either generated based on a seed asset (to search for online content related to a seed asset) or based on a preference profile (to search for online content related to the preferences of a subscriber). In both cases Ncanto returns a list of API queries for the selected video source. It is the responsibility of Ncanto's client to send those queries to the corresponding APIs.

### 2.3.15 Program guide API
With the help of flexible filtering queries to the asset store Ncanto provides electronic program guide functions from simple grid presentation to sophisticated faceted browsing.
The program guide API offers configurable annotations and flexible filtering and sorting criteria, very similar to the generic postprocessing applicable to recommendation requests too.
In addition, the actual values of asset attributes together with their counts are available, to be used e.g. by a faceted user interface.

### 2.3.16 Asset valuation
The asset valuation features help service providers to decide which new (typically out of entitlement) content, e.g. a new pay TV channel, is the most promising for a subscriber and should be offered. In addition the valuation features can also be used to estimate the potential impact and revenue of new

content before the service provider licenses the content for offering to its subscribers. The asset valuator of Ncanto can use the preference profiles to perform the valuation. The selection of assets to be valuated can be based on sourceId, sourceGroupId, or asset store ID.
The valuator either returns the average score of the assets or a list of highest scoring assets.

# 3 Application Programming Interface

Ncanto's APIs are designed in the REST (Representational State Transfer) architectural style, using HTTP as the underlying protocol. The main implications of this are:

- Resources (server-side state) are identified by URI.
- The HTTP methods GET, POST, PUT, and DELETE are the verbs to operate on these resources, i.e. to retrieve and change state.
- Communication is stateless in the sense that no contexts between calls exist, other than what is expressed through explicit changes on the resources (POST, PUT, DELETE methods). Specifically, this means there is no notion of a "session" with a "logon" and "logoff". Any API call is possible anytime, without an enforced sequencing.
- Resource state is transferred as documents in JSON format, all requests shall set http header Content-Type:application/json.
- Json responses are compressed if the client provides an Accept-encoding: gzip header in the request.
- CORS (Cross-origin resource sharing, http://www.w3.org/TR/cors/) is supported to bypass the same origin security policy of browser based clients
- http compression is supported for all Ncanto responses
- GZIP compression of request bodies is supported in incoming direction

## *3.1 Response codes*

Errors are signaled using HTTP Status codes. The standard status codes are used as defined in RFC2616. The corresponding error messages are delivered in JSON format and specified in Section 3.

| Result | Description | Comment |
|---|---|---|
| | | |
| **2xx** | **Success** | |
| 200 | OK | Successful request |
| 202 | Accepted | The request was received but not yet fully processed (e.g. batch). |
| 204 | No response | Request successful, but there is no information to send back |
| | | |
| **4xx** | **Client Error** | |
| 400 | Bad request | The request could not be understood due to malformed syntax. Explanatory error messages are delivered in the following generic format:<br>`{"description": string, required, error message`<br>`"what": string required, reference to errored object`<br>`}` |
| 401 | Unauthorized | The request was sent without valid authentication credentials for the requested action |

| | | |
|---|---|---|
| 403 | Forbidden | The request is not supported at the given URL or your client IP address is blocked. |
| 404 | Resource not found | The requested resource was not found. This can apply to one or multiple resources required, details are given in the error message.<br>Explanatory error messages are delivered in the following generic format:<br>`{"description": string, required, error message`<br>`"what": string required, reference to errored`<br>`object`<br>`}` |
| 409 | Conflict | The request cannot be fulfilled because it conflicts with the current state of the system. For example, a certain resource already exists and cannot be created twice.<br>Explanatory error messages are delivered in the following generic format:<br>`{"description": string, required, error message`<br>`"what": string required, reference to errored`<br>`object`<br>`}` |
| 415 | Unsupported media type | A media type other than application/json or no media type has been specified. |
| 429 | Too many requests | The client has exceeded the rate limit set (only used by test and demo instances). Headers X-Ratelimit-Limit: n (requests/sec), X-Ratelimit-Remaining: 0..n (requests) and X-Ratelimit-Reset: 0..1.0 (seconds until the next reset of the bucket) inform the client about the rate limiting configuration. In addition a body Retry-After: 0..1.0s is returned if the limit is exceeded.<br><br>Note that asset update (POST /store/<storeId>/asset) requests may return 429 even if the rate limit is not yet exhausted. The meaning is the same, the client should retry after a short period. The resources used by update assets requests are limited to avoid impacting the performance of Ncanto. |
| | | |
| **5xx** | **Server Error** | |
| 500 | Internal server error | |
| 503 | Service not available | |

## *3.2 Customer-Specific Endpoint*

Ncanto is presented as an exclusive, single-tenant system to each customer. Therefore each customer shall use one specific base URI to access the system. For instance, a customer "*tvbox*" is accessing the recommendation API of the system via the base URI https://api-tvbox.xroadmedia.com/.
For scalability and cost reasons multi-tenant deployments are supported but hidden from the customer.

## 3.3 Authentication and Authorization

Ncanto offers the following optional authentication and authorization methods:

- Based on network level security, i.e. access from defined source IP addresses only.
- HTTP Basic Authentication on each request with a technical user account defined for the customer.
- Based on a TLS client certificate.
- Based on API key, i.e. a key provided as HTTP request header api_key or as query parameter api_key

HTTP basic authentication is intended for a limited number of customer servers only, does not support a large number of distributed clients. An authorization header has to be included in each API request by the client (see http://en.wikipedia.org/wiki/Basic_access_authentication).

Another option is authentication via client certificate signed by single sign-on server of customer or other trusted authorization server. Ncanto is keeping a list of trusted authorization servers. No revocation of certificates is supported, certificates are valid until they expire.

## 3.4 Transport Layer Security

Unless otherwise specified and agreed with the customer, the API shall be accessed using HTTPS (HTTP over a TLS connection). This prevents eavesdropping and allows clients to verify that they are connected to a legit server.

## 3.5 Specification Syntax

The API is specified in JSON notation. As there is no standard JSON schema specification, the grammar is given in an ad-hoc notation, using the simple data types specified in the JSON grammar (http://json.org) and @object to name specific objects defined here.

The following object types are used in the JSON specification:

| type | description |
|---|---|
| string | string according to XSD string data type. If used in URLs, IDs have to be encoded according to RFC3986 Section 2.5: the string should first be encoded as octets according to the UTF-8<br>character encoding; then only those octets that do not correspond to characters in the unreserved set (ASCII letters, digits, hyphen, dot, underscore, tilde) should be percent-encoded.<br>In JSON objects the ID strings are encoded according to http://www.json.org |
| nonEmptyString | Any string except "". |
| string512 | Like string, but maximum 512 bytes long after UTF-8 encoding. string512 is introduced from R4.8 for performance improvement reasons. The length is not validated until R5.3 but IDs longer than 512 bytes are deprecated. Validation is introduced from R5.4. |
| storeName | string of maximum length 200 characters, composed of lowercase ASCII letters and of digits 0-9. Starting from R4.1 store names may contain dots (.) and underscores (_) too. |

| | |
|---|---|
| Int | integer number according to XSD integer data type |
| decimal | decimal number according to XSD decimal data type, e.g. "1.347" |
| boolean | true\|false |
| enum | one of the specified values |
| timestamp | time according to XSD dateTime data type, format "CCYY-MM-DDThh:mm:ss[Z\|(+\|-)hh:mm]", e.g. "2001-10-26T21:32:52", "2001-10-26T21:32:52+02:00", "2001-10-26T19:32:52Z", "2001-10-26T19:32:52+00:00", or "2001-10-26T21:32:52.12679" |
| duration | time interval according to XSD duration data type, format "PnYnMnDTnHnMnS", e.g. "P1Y6M15DT12H" (one year, 6 months, 15 days, 12 hours) or "-PT1H" (minus one hour). |
| time | time of day, format "hh:mm:ss", e.g. "21:32:52", or "21:32:52.12679" |
| json | JSON object (http://www.json.org) |
| noDotsJson | JSON object with the limiation that no dots (.) are allowed in keys |
| langCode | Non-empty string composed of ASCII letters, digits, hyphen '-', and – from R5.0 – underscore '_'. This data type is used for language specifications and is wide enough to support all commonly used language codes, e.g. ISO 639 and BCP 47. |
| timezone | Timezone info in tz database format (e.g. "Europe/Vienna") or as fixed offset [+-]hh:mm (e.g. "+03:00"). See http://en.wikipedia.org/wiki/List_of_tz_database_time_zones for a complete list of tz database IDs. |

## *3.6 Metadata Management*

See Section 2.3.1 for an introduction to metadata management.

### 3.6.1 Create or reset Store, change active Store

This request creates an empty store with the given storeId, or, if storeId already exists, updates the store's active state. By default, autoDeactivate=true, i.e. if a store is created in active state or changed to active state, all other stores are automatically deactivated (set to "active": False). To activate multiple stores at the same time, each of them has to be activated independently using the query parameter autoDeactivate=false. Multiple active stores are treated as "one big store".
Some store parameters, e.g. the import status or the earliest/latest availability of assets in the store, are calculated on the fly based on the content and state of the store at the time the query is sent. Create or update store requests (PUT) ignore those parameters in the body.

| **PUT** | `/store/<storeId>?autoDeactivate=false` | |
|---|---|---|
| ***body*** | `@store` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400** | Request body is syntactically incorrect |

```
@store =
{"storeId": storeName,                  optional, unique ID of the store, ignored by PUT
                                        method
 "active": boolean,                     optional, activates the store if set to true. If
                                          omitted, the status of an existing store remains
                                          unchanged, a new store is created by default in
                                          passive state.
 "importing": boolean,                  optional, current import state, ignored by PUT
```

```
                                        method
 "importStatus": string,               optional, detailed information about import
                                        Progress, ignored by PUT method
 "noOfAssets": int,                    optional, number of assets in the store, ignored
                                        by PUT method
 "availabilityFrom": timestamp,        optional, beginning of availability of the earliest
                                        asset in the store, ignored by PUT method
 "availabilityTo": timestamp,          optional, beginning of availability of the latest
                                        asset in the store, ignored by PUT method
 "customProperties": noDotsJson        optional, arbitrary JSON object containing customer
                                        specific properties of the asset store. Please
                                        note that it is not possible to change the data
                                        type of an existing value in the JSON object
                                        without deleting and recreating the asset store.
                                       Supported from R4.2.

}
```

### 3.6.2 Retrieve Store(s)

This request returns the actual state of the given storeId.

| **GET** | `/store/<storeId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | storeId does not exist |
| ***response*** | `@store`<br><br>See Section 3.6.1 for a definition of @store. | |

If storeId is omitted in the request URL, the response comprises a list of all stores of Ncanto.

| **GET** | `/store?count=<no_of_stores>&startAfter=<storeId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@store,...]`<br><br>See Section 3.6.1 for a definition of @store.<br>Stores are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

Up to R3.3 the properties of the active store can be queried with:

| **GET** | `/store-active` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | there is no active store available |
| ***response*** | `@store`<br><br>See Section 3.6.1 for a definition of @store. | |

Starting from R4.0 multiple active stores are supported. The above API call is still supported for backward compatibility reasons but is deprecated and, in case of multiple active stores, will only return one of them.

The following request shall be used to retrieve a list of all active stores:

| **GET** | `/store-active?all=true` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | there are no active stores present |
| ***response*** | `[@store]` | |

| | |
|---|---|
| | See Section 3.6.1 for a definition of @store. |

### 3.6.3 Delete a Store

Deleting a Store automatically deletes all Assets in the Store and the corresponding metadata.

| **DELETE** | `/store/<storeId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **409 CONFLICT** | Request cannot be executed because an ingest is ongoing. |

### 3.6.4 Retrieve custom properties of an asset store

Supported from R4.2. This request is used to retrieve the custom properties of the asset store.

| **GET** | `/store/<storeId>/customproperties` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | storeId does not exist |
| ***response*** | `customProperties json object` | |

### 3.6.5 Update custom properties of an asset store

Supported from R4.2. This request is used to create or to update the custom properties of a profile.

| **PUT** | `/store/<storeId>/customproperties` | |
|---|---|---|
| ***body*** | `Arbitrary JSON object with no dots in keys (noDotsJson). Shall not exceed 10kbyte per store.` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | storeId does not exist |

### 3.6.6 Activate and deactivate stores

Supported from R4.1.
With the help of this request, asset stores can be activated and deactivated individually.

| **POST** | `/store-active` | |
|---|---|---|
| ***body*** | `{`<br>`"add":[<storeId>],  optional, activate the given stores, leave others unchanged`<br>`"remove": [<storeId],  optional, deactivate the given stores, leave others unchanged`<br>`"set":[<storeId>]  optional, activate the given stores and deactivate all others`<br>`}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | At least one <storeId> in the "add" or "set" list is not found. Note that not found store IDs in the "remove" list are ignored. |

The attributes are processed in the order: add, remove, set, i.e. if set is present, add+remove are overruled, if the same storeId is present in add and remove, it will be removed.

### 3.6.7 Update or delete Assets

This request is used to create, update, or delete the metadata of a single or of a group of Assets in the Store identified by storeId. Note that assets can be modified in any of the existing stores, including the active one. Deleting assets is performed first, updating as second step. A generic filter to delete all matching assets is supported from R4.4. See 3.6.11for the definition of @genericFilter. Please note that indexing new assets is a heavy operation and may impact the performance of other API calls. The amount of assets created/updated in one API call has to be selected carefully. 200 assets per request is typically a good starting point. In case the number of assets is too high Ncanto may return 503 (unavailable) or 504 (gateway timeout) responses. Update asset requests shall be sent sequentially from a single thread. In case of multiple requests in parallel Ncanto may return 429 (too many requests).

| **POST** | /store/<storeId>/asset | |
|---|---|---|
| ***body*** | {"update":[@asset,...],<br>"deleteFilter": @genericFilter<br>"delete":[assetId,...]<br>}<br><br>• <storeId>: unique ID of the store updated<br>• <assetId>: unique ID (type: string512) of the Asset.<br>• See Appendix 1 for a detailed explanation of the metadata attributes of @asset. | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | storeId does not exist |
| | **429** | The update API is temporarily unavailable due to overload, retry after a short period. |
| | **503, 504** | The update API is temporarily unavailable due to overload, reduce the number of assets updated in one step. |

```
@asset =
{"assetId": string512,                  required, unique ID of the asset. Note that
                                          assetIds have to be unique across all active
                                          stores in the system.
 "assetGroups": [string],               optional, from R4.7, used to filter search phrase
                                          suggestions.
 "sourceGroupId": string,               optional, source group ID of the Source, e.g.
                                          "BBC_TV_channels", deprecated from R4.1
 "sourceGroupIds": [string],            optional, list of source group IDs of the source,
                                          e.g. ["BBC_TV_channels",
                                          "English_language_channels"]
 "sourceId": string,                    required, source ID of the asset, e.g. "BBC1"
 "sourceLogoUrl": string,               optional, URL of source logo)
 "availabilityStartTime": timestamp,    required, beginning of availability
 "availabilityEndTime": timestamp,      required, end of availability
 "durationSeconds": int,                required, duration in seconds
 "sequenceNo": int,                     optional, from R4.1, sequence number of the asset
                                          to be used for sorting
 "titles":{<lang>: <title>,             required (at least one title)
           <lang>: <title>,               <lang>: langCode, <title>:string
           ...},
 "programType": string,                 required, program type of the asset
 "assetType": string,                   required, distinguishing TV, VoD, OTT, ad
                                          assets, by default, recommendation engines use
                                          assets with assetType!="ad", ad engine and the ad
```

```
                                        interleaver only work with assets of type="ad".
                                        The used asset types are configurable by
                                        engine.
"assetFamilies": [string],              optional, from R5.4, multi-valued alternative to
                                        assetType.
"programId": string,                    optional, identifying assets representing e.g. the
                                        same movie
"genreFine":[string,...],               optional, list of fine granularity genres
"genreMedium":[string,...],             optional, list of medium granularity genres
"genreBroad":[string,...],              required, list of broad granularity genres
"originalStartTime": timestamp,         optional, original (broadcast) start time
"originalEndTime": timestamp,           optional, original (broadcast) end time
"publicationTime": timestamp,           optional, from R4.1, date/time of publication of
                                        the asset
"lastUpdateTime": timestamp,            optional, from R4.1, date/time of last modification
                                        of the asset, if missing, it is set automatically
                                        by update and ingest asset requests

"first": boolean,                       optional, indicates first time airing
"last": boolean,                        optional, indicates last airing
"live": string,                         optional, indicates if the asset is live/ live
                                        recorded, or not live
"pay": boolean,                         optional, pay flag
"onDemand": boolean,                    optional, indicates VoD assets
"seriesId": string,                     optional, identifies all episodes of a series
"season": int,                          optional, season number, >=0
"episode": int,                         optional, episode number, >=0
"episodesInSeason": int,                optional, number of episodes in the season, from
                                        R4.1, >=1
"ageCountry":{<country>: <age>,         optional, collection of country / age pairs
              <country>: <age>,           <country>: nonEmptyString, <age>: int
              ...},
"parentalGuidance": noDotsJson,         optional, from R4.1, arbitrary JSON object
                                        describing parental guidance rules
"tip":[string,...],                     optional, editorial tips, deprecated from R5.2
"structTip": [                          optional,  a structured alternative to the tip
                                        attribute, supports arbitrary integer/decimal
  {                                     number and string fields. Ignored by the
                                        preference engine if tip is set.
   <nonEmptyString>: int,               optional, integer type attribute of the tip
   <nonEmptyString>: decimal,           optional, decimal type attribute of the tip
   <nonEmptySstring>: string,           optional, string type attribute of the tip,
                                        e.g. vendor, type, etc. Note that numbers in
                                        double quotes are interpreted as strings.
     ...,                               optional, additional attribute of the tip
  },
  ...                                   optional, additional tips
],
"editorialRating":[string,...],         optional, editorial ratings
"structEditorialRating": [              optional, a structured alternative to the
                                        editorialRating attribute, supports arbitrary
  {                                     integer/decimal number and string fields. Ignored
                                        by the preference engine if editorialRating is
                                        set.
   nonEmptyString: int,                 optional, integer type attribute of the ed. rating
   nonEmptyString: decimal,             optional, decimal type attribute of the ed. rating
   nonEmptyString: string,              optional, string type attribute of the ed. rating,
                                        e.g. vendor, type, etc. Note that numbers in
                                        double quotes are interpreted as strings.
   ...,                                 optional, additional attribute of the ed. rating
```

```
   },
   ...                                  optional, additional editorial ratings
 ],
 "communityRating": decimal,            optional, calculated from the ratings of all
                                          subscribers, normalized to range [0:1]
 "communityViewCount": integer,         optional, calculated from all play interactions of
                                          all subscribers, from R4.4
"communityShareCount": integer,         optional, calculated from all share interactions of
                                          all subscribers, from R4.4
"communityTrendScore": decimal,         optional, calculated from the number of all
                                          interactions of all subscribers in the most
                                          recent 12h period compared to the pervious
                                          12h period, normalized to range [0:1],
                                          from R4.4
 "stereo": boolean,                     optional, stereo sound available
 "surroundSound": boolean,              optional, surround audio available
 "multiChannelAudio": boolean,          optional, indicates the availability of multiple
                                          independent audio channels, from R5.3
 "bw": boolean,                         optional, black & white
 "subtitled": boolean,                  optional, indicates availability of subtitles
 "wideScreen": boolean,                 optional, aspect ratio higher than 4:3
 "encoded": boolean,                    optional, encoded asset
 "forBlind": boolean,                   optional, audio commentary for sightless
 "hd": boolean,                         optional, high definition
 "interactive": boolean,                optional, interactive asset
 "threeD": boolean,                     optional, 3D
 "originalTitle": string,               optional, original title
 "originalSubtitle": string,            optional, original subtitle
 "seasonOriginalTitle": string,         optional, from R4.8, original title of the season
 "audioLang":[string,...],              optional, list of audio languages
 "defaultLanguage": langCode,           optional, from R4.5, the default language of the
                                          metadata
 "subtitleLang": [string],              optional, list of subtitle (closed caption)
                                          languages (R4.0)
 "themeCircle":[string,...],            optional, genre of theme channel assets
 "keywords":{<lang>: [<keyword>,...],   optional, list of language/list of keyword pairs
             <lang>: [<keyword>,...]      <lang>: langCode, <keyword>: string
            ...},
 "persons":{                            optional, cast of the asset
    <function>:[                        required, <function>: nonEmptyString, e.g. "actor"
                                          or "director", "script" or corresponding ID
      {
       "canonical": string,             required, main name of the person used by
                                          the engines
       "perLanguage":{                  optional, name of the person in different languages
          <lang>:<name>,                  <lang>: langCode, <name>: string
          ...
       },
       "character": {                   optional, from R4.1, the character the person is
                                          playing
         <lang>:<string>,               language / character name pairs
         ...
       }
      },
      ...                               additional persons of the same function
    ]
  ...                                   additional functions
 },
 "uiFunctionNames": {                   optional, from R4.1, language:function name pairs
```

```
    nonEmptyString: {                   function ID
       <lang>:<function_name>,          language - function name pairs
       ...
       },
       ...
},
"imageUrl":[string,...],              optional, URL of thumbnail images
"contentUrl":[string],                optional, streaming, download, etc. URL of asset
"media": [                            optional, supported from R4.1, attribute to
                                        describe any media (images, streaming content,
                                        trailers, etc.) associated with the asset.
                                        supports arbitrary integer/decimal number and
                                        string fields
  {
   nonEmptyString: int,               optional, integer attribute of the the medium
                                        (e.g. width pixels)
   nonEmptyString: decimal,           optional, decimal number attribute of the medium
   nonEmptyString>: string,           optional, string attribute of the medium, e.g.
                                        "url", "type", "encoding", ... Note that numbers
                                      in double quotes are interpreted as strings.
   ...                                optional, additional media attributes
  },
  ...                                 optional, additional media
],
"leaseDurationMinutes": int,          optional, lease duration in minutes
"viewCount": int,                     optional, maximum view count
"copyControl":[string,...],           optional, list of DRM attributes
"productionYearFirst": int,           optional, first year of production
"productionYearLast": int,            optional, last year of production
"productionCountry": [string,...],    optional, list of production countries
"uiProductionCountry": {              optional, from R4.1, to be used if
                                        productionCountry contains IDs
   <lang>: [string],                  language / list of production country pairs
   ...
},
"price":{<currency>: <price>,         optional, map of currency / price pairs
         <currency>: <price>,           <currency>: nonEmptyString, <price>: decimal
         ...},
"priceBin": int,                      optional, one of 0="free", 1="cheap",
                                        2="inexpensive", 3="not too expensive",
                                        4="premium", 5="expensive"
"profitability": decimal,             optional, indicates profitability of the asset
"targetDemographic":[string,...],     optional, demographic criteria of target viewer
"targetDevice":[string,...],          optional, list of devices the assets is suitable
                                        for
"subtitles":{<lang>: <subtitle>,      optional, list of language/subtitle pairs
             <lang>: <subtitle>,        <lang>: langCode, <subtitle>: string
             ...},
"seasonTitles":{<lang>: <seasonTitle>,optional, list of language/season title pairs
             <lang>: <seasonTitle>,     <lang>: langCode, <seasonTitle>: string
             ...},
"categories":{<lang>: <category>,     optional, list of language/category pairs
              <lang>: <category>,       <lang>: langCode, <category>: string
             ...},
"synopses":{                          optional, textual description of the asset
  <type>:                             required, <type>: nonEmptyString, e.g. "long" or
                                        "short to distinguish different synopsis types
        {<lang>:<synopsis>,           at least one required, <lang>: langCode,
                                        <synopsis>: string
```

```
          <lang>:<synopsis>,
          ...},
  <type>:
         {<lang>:<synopsis>,
          ...},
  ...},
"sourceNames":{                          optional, name of the source, e.g. "CNN"
  <type>:                                required, <type>: nonEmptyString, e.g. "long" or
                                           "short", to distinguish name spaces ("CNN" vs.
                                           "Cable News Corporation")
         {<lang>:<sourceName>,           at least one required, <lang>: langCode,
                                           <sourceName>: string
          <lang>:<sourceName>,
          ...},
  <type>:
         {<lang>:<sourceName>,
          ...},
  ...},
"sourceGroupNames":[                     optional, from R4.4, name of the source group
  {<lang>:<sourceGroupName>,             at least one required, <lang>: langCode,
                                           <sourceGroupName>: string
  <lang>:<sourceGroupName>,
  ...
  },
  ...],
"uiGenres":{                             optional, additional genre classifications for
                                           displaying purposes (annotation)
  <type>: [                              required, <type>: nonEmptyString, identifier of the
                                           classification, e.g. "long", "short". Types
                                           "broad", "medium", and "fine" shall contain the
                                           human-readable genre names. If present, those
                                           uiGenres are used in preference engine
                                           explanations.
           {<lang>:<uiGenre>,            at least one required, <lang>: langCode, <uiGenre>:
            <lang>:<uiGenre>,              string
            "id": <genreId>              optional, if genreBroad/Medium/Fine contain IDs,
                                           this should contain the ID to allow for correct
                                           human-readable explanations.
            ...},
           ...
          ],
  <type>: [
           {<lang>:<uiGenre>,
            ...}
          ],
  ...},
"uiProgramType":{<lang>: <prgT>,         optional, program type in human readable form
                 <lang>: <prgT>,           <lang>: langCode, <prgT>: string
                 "id": <prgtId>          optional, if programType contains IDs,
                                           this should contain the ID to allow for correct
                                           human-readable explanations.
                 ...},
"adProduct": string,                     optional, description of advertised product
"adCompany": string,                     optional, advertiser
"adTaxonomy":[string,...],               optional, ad classification
"customProperties": noDotsJson,          optional, any customer defined properties. Learned
                                           by Ncanto in preference profiles. Note that if
                                           any special mapping of attributes is required
                                           (e.g. timestamp, language map, or nested), you
```

```
                                        have to ask XroadMedia personnel to configure it.
 "clientCustomProperties": noDotsJson,  optional, from R4.5, any customer defined
                                        properties. Only useable for filtering, sorting,
                                        and as annotation. Note that if
                                        any special mapping of attributes is required
                                        (e.g. timestamp, language map, or nested), you
                                        have to ask XroadMedia personnel to configure it.

 "lightness": decimal,                  optional, describing the mood of the asset along
                                        the axis sad-serious-lighthearted-humorous
 "pace": decimal,                       optional, describing the mood of the asset along
                                        the axis slow-fast
 "relatedAssets": noDotsJson,           optional, from R4.1, links to related assets, e.g.
                                        trailers or bonus material related to the asset
                                        {"trailers":[<assetId>],"bonus":[<assetId>]}
 "metadataQuality": decimal,            optional, from R5.3, indicator of the richness and
                                        of the quality of the metadata. Assets with
                                        metadataQuality < 0.1 are masked in preference
                                        profiles. The threshold is configurable by
                                        XroadMedia personnel.
 "awards": [                            optional, from R5.2
  {
   "name": string,                      required, name of the award
   "uiName": {                          optional, name of the award in different languages
    <langCode>: <string>
   },
   "category": string,                  optional, category (e.g. best actress)
   "uiCategory": {                      optional, category in different languages
    <langCode>: <string>
   },
   "year": int,                         optional, year of the award
   "grade": string,                     optional, nominated / won / etc.
   "person": string                     optional, person that won the award
  }
 ]
}
```

### 3.6.8 Retrieve Assets by asset ID

This request returns all the metadata about the corresponding assets from a specific asset store. No annotation or sorting is supported, the request is intended for debugging purposes only. Please use the "retrieve assets by filter" request described in 3.6.11 with the "assetIds" parameter instead.

| **GET** | `/store/<storeId>/asset?id=<assetId>&id=<assetId>&id=<assetId>...` | |
|---|---|---|
| ***result*** | **200 OK** | All the given Assets have been returned. |
| | **400 BAD REQUEST** | Malformed request or body |
| | **404 NOT FOUND** | storeId does not exist or one or more of the assetIds do not exist |
| ***response*** | `[@asset,...]`<br><br>See Section 3.6.7 for the definition of @asset. | |

To retrieve asset metadata from the active asset store, use the following request (supported from R3.2.9):

| **GET** | `/store-active/asset?id=<assetId>&id=<assetId>&id=<assetId>...&includePlaylistAssets=true` |
|---|---|

| | | |
|---|---|---|
| ***result*** | **200 OK** | All the given Assets have been returned. |
| | **400 BAD REQUEST** | Malformed request or body |
| | **404 NOT FOUND** | storeId does not exist or one or more of the assetIds do not exist |
| ***response*** | `[@asset,...]`<br><br>See Section 3.6.7 for the definition of @asset. If includePlaylistAssets=true, assets are searched for in the active asset store as well as in the database of playlist asset metadata. | |

Note that the length of the parameter section of the retrieve assets URL is limited to 7000 characters, which limits the number of assets the metadata of which can be retrieved in one request.

### 3.6.9 Delete all Assets in a Store

Note that delete all can be applied to any store, including the active one.

| **DELETE** | `/store/<storeId>/asset` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | All the assets have been deleted. |
| | **404 NOT FOUND** | storeId does not exist |

### 3.6.10 Ingest Assets

This request deletes all Assets in the Store and triggers an asynchronous full import of metadata. The mood attributes "lightness" and "pace" are not available in Funke metadata but will be generated automatically during ingestion by Ncanto.

The lastUpdateTime attribute of all assets is set to the current time. The following asset types are ingested:

- broadcast assets from the epg section of the OPML file
- vod assets from the vod section of the OPML file
- advertising assets from the ad section of the OPML file
- theme channel assets from the channel section of the OPML file (starting from R4.1)

Starting from R4.1 the type of assets ingested may be limited via the “types” attribute.

| **POST** | `/store/<storeId>` | |
|---|---|---|
| ***body*** | `{"protocol":"IPDS",` | `required`<br>`required`<br>`optional, list of OPML outlines to be processed` |
| | `"feed":<feedUrl>`<br>`"types":[<enum>]`<br><br>`}`<br><br>• protocol: identifies the protocol to be used, currently protocol "IPDS", corresponding to Funke IPDS format, is supported<br>• <feedUrl>: URL of metadata feed<br>• types: determines which asset types (which OPML sections) are processed | |
| ***result*** | **202 ACCEPTED** | Request successful, import is ongoing in the background |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

From R3.3.4, a full import can be done from a compressed metadata file too. The metadata has to be included in the request body. Note that ingest from file uses all metadate files in the archive, an OPML file is not required. Hence filtering to specific types is not supported.

| **POST** | /store/<storeId>/ingest/archive | |
|---|---|---|
| ***body*** | multipart/form-data:<br><br>Part 1:<br>type: application/json<br>{"protocol":"IPDS", required<br>"feed":<string>, required, ignored<br>"types": [<enum>] ignored,<br>}<br><br>Part 2:<br>type: application/octet-stream<br>metadata_file.zip in Funke EPGdata format. | |
| ***result*** | **202 ACCEPTED** | Request successful, import is ongoing in the background |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

The following example based on command line tool curl illustrates how the multipart/data-form request has to look like for file based import:

```
curl -s -i -F 'json={"protocol":"IPDS", "feed": "epg.zip"};type=application/json' -F
'file=@<filename.zip>;type=application/octet-stream' -XPOST
http://<hostname>/store/<storeId>/ingest/archive
```

### 3.6.11 Retrieve Assets by filter criteria

This request returns metadata of selected assets from the active asset store and is typically used to populate program guides. Optionally, playlist assets can be included in the response too.
In contrast to the retrieve assets by asset ID request (cf. Section 3.6.8), the assets can be filtered by arbitrary criteria and the response can be fine tuned using an annotation template.

Starting from R4.1 pagination of results via the paginationOffset and paginationLimit query parameters is supported. Note that the numberOfItems specification of the context is still observed, but only a subset of the result is returned.

| **GET** | /store-active/asset?adHocContext=@global&includePlaylistAssets=boolean&subscriber=<string>&paginationLimit=<integer>&paginationOffset=<integer>&trackSearchTerm=<string> | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | adHocContext | Mandatory, type @global. |
| | includePlaylistAssets | Type: boolean. If true, the request uses all playlist assets as candidates too. |
| | subscriber | Type: string. Mandatory if sorting by lineup or lineupFilter are used in the adHocContext. If present, the subscriber's entitlement will be applied to the results. |
| | paginationLimit | Type: positive integer. Optional, if present, only the specified number of results is returned. |
| | paginationOffset | Type: non-negative integer. Optional, if present, the first offset results will be omitted from the response. |

| | | |
|---|---|---|
| | `trackSearchTerm` | Type: string, optional, supported from R4.8.6. If present, the given string will be tracked as search term. |
| ***body*** | - | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | One or more of the Ids referenced in the adHocContext or the subscriberId do not exist |
| ***response*** | `[@augmentedAsset,...] if filterFacets are not included in the adHocContext {"recommendations": [@augmentedAsset], "attributeValues": [@attributeValues]} if filterFacets are included in the adHocContext`<br><br>The metadata of assets is filtered according to the given annotation. If includePlaylistAssets=true, assets are searched for in the active asset store as well as in the database of playlist asset metadata. The subscriber ID is used for blacklisting only. | |

| ***POST*** | `/store-active/asset?includePlaylistAssets=boolean&subscriber=<string>&paginationLimit=<integer>&paginationOffset=<integer>&trackSearchTerm=<string>` | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | `includePlaylistAssets` | Type: boolean. If true, the request uses all playlist assets as candidates too. |
| | `subscriber` | Type: string. Mandatory if sorting by lineup or lineupFilter are used in the adHocContext. If present, the subscriber's entitlement will be applied to the results. |
| | `paginationLimit` | Type: positive integer. Optional, if present, only the specified number of results is returned. |
| | `paginationOffset` | Type: non-negative integer. Optional, if present, the first offset results will be omitted from the response. |
| | `trackSearchTerm` | Type: string, optional, supported from R4.8.6. If present, the given string will be tracked as search term. |
| ***body*** | `@global mandatory, the adHocContext` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | One or more of the Ids referenced in the adHocContext or the subscriberId do not exist |
| ***response*** | `[@augmentedAsset,...] if filterFacets are not included in the adHocContext {"recommendations": [@augmentedAsset], "attributeValues": [@attributeValues]} if filterFacets are included in the adHocContext`<br><br>The metadata of assets is filtered according to the given annotation. If includePlaylistAssets=true, assets are searched for in the active asset store as well as in the database of playlist asset metadata. The subscriber ID is used for blacklisting only. | |

The request's adHocContext is based on the global context parameters (cf. Section 3.16.1). In addition to the global context attributes, the following special program guide attributes are supported from R4.1:

- "beforeAfter" to implement "what's on now and what's next" features. In contrast to simple availability window filtering, the beforeAfter feature allows to retrieve a specified number of previous and/or next assets per source.
- "assetIds" attribute to retrieve the metadata of selected assets. Note that "assetIds"=["id1",´"id2"] is equivalent to filter={"term":"assetId", "operator":"in", "values":["id1", "id2"]}

```
@global =
{"assetTypes": [string],              optional, defines the asset types to be retrieved.
                                        Note that for compatibility to earlier releases
                                        the asset type "ad" is reserved for advertising
                                        assets and is never returned by retrieve assets
                                        requests. If "assetTypes" is not specified, all
                                        asset types with the exception of "ad" are used.
                                        Supported from R4.0
 "inputs": [string,...],              optional, list of sourceGroupIds used as input,
                                        empty list means all sourceGroups will be used
 "grouping": enum,                    optional, one of "none", "program", "series",
                                        default="none", deprecated from R4.6
 "groupingAttributes": [string],      optional, list of asset attribute names in dot
                                        notation. The first attribute that is present in
                                        an asset will define the asset's group. Only leaf
                                        attributes are supported, no wildcards. Use
                                        "groupingAttributes":["seriesId","programId"] as
                                        replacement of the deprecated "grouping":"series"
                                        configuration. Supported from R4.6.  Overrules
                                        "grouping" if both are present in the context.
 "groupingPriority": enum,            optional, only episode & episodeForced supported
                                        before R5.6, allowed values
                                        "businessScore", "firstUnwatchedEpisode",
                                        "episode" (deprecated, same as
                                        firstUnwatchedEpisode),
                                        "firstUnwatchedEpisodeForced", "episodeForced"
                                        (deprecated, same as
                                        firstUnwatchedEpisodeForced), "firstEpisode",
                                       "lastEpisode", "latestArrival", "earliestArrival",
                                        default "businessScore". Determines which
                                        episodes are first selected to populate each
                                        series group. By default the assets are selected
                                        randomly if the business scores of the episodes
                                        are identical (groupingPriority=businessSCore).
                                        If firstUnwatchedEpisode priority is selected,
                                        the first not yet watched episode of each series
                                        are used. If firstUnwatchedEpisodeForced is
                                        specified, the first not yet watched episode is
                                        selected even if there are other episodes with
                                        higher business scores. firstEpisode and
                                        lastEpisode use the lowest / highest season /
                                        episode numbers available, ignoring which of them
                                        have already been watched. latestArrival /
                                        earliestArrival use the episodes with earlierst /
                                        latest availability star time to populate the
                                        group. Only effective if type of items =
                                        groups and if seriesId is among the grouping
                                        attributes.
 "numberOfItems":int,                 required, number of items to be returned
 "typeOfItems": enum,                 optional, one of "assets", "groups", default =
                                        "assets"
 "maxGroupSize": int,                 optional, defines the max. No. of assets per group,
                                        ignored if typeOfItems = assets
 "windowOfAvailabilityType": enum,    optional, one of "overlap", "strictStart",
                                        "strictEnd", "strict". Default: "overlap".
                                        Defines the interpretation of window of
                                        availability and relative window of availability.
                                        If set to overlap, any overlap between the
                                        asset's availability and the specified window of
```

```
                                        availability will be allowed. If set to
                                        "strictStart" or "strictEnd", the availability
                                        start / end of the asset has to lie within the
                                        specified window of availability. If set to
                                        "strict", both the asset's availability start and
                                        also availability end have to lie within the
                                        specified window of availability. In case of
                                        multiple start / end times one of them has to
                                        fulfill the availability conditions.
"windowOfAvailabilityPriority": enum,  optional, supported from R4.2, allowed values
                                        "relative", "tightest", default value "relative"
                                        Relative means that if a relative window is
                                        specified, it overrides any absolute window.
                                        Thightest means that the more stringent of the
                                        relative and the absolute limits is used.
                                        Applies to the generic as well as to the per
                                        source window of availability.
"windowOfAvailabilityStart": timestamp, optional, default –infinity, defines the
                                          absolute time window in which results shall be
                                          available.
"windowOfAvailabilityEnd": timestamp,  optional, default +infinity, defines the
                                        absolute time window in which results shall be
                                        available.
"relativeWindowOfAvailabilityStart":   optional,, defines the relative time
                                        window in which recommendations shall be
                                        available.
   {"origin": enum,                    required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": duration                 required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, defines the relative time
                                        window in which recommendations shall be
                                        available.
   {"origin": enum,                    required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": duration                 required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, supported from R4.2, overrules the
                                        generic window of availability definition for
                                        selected source IDs.
     <sourceId>: {                      the sourceId the special window is applicable to
     "start": timestamp,                optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": duration               required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": duration               required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
```

```
      <sourceGroupId>: {              the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
                                        used)
      "start": timestamp,               optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
},
"availabilityWindows": [enum],        optional, allowed values: "originalStartEndTime" or
                                        "availabilityStartEndTime", default value
                                        ["availabilityStartEndTime"]. Defines which asset
                                        attribute pairs determine the asset's
                                        availability. By default only
                                        availabilityStartTime / availabilityEndTime are
                                        used. This can be replaced or extended by
                                        originalStartTime / originalEndTime. Supported
                                        from R5.1.
"country": string,                    optional, used to select the desired age
                                        rating
"language": langCode                  optional, determines the language of annotations
                                        and explanations. If language is present,
                                        "languages" and "fallbackLanguages" are ignored
                                        and the legacy behavior applies: all language
                                        maps are filtered to the single language desired,
                                        if that is not available an empty object is
                                        returned and no fallbacks apply.
"languages": {                        optional, from R4.5, ignored if "language" is given
  "sequence": [langCode],             required, the first available <count> languages
                                        from the list are included in the response. If
                                        none of the languages is available in the
                                        metadata, the "fallbackLanguages" are used.
  "count": int                        required,
 },
"fallbackLanguages": {                optional, from R4.5, ignored if "language" is given
  "sequence": [langCode],             required, the first available <count> languages
                                        from the list are included in the response if
                                        none of the preferred languages is available in
                                        the metadata. If none of the fallbackLanguages is
                                        available in the metadata, the defaultLanguage of
                                        the asset is used. If the asset has no
                                        defaultLanguage or if the defaultLanguage is not
                                        available, no language filtering is done at all,
                                        all available languages are returned.
  "count": int                        required,
 },
"minAge":int,                         optional, if "country" is specified, the
                                        corresponding age rating will be used. If
                                        "country" is not specified, the highest minAge of
                                        all countries available will be used. Note that
```

```
                                    the global minAge limit may be overruled by per-
                                    engine minAge values.
"filter": @genericFilter            optional, allows to filter recommendations to any
                                    asset attribute value
"subscriberBlacklisting": boolean,  optional, default = false. If
                                    subscriberBlacklisting is enabled, the subscriber
                                    ID becomes a required parameter of the
                                    recommendation request
"sorting": enum,                    optional default = "byTime", supported
                                    values: "byTime", "byLineup", "byEpisode",
                                    "multi" or any asset attribute in dot notation.
                                    From R4.8 "random" sorting is supported, too. See
                                    Section 2.3.6 for details. You can change the
                                    sorting direction using sorting=multi.
"multiSorting": [@sortingCriterion], required if "sorting"="multi", ignored otherwise
"annotation": string,               optional, annotationId to be used for all asset
                                    types
"annotationAttributes":[string],    optional, from R4.1, list of asset attributes in
                                    dot notation, will be merged with "annotation" if
                                    both are present
"conditionalAnnotationAttributes":[@condAnn],optional, from R4.1, list of conditional
                                    annotation attributes, see Section 3.10.1, will
                                    be merged with "annotation" if both are present
"annotationPerAssetType": {         optional, supported from R4.0, individual
  <assetType>: string,              annotations per asset type. If present, the
  ...                               global annotation is selectively overruled for
 }                                  the corresponding asset types.
"entitlement":[string,...],         optional, list of entitlements, empty list means
                                    all sources will be used. If provided, an
                                    entitlement defined here is intersected with the
                                    subscriber's default entitlement (until R3.3) or
                                    overrules the default (from R4.1) entitlement of
                                    the subscriber. Starting from R4.1 the union of
                                    entitlement, entitlementSources and
                                    entitilementSourceGroups is used.
"entitlementSources": [string],     optional, supported from R4.1, list of source IDs
                                    to be used as entitlement. The union of
                                    entitlement, entitlementSources and
                                    entitilementSourceGroups is used.
"entitlementSourceGroups": [string], optional, supported from R4.1, list of sourceGroup
                                    IDs to be used as entitlement. The union of
                                    entitlement, entitlementSources and
                                    entitilementSourceGroups is used.
"device":[string,...],              optional, list of device identifiers
"timeZone": timezone,               optional, time zone of the client used to evaluate
                                    time of day based business rules
"sourceLineupName": string,         optional, reference to one of the subscriber's
                                    source lineups. If byLineup is one of the sorting
                                    criteria, either sourceLineupName or sourceLineup
                                    has to be specified.
"sourceLineup": [sourceId],         optional, list of source IDs to be used for
                                    sorting. If byLineup is one of the sorting
                                    criteria, either sourceLineupName or sourceLineup
                                    has to be specified. If both are present,
                                    sourceLineup overrules sourceLineupName.
"lineupFilter": {                   optional, if present, the results are filtered to
                                    (a subset of) the lineup filter defined by
                                    sourceLineupName or sourceLineup. Supported from
                                    R4.7.
```

```
   "startIndex": int,                optional, default 0, specifies the index of the
                                       first sourceId in the lineup which should be
                                       included in the results
   "endIndex": int                   optional, default infinity, specifies the index of
                                       the last sourceId in the lineup which should be
                                       included in the results
 },
 "beforeAfter":                      optional, supported from R4.1, specifies to return
                                       a number of assets around the origin. Typically
                                       used for "what's on now and what's next"
                                       features. Origin may be an absolute timestamp or
                                       "currentTime". Offset points to the first asset
                                       to be returned, limit defines the number of
                                       assets returned. Note that "limit" assets per
                                       source (typically per TV channel) are returned.
    {"origin": timestamp or enum,    required, either a timestamp in xsd:datetime format
                                       or "currentTime"
     "offset": int,                  required, first asset to be returned. 0: asset
                                       aired at "origin" time, 1: next asset, -1:
                                       previous asset, ...
     "limit": int                    required, number of assets to be returned per
                                       sourceId
    },
 "assetIds": [string512],            optional, supported from R4.1, returns the metadata
                                       of the given assets only.
 "assetStores": [storeid],           optional, if provided, the request will use
                                       the given asset store(s) instead of the active
                                       asset store(s).
                                       Supported from R4.1.
  "filterFacets": {                  optional, from R4.4, returns the specified number
                                       of attribute values of the given asset
                                       attributes. Note that if filterFacets are
                                       requested, the response format changes from
                                       [@augmentedAsset] to ["recommendations":
                                       [@augmentedAsset], "attributeValues":
                                       [@attributeValues].
    <attribute>: <count>               leaf attribute of @asset in dot notation: number
  },                                   of values to be returned (0=unlimited)
                                       Only leaf attribute names without wildcards are
                                       supported, branch attributes and attributes
                                       including wildcards are ignored. <attribute>:
                                       string, <count>: int.
  "personalization": enum            optional, from R4.5, one of "none" or "subscriber",
                                       if set to "subscriber", the results are scored in
                                       all profiles of the subscriber. The
                                       subscriber ID has to be present in the
                                       recommendation request in that case. Default
                                       value "none".
}
```

**@genericFilter** =

```
{   "term": string,                  required if operator is not "and", "or", or
                                       "nestedAnd", name of the asset attribute, e.g.
                                       "uiGenre.<type>.<lang>". All asset attributes are
                                       supported. If term is specified, the filter
                                       consists of one single criterion and a value has
                                       to be specified.
    "value": string,                 required if "term" is defined and operator is not
                                       "in", "exists", or "notExists", value of the
```

```
                                        asset attribute, not allowed if operator is "in",
                                        "exists", "notExists".
    "values": [string],                 required if "term" is defined and "operator"="in",
                                        not allowed for other operators, supported from
                                        R4.0
    "operator": enum,                   optional, one of "and", "or", "equals",
                                        "notEquals", "contains", "notContains", "atMost",
                                        "atLeast", "smaller", "larger", "in", "exists",
                                        "notExists", starting from R4.4 "notIn", starting
                                        from R4.6 "nestedAnd". If operator is one of
                                        "and", "or", "nestedAnd": "criteria" have
                                        to be specified. If "operator" is one of
                                        "equals", "notEquals", "contains", "notContains",
                                        "atMost", "atLeast", "smaller", "larger": "term"
                                        and "value" have to be specified. If operator is
                                        "in", "notIn": "term" and "values" have to be
                                        specified. If operator is "exists", "notExists":
                                        "term" has to be specified.If "operator" is
                                        missing, the default value "equals" is assumed.
                                        The allowed combinations of "term" and "operator"
                                        are listed in Table 1. Note that "notEquals"
                                        returns true if the given metadata attribute is
                                        not present at all.
    "criteria": [@genericFilter]        required if "operator" equals "and", "nestedAnd",
                                        or "or".
  }
```

If a filter criterion is applied to an attribute of type list, the criterion yields true if any of the list elements fulfills it. If multiple filter criteria combined by operator "and" are applied to a list, the result is true if each of the individual criteria is matched by at any list element. The customProperties and clientCustomProperties attributes of the asset may contain list of object data structures where multiple criteria shall be fulfilled by the same list element. In such cases the "nestedAnd" operator shall be used.

Example to illustrate the difference between "and" and "nestedAnd":
Given the two documents

```
{
  "documentId": 1,
  "contributions": [
    {
      "firstName": "Alice",
      "lastName": "Smith"
    },
    {
      "firstName": "Bob",
      "lastName": "Taylor"
    }
  ]
}
{
  "documentId": 2,
  "contributions": [
    {
      "firstName": "Alice",
      "lastName": "Taylor"
    }
  ]
```

```
}
```

a filter

```
{
  "operator": "and",
  "criteria": [
    {
      "term":"firstName",
      "value":"Alice"
    },
    {
      "term":"lastName",
      "value":"Taylor"
    }
  ]
}
```

would return both documents (in document 1 the first contributor matches Alice, the second matches Taylor), while the filter

```
{
  "operator": "nestedAnd",
  "criteria": [
    {
      "term":"firstName",
      "value":"Alice"
    },
    {
      "term":"lastName",
      "value":"Taylor"
    }
  ]
}
```

would only return document 2 (because there the same list element matches both criteria).
Note that the "nestedAnd" operator can only be used for asset atttributes which have been configured for nested mapping. Please consult XroadMedia if you want to use the "nestedAnd" functionality.

| attribute data type | allowed operators |
|---|---|
| string | equals, notEquals, exists, notExists, in, notIn. Until R4.7 operators contains and notContains exhibit the same behavior as equals/notEquals. From R4.8 contains/notContains execute a substring filtering at token level, i.e. at complete word level. If "programType": "live sports", programType contains sports will return true, but programType contains sport will be false. Contains and notContains operators only work in searchable string attributes, i.e. titles, uiGenres, genreBroad, genreMedium, genreFine, originalTitle, keywords, categories, persons, subtitles, synopses, sourceNames, productionCountry. |
| int | equals, notEquals, atMost, smaller, atLeast, larger, exists, notExists, in, notIn |
| decimal | equals, notEquals, atMost, smaller, atLeast, larger, exists, notExists, in, notIn |
| boolean | equals, notEquals, exists, notExists, in, notIn |
| timestamp | equals, notEquals, atMost, smaller, atLeast, larger, exists, notExists, in, notIn |
| enum | equals, notEquals, exists, notExists, in, notIn |

| | |
|---|---|
| Json, noDotsJson | equals, notEquals, exists, notExists, in, notIn. Until R4.7 operators contains and notContains exhibit the same behavior as equals/notEquals. From R4.8 contains/notContains execute a substring filtering at token level, i.e. at complete word level. If "programType": "live sports", programType contains sports will return true, but programType contains sport will be false. Contains and notContains operators only work in searchable string attributes, i.e. titles, uiGenres, genreBroad, genreMedium, genreFine, originalTitle, keywords, categories, persons, subtitles, synopses, sourceNames, productionCountry. |

**Table 1: Allowed data type / operator combinations in @genericFilter**

```
@sortingCriterion =
{"criterion":enum,                    required, one of "byScore", "byTime",
                                        "byLineup",  "byEpisode", "byEngine",
                                        "byGroupMaxScore", "byBusinessScore".
                                        Any asset attribute in dot notation is supported
                                        too. See Section 2.3.6 for details. From R4.8
                                        "random" sorting is supported, too.
 "condition": @genericFilter,         optional, from R4.6. If the criterion points to a
                                        list of objects with nested mapping the condition
                                        may be used to select which element of the list
                                        should be used for sorting. Please note that this
                                        only applies to data structures inside
                                        customProperties and clientCustomProperties, and
                                        that nested mapping has to be configured by
                                        XroadMedia personnel.
 "direction": enum,                   optional, one of "ascending", "descending", default
                                        value of criterion used if missing, see Section
                                        2.3.6 for details.
 "sequence": [string]                 optional, supported from R4.5, Values in the
                                        sorting sequence are always sorted to the
                                        beginning of the result list, in the order of
                                        their appearance in the sequence. The remaining
                                        values are sorted according to the specified
                                        direction. Note that the sorting sequence only
                                        works for single value leaf attributes.
}

@condAnn =
{<attribute>:<@genericFilter,         at least one attribute:criterion pair required
                                        <attribute>: asset attribute name in dot notation
                                        @genericFilter: condition which has to be met for
                                        the attribute to be returned, syntax see above.
}


@augmentedAsset =
  {"asset": @asset  ,                 assetId and annotation of the asset based on the
                                        annotationId defined in the context.
                                        availabilityStartTime and availabilityEndTime are
                                        converted to the subscriber's timezone (if known)
   "groupId": string,                 ID of the recommendation group the asset belongs
                                        to. groupIds are consecutive integer number
                                        strings "0", "1", "2", ... generated by Ncanto to
                                        indicate assets which represent multiple
```

```
                                        occurrences of the same program or episodes of
                                        the same series
   "groupSize": int                   size of the group, i.e. the number of assets having
                                        the same groupId as this one. Supported from R5.3
  }

@attributeValues = {
   "attribute": <attribute_name>,      name of the attribute in dot notation
   "values":[
      {"value": <value>,               value of the attribute
       "count": integer                number of assets carrying the attribute value
      }
    ]
}
```

### 3.6.12 Retrieve metadata attribute values

This request returns the list of values of a specific asset attribute currentl y used in the active asset store or in the active asset store and by playlist assets.

| **GET** | `/store-active/attribute/<attribute_name>?includePlaylistAssets=true&assetStores=[<storeid>&condition=@genericFilter` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| ***parameters*** | If `includePlaylistAssets=true` is set, not only the active asset store but also playlist assets are taken into account.<br>If `assetStores` are specified, only the assets in the given stores will be considered.<br>If a `condition` is specified, only the assets which fulfil the condition will be considered in the attribute value search. The condition syntax `@genericFilter` is defined in Section 3.6.11. | |
| ***response*** | `{"attribute": <attribute_name>, "values":[{"value": <value>, "count": integer},...]}`<br><br><attribute_name> is the name of one asset attribute specified in dot notation, e.g. "genreBroad" or "persons.actor.english".<br><br>The response contains a list of values of the given attribute together with the number of matching assets. Example: {"attribute":"persons.actor.english", "values":[{"value":"Clint Eastwood", "count": 100},{"value":"Robert De Niro", "count": 200}]}<br><br>If the attribute name given is not pointing to values but represents an intermediate level of the data structure, the corresponding keys are still returned but the counts are omitted. Example: {"attribute":"persons.actor", "values":[{"value":"english},{"value":"german}]} | |

### 3.6.13 Suggest metadata attribute values (search phrase suggestion)

This request returns search phrase suggestions together with the asset attribute name the suggestion was found in. Typical use case of the feature is to provide autocompleted suggestions while the user is typing a search query. If the same suggestion is found in multiple asset attributes, only one is returned and the duplicates are removed. Starting from R4.7 independent search phrase databases are created per asset group (see the assetGroups attribute in Section 3.6.7). The assetGroups parameter allows to select which asset groups the search phrase suggestions should be made from.
Starting from R4.8 suggestions can be filtered to one language. Please note that the default duplicate removal is deactivated if language filtering is active.

| **GET** | /store-active/search-phrase/<search_term>?numberOfItems=<int>&attributeContributions={<asset_attribute_dot_notation_with_wildcards>:<decimal>,...}&assetGroups=[string]&language=<langcode>&languages=[<langcode>] | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| ***parameters*** | numberOfItems: required if attributeContributions are set. Defined the total number of items to be returned. The results will be distributed evenly among the attributes.<br>attributeContributions: if missing, the results are derived from all default attributes used for search. If present, only the specified subset of the search attributes is used. If the attribute is not a leaf of the asset object tree, a wildcard has to be used, e.g. persons.* or persons.*.perLanguage.english. The relative contribution of the attribute to the numberOfAsset results is determined by the decimal value.<br>assetGroups: supported from R4.7. If provided, only those assets which carry the given asset group value will be used for suggestions. If missing, all assets are used (the ones assigned to asset groups as well as the unassigned ones).<br>language: deprecated from R5.0, replaced by languages. If provided, the phrase suggestions are provided from the given language only.<br>languages: if provided, the phrase suggestions are provided from the given languages only. | |
| ***response*** | [{"attribute": <attributeName>, "phrase": <phrase>}, …]<br><br><attribute_name>: asset attribute name in dot notation, e.g. "genreBroad" or "persons.actor.perLanguage.english".<br><br><phrase>: auto-completed search term, e.g. "Brad Pitt" | |

## *3.7 Entitlement Template Management*

Entitlement templates define which sources Ncanto may use to recommend assets. Entitlement templatess can be associated with a subscriber (default entitlement of the subscriber) and/or with a context. If both are specified, Ncanto will use the entitlement from the context only. From R4.6 a entitlements can also be specified in generic filter syntax.
As an alternative to defining entitlement templates, the entitlement may be specified directly in the context, too. See the entitlementSources, entitlementSourceGroups, and entitlementFilter attributes of @subscriber and @context.

### 3.7.1 Create or Update an Entitlement Template

This request is used to create a new or to update an existing entitlement template.

| **PUT** | /entitlement/<entId> | |
|---|---|---|
| ***body*** | @entitlement | |
| ***result*** | **204 NO_RESPONSE** | Request successful |
| | **400 BAD_REQUEST** | Request body is syntactically incorrect |

```
@entitlement =
{"entitlementId": string512,          optional, unique identifier of the entitlement,
                                        ignored by the PUT method
 "sources": [string,...],             at least one required, list of source IDs the
                                        entitlement is composed of. If "sourceGroups" or
                                        "filter" are also present, the union is used.
 "sourceGroups": [string,...],        optional, list of sourceGroupIds the entitlement is
                                        composed of. Supported from R4.3. If "sources" or
                                        "filter" are also present, the union is used.
 "filter": @genericFilter             optional, from R4.6, generic filter expressing the
                                        entitlement. This allows any asset attribute to
```

```
                                        be used to define the entitlement. If
                                        "sources" or "sourceGroups" are also present, the
                                        union is used.
}
```

### 3.7.2 Retrieve one or all Entitlement Template(s)

This request is used to retrieve an entitlement template.

| **GET** | /entitlement/<entId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | entId does not exist |
| ***response*** | @entitlement | |

If entId is omitted from the request URL, a list of all entitlement templates in the database is returned:

| **GET** | /entitlement?count=<no_of_entIds>&startAfter=<entId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | [@entitlement,...]<br>Entitlement IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000. Even if count is not specified, maximum 1000 are returned. | |

### 3.7.3 Delete an Entitlement Template

This request is used to delete one entitlement template.

| **DELETE** | /entitlement/<entId> | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | entId successfully deleted or not found |

## *3.8 Subscriber Management*

### 3.8.1 Create or Update Subscriber

This request is used to create a new subscriber or to update the attributes of an existing one.

| **PUT** | /subscriber/<subId> | |
|---|---|---|
| ***body*** | @subscriber | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request or body |

```
@subscriber =
{"subscriberId": string512,             optional, unique ID of the subscriber, ignored by
                                          PUT request
 "customProperties": noDotsJson,        optional, arbitrary JSON object without dots in the
                                          keys, may not be used to store any personally
                                          identifiable information and shall not exceed
                                          1kbyte per subscriber.
 "subscriberGroups":[string,...],       optional, list of (static) subscriber groups the
                                          subscriber belongs to
 "dynamicSubscriberGroups": [string],   optional, list of dynamic subscriber groups the
                                          subscriber belongs to. Ignored by PUT requests,
                                          calculated on the fly from the dynamic subscriber
                                          group definitions (see Section 0) by GET
                                          requests. Supported from R5.4.
 "blacklist": @blacklist,               optional, see definition below
 "entitlement":[string,...],            optional, list of entitlement (template) IDs of the
                                          subscriber, See Section 3.7 for details. The
```

```
                                        union of entitlement, entitlementSources,
                                        entitilementSourceGroups, and entitlementFilter
                                        is used.
 "entitlementSources": [string],        optional, list of source IDs to be used as
                                        entitlement. The union of entitlement,
                                        entitlementSources, entitilementSourceGroups, and
                                        entitlementFilter is used
 "entitlementSourceGroups": [string],   optional, list of sourceGroup  IDs to be used as
                                        entitlement. The union of entitlement,
                                        entitlementSources, entitilementSourceGroups, and
                                        entitlementFilter is used.
 "entitlementFilter": @genericFilter    optional, from R4.6, generic filter expression
                                        defining the entitlement. The union of
                                        entitlement, entitlementSources,
                                        entitlementSourceGroups, and entitlementFilter is
                                        used.
 "sourceLineups": {                     optional, supported from R4.0, defines the source
   <lineupName>: [string],              lineups of the subscriber used for sorting. Each
   ...                                  source lineup comprises a list of source IDs
 }
 "profiles":[string,...],               optional, list of profile IDs of the subscriber,
                                        see  Section 3.8.42 for details
 "topics":[@topic,...],                 optional, list of topics of the subscriber
 "playlists": [string],                 optional, list of playlist IDs
                                        of the subscriber
 "bookmarkLists": [string],             optional, list of bookmark list IDs
                                        of the subscriber, supported from R5.6
 "expiryTime": timestamp OR duration,   optional, from R4.1, subscribers are automatically
                                        deleted after expiry if the feature is enabled
                                        via the provisioning system. Durations have to be
                                        positive values.
 "lastActivityTime": timestamp          optional, from R4.1, automatically updated if
                                        enabled in the provisioning system
}
```

**@blacklist (See Section 4.8 for details) =**

```
{"assets":[                             optional, list of blacklisted assets
           {"time": timestamp,          optional, time the asset has been blacklisted
            "assetId": string512        required, asset ID
           },...
          ],
 "programs":[                           optional, list of blacklisted programs
             {"time": timestamp,        optional, time the program has been blacklisted
              "programId": string       required, program ID
             },...
            ],
 "series":[                             optional, list of blacklisted series
           {"time": timestamp,          optional, time the series has been blacklisted
            "seriesId": string          required, series ID
           },...
          ]
}
```

**@topic (see 3.8.24 for details) =**

```
{"owner":string,                        required
 "topicId":string,                      required
 "topicName": string,                   optional, the clear text title of the topic
 "topicUrl": [string]                   optional, URLs of the social network pages the
```

```
                                        topic was derived from
}
```

Note that creating or updating a subscriber with a new profileId or topicId does not automatically set up the profile or the topic. Update topics (3.8.23) and update profile (3.9.1) have to be called by the client separately.

### 3.8.2 Retrieve one or multiple Subscriber(s)

| **GET** | `/subscriber/<subId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `@subscriber`<br><br>Empty attributes are omitted from the response. See Section 3.8.1 for the definition of @subscriber. | |

| **GET** | `/subscriber?count=<no_of_subs>&startAfter=<subId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `[<subId>,...]`<br><br>Subscriber IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

Expanded information about one subscriber's properties can be retrieved with one single request. If the elements parameter is used, Ncanto will

- filter the subscriber's properties and only return those addressed by @elements
- if requested, retrieve the subscriber's profiles, entitlements, and playlists, and return the addressed attributes of those objects too.

The elements to be returned are addressed in dot notation, i.e. profiles.filter will return the filters of all profiles. Wildcards or indexing in lists are not supported.

| **GET** | `/subscriber/<subId>?elements=@elements` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `@expandedSubscriber`<br><br>Empty attributes are omitted from the response. | |

```
@elements =
[
 "<element>.<element>"...,            at least one required, path to the element which
                                        should be returned. E.g.
                                        ["profiles.profileId",
                                        "profiles.name",
                                        "customProperties"] will omit all
                                        subscriber attributes except customProperties and
                                        profiles, and will, for all profiles, return
                                        profile ID and profile name.
 ...
]
```

```
@expandedSubscriber =
{"subscriberId": string512,            optional, unique ID of the subscriber, ignored by
                                         PUT request
 "customProperties": noDotsJson,       optional, arbitrary JSON object, May not be used to
                                       store any personally identifiable information and
                                       shall not exceed 1kbyte per subscriber.
 "subscriberGroups":[string,...],      optional, list of subscriber groups the subscriber
                                       belongs to
 "blacklist": @blacklist,              optional, see definition below
 "entitlement":[@entitlement,...],     optional, list of entitlement IDs of the
                                       subscriber, See Section 3.7 for details.
 "profiles":[@profile,...],            optional, list of profile IDs of the subscriber,
                                       see  Section 3.8.42 for details
 "topics":[@topic,...],                optional, list of topics of the subscriber
 "playlists": [@assetList]             optional, list of playlist IDs
                                         of the subscriber
}
```

### 3.8.3 Delete Subscriber

| **DELETE** | /subscriber/<subId>?recursive=boolean | |
|---|---|---|
| ***result*** | **204 NO_RESPONSE** | subId has been successfully deleted or was not found |

If the recursive=true option is used, the playlists and the profiles of the subscriber will be deleted too. If recursive = false or missing, the profiles and playlists are kept.

### 3.8.4 Retrieve Blacklist

This request is used to retrieve the blacklist of a subscriber. In addition to the subscriber level, blacklists exist at profile level too.

| **GET** | /subscriber/<subId>/blacklist | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | @blacklist<br><br>See Section 3.8.1 for the defintion of @blacklist. | |

### 3.8.5 Update Blacklist

This request is used to create or to update the blacklist of a subscriber. In addition to the subscriber level, blacklists exist at profile level too.

| **POST** | /subscriber/<subId>/blacklist | |
|---|---|---|
| ***body*** | {"add": @blacklist,<br>"remove": @blacklist<br>}<br><br>See Section 3.8.1 for the defintion of @blacklist. | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId  does not exist |

Adding to the blacklist is performed first, removing after.

### 3.8.6 Retrieve custom properties of subscriber

This request is used to retrieve the custom properties of a subscriber.

| GET | /subscriber/<subId>/customproperties | |
|---|---|---|
| *result* | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| *response* | customProperties noDotsJson object | |

### 3.8.7 Update custom properties of subscriber

This request is used to create or to update the custom properties of a subscriber. If customProperties existed before, they are replaced by the request body.

| PUT | /subscriber/<subId>/customproperties | |
|---|---|---|
| *body* | Arbitrary JSON object. May not be used to store any personally identifiable information and shall not exceed 1kbyte per subscriber. | |
| *result* | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.8 Update individual custom properties attributes of subscriber

This request is used to update individual attributes the custom properties of a subscriber. In contrast to the PUT call above, the body the POST request is merged with the existing customProperties (instead of replacing it).
The following merging rules are applied:

- replace value if key already exists
- add key:value pair if key does not exist
- lists (JSON arrays) are treated as if they were single values, i.e. list elements are never merged but the whole list is replaced.
- the above rules apply at any hierarchical level of customProperties, not only at the top level

| POST | /subscriber/<subId>/customproperties | |
|---|---|---|
| *body* | Arbitrary JSON object without dots in the key names. May not be used to store any personally identifiable information and shall not exceed 1kbyte per subscriber. | |
| *result* | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.9 Retrieve entitlement of subscriber

This request is used to retrieve the entitlement (list of entitlement IDs) of a subscriber.

| GET | /subscriber/<subId>/entitlement | |
|---|---|---|
| *result* | **200 OK or 204 NO DATA** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| *response* | [<string>] list of entitlement IDs | |

### 3.8.10 Update entitlement of subscriber

This request is used to create or to update the entitlement (list of entitlement IDs) of a subscriber. If an entitlement existed before, they are replaced by the request body.

| **PUT** | `/subscriber/<subId>/entitlement` | |
|---|---|---|
| ***body*** | `[<string>] list of entitlement IDs` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.11 Delete entitlement of subscriber

This request is used to delete the entitlement (list of entitlement IDs) of a subscriber.

| **DELETE** | `/subscriber/<subId>/entitlement` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.12 Retrieve entitlement sources of subscriber

This request is used to retrieve the entitlement sources (list of source IDs) of a subscriber.

| **GET** | `/subscriber/<subId>/entitlementSources` | |
|---|---|---|
| ***result*** | **200 OK or 204 NO DATA** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `[<string>] list of source IDs` | |

### 3.8.13 Update entitlement sources of subscriber

This request is used to create or to update the entitlement sources (list of source IDs) of a subscriber. If entitlementSources existed before, it is replaced by the request body.

| **PUT** | `/subscriber/<subId>/entitlementSources` | |
|---|---|---|
| ***body*** | `[<string>] list of source IDs` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.14 Delete entitlement sources of subscriber

This request is used to delete the entitlement sources (list of source IDs) of a subscriber.

| **DELETE** | `/subscriber/<subId>/entitlementSources` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.15 Retrieve entitlement source groups of subscriber

This request is used to retrieve the entitlement source groups (list of source group IDs) of a subscriber.

| **GET** | `/subscriber/<subId>/entitlementSourceGroups` | |
|---|---|---|
| ***result*** | **200 OK or 204 NO DATA** | Request successful |

| | | |
|---|---|---|
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `[<string>] list of source group IDs` | |

### 3.8.16 Update entitlement source groups of subscriber

This request is used to create or to update the entitlement source groups (list of source group IDs) of a subscriber. If entitlementSourceGroups existed before, it is replaced by the request body.

| **PUT** | `/subscriber/<subId>/entitlementSourceGroups` | |
|---|---|---|
| ***body*** | `[<string>] list of source group IDs` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.17 Delete entitlement source groups of subscriber

This request is used to delete the entitlement source groups (list of source group IDs) of a subscriber.

| **DELETE** | `/subscriber/<subId>/entitlementSourceGroups` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.18 Retrieve entitlement filter of subscriber

This request is used to retrieve the entitlement filter of a subscriber.

| **GET** | `/subscriber/<subId>/entitlementFilter` | |
|---|---|---|
| ***result*** | **200 OK or 204 NO DATA** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `@genericFilter` | |

### 3.8.19 Update entitlement filter of subscriber

This request is used to create or to update the entitlement filter of a subscriber. If entitlementFilter existed before, it is replaced by the request body.

| **PUT** | `/subscriber/<subId>/entitlementFilter` | |
|---|---|---|
| ***body*** | `@genericFilter` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.20 Delete entitlement filter of subscriber

This request is used to delete the entitlement filter of a subscriber.

| **DELETE** | `/subscriber/<subId>/entitlementFilter` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.21 Update Profiles

This call is used to assign a Profile to a Subscriber or to remove profiles from a subscriber. The assignment is stored for administrative purposes only and is not mandatory, stand alone profiles are allowed too. Ncanto does not check for conflicts (e.g if one Profile is assigned to multiple Subscribers). From R4.8, if recursive is missing or false, the profile is only removed from the profile list of the subscriber, but not delete. If recursive==true, the profile itself is deleted too. If removeByName is used instead of remove, all profiles with the matching names (multiple profiles of the subscriber may have the same name) are deleted.

| **POST** | `/subscriber/<subId>/profiles?recursive=boolean` | |
|---|---|---|
| ***body*** | `{"add":[<profileId>,...],`<br>`"remove":[<profileId>,...],`<br>`"removeByName": [<profileName>]}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.22 Retrieve Profiles

This request returns the list of profiles associated with a given subscriber.

| **GET** | `/subscriber/<subId>/profiles` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `[<profileId>,...]` | |

### 3.8.23 Update Topics

The update topics call takes as its input information from a person's social profile. The social profile is an expression of a set of interests ("likes") which is used to generate *topics*. The topic IDs are then used to request recommendations through the uniform Recommendation API calls (see 3.17).

To enable the client to distinguish between the subscriber's own and their contacts' likes, the API accepts multiple social profiles in a single call, identified by the name of the owner. This name is a string only used for identifying which topic belongs to which social profile, it is used in the explanation of recommendations (see 3.17). Ncanto analyzes the social profiles and generate a number of topics for the Subscriber.

There is no guaranteed number of topics to expect and the client must not assume any relation between likes and topics (i.e. one like may result in zero, one, or more topics, and one topic may be the result of one or more likes). Topics are not "owned" by any user – a topic is a self-sufficient key to create recommendations.

As of the current API version, only social profiles from Facebook are accepted. The social profile type FACEBOOK is largely identical to the format delivered by Facebook's REST API: a list of Facebook Likes as JSON.

| **POST** | `/subscriber/<subId>/topics` |
|---|---|
| ***body*** | `[{"owner":string,  optional, name of the owner of the profile` |

| | | |
|---|---|---|
| | "type":enum, required, only "facebook" supported<br>"socialProfile":[{ at least one required<br>"name":string, required, name of the Facebook page<br>"category":string, required, category of the Facebook page<br>"id":string, required, ID of the Facebook page<br>"created_time":timestamp}, optional,<br>...<br>]},<br>...<br>] | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | [@topic,...]<br><br>See Section 3.8.1 for the definition of @topic. | |

### 3.8.24 Retrieve Topics

This request returns the list of topics associated with a subscriber.

| **GET** | /subscriber/<subId>/topics | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | [@topic,...]<br><br>See Section 3.8.1 for the definition of @topic. | |

### 3.8.25 Update Source Lineups of the Subscriber

This request is used to create or to update all lineups of a subscriber in one step. Source lineups are used to filter and sort results.

| **PUT** | /subscriber/<subId>/sourceLineups | |
|---|---|---|
| ***body*** | {<br><lineupName>: [<sourceId>], <lineupName>: string, <sourceId>: string<br>...<br>} | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.26 Retrieve Source Lineups of subscriber

This request is used to retrieve all source lineups of a subscriber.

| **GET** | /subscriber/<subId>/sourceLineups | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | {<br><lineupName>: [<sourceId>], <lineupName>: string, <sourceId>: string<br>...<br>} | |

### 3.8.27 Delete Source Lineups of Subscriber

This request deletes all source lineups of the subscriber.

| **DELETE** | `/subscriber/<subId>/sourceLineups` | |
|---|---|---|
| ***result*** | **204 NO_RESPONSE** | Source lineups have been successfully deleted or were not found |

### 3.8.28 Update Single Source Lineup of Subscriber

This request is used to create or to update one lineup of a subscriber.

| **PUT** | `/subscriber/<subId>/sourceLineups/<lineupName>` | |
|---|---|---|
| ***body*** | `[<sourceId>]  <sourceId>: string` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |

### 3.8.29 Retrieve Single Source Lineup of subscriber

This request is used to retrieve one source lineup of a subscriber.

| **GET** | `/subscriber/<subId>/sourceLineups/<lineupName>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist or lineupName does not exist |
| ***response*** | `[<sourceId>]  <sourceId>: string` | |

### 3.8.30 Delete Single Source Lineup of Subscriber

This request deletes one source lineup of the subscriber.

| **DELETE** | `/subscriber/<subId>/sourceLineups/<lineupName>` | |
|---|---|---|
| ***result*** | **204 NO_RESPONSE** | Source lineup has been successfully deleted or was not found |

### 3.8.31 Retrieve Source Lineup names of subscriber

This request is used to retrieve all source lineup names of a subscriber.

| **GET** | `/subscriber/<subId>/sourceLineupNames` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId does not exist |
| ***response*** | `[<sourceLineupName>]  <sourceLineupName>: string` | |

### 3.8.32 Score Assets in all Profiles of a Subscriber

For every individual asset this request selects the Profile best matching the asset and scores the asset in that Profile. Cf. Section 3.9.11.

| **POST** | `/subscriber/<subId>/profiles?errorTolerant=true&recordingSuggestion=boolean&scoringProfiles=enum` |
|---|---|
| ***body*** | `{"toScore":[<assetId>,...], "maxExplanationLevel": integer, "explanationConfig": {"language": <langcode>, "types": [<type_enum>], "metadataLanguages": [<langcode>]}}` |

| | | |
|---|---|---|
| | Request parameter errorTolerant=true may be added to ignore assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store. Starting from R4.5, feature level explanations of the scores are provided if maxExplanationLevel is set to 3. Note that other explanation levels are not supported. maxExplanationLevel is deprecated from R4.6 and the more flexible "explanationConfig" parameter shall be used. "explanationConfig" supports independent specifications of explanation language (default: English), of the explanation types (only "feature" supported), and of the language of asset attributes in the explanation (metadataLanguages, the first language from the list which is available in the asset is used, fallback to canonical names and IDs if none ie available). If explanationConfig is provided, maxExplanationLevel will be ignored.<br><br>Starting from R4.5 the profile selection algorithm for scoring is configurable via the parameter scoringProfiles. If scoringProfiles is omitted or set to "prefilter", all profiles the prefilter of which pass the asset are used for scoring. If there are multiple such profiles, the maximum of the scores is returned. If scoringProfiles is set to "filter", the profile filter is taken into account in addition. By using the value "filter" one can achieve that only those assets receive a score which would also be returned as blending preference engine recommendations. | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId or assetId or subId does not exist |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>] | |

### 3.8.33 Rating Significance / Rate an asset in all matching profiles of a subscriber

Estimates, how significant the asset provided in the request body is for the different Profiles of the Subscriber. Significance values are normalized to the range [0,1], 0 meaning poor match between the asset and the profile and 1 corresponding to a perfect match between the two. Starting from R4.2 two algorithms are supported: by default, the cosine similiarity between the asset and the profile's filter is calculated. If significanceAlgorithm=prefilter is specified, the filters are ignored and only the prefilters are taken into account. If the asset passes the profile's prefilter, the significance is 1.0, otherwise the significance is 0.0.

| **POST** | /subscriber/<subId>/profiles?significanceAlgorithm=prefilter or cosine | |
|---|---|---|
| ***body*** | {<br>"assetId":<assetId>, new from R5.0<br>"asset":<assetId> deprecated from R5.0<br>} | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | assetId or subId does not exist or subscriber does not have profiles |
| ***response*** | [{"profileId": string512, "significance": decimal},...] | |

Starting from R4.1 the rating significance request can automatically assign the rating to matching profiles. By default the rating is automatically applied if there is exactly one profile matching (i.e. with significance > 0). If no unique matching profile can be found, the request returns 409 response code. The assets given in toScore are then scored with all profiles of the subscriber.
If the query parameter allowMultipleMatching is set to true, the rating is applied to all profiles which are matching.

| **POST** | /subscriber/<subId>/profile-matching?errorTolerant=true&recordingSuggestion=boolean&allowMultipleMatching=boolean&scoringProfiles=enum | |
|---|---|---|
| ***body*** | {"assetId": string512, | required, ID of the asset rated |
| | "rating": decimal, | required, range [-1,...+1],+1 for explicit "like", -1 for explicit "dislike" |
| | "ratingAggregation": enum, | optional, from R5.1, one of "accumulate", "overwrite", default value: "accumulate". Specifies if implicit ratings (-1 < ratingValue < 1) replace the previous rating or are added to the previous rating |
| | "type": enum, | optional, one of "single", "series", if missing, the type is determined automatically: if seriesId is present: type=series, else: type=single |
| | "toScore":[string,...], | optional, list of asset IDs to be re-scored |
| | "setting": { | optional, viewing setting of the rating |
| | "timeOfDayBin": <string>, | optional, e.g. "morning", "evening" |
| | "timeOfWeekBin": <string>, | optional, e.g. "weekday", "weekend" |
| | "device": <string>, | optional, the device used |
| | "locationBin": <string>, | optional, e.g. "home", "work" |
| | "moodBin": <string>, | optional, e.g. "happy", "sad" |
| | "activity": <string>, | optional, e.g. "traveling", "relaxing" |
| | "autoTimeOfDayBin": { | optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair |
| | }, | |
| | "autoTimeOfWeekBin: { | optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning of the week (xsdDuration) bin name (string) pair |
| | } | |
| | }, | |
| | "interaction": <string>, | optional, if provided, the interaction log entry will contain interaction type <string> instead of the default value "rate". Supported from R4.5. |
| | "contextId": <string>, | optional, if provided, the given contextId will be logged in the interaction log. Supported from R5.2. |
| | "revenue": <decimal>, | optional, if provided the given value will be logged for the interaction. Supported from R5.6. |

<table>
<tr><td></td><td colspan="2">
<pre>
 "currency": <string>,            optional, if provided the given value will
                                      be logged for the interaction. Supported
                                      from R5.6.
 "maxExplanationLevel": integer   optional, if set to 3, the response contains
                                      feature level explanations of the score.
                                      Supported from R4.5, deprecated from R4.6
 "explanationConfig": {           optional, supported from R4.6, if present,
                                      maxExplanationLeve is ignored
   "language": <langcode>,        optional, the desired language of the
                                      explanation, default: English
   "types": [<type_enum>],        only "feature" supported for feature level
                                      explanations equivalent to those of the
                                      preference engine
   "metadataLanguages": [<langcode>] optional, preferred language of the asset
                                      attributes used in the explanation,
                                      fallback to IDs and canonical names if
                                      none of the languages is available.
 }
}
</pre>
Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store.

Starting from R4.5 the profile selection algorithm for scoring is configurable via the parameter scoringProfiles. If scoringProfiles is omitted or set to "prefilter", all profiles the prefilter of which pass the asset are used for scoring. If there are multiple such profiles, the maximum of the scores is returned. If scoringProfiles is set to "filter", the profile filter is taken into account in addition. By using the value "filter" one can achieve that only those assets receive a score which would also be returned as blending preference engine recommendations.
</td></tr>
<tr><td rowspan="3">result</td><td>200 OK</td><td>Request successful</td></tr>
<tr><td>400 BAD REQUEST</td><td>Malformed request</td></tr>
<tr><td>404 NOT FOUND</td><td>assetId or subscriber ID does not exist or subscriber does not have profiles</td></tr>
<tr><td></td><td>409 CONFLICT</td><td>no single matching profile could be found, use POST /subscriber/<subId>/profiles and assign rating manually.</td></tr>
<tr><td>response</td><td colspan="2">
<pre>
[
 {
  "assetId": string512,
  "score": decimal,
  "rating": decimal,
  "recordingSuggestion": boolean,
  "explanations": [@explanation,...],  see 3.17 for the definition
 },
 ...
]
</pre>
</td></tr>
</table>

### 3.8.34 Rate an Asset / score assets in a profile identified by its name

This request is supported from R4.1. It is equivalent to the Rate an Asset request described in Section 3.9.10, the only difference is that the profile to which the rating shall be applied is not identified by its unique profile ID but is addressed via subscriber ID and profile name. The asset IDs provided in "toScore" are scored in the selected profile only.

| POST | /subscriber/<subId>/profile-matching?profileName=string&errorTolerant=true&recordingSuggestion=boolean | |
|---|---|---|
| ***body*** | {"assetId": string512, | optional, ID of the asset rated |
| | "rating": decimal, | required if assetId is present, range [-1, ...+1],+1 for explicit "like", -1 for explicit "dislike" |
| | "ratingAggregation": enum, | optional, from R5.1, one of "accumulate", "overwrite", default value: "accumulate". Specifies if implicit ratings (-1 < ratingValue < 1) replace the previous rating or are added to the previous rating |
| | "type": enum, | optional, one of "single", "series", if missing, the type is determined automatically: if seriesId is present: type=series, else: type=single |
| | "toScore":[string,...], | optional, list of asset IDs to be re-scored |
| | "setting": { | optional, viewing setting of the rating |
| | "timeOfDayBin": <string>, | optional, e.g. "morning", "evening" |
| | "timeOfWeekBin": <string>, | optional, e.g. "weekday", "weekend" |
| | "device": <string>, | optional, the device used |
| | "locationBin": <string>, | optional, e.g. "home", "work" |
| | "moodBin": <string>, | optional, e.g. "happy", "sad" |
| | "activity": <string>, | optional, e.g. "traveling", "relaxing" |
| | "autoTimeOfDayBin": { | optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair |
| | }, | |
| | "autoTimeOfWeekBin: { | optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning of the week (xsdDuration) bin name (string) pair |
| | } | |
| | }, | |
| | "interaction": <string>, | optional, if provided, the interaction log entry will contain interaction type <string> instead of the default value "rate". Supported from R4.5. |
| | "contextId": <string>, | optional, if provided, the given contextId will be logged in the interaction log. Supported from R5.2. |
| | "revenue": <decimal>, | optional, if provided the given value will be logged for the interaction. Supported from R5.6. |
| | "currency": <string>, | optional, if provided the given value will be logged for the interaction. Supported from R5.6. |
| | "maxExplanationLevel": integer | optional, if set to 3, the response contains feature level explanations of the score. Supported from R4.5, deprecated from R4.6 |
| | "explanationConfig": { | optional, supported from R4.6, if present, maxExplanationLeve is ignored |
| | "language": <langcode>, | optional, the desired language of the |

| | | |
|---|---|---|
| | explanation, default: English<br>"types": [<type_enum>], only "feature" supported for feature level explanations equivalent to those of the preference engine<br>"metadataLanguages": [<langcode>] optional, preferred language of the asset attributes used in the explanation, fallback to IDs and canonical names if none of the languages is available.<br>}<br>}<br><br>Request parameter profileName identifies the profile. Ncanto will search for a profile with the given name among the profiles of the subscriber. If there is no profile or if there are multiple profiles with the given name, an exception is thrown.<br><br>Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store. | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | assetId or subscriber ID does not exist or subscriber does not have a profile with the given names |
| | **409 CONFLICT** | the subscriber has multiple profiles with the given name |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>] | |

### 3.8.35 Retrieve Playlists

This request returns the playlists associated with a subscriber.

| **GET** | /subscriber/<subId>/playlists | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subscriber ID does not exist |
| ***response*** | [<listId>] | |

### 3.8.36 Update Playlists

This request is used to update the playlists associated with a subscriber.

| **POST** | /subscriber/<subId>/playlists | |
|---|---|---|
| ***body*** | {"add": [<listId>],<br>"remove": [<listId>]<br>} | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |

| | | |
|---|---|---|
| | **404 NOT FOUND** | subscriber ID does not exist |

Adding of playlists is performed first, removing after. Note that update playlists does not check if the added playlists actually exist.

### 3.8.37 Clone a Subscriber

The clone subscriber request creates a copy of the given template subscriber. The profiles, playlists and bookmark lists of the template subscriber will be cloned, the IDs of the new entities will be <newSubId>_<oldProfile/Playlist/BookmarklistId>. All other properties of the template subscriber (topics, subscriberGroups, etc.) will be copied.
However, the previous ratings of the template subscriber will not be cloned to update the collaborative matrix and statistical lists.

| **POST** | `/subscriber/<newSubId>` | |
|---|---|---|
| ***body*** | `{"template":<templateSubId>}` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId of the template does not exist |
| ***response*** | `@subscriber`<br><br>See Section 3.8.1 for the definition of @subscriber. | |

### 3.8.38 Merge Subscribers

The merge subscribers request copies the blacklists, playlists, source lineups, topics, and profiles of the subscriber to be merged into the combined subscribers. Playlists and topics are merged at ID level, i.e. the playlist and topic IDs of the merged subscriber are added to the combined subscriber. The lineups of the merged subscriber are copied to the combined subscriber too, unless there is already a lineup of the same name. In that case the lineup is not merged. The custom properties of the subscriber are merged using the logic described in Section 3.8.8.

Each profile of the merged subscriber which does not match a profile of the combined subscriber with the same name is copied into the profile list of the combined subscriber. If a profile of the same name exists, the profiles are merged by adding all the rated assets of the merged profile to the combined profile.

At the end of the merging process the merged subscriber is deleted, as well as its merged profiles (i.e. those which matched a profile of the same name and have therefore been merged at rated asset level).

| **POST** | `/subscriber/<combinedSubId>` | |
|---|---|---|
| ***body*** | `{"mergeSubscriber":<toBeMergedSubId>}` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | Either <combinedSubId> or <toBeMergedSubId> do not exist |
| ***response*** | Merged subscriber object | |

### 3.8.39 Calculate collaborative scores

Returns the collaborative scores of a list of assets. Collaborative scores are calculated across all collaborative matrices, a specific matrix does not have to be selected. Note that the feature is deprecated from R4.7.1 and will be removed from R4.8.

| **POST** | /subscriber/<subId>/collaborative | |
|---|---|---|
| ***body*** | {"toScore":[<assetId>,...]} | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | subId ID does not exist |
| ***response*** | [{"assetId": string512, "score": decimal, "rating": decimal},...] | |

### 3.8.40 Source, sourceGroup, and asset store valuation

The valuation request returns the average score of all assets or a list of all assets with a score greater than or equal to the given minimum score, for all assets with a specified sourceId, a specified sourceGroupId, or for all assets which are in a specified asset store.
Scoring is either done using the preference engine and all profiles of the subscriber (see Section 3.8.32) or using the collaborative engine (see Section 3.8.39). It is possible to specify a list of source IDs, a list of sourceGroup IDs and a list of asset stores. In that case the results are calculated individually for each sourceId, sourceGroupId, or asset store.
Note that the valuation request accepts inactive store IDs too.

| **POST** | /subscriber/<subId>/valuator | |
|---|---|---|
| ***body*** | {<br>"algorithm": enum, required, only "preference" supported from R4.8<br>"result": [enum], required, allowed "averageScore", "highestScoreAssets"<br>"minScore": decimal, required if "result" contains "highestScoreAssets"<br>"sources":[<sourceId>], optional, list of sources to be valuated individually, at least one of sources/sourceGroups/stores required<br>"sourceGroups": [<sGId>], optional, list of sourceGroups to be valuated individually, at least one of sources/sourceGroups/stores required<br>"stores": [<storeId>] optional, list of asset stores to be valuated individually, may include inactive stores. At least one of sources/sourceGroups/stores required<br>} | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | storeId does not exist |
| ***response*** | {<br>"algorithm": enum, see body<br>"sources": {<br><sourceId>: { sourceId: see body<br>"averageScore": <decimal>, average score of all assets matching the given sourceId<br>"highestScoreAssets": [<assetId>] list of asset IDs with score greater than or equal to minScore<br>},<br>...<br>},<br>"sourceGroups": {<br><sourceGroupId>: { sourceGroupId: see body | |

```
        "averageScore": <decimal>,        average score of all assets
matching the given sourceGroupId
        "highestScoreAssets": [<assetId>] list of asset IDs with score
                                          greater than or equal to minScore
    },
    ...
 },
 "stores": {
    <storeId>: {                          storeId: see body
        "averageScore": <decimal>,        average score of all assets in the
given asset store
        "highestScoreAssets": [<assetId>] list of asset IDs with score
                                          greater than or equal to minScore
    },
    ...
 }
}
```

### 3.8.41 Retrieve like degrees of all profiles of a subscriber

The like degree is Ncanto's estimate how well a feature / feature value pair (e.g. genreBroad: romance) is perceived. Like degrees are decimal values normalized to the range [0:1], values below 0.5 indicate rejection, values above 0.5 liking.
The GET version of the request returns the feature / feature value pairs of all profiles of the subscriber which have the highest like degrees.

| **GET** | `/subscriber/<subscriberId>/like-degrees?count=<integer>` | |
|---|---|---|
| ***path parameters*** | `subscriberId` | Unique ID of the subscriber the like degrees are requested of. |
| ***query parameters*** | `count` | Optional, type positive integer. The maximum number of like degrees to be returned. |
| ***body*** | - | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Count is negative |
| | **404 NOT FOUND** | subscriberId not found |

***response***

```
{
  "totalNumberOfFeatureValuePairs": <int>, the total number of feature
                                             value pairs in all preference
                                             profiles of the subscriber.

  "likeDegrees": [
    {
      "feature": <string>,                 the feature name (see section
                                             Table 4 for the list of supported
                                             feature names, e.g. genreBroad

      "featureValue": <any>,               the feature value, the data type
                                             depends on the feature's data
                                             type. E.g. "romance"

      "likeDegree": <decimal>              the like degree in range [0:1]
    },
    ...
  ]
}
```

The POST request allows to filter to individual features (e.g. genre like degrees only). From R5.0 a viewing setting can be provided, too. If a viewing setting is specified, Ncanto will return like degrees learned in the s
ame viewing setting. If there is no or not enough information about the desired viewing setting available, like degrees will be blended from all learned viewing settings, emphasizing the desired viewing setting.

| **POST** | `/subscriber/<subscriberId>/like-degrees?count=<integer>` | |
|---|---|---|
| ***path parameters*** | `subscriberId` | Unique ID of the subscriber the like degrees are requested of. |
| ***query parameters*** | `count` | Optional, type positive integer, supported from R5.0. The maximum number of like degrees to be returned. |
| ***body*** | `{`<br>` "features": [<featureName>],` optional. The supported feature names correspond to the root features listed in Table 43.9.1, e.g. "genreBroad" or "persons" but not "persons.actor". If no features are provided, the response will contain all features of the profile.<br>` "setting": {` optional, supported from R5.0, the viewing setting the like degrees of which are requested.<br>`  "timeOfDayBin": <string>,` optional, e.g. "morning", "evening"<br>`  "timeOfWeekBin": <string>,` optional, e.g. "weekday", "weekend"<br>`  "device": <string>,` optional, the device used<br>`  "locationBin": <string>,` optional, e.g. "home", "work"<br>`  "moodBin": <string>,` optional, e.g. "happy", "sad"<br>`  "activity": <string>,` optional, e.g. "traveling", "relaxing"<br>`  "autoTimeOfDayBin": {` optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time<br>`  <offsetFrom>: <binName>` at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair<br>`  },`<br>`  "autoTimeOfWeekBin: {` optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time<br>`  <offsetFrom>: <binName>` at least one required, offset wrt beginning of the week (xsdDuration) bin name (string) pair<br>`  }`<br>` }`<br>`}` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Invalid feature name |
| | **404 NOT FOUND** | subscriberId not found |
| ***response*** | `{`<br>`  "totalNumberOfFeatureValuePairs": <int>,` the total number of feature value pairs in the preference profile<br>`  "likeDegrees": [`<br>`  {` | |

| | | |
|---|---|---|
| | "feature": <string>,<br><br>"featureValue": <any>,<br><br>"likeDegree": <decimal><br>},<br>...<br>]<br>} | the feature name (see section 3.9.1 for the list of supported feature names, e.g. genreBroad<br>the feature value, the data type depends on the feature's data type. E.g. "romance"<br>the like degree in range [0:1] |

### 3.8.42 Retrieve recently used search terms of the subscriber

This request returns the recently used tracked search terms used by the subscriber. The tracking of search terms is supported by recommendation requests (if a search engine is part of the context) as well as by retrieve assets by filter requests (e.g. for use cases where the client combines the search phrase suggestions feature with a filter request).

Note that the blacklist/whitelist described in Sections 3.23.1 and 3.23.2 is applicable to the personal search terms of one subscriber too.

The search terms are ordered by descending timestamp in the response.

| **GET** | /subscriber/<subscriberId>/search-terms?limit=<positiveInteger>&timeWindow=<duration>&actualCounts=<boolean>&startsWith=<string> | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | limit | Type: positive integer, defines the number of most recent search terms to be returned |
| | timeWindow | Type: positive duration, defines the time window from which the recent search terms should be collected |
| | actualCounts | Type: boolean. If true, Ncanto checks how many assets matching the search term are available in the active store at the time of querying. |
| ***body*** | - | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD_REQUEST** | Request parameters are syntactically incorrect |
| ***response*** | [<br>{<br>"searchTerm":<string>, the search term (lowercased)<br>"timestamo": <timestamp>, the last time the term was searched for<br>"actualCount": <integer> the number of assets in the active store matching the search term<br>}<br>]<br><br>Note that actualCount is omitted from the response if the query parameter is missing or set to false. | |

## 3.9 Profile Management

### 3.9.1 Create or Update Profile

This call is used to create a new profile or to update an existing one. Starting from R4.0 the metadata of ratedAssets is not mandatory, Ncanto tries to look up missing metadata in the active asset store and in playlist assets. Please note that if a profile is created / updated with new rated assets, the new ratings will not be applied to the collaborative engine.

| **PUT** | /profile/<profileId> | |
|---|---|---|
| ***body*** | @profile | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request or body |
| | **404 NOT FOUND** | The metadata of at least one ratedAsset is missing in the body and not found in the active asset store and in playlist assets. |

```
@profile =
{"profileId": string512,                optional, unique ID of the profile, ignored by the
                                          PUT request
 "customProperties": noDotsJson,        optional, arbitrary, customer defined properties of
                                          the Profile as JSON object without dots in the
                                          key names. May not be used to store any
                                          personally identifiable information and shall not
                                          exceed 1kbyte per profile.
 "calibrationFactor": decimal           required, calibration factor of the Profile. Use
                                          0.5 as initial value when creating a new
                                          channel.
 "errorRatio": decimal,                 optional, last error ratio value
 "name": string,                        optional, name of the profile, e.g. "my movies" to
                                          be used in explanations
 "prefilter": @filter,                  optional, the pre-filter of
                                          the profile
 "genericPrefilter": @genericFilter,    optional, from R4.4, new syntax offering more
                                          flexibility than the @filter. If
                                          "genericPrefilter" is present, "prefilter" is not
                                          allowed.
 "filter": @filter,                     required, the filter of the Profile. From R4.5 the
                                          filter can be omitted if a prefilter or a
                                          genericPrefilter is present.
 "ratedAssets": [@ratedAsset,...],      optional
 "ratedFeatures": [@ratedFeature,...]   optional,
 "blacklist": @blacklist,               optional, see 3.8.1 for the definition of
                                          @blacklist
 "playlists": [string]                  optional, list of playlist IDs
                                          associated with the profile
}

@filter =
[@disjunct,...]                         at least one disjunct required, list of disjuncts
                                          of the filter

@disjunct =
{"disjunctType": enum,                  required, "operatorDefined" or "likedSingle" or
                                          "likedSeries"
 "conjuncts: [@conjunct,...]            at least one conjunct required
}

@conjunct =
{"constraintType": enum,                required, see table below for details
 "operator": enum,                      required, see table below for details
 "value": string                        required, see table below for details
}
```

```
@ratedAsset =
{"timestamp": timestamp,              required, time of rating
 "assetId": string512,                required, ID of the rated asset
 "seriesId": string,                  optional, series ID of the rated asset
 "type": enum                         optional, one of "single", "series", default:
                                        "single"
 "rating": decimal,                   required, rating value (e.g. 1.0 for explicit
                                        like, -1.0 for explicit dislike)
 "metadata": @asset                   optional, full metadata of the asset. See 3.6.7 for
                                        a definition of @asset. If missing, Ncanto tries
                                        to load the asset metadata from the active asset
                                        store and from the playlist asset store. A 404
                                        exception is thrown if the metadata of one or
                                        more assets is not found.
}

@ratedFeature =                       optional,
{"timestamp": timestamp,              required, time of rating
 "feature": string,                   required, name of the rated feature in dot
                                        notation, e.g. "persons.actor"
 "featureValue": string,              required, value of the rated feature, e.g. "Clint
                                        Eastwood", corresponding to the canonical name of
                                        the person (see @asset)
 "rating": decimal,                   required, like degree of the feature/value pair
                                        0 <= likeDegree <= 1
}
```

| constraint type | allowed operators |
|---|---|
| sourceId | equals, notEquals |
| startBin | equals, notEquals, atMost, smaller, atLeast, larger |
| durationBin | equals, notEquals, atMost, smaller, atLeast, larger |
| first | Equals |
| last | Equals |
| live | equals, notEquals |
| pay | Equals |
| onDemand | Equals |
| priceBin | equals, notEquals, atMost, smaller, atLeast, larger |
| titles | contains, notContains, equals, notEquals. equals/notEquals searches for assets whose title, at least in one language, exactly matches the "value". contains/notContains only works with single word "values" and does a full text search in all titles. |
| originalTitle | contains, notContains, equals, notEquals. equals/notEquals searches for assets whose original title exactly matches the "value". contains/notContains only works with single word "values" and does a full text search in the original title. |
| seriesId | Equals |
| season | equals, notEquals, atMost, smaller, atLeast, larger |
| episode (from R4.0) | equals, notEquals, atMost, smaller, atLeast, larger |
| productionYearFirst | equals, notEquals, atMost, smaller, atLeast, larger |
| productionYearLast | equals, notEquals, atMost, smaller, atLeast, larger |

| | |
|---|---|
| productionCountry | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| tip | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| editorialRating | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| communityRating | atMost, smaller, atLeast, larger |
| surroundSound | Equals |
| bw | Equals |
| subtitled | Equals |
| forBlind | Equals |
| hd | Equals |
| interactive | Equals |
| threeD | Equals |
| programType | equals, notEquals |
| assetType | equals, notEquals |
| genreFine | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| genreMedium | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| genreBroad | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| themeCircle | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| audioLang | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| keywords | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| persons | contains, notContains. Selects assets which include the given person in their metadata (in any function and in any language). Full match is required, i.e. "Clint" will not match "Clint Eastwood". |
| subtitles | contains, notContain. Only works with single word “values” and does a full text search in all subtitles. |
| categories | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| synopses | contains, notContains. Only works with single word “values” and does a full text search in all synopses. |

| | |
|---|---|
| sourceNames | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| uiGenres | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| copyControl | contains, notContains |
| adProduct | contains, notContains |
| adCompany | contains, notContains |
| adTaxonomy | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| targetDevice | contains, notContains, selects assets which include / do not include the specified single value in the asset attributes (list of values). |
| sourceGroupIds (from R4.5) | contains, notContains |
| communityViewCount (from R4.5) | atMost, smaller, atLeast, larger |
| communityShareCount (from R4.5) | atMost, smaller, atLeast, larger |
| communityTrendScore (from R4.5) | atMost, smaller, atLeast, larger |
| stereo (from R4.5) | Equals |
| wideScreen (from R4.5) | Equals |
| encoded (from R4.5) | Equals |
| defaultLanguage (from R4.5) | equals, notEquals |
| subtitleLang (from R4.5) | contains, notContains |
| themeCircle (from R4.5) | contains, notContains |
| leaseDurationMinutes (from R4.5) | atMost, smaller, atLeast, larger |
| viewCount (from R4.5) | atMost, smaller, atLeast, larger |
| profitability (from R4.5) | atMost, smaller, atLeast, larger |
| targetDemographic (from R4.5) | contains, notContains |
| targetDevice (from R4.5) | contains, notContains |
| lightness (from R4.5) | atMost, smaller, atLeast, larger |
| pace (from R4.5) | atMost, smaller, atLeast, larger |
| | |

**Table 2: Supported constraint type and operator combinations**

| | |
|---|---|
| Start time bins | 0 = "night": [01:00, 06:00[<br>1 = "morning": [06:00, 11:00[<br>2 = "noon": [11:00, 15:00[<br>3 = "afternoon": [15:00, 18:00[<br>4 = "early evening": [18:00, 22:00[<br>5 = "late evening": all others |
| Duration bins | 0 = "very short": [000, 015[ minutes<br>1 = "short": [015, 030[ minutes<br>2 = "fair": [030, 060[ minutes |

| | |
|---|---|
| | 3 = “medium": [060, 090[ minutes<br>4 = "long": [090, 120[ minutes<br>5 = “very long"; [120,+INF[ minutes |

Table 3: Attribute binning used in profile filters

### 3.9.2 Create a Profile from Seed

This call is used to create a new Profile based on the characteristics of a seed asset. Ncanto automatically creates the filter of the Profile and adds the seed asset to ratedAssets with rating=1. calibrationFactor, ratedFeatures, and blacklist are initialized.

| **POST** | /profile/<profileId> | |
|---|---|---|
| ***body*** | {"name": string, optional<br>"customProperties":<customer_defined_properties>, optional<br>"seed":<assetId> required<br>"setting": { optional, from R4.2, viewing setting of the rating<br>"timeOfDayBin": <string>, optional, e.g. "morning", "evening"<br>"timeOfWeekBin": <string>, optional, e.g. "weekday", "weekend"<br>"device": <string>, optional, the device used<br>"locationBin": <string>, optional, e.g. "home", "work"<br>"moodBin": <string>, optional, e.g. "happy", "sad"<br>"activity": <string>, optional, e.g. "traveling", "relaxing"<br>"autoTimeOfDayBin": { optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time<br><offsetFrom>: <binName> at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair<br>},<br>"autoTimeOfWeekBin: { optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time<br><offsetFrom>: <binName> at least one required, offset wrt beginning of the week (xsdDuration) bin name (string) pair<br>}<br>},<br>"prefilter": @filter, optional, from R4.5, if present the prefilter is added to the profile<br>"genericPrefilter": @genericFilter optional, from R4.5, if present, the generic prefilter is added to the profile<br>"interaction": <string> optional, if provided, the interaction log entry will contain interaction type <string> instead of the default value "rate". Supported from R4.5.<br>} | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request or body |
| | **404 NOT FOUND** | assetId not found |

### 3.9.3 Retrieve a Profile

This request returns the full description of a profile.

| **GET** | `/profile/<profileId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request or body |
| | **404 NOT FOUND** | profileId not found |
| ***response*** | `@profile` | |

### 3.9.4 Delete a Profile

| **DELETE** | `/profile/<profileId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | profileId has been successfully deleted or was not found |

### 3.9.5 Retrieve Blacklist

This request returns the blacklist of a profile.

| **GET** | `/profile/<profileId>/blacklist` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |
| ***response*** | `@blacklist` | |

### 3.9.6 Update Blacklist

This request is used to create or update the blacklist of a profile.

| **POST** | `/profile/<profileId>/blacklist` | |
|---|---|---|
| ***body*** | `{"add": @blacklist,`<br>`"remove": @blacklist`<br>`}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

Adding to the blacklist is performed first, removing after.

### 3.9.7 Retrieve custom properties of profile

This request is used to retrieve the custom properties of a profile.

| **GET** | `/profile/<profileId>/customproperties` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |
| ***response*** | `customProperties noDotsJson object` | |

### 3.9.8 Update custom properties of profile

This request is used to create or to update the custom properties of a profile.

| **PUT** | `/profile/<profileId>/customproperties` |
|---|---|

| | | |
|---|---|---|
| ***body*** | `Arbitrary JSON object without dots in the key names. May not be used to store any personally identifiable information and shall not exceed 1kbyte per profile.` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

### 3.9.9 Update individual custom properties attributes of profile

This request is used to update individual attributes the custom properties of a profile. In contrast to the PUT call above, the body the POST request is merged with the existing customProperties (instead of replacing it).
The following merging rules are applied:

- replace value if key already exists
- add key:value pair if key does not exist
- lists (JSON arrays) are treated as if they were single values, i.e. list elements are never merged but the whole list is replaced.
- the above rules apply at any hierarchical level of customProperties, not only at the top level

| **POST** | `/profile/<profileId>/customproperties` | |
|---|---|---|
| ***body*** | `Arbitrary JSON object without dots in the key names. May not be used to store any personally identifiable information and shall not exceed 1kbyte per profile.` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

### 3.9.10 Rate an Asset in a single profile

This request is used to communicate a rating (like or dislike) of an asset to Ncanto. If a list of assets is provided in the request body, the recommendation scores of the assets will be recalculated after updating the profile with the new rating. Rating values -1, 0, and +1 replace (overwrite) any previous rating of the same asset, rating values in the range ]-1, 0[ U ]0, +1[ are accumulated within the interval [-0.9, 0.9] but don't override a previous -1 or +1 rating.

| **POST** | `/profile/<profileId>/scorer?errorTolerant=true&recordingSuggestion=boolean` |
|---|---|
| ***body*** | `{`<br>`"assetId": string512, required, ID of the asset rated`<br>`"rating": decimal, required, range [-1,...+1],+1 for explicit "like", -1 for explicit "dislike"`<br>`"ratingAggregation": enum, optional, from R5.1, one of "accumulate", "overwrite", default value: "accumulate". Specifies if implicit ratings (-1 < ratingValue < 1) replace the previous rating or are added to the previous rating`<br>`"type": enum, optional, one of "single", "series", default: "single". If missing, the type is determined automatically: if seriesId is present: type=series, else: type=single`<br>`"subscriber": string, optional, subscriber ID`<br>`"toScore":[string,...], optional, list of asset IDs to be re-scored` |

```
 "setting": {                          optional, viewing setting of the rating
  "timeOfDayBin": <string>,            optional, e.g. "morning", "evening"
  "timeOfWeekBin": <string>,           optional, e.g. "weekday", "weekend"
  "device": <string>,                  optional, the device used
  "locationBin": <string>,             optional, e.g. "home", "work"
  "moodBin": <string>,                 optional, e.g. "happy", "sad"
  "activity": <string>,                optional, e.g. "traveling", "relaxing"
  "autoTimeOfDayBin": {                optional, from R5.6, ignored if timeOfDayBin
                                         is set, defines the time of day bin values
                                         to be set automatically depending on
                                         current time
    <offsetFrom>: <binName>            at least one required, offset wrt beginning
                                         of the day (xsdDuration) bin name (string)
                                         pair
  },
  "autoTimeOfWeekBin: {                optional, from R5.6, ignored if
                                         timeOfWeekBin is set, defines the time of
                                         week bin values to be set automatically
                                         depending on current time
    <offsetFrom>: <binName>            at least one required, offset wrt beginning
                                         of the week (xsdDuration) bin name
                                         (string) pair
  }
 },
 "interaction": <string>,              optional, if provided, the interaction log
                                         entry will contain interaction type
                                         <string> instead of the default value
                                         "rate". Supported from R4.5.
 "contextId": <string>,                optional, if provided, the given contextId
                                         will be logged in the interaction log.
                                         Supported from R5.2.
 "revenue": <decimal>,                 optional, if provided the given value will
                                         be logged for the interaction. Supported
                                         from R5.6.
 "currency": <string>,                 optional, if provided the given value will
                                         be logged for the interaction. Supported
                                         from R5.6.
 "maxExplanationLevel": integer        optional, if set to 3, the response contains
                                         feature level explanations of the score.
                                         Supported from R4.5, deprecated from R4.6.
 "explanationConfig": {                optional, supported from R4.6, if present,
                                         maxExplanationLeve is ignored
   "language": <langcode>,             optional, the desired language of the
                                         explanation, default: English
   "types": [<type_enum>],             only "feature" supported for feature level
                                         explanations equivalent to those of the
                                         preference engine
   "metadataLanguages": [<langcode>] optional, preferred language of the asset
                                         attributes used in the explanation,
                                         fallback to IDs and canonical names if
                                         none of the languages is available.
 }
}
```

Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store.

| ***result*** | **200 OK** | Request successful |
|---|---|---|

| | | |
|---|---|---|
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId or assetId or subscriber ID does not exist |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>] | |

### 3.9.11 Rate an Asset in multiple Profiles

By default the rating is automatically applied to one of the given profiles if there is exactly one profile matching (i.e. with rating significance > 0). If no unique matching profile can be found, the request returns 409 response code. If the query parameter allowMultipleMatching is set to true, the rating is applied to all profiles which are matching.
The assets given in toScore are then scored in all profiles specified.

| **POST** | /profile/<profileId1>,<profileId1>,.../scorer?errorTolerant=true&recordingSuggestion=Boolean&allowMultipleMatching=boolean&scoringProfiles=enum | |
|---|---|---|
| ***body*** | {"assetId": string512, | required, ID of the asset rated |
| | "rating": decimal, | required, range [-1,...+1],+1 for explicit "like", -1 for explicit "dislike" |
| | "ratingAggregation": enum, | optional, from R5.1, one of "accumulate", "overwrite", default value: "accumulate". Specifies if implicit ratings (-1 < ratingValue < 1) replace the previous rating or are added to the previous rating |
| | "type": enum, | optional, one of "single", "series", default: "single". If missing, the type is determined automatically: if seriesId is present: type=series, else: type=single |
| | "subscriber": string, | optional, subscriber ID |
| | "toScore":[string,...], | optional, list of asset IDs to be re-scored |
| | "setting": { | optional, viewing setting of the rating |
| | "timeOfDayBin": <string>, | optional, e.g. "morning", "evening" |
| | "timeOfWeekBin": <string>, | optional, e.g. "weekday", "weekend" |
| | "device": <string>, | optional, the device used |
| | "locationBin": <string>, | optional, e.g. "home", "work" |
| | "moodBin": <string>, | optional, e.g. "happy", "sad" |
| | "activity": <string>, | optional, e.g. "traveling", "relaxing" |
| | "autoTimeOfDayBin": { | optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair |
| | }, | |
| | "autoTimeOfWeekBin: { | optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time |
| | <offsetFrom>: <binName> | at least one required, offset wrt beginning |

| | |
|---|---|
| | of the week (xsdDuration) bin name (string) pair<br>}<br>},<br>"interaction": <string>, optional, if provided, the interaction log entry will contain interaction type <string> instead of the default value "rate". Supported from R4.5.<br>"contextId": <string>, optional, if provided, the given contextId will be logged in the interaction log. Supported from R5.2.<br>"revenue": <decimal>, optional, if provided the given value will be logged for the interaction. Supported from R5.6.<br>"currency": <string>, optional, if provided the given value will be logged for the interaction. Supported from R5.6.<br>"maxExplanationLevel": integer optional, if set to 3, the response contains feature level explanations of the score. Supported from R4.5, deprecated from R4.6.<br>"explanationConfig": { optional, supported from R4.6, if present, maxExplanationLeve is ignored<br>"language": <langcode>, optional, the desired language of the explanation, default: English<br>"types": [<type_enum>], only "feature" supported for feature level explanations equivalent to those of the preference engine<br>"metadataLanguages": [<langcode>] optional, preferred language of the asset attributes used in the explanation, fallback to IDs and canonical names if none of the languages is available.<br>}<br>}<br><br>Starting from R4.5 the profile selection algorithm for scoring is configurable via the parameter scoringProfiles. If scoringProfiles is omitted or set to "prefilter", all profiles the prefilter of which pass the asset are used for scoring. If there are multiple such profiles, the maximum of the scores is returned. If scoringProfiles is set to "filter", the profile filter is taken into account in addition. By using the value "filter" one can achieve that only those assets receive a score which would also be returned as blending preference engine recommendations. |
| ***result*** | **200 OK** Request successful<br>**400 BAD REQUEST** Malformed request<br>**404 NOT FOUND** one of the profile IDs or assetIds or does not exist |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>]<br><br>Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose |

| | |
|---|---|
| | code will be 200 and the response will only include the assets found in the store. |

### 3.9.12 Score Assets in single Profile

This request returns the recommendation scores of the assets in the request body. The scores are calculated using a single profile. Cf. Section 3.8.32. The request body parameters are explained in detail in Section 3.9.10.

| **POST** | /profile/<profileId>/scorer?errorTolerant=true&recordingSuggestion=boolean&scoringProfiles=enum | |
|---|---|---|
| ***body*** | {"toScore":[<assetId>,...], "maxExplananationLevel": integer, "explanationConfig": {"language": <langcode>, "types": [<type_enum>], "metadataLanguages": [<langcode>]}, "setting": @setting}<br><br>Feature level explanations of the scores are provided if maxExplanationLevel is set to 3 or if explanationConfig.types contains "feature". Note that other explanation levels are not supported. maxExplanationLevel is deprecated from R4.6 and the more flexible "explanationConfig" parameter shall be used. "explanationConfig" supports independent specifications of explanation language (default: English), of the explanation types (only "feature" supported), and of the language of asset attributes in the explanation (metadataLanguages, the first language from the list which is available in the asset is used, fallback to canonical names and IDs if none ie available). If explanationConfig is provided, maxExplanationLevel will be ignored. | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId or assetId does not exist |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>]<br>Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store.<br><br>Starting from R4.5 the profile selection algorithm for scoring is configurable via the parameter scoringProfiles. If scoringProfiles is omitted or set to "prefilter", all profiles the prefilter of which pass the asset are used for scoring. If there are multiple such profiles, the maximum of the scores is returned. If scoringProfiles is set to "filter", the profile filter is taken into account in addition. By using the value "filter" one can achieve that only those assets receive a score which would also be returned as blending preference engine recommendations. | |

### 3.9.13 Score Assets in multiple Profiles

This request returns the recommendation scores of the assets in the request body. The scores are calculated by selecting, for each asset individually, the best matching one from a list of profiles and

calculating the score in that Profile. The rating corresponds to the rating in the best matching Profile, too. Cf. Section 3.8.32. The request body parameters are explained in detail in Section 3.9.11

| **POST** | /profile/<profileId1>,<profileId1>,.../scorer?errorTolerant=true&recordingSuggestion=boolean&scoringProfiles=enum | |
|---|---|---|
| ***body*** | {"toScore":[<assetId>,...], "maxExplananationLevel": integer,<br>"explanationConfig": {"language": <langcode>, "types": [<type_enum>],<br>"metadataLanguages": [<langcode>]}, "setting": @setting}<br><br>Feature level explanations of the scores are provided if maxExplanationLevel is set to 3 or if explanationConfig.types contains "feature". Note that other explanation levels are not supported.<br>maxExplanationLevel is deprecated from R4.6 and the more flexible "explanationConfig" parameter shall be used. "explanationConfig" supports independent specifications of explanation language (default: English), of the explanation types (only "feature" supported), and of the language of asset attributes in the explanation (metadataLanguages, the first language from the list which is available in the asset is used, fallback to canonical names and IDs if none ie available). If explanationConfig is provided, maxExplanationLevel will be ignored.<br><br>The profile selection algorithm for scoring is configurable via the parameter scoringProfiles. If scoringProfiles is omitted or set to "prefilter", all profiles the prefilter of which pass the asset are used for scoring. If there are multiple such profiles, the maximum of the scores is returned. If scoringProfiles is set to "filter", the profile filter is taken into account in addition. By using the value "filter" one can achieve that only those assets receive a score which would also be returned as blending preference engine recommendations. | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | one of the profile IDs or assetIds or does not exist |
| ***response*** | [<br>{<br>"assetId": string512,<br>"score": decimal,<br>"rating": decimal,<br>"recordingSuggestion": boolean,<br>"explanations": [@explanation,...], see 3.17 for the definition<br>},<br>...<br>]<br>Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store. | |

### 3.9.14 **Retrieve Playlists**

This request returns the playlists associated with a profile.

| **GET** | /profile/<profileId>/playlists | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |
| ***response*** | [<listId>] | |

### 3.9.15 Update Playlists

This request is used to update the playlists associated with a profile.

| **POST** | /profile/<profileId>/playlists | |
|---|---|---|
| ***body*** | {"add": [<listId>],<br>"remove": [<listId>]<br>} | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

Adding of playlists is performed first, removing after. Note that update playlists does not check if the added playlists actually exist.

### 3.9.16 Retrieve Bookmark Lists

This request returns the bookmark lists associated with a profile. Supported from R5.6.

| **GET** | /profile/<profileId>/bookmarkLists | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |
| ***response*** | [<listId>] | |

### 3.9.17 Update Bookmark Lists

This request is used to update the bookmark lists associated with a profile. Supported from R5.6.

| **POST** | /profile/<profileId>/bookmarkLists | |
|---|---|---|
| ***body*** | {"add": [<listId>],<br>"remove": [<listId>]<br>} | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

Adding of bookmark lists is performed first, removing after. Note that update bookmark lists does not check if the added bookmark lists actually exist.

### 3.9.1 Rate a Feature

This request is used to communicate an explicit rating (like degree) of a feature-value pair (e.g. persons.actor = Clint Eastwood) to Ncanto. If a list of assets is provided in the request body, the recommendation scores of the assets will be recalculated after updating the profile with the new rating.

| **POST** | /profile/<profileId>/scorer?errorTolerant=true&recordingSuggestion=boolean |
|---|---|
| ***body*** | {"feature": string, required, name of the feature rated in dot notation<br>"featureValue": sting, required, value of the feature rated<br>"rating": decimal, optional, range [-1,...+1]. If missing or null, a previously present rating for the feature-value pair will be removed.<br>"subscriberId": string512,optional, subscriber ID<br>"toScore":[string,...] optional, list of asset IDs to be re-scored<br>"maxExplanationLevel": integer optional, if set to 3, the response contains feature level explanations of the score. |

| | |
|---|---|
| | Supported from R4.5, deprecated from R4.6.<br>"explanationConfig": { optional, supported from R4.6, if present, maxExplanationLeve is ignored<br>"language": <langcode>, optional, the desired language of the explanation, default: English<br>"types": [<type_enum>], only "feature" supported for feature level explanations equivalent to those of the preference engine<br>"metadataLanguages": [<langcode>] optional, preferred language of the asset attributes used in the explanation, fallback to IDs and canonical names if none of the languages is available.<br>}<br>}<br><br>Request parameter errorTolerant=true may be added to ignore "toScore" assetIds which are missing from the asset store. If the parameter is set, the respose code will be 200 and the response will only include the assets found in the store.<br><br>The supported feature names are summarized in |

| | | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId or feature or subscriber ID does not exist |

***response***

```
[
 {
  "assetId": string512,
  "score": decimal,
  "rating": decimal,
  "recordingSuggestion": boolean,
  "explanations": [@explanation,...],  see 3.17 for the definition
 },
 ...
]
```

| ***feature name*** | ***remark*** |
|---|---|
| sourceId | see corresponding asset attribute in section 3.6.7 |
| first | see corresponding asset attribute in section 3.6.7 |
| live | see corresponding asset attribute in section 3.6.7 |
| onDemand | see corresponding asset attribute in section 3.6.7 |
| startBin | see Table 3 |
| durationBin | see Table 3 |
| priceBin | see corresponding asset attribute in section 3.6.7 |
| ageBin | see Table 5 |
| yearBin | see Table 5 |
| productionCountry | see corresponding asset attribute in section 3.6.7 |
| tip | see corresponding asset attribute in section 3.6.7 |
| audioLang | see corresponding asset attribute in section 3.6.7 |
| editorialRating | see corresponding asset attribute in section 3.6.7 |
| surroundSound | see corresponding asset attribute in section 3.6.7 |
| subtitled | see corresponding asset attribute in section 3.6.7 |
| assetType | see corresponding asset attribute in section 3.6.7 |
| communityRatingBin | see Table 5 |
| forBlind | see corresponding asset attribute in section 3.6.7 |

| | |
|---|---|
| bw | see corresponding asset attribute in section 3.6.7 |
| hd | see corresponding asset attribute in section 3.6.7 |
| interactive | see corresponding asset attribute in section 3.6.7 |
| threeD | see corresponding asset attribute in section 3.6.7 |
| programType | see corresponding asset attribute in section 3.6.7 |
| genreFine | see corresponding asset attribute in section 3.6.7 |
| genreMedium | see corresponding asset attribute in section 3.6.7 |
| genreBroad | see corresponding asset attribute in section 3.6.7 |
| themeCircle | see corresponding asset attribute in section 3.6.7 |
| keywords.<langcode> | see below |
| persons.<function> | see below |
| adProduct | see corresponding asset attribute in section 3.6.7 |
| adCompany | see corresponding asset attribute in section 3.6.7 |
| adTaxonomy | see corresponding asset attribute in section 3.6.7 |
| targetDemographic | see corresponding asset attribute in section 3.6.7 |
| customProperties.<attributeName>.... | see below |

**Table 4: Supported Feature Names**

In case of keyword, person, and customProperties ratings the feature names and values have to be generated according to the following rules:

Keywords: keywords.<language>:<keyword>, e.g. for "keywords": {"eng":"mykeyword"} the feature name is "keywords.eng", the value is " mykeyword".
Persons: persons.<function>:<canonicalName>, e.g. for "persons": {"actor":[{"canonical":"Robert De Niro"}]} the feature name is "persons.actor", the value " Robert De Niro".
Custom properties: customProperties.<pathElement>.<pathElement>:<value>
E.g. for "customProperties" :

```
{
  "property1": ["a", "b", "c"],
  "property2": 12.0,
  "property3": {
    "typeA": 1,
    "typeB": 2
  }
}
```

the feature names/value pairs are "customProperies.property1": "a" or "customProperties.property3.typeB": "2".

| | |
|---|---|
| Production year bins | bin 0 ... 0-1950<br>bin 1 ... 1945-1955<br>bin 2 ... 1950-1960<br>bin 3 ... 1955-1965<br>bin 4 ... 1960-1970<br>bin 5 ... 1965-1975<br>bin 6 ... 1970-1980<br>bin 7 ... 1975-1985<br>bin 8 ... 1980-1990 |

| | |
|---|---|
| | bin 9 ... 1985-1995<br>bin 10 ... 1990-2000<br>bin 11 ... 1995-2005<br>bin 12 ... 2000-2010<br>bin 13 ... 2005-2015<br>bin 14 ... 2010-2020<br>... |
| Age bins | 0 = [0, 4[ years<br>1 = [4, 7[ years<br>2 = [7, 14[ years<br>3 = [14, 18[ years<br>4 = [18, inf] years |
| Community rating bins | 0 = "hated": [0.0, 0.2[<br>1 = "unpopular": [0.2, 0.4[<br>2 = "ignored": [0.4, 0.6[<br>3 = "popular": [0.6, 0.8[<br>4 = "loved": [0.8, 2.0] |
| Start time bins | 0 = "night": [01:00, 06:00[<br>1 = "morning": [06:00, 11:00[<br>2 = "noon": [11:00, 15:00[<br>3 = "afternoon": [15:00, 18:00[<br>4 = "early evening": [18:00, 22:00[<br>5 = "late evening": all others |
| Awards | <awardName>_0: 0...3 awards of type <awardName> present in the asset<br><awardName>_1: 2...5 awards of type >awardName> present in the asset<br><awardName>_2: 3...10 awards of type >awardName> present in the asset<br><awardName>_3: 5 or more awards of type >awardName> present in the asset |

Table 5: Binning of feature values

### 3.9.2 Retrieve name of profile

This request is used to retrieve the name attribute of a profile.

| **GET** | `/profile/<profileId>/name` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |
| ***response*** | `{"name": <profileName>}` | |

### 3.9.3 Update name of profile

This request is used to create or to update the name attribute of a profile.

| **PUT** | `/profile/<profileId>/name` | |
|---|---|---|
| ***body*** | `{"name": <profileName>}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | profileId does not exist |

### 3.9.4 Retrieve profile strengths

The profile strength feature returns an indication of the richness of one or multiple preference profiles. The aim is to motivate the user to provide more rating feedback. Note that in comparison to the profile's error ratio the profile strength is

- Always available in real time (the error ratios are only calculated once per day if the profile has received at least 10 positive plus 10 negative ratings)
- More deterministic: additional feedback always increases the profile strength while the error ratio may even increase if the new feedback contradicts to previous knowledge

The profile strength algorithm takes into account asset likes (rating value = 1.0), asset dislikes (rating value=-1.0), positive implicit asset feedback (0.0 < rating value < 1.0), negative implicit asset feedback (-1.0 < rating value < 0.0), positive feature ratings (rating value > 0.0) and negative feature ratings (rating value < 0.0) independently.

The profile strength is a decimal value in the range [0:1], composed of the following contributions:

- Likes: maximum contribution 0.2
- Dislikes: maximum contribution 0.2
- Positive implicit feedback: maximum contribution 0.1
- Negative implicit feedback: maximum contribution 0.1
- Positive feature ratings: maximum contribution 0.2
- Negative feature ratings: maximum contribution 0.2

As a consequence if a client does not use all feedback types, the maximum value of the profile strength will be lower than 1.0. Some examples:

- Only asset ratings are possible, no feature ratings: maximum profile strength value 0.6
- Only likes and dislikes are used: maximum profile strength value 0.4
- Only positive implicit asset ratings (e.g. viewing information) are available: maximum profile strengh value 0.1

If multiple profile IDs are specified, the profile strength values are calculated independently and returned as a list.

| **GET** | `/profile/<profileId1>,<profileId2>,.../strength` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | At least one of the profiles not found |
| ***response*** | | |

```
[
  {
    "profileId": <profileId1>,
    "strength": <decimal>
  },
  {
    "profileId": <profileId2>,
    "strength": <decimal>
  },
  ...
]
```

### 3.9.5 Retrieve like degrees of a profile

The like degree is Ncanto's estimate how well a feature / feature value pair (e.g. genreBroad: romance) is perceived. Like degrees are decimal values normalized to the range [0:1], values below 0.5 indicate rejection, values above 0.5 liking.
The GET version of the request returns the feature / feature value pairs of the profile which have the highest like degrees.

| **GET** | `/profile/<profileId>/like-degrees?count=<integer>` | |
|---|---|---|
| ***path parameters*** | `profileId` | Unique ID of the profile the like degrees are requested of. |
| ***query parameters*** | `count` | Optional, type positive integer. The maximum number of like degrees to be returned. |
| ***body*** | - | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Count is negative |
| | **404 NOT FOUND** | profileId not found |
| ***response*** | `{`<br>`  "totalNumberOfFeatureValuePairs": <int>, the total number of feature value pairs in the preference profile, the max. meaningful count value.`<br>`  "likeDegrees": [`<br>`  {`<br>`  "feature": <string>, the feature name (see Table 4 for the list of supported feature names, e.g. genreBroad`<br>`  "featureValue": <any>, the feature value, the data type depends on the feature's data type. E.g. "romance"`<br>`  "likeDegree": <decimal> the like degree in range [0:1]`<br>`  },`<br>`  ...`<br>`  ]`<br>`}` | |

The POST request allows to filter to individual features (e.g. genre like degrees only). From R5.0 a viewing setting can be provided, too. If a viewing setting is specified, Ncanto will return like degrees learned in the same viewing setting. If there is no or not enough information about the desired viewing setting available, like degrees will be blended from all learned viewing settings, emphasizing the desired viewing setting.

| **POST** | `/profile/<profileId>/like-degrees?count=<integer>` | |
|---|---|---|
| ***path parameters*** | `profileId` | Unique ID of the profile the like degrees are requested of. |
| ***query parameters*** | `count` | Optional, type positive integer, supported from R5.0. The maximum number of like degrees to be returned. |
| ***body*** | `{`<br>`  "features": [<featureName>], optional. The supported feature names correspond to the root features listed in Table 4, e.g. "genreBroad" or "persons" but not "persons.actor". If no features are provided, the response will contain all features of the profile.`<br>`  "setting": { optional, supported from R5.0, the viewing setting the like degrees of which are requested.`<br>`  "timeOfDayBin": <string>, optional, e.g. "morning", "evening"` | |

| | | |
|---|---|---|
| | "timeOfWeekBin": <string>, optional, e.g. "weekday", "weekend"<br>"device": <string>, optional, the device used<br>"locationBin": <string>, optional, e.g. "home", "work"<br>"moodBin": <string>, optional, e.g. "happy", "sad"<br>"activity": <string>, optional, e.g. "traveling", "relaxing"<br>"autoTimeOfDayBin": { optional, from R5.6, ignored if timeOfDayBin is set, defines the time of day bin values to be set automatically depending on current time<br><offsetFrom>: <binName> at least one required, offset wrt beginning of the day (xsdDuration) bin name (string) pair<br>},<br>"autoTimeOfWeekBin: { optional, from R5.6, ignored if timeOfWeekBin is set, defines the time of week bin values to be set automatically depending on current time<br><offsetFrom>: <binName> at least one required, offset wrt beginning of the week (xsdDuration) bin name (string) pair<br>}<br>}<br>} | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Invalid feature name |
| | **404 NOT FOUND** | profileId not found |
| ***response*** | {<br>"totalNumberOfFeatureValuePairs": <int>, the total number of feature value pairs in the preference profile<br>"likeDegrees": [<br>{<br>"feature": <string>, the feature name (see section 3.9.1 for the list of supported feature names, e.g. genreBroad<br>"featureValue": <any>, the feature value, the data type depends on the feature's data type. E.g. "romance"<br>"likeDegree": <decimal> the like degree in range [0:1]<br>},<br>...<br>]<br>} | |

## *3.10 Annotation Management*

### 3.10.1 Create or Update an Annotation Template

This request creates or updates the set of metadata attributes which accompany recommendations.

| | | |
|---|---|---|
| **PUT** | /annotation/<annotationId> | |
| ***body*** | @annotation | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

**@annotation =**
{"annotationId": string512, optional, unique ID of the annotation template, is

```
                                        ignored in the PUT request

 "uiGenreType": [string,...]           optional, list of the UI genre types
                                         to be returned (see uiGenre <type> in @asset)
 "sourceNameType": [string,...]        optional, list of the source name
                                         types to be returned (see sourceName
                                         <type> in @asset
 "synopsisType": [string,...]          optional, list of the synopsis <type>s to be
                                         returned
 "attributes":[enum,...],              optional, list or metadata attributes to be
                                         returned as annotation
 "conditionalAttributes": {            optional, supported from R4.2, defines attributes
                                         which should only be returned if the condition is
                                         met
  <attribute>: <@genericFilter>        at least one attribute/condition pair required
                                         <attribute>: asset attribute in dot notation
                                         @genricFilter: condition according to
                                         genericFilter syntax (see Section 3.6.11). Note
                                         that in annotation conditions the "and" operator
                                         does not make sense, it is always interpreted as
                                         "nestedAnd", i.e.in case of list attributes all
                                         criteria have to be matched by the same list
                                         element.
}

Supported attributes are all top level attributes of @asset.
```

Starting from R4.0 direct addressing of attribute types using the dot notation is supported too. I.e. `{"attributes": ["synopses.short"]}` may be used instead of `{"attributes":["synopses"], "synopsisType": "short"}`
Please note that the dot notation allows to select a specific language too (e.g. `{"attributes": ["synopsis.short.eng"]}`), but this is not recommended. Language selection shall be done via the context.

### 3.10.2 Retrieve one or multiple Annotation Template(s)

| **GET** | `/annotation/<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | annotationId does not exist |
| ***response*** | `@annotation` | |

| **GET** | `/annotation?count=<no_of_annotationIds>&startAfter=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@annotation,...]`<br>Annotation IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.10.3 Delete Annotation Template

| **DELETE** | `/annotation/<annotationId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | annotationId successfully deleted or not found |

## *3.11 Moderated List Management*

### 3.11.1 Create or Update a Moderated List

This request is used to create or update a list of editorial recommendations. The corresponding recommendation scores have to be provided by the editor, too.
Note that Ncanto does not check if the assetId which is put into the moderated list exists in the active asset stores or not.

| **PUT** | `/list/moderated/<listId>` | |
|---|---|---|
| ***body*** | `@assetList` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

```
@assetList =
{"listId": string512,                  optional, unique ID of the moderated list
 "assets": [@scoredAsset,...],         optional, empty lists are allowed
 "customProperties": noDotsJson        optional, arbitrary customer defined properties of
                                         the moderated list as JSON object without dots in
                                         the key names
}

@scoredAsset =
{"assetId": string512,                 required, asset ID
 "score": decimal,                     required, editorial score in range [0,1]
 "timestamp": timestamp,               optional, time of adding / modifying the asset.
                                         Note that the timestamp is not set automatically
                                         by Ncanto.
 "customProperties": noDotsJson,       optional, customer defined properties of the
                                         editorial recommendation
 "asset": @asset,                      optional, metadata of the asset. "asset" is ignored
                                         by PUT requests and returned filtered by the
                                         annotation template by GET requests
 "customExplanations": {               optional, from R5.0, custom explanations applied to
                                         the moderated asset if recommended by a moderated
                                         engine.
  <type>: {                            mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>       mandatory, language / explanation text pairs. In
                                         contrast to the customExplanations of the context
                                         no variables are supported here.
  }
 }
}
```

### 3.11.2 Retrieve one or multiple Moderated List(s)

| **GET** | `/list/moderated/<listId>?annotation=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | listId not found |
| ***response*** | `@assetList` | |

| **GET** | `/list/moderated?count=<no_of_lists>&startAfter=<listId>&annotation=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@assetList]` | |

| | |
|---|---|
| | Lists are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. |

An optional parameter `annotation` can be used to retrive selected metadata fields of the assets. Note if a moderated list is retrieved without annotation, Ncanto does not check if the asset IDs are present in the active stores. In contrast, if an annotation is specified, only assets which are part of an active store will be returned.

### 3.11.3 Delete a moderated list

| **DELETE** | `/list/moderated/<listId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | listId successfully deleted or not found |

## *3.12 Statistical List Management*

Statistical Lists are updated by Ncanto automatically in one hour intervals. Until Ncanto R5.2, statistical lists with fixed time windows are supported. Table 6 summarizes the allowed statistical list IDs supported until R5.2, which are deprecated and replaced by list type / time window starting from R5.3.

| **listId** | **description** |
|---|---|
| `mostWatched8h` | Most watched assets of the last 8 hours (sliding window, updated each hour). Takes into account all interactions of type "play". |
| `mostWatched1d` | Most watched assets of the last day (sliding window, updated each hour). Takes into account all interactions of type "play". |
| `mostWatched1w` | Most watched assets of the last week (sliding window, updated each hour). Takes into account all interactions of type "play". |
| `bestRated8h` | Best rated assets of the last 8 hours (sliding window, updated each hour). Takes into account the average rating of an asset as well as the number of ratings the asset received. |
| `bestRated1d` | Best rated assets of the last day (sliding window, updated each hour). Takes into account the average rating of an asset as well as the number of ratings the asset received. |
| `bestRated1w` | Best rated assets of the last week (sliding window, updated each hour). Takes into account the average rating of an asset as well as the number of ratings the asset received. |
| `hottest4h, aliased as trending4h from R4.0` | Trending assets of the last 4 hours (sliding window, updated each hour), i.e. assets which received the most attention in the last observation period compared to the previous observation period. Note that "hottest" and "trending" are aliases and refer to the same statistical lists. All interactions with an asset, e.g. of types play, rate, etc. are included in the hottest list calculation. |
| `hottest12h, aliased as trending12h from R4.0` | Trending assets of the last 12 hours (sliding window, updated each hour), i.e. assets which received the most attention in the last observation period compared to the previous observation period. Note that "hottest" and "trending" are aliases and refer to the same statistical lists. All interactions |

| | |
|---|---|
| | with an asset, e.g. of types play, rate, etc. are included in the hottest list calculation. |
| `hottest3d, aliased as trending3d from R4.0` | Trending assets of the last 3 days (sliding window, updated each hour), i.e. assets which received the most attention in the last observation period compared to the previous observation period. Note that "hottest" and "trending" are aliases and refer to the same statistical lists. All interactions with an asset, e.g. of types play, rate, etc. are included in the hottest list calculation. |
| `latestArrivals1d` | Assets, the availabilityStartTime of which has passed during the last day, sorted by ascending elapsed availability. This statistical list is limited to a maximum of 100 assets. |
| `latestArrivals1w` | Assets, the availabilityStartTime of which has passed during the last week, sorted by ascending elapsed availability. This statistical list is limited to a maximum of 100 assets. |
| `latestArrivals4w` | Assets, the availabilityStartTime of which has passed during the last four weeks, sorted by ascending elapsed availability. This statistical list is limited to a maximum of 100 assets. |
| `mostPopular8h` | Assets which received most attention in the last 8 hours. All interactions with an asset, e.g. of types play, rate, etc. are included in the popular list calculation. |
| `mostPopular1d` | Assets which received most attention in the last 24 hours. All interactions with an asset, e.g. of types play, rate, etc. are included in the popular list calculation. |
| `mostPopular1w` | Assets which received most attention in the last 7 days. All interactions with an asset, e.g. of types play, rate, etc. are included in the popular list calculation. |
| `mostShared8h` | Assets which received the most “share” interactions in the last 8 hours. |
| `mostShared1d` | Assets which received the most “share” interactions in the last 24hours. |
| `mostShared1w` | Assets which received the most “share” interactions in the last 7 days. |

**Table 6: Statistical list IDs supported by Ncanto (until R5.2, deprecated from R5.3)**

Starting from R5.3 a generalized statistical list approach is introduced and the fixed list IDs above are deprecated. The generalized approach comprises separate list type and time window parameters. The supported list types are summarized in Table 8. The time window is specified in XSD:duration format.

| list types | description |
|---|---|
| `mostWatched` | Most watched. Takes into account all interactions of type "play". |
| `bestRated` | Best rated. Takes into account the average rating of an asset as well as the number of ratings the asset received. |
| `trending` | Trending, i.e. assets which received the most attention in the last time window compared to the previous time window. All interactions with an asset, e.g. of types play, rate, etc. are included in the hottest list calculation. |

| | |
|---|---|
| latestArrivals | Assets, the availabilityStartTime of which has passed during the time window, sorted by ascending elapsed availability. This statistical list is limited to a maximum of 100 assets. |
| mostPopular | Assets which received most attention during the time window. All interactions with an asset, e.g. of types play, rate, etc. are included in the popular list calculation. |
| mostShared | Assets which received the most "share" interactions in the last 8 hours. |
| mostShared | Assets which received the most "share" interactions during the time window. |

**Table 7: Statistical list types supported by Ncanto from R5.3**

### 3.12.1 Retrieve an asset level Statistical List

The following request is supported from R5.3:

| **GET** | /list/statistical/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]<br>&subscriberGroups=[<subGroupId>]&annotation=<annotationId> | |
|---|---|---|
| ***query parameters*** | assetTypes | Optional, type list of strings. If provided, only assets of the given assetType will be included in the statistical list. |
| | programTypes | Optional, type list of strings. If provided, only assets of the given programType will be included in the statistical list. |
| | assetFamilies | Optional, from R5.4, type list of strings. If provided, only assets of the given assetFamilies will be included in the statistical list. |
| | subscriberGroups | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. Not supported in case of listType latestArrivals. |
| | annotation | Optional, type string. Reference to the annotation template specifying which asset attributes should be returned along with the list. |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Wrong list type or time window, or invalid query parameter. |
| | **404 NOT FOUND** | annotation not found |
| ***response*** | @assetList | |

Note that if a statistical list is retrieved without annotation, Ncanto does not check if the asset IDs are still present in the active stores. In contrast, if an annotation is specified, only assets which are part of an active store will be returned.

The following request using the list IDs is deprecated from R5.3:

| **GET** | /list/statistical/<listId>?annotation=<annotationId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Invalid query parameter |
| | **404 NOT FOUND** | listId not found |
| ***response*** | @assetList | |

The following request returning multiple statistical lists in one call is deprecated from R5.0:

| **GET** | /list/statistical?count=<no_of_lists>&startAfter=<listId>&annotation=<annotationId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | [@assetList]<br>Lists are returned in a deterministic but unsorted way. Note that for performance reasons latestArrivals lists are not included in the response but have to be queried individually. | |

### 3.12.2 **Retrieve program and series level Statistical Lists**

Starting from R5.4 Ncanto maintains statistical lists at programId and seriesId levels, too. All the list types of Table 7 except "latestArrivals" are supported. The statistical lists at program and series level are generated from the same (asset level) interaction data which is also used for the asset level statistical lists, but using the programId or seriesId of the assets.

| **GET** | `/list/statisticalProgram/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]&subscriberGroups=[string]` | |
|---|---|---|
| ***query parameters*** | `assetTypes` | Optional, type list of strings. If provided, only assets of the given assetType will be considered when creating the statistical list. |
| | `programTypes` | Optional, type list of strings. If provided, only assets of the given programType will be considered when creating the statistical list. |
| | `assetFamilies` | Optional, type list of strings. If provided, only assets of the given assetFamilies will be considered when creating the statistical list. |
| | `subscriberGroups` | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Wrong list type, time window, or query parameter. |
| ***response*** | `{"programs": [{"programId": string, "score": decimal}]}` | |

| **GET** | `/list/statisticalSeries/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]&subscriberGroups=[string]` | |
|---|---|---|
| ***query parameters*** | `assetTypes` | Optional, type list of strings. If provided, only assets of the given assetType will be considered when creating the statistical list. |
| | `programTypes` | Optional, type list of strings. If provided, only assets of the given programType will be considered when creating the statistical list. |
| | `assetFamilies` | Optional, type list of strings. If provided, only assets of the given assetFamilies will be considered when creating the statistical list. |
| | `subscriberGroups` | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Wrong list type, time window, or query parameter. |
| ***response*** | `{"series": [{"seriesId": string, "score": decimal}]}` | |

### 3.12.3 **Recalculate Statistical List**

For performance reasons statistical lists are cached for a short period of time depending on the time window. The caching durations are described in Section 2.3.3. In order to clear the cache and to force the recalculation for one specific statistcal list the following request is supported from R5.3:

| **DELETE** | `/list/statistical/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]&subscriberGroups=[string]` | |
|---|---|---|
| ***query parameters*** | `assetTypes` | Optional, type list of strings. If provided, only assets of the given assetType will be included in the statistical list. |
| | `programTypes` | Optional, type list of strings. If provided, only assets of the given programType will be included in the statistical list. |
| | `assetFamilies` | Optional, from R5.4, type list of strings. If provided, only assets of the given assetFamilies will be included in the statistical list. |
| | `subscriberGroups` | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. |
| ***result*** | **200 OK** | Request successful |

From R5.4 the following API calls are supported, too:

| **DELETE** | `/list/statisticalProgram/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]&subscriberGroups=[string]` | |
|---|---|---|
| ***query parameters*** | `assetTypes` | Optional, type list of strings. If provided, only assets of the given assetType will be included in the statistical list. |
| | `programTypes` | Optional, type list of strings. If provided, only assets of the given programType will be included in the statistical list. |
| | `assetFamilies` | Optional, type list of strings. If provided, only assets of the given assetFamilies will be included in the statistical list. |
| | `subscriberGroups` | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. |
| ***result*** | **200 OK** | Request successful |

| **DELETE** | `/list/statisticalSeries/<listType>/<timeWindow>?assetTypes=[<string>]&programTypes=[<string>]&assetFamilies=[string]&subscriberGroups=[string]` | |
|---|---|---|
| ***query parameters*** | `assetTypes` | Optional, type list of strings. If provided, only assets of the given assetType will be included in the statistical list. |
| | `programTypes` | Optional, type list of strings. If provided, only assets of the given programType will be included in the statistical list. |
| | `assetFamilies` | Optional, type list of strings. If provided, only assets of the given assetFamilies will be included in the statistical list. |
| | `subscriberGroups` | Optional from R5.6, type list of strings. If provided, only interactions of the subscribers which are member of at least one of the given groups are taken into account. |
| ***result*** | **200 OK** | Request successful |

The legacy syntax specifying the statistical list via list ID is supported, too, but it is deprecated from R5.3:

| **DELETE** | `/list/<listId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |

## *3.13 Playlist Management*

The metadata of assets in a playlist is persisted by Ncanto in order to allow for rating and scoring of the playlist assets even if they are not available in the active store anymore. Assets remain in a playlist until they are removed by the client, no automatic purging is supported.
Assets in a playlist may be used as candidates for recommendations.

### 3.13.1 **Create or update a playlist**

This request is used to create or restore a playlist. Note that because the metadata of playlist assets is stored by Ncanto, all assets on the playlist have to be available in the active or playlist asset stores, otherwise a not found exception is thrown.

| **PUT** | `/list/playlist/<listId>` | |
|---|---|---|
| ***body*** | `@assetList` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | At least one of the asset IDs in the @assetList has not been found. |

### 3.13.2 Add assets to and remove assets from a playlist

To add assets to or remove assets from a playlist, use the POST method described below. Again, the metadata of the added assets is saved by Ncanto in a special asset store, hence the assets to be added have to be available in the active or playlist asset store.

| **POST** | `/list/playlist/<listId>` | |
|---|---|---|
| ***body*** | `{"add": [@scoredAssetId],`<br>`"remove": [@scoredAssetId]`<br>`}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | at least one of the assets to be added has not been found. |

Adding of assets is performed first, removing after. Note that if an asset is added to a non-existing playlist, the list will be created automatically.

```
@scoredAssetId =
{"assetId": string512,                  required, asset ID
 "score": decimal,                      optional, personal score in range [0;+1]
 "rating": decimal,                     optional, personal rating in range [-1;+1]
 "customProperties": noDotsJson         optional, customer defined properties of the
                                           playlist asset
}
```

### 3.13.3 Retrieve one or multiple playlist(s)

Retrieve single playlist:

| **GET** | `/list/playlist/<listId>?annotation=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | listId not found |
| ***response*** | `@assetList` | |

Retrieve a group of playlists:

| **GET** | `/list/playlist?listId=<listId>&listId=<listId>&...&annotation=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | one of the listIds not found |
| ***response*** | `[@assetList]` | |

Retrieve all playlists:

| **GET** | `/list/playlist?count=<no_of_lists>&startAfter=<listId>&annotation=<annotationId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@assetList]`<br>Lists are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.13.4 Delete a playlist

| **DELETE** | `/list/playlist/<listId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | listId successfully deleted or not found |

## *3.14 Bookmark List Management*

In contrast to playlists bookmark lists can only refer to assets still available in the active store. Hence the metadata of bookmark list assets is not persisted, and assets are automatically removed from a bookmark list as soon as they are removed from the active store. In addition an expiry timestamp may be defined for each asset in the list, after which the asset is removed automatically, too.

### 3.14.1 **Create or update a bookmark list**

This request is used to create or restore a bookmark list. Note that assets on the bookmark list have to be available in the active stores, otherwise a not found exception is thrown.

| **PUT** | `/list/bookmarklist/<listId>` | |
|---|---|---|
| ***body*** | `@assetListTtl` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | At least one of the asset IDs in the @assetList was not found. |

```
@assetListTtl =
{"listId": string512,                  optional, unique ID of the moderated list
 "assets": [@scoredAssetTtl,...],      optional, empty lists are allowed
 "customProperties": noDotsJson        optional, arbitrary customer defined properties of
                                         the moderated list as JSON object without dots in
                                         the key names
}

@scoredAssetTtl =
{"assetId": string512,                 required, asset ID
 "score": decimal,                     required, editorial score in range [0,1]
 "timestamp": timestamp,               optional, time of adding / modifying the asset.
                                         Note that the timestamp is not set automatically
                                         by Ncanto.
 "expiryDateTime": timestamp,          optional, the expiry date after which the asset
                                         shall be removed automatically from the list.
 "customProperties": noDotsJson,       optional, customer defined properties of the
                                         editorial recommendation
 "asset": @asset,                      optional, metadata of the asset. "asset" is ignored
                                         by PUT requests and returned filtered by the
                                         annotation template by GET requests
 "customExplanations": {               optional, from R5.0, custom explanations applied to
                                         the moderated asset if recommended by a moderated
                                         engine.
  <type>: {                            mandatory, explanation type (string)
   <langcode>: <explanationText>       mandatory, language / explanation text pairs. In
                                         contrast to the customExplanations of the context
                                         no variables are supported here.
  }
 }
}
```

### 3.14.2 **Add assets to and remove assets from a bookmark list**

To add assets to or remove assets from a bookmark list, use the POST method described below. Again, the the assets to be added have to be available in the active stores.

| **POST** | `/list/bookmarklist/<listId>` |
|---|---|

| | | |
|---|---|---|
| ***body*** | `{"add": [@scoredAssetIdTtl],`<br>`"remove": [@scoredAssetIdTtl]`<br>`}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | at least one of the assets to be added has not been found. |

Adding of assets is performed first, removing after. Note that if an asset is added to a non-existing bookmark list, the list will be created automatically.

```
@scoredAssetIdTtl =
{"assetId": string512,                  required, asset ID
 "score": decimal,                      optional, personal score in range [0;+1]
 "rating": decimal,                     optional, personal rating in range [-1;+1]
 "customProperties": noDotsJson,        optional, customer defined properties of the
                                          playlist asset
 "expiryDateTime": timestamp            optional, the expiry date after which the asset
                                          shall be removed automatically from the list.
}
```

### 3.14.3 Retrieve one or multiple bookmark list(s)

Retrieve single bookmark list:

| | | |
|---|---|---|
| **GET** | `/list/bookmarklist/<listId>?annotation=<annotationId>` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | listId not found |
| ***response*** | `@assetListTtl` | |

Retrieve a group of bookmark lists:

| | | |
|---|---|---|
| **GET** | `/list/bookmarllist?listId=<listId>&listId=<listId>&...&annotation=<annotationId>` | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | one of the listIds not found |
| ***response*** | `[@assetListTtl]` | |

Retrieve all bookmark lists:

| | | |
|---|---|---|
| **GET** | `/list/bookmarklist?count=<no_of_lists>&startAfter=<listId>&annotation=<annotationId>` | |
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@assetListTtl]`<br>Lists are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.14.4 Delete a bookmark list

| | | |
|---|---|---|
| **DELETE** | `/list/bookmarklist/<listId>` | |
| ***result*** | **204 NO RESPONSE** | listId successfully deleted or not found |

## *3.15 Ad Pool Management*

Ad Pools define which advertising assets are available and applicable to (may be placed into) which asset. Ad Pools are typically an output of the customer's ad campaign manager. An Ad Pool comprises two parts:

1. a list of ad assets available for placement into any asset
2. a list of ad assets dedicated for placement into individual assets

Ad Pools are expected to be updated in real time by the campaign manager.

### 3.15.1 **Create or Update an Ad Pool**

This request creates or updates an ad pool

| **PUT** | `/adpool/<adpoolId>` | |
|---|---|---|
| ***body*** | `@adpool` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

```
@adpool =
{"adpoolId": string512,                 optional, unique ID of the ad pool, ignored
                                          by the PUT request
 "genericAds": [<adAssetId>],           optional, list of ads which can be placed into any
                                          asset
 "dedicatedAds": {
      <assetId>: [<adAssetId>,...],     optional, list of ads per asset
      }
}
```

### 3.15.2 **Retrieve one or multiple Ad Pool(s)**

Retrieve single ad pool:

| **GET** | `/adpool/<adPoolId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | adPoolId not found |
| ***response*** | `@adpool` | |

Retrieve a group of ad pools:

| **GET** | `/adpool?adpoolId=<adpoolId>&adpoolId=<adpoolId>&...` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | one of the adPoolds not found |
| ***response*** | `[@adpool]` | |

Retrieve all adPools:

| **GET** | `/adpool?count=<no_of_adpools>&startAfter=<adpoolId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@adpool]`<br>Ad pools are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.15.3 **Delete an Ad Pool**

| **DELETE** | `/adpool/<adpoolId>` |
|---|---|

| | | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | ad pool successfully deleted or not found |

### 3.15.4 Ingest Ad Pool

This request deletes the ad pool and triggers an asynchronous full import of the ad pool.

| **POST** | `/adpool/<adpoolId>` | |
|---|---|---|
| ***body*** | `{"protocol":"IPDS",` | `required, only "IPDS" corresponding to Axel Springer IPDS format is supported` |
| | `"feed":<feedUrl>` | `required, URL of the IPDS stream` |
| | `}` | |
| ***result*** | **202 ACCEPTED** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| | **404 NOT FOUND** | adPoolId does not exist |

## *3.16 Context Management*

See 2.3 for an explanation of the concepts used in this chapter.

### 3.16.1 Create or Update a Context

| **PUT** | `/context/<contextId>` | |
|---|---|---|
| ***body*** | `@context` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

Please note that most of the attributes of a Context are optional because the pre-defined Context is combined with the ad-hoc Context at runtime.
Which attributes are required to arrive at a valid configuration at runtime, is summarized in Section 2.3.9.
Note that all unconditional rules have to be placed in front of conditional rules in the context.

```
@context =
{"contextId": string512,                optional, unique name of the context, ignored by
                                          the PUT request
 "contextNames": {                      optional, from R5.3, name of the context (to be
                                          used e.g. as container title in the UI of the
                                          client)
   <langcode>: <contextName>            mandatory, language / context name pairs. The
                                          context name may include references to context
                                          variables in the syntax
                                          ${<variableName!"<defaultValue>"}.
                                          Variable names which cannot be resolved are
                                          replaced by the <defaultValue> if one is
                                          specified, or by "" if no default is given.
 },
 "presentationProperties": noDotsJson,  optional, supported from R5.6. If present, the
                                          contents are copied into the recommendation
                                          response. Intended to communicate presentation
                                          and layout parameters to the client.
 "variables": {                         optional, supported from R5.0, definition of
                                          variables used in the context.
   <variableName>: @variableDef         required, variable name/definition pairs, names are
                                          of type non-empty string
 }
 "rules": [@rule,...]                   required, business rules of the context
```

```
}
```

The attributes required to define a variable depend on the variable type which can be one of the following values: "likeDegree", "queryParameter" (from R5.1), "timestamp" (from R5.3), "subscriberProperty" (from R5.6), "seedProperty" (from R5.6).

```
@variableDef =
{
  "type": "likeDegree",                 required
  "profileName":    string,            optional for variables of type "likeDegree",
                                          defines which profile of the subscriber
                                          the like degree has to be taken from. Ignored
                                          if profileId is present, too.
  "profileId": string,                  optional for variables of type "likeDegree",
                                          defines which profile to be used via
                                          reference to the query parameter of the
                                          recommendation request.
  "feature": string,                    required for variables of type "likeDegree", same
                                          as in like degree request, see Section 3.8.41 for
                                          details.
  "setting": @setting,                  optional for variables of type "likeDegree", same
                                          as in like degree request, see Section 3.8.41 for
                                          details.
  "index": integer,                     optional for variables of type "likeDegree",
                                          default=0, non-negative integer, defines
                                          which like degree should be used. By default
                                          (index=0) use the featureValue with the highest
                                          like degree. If index=1 use the featureValue with
                                          the second highest like degree
  "minLikeDegree": <decimal>,           optional, range [0:1], default value 0. If the like
                                          degree of the feature value addressed by the
                                          index is lower than minLikeDegree, the variable
                                          is not resolved. Supported from R5.1.
  "default": string,                    optional, from R5.4, specifies a default value for
                                          the variable, to be used if the variable cannot
                                          be resolved
  "patches": [                          optional, from R5.1, defines where the variable
                                          will be used in the context
    {
      "operation": <enum>,              optional, allowed value = default value "replace"
      "path": <string>                  required, pointer in dot notation to a context
                                          parameter which should be replaced by the
                                          variable. Rules are addressed by their title,
                                          engines by the engine name. Example: "rules.
                                          <ruleTitle>.then.similarityEngines.
                                          <simEngineName>.global.contribution".
    }
  ]
}

or

{
  "type": "queryParameter",             required
  "parameterName": string,              required if "type"=="queryParameter", specifies the
                                          value of which query parameter of the
                                          recommendation request shall be used
  "default": string,                    optional, from R5.4, specifies a default value for
                                          the variable, to be used if the variable cannot
```

```
                                        be resolved
  "patches": [                        optional, from R5.1, defines where the variable
                                        will be used in the context
    {
      "operation": <enum>,            optional, allowed value = default value "replace"
      "path": <string>                required, pointer in dot notation to a context
                                        parameter which should be replaced by the
                                        variable. Rules are addressed by their title,
                                        engines by the engine name. Example: "rules.
                                        <ruleTitle>.then.similarityEngines.
                                        <simEngineName>.global.contribution".
    }
  ]
}

or

{
  "type": "timestamp",                required
  "origin": enum,                     required if "type"=="timestamp", origin of the
                                        relative time specification, allowed values
                                        "currentTime", "startOfDay", "startOfWeek"
  "offset": duration,                 required if "type"=="timestamp", offset with
                                        respect to the "origin" in XSD duration format
  "default": string,                  optional, from R5.4, specifies a default value for
                                        the variable, to be used if the variable cannot
                                        be resolved
  "patches": [                        optional, from R5.1, defines where the variable
                                        will be used in the context
    {
      "operation": <enum>,            optional, allowed value = default value "replace"
      "path": <string>                required, pointer in dot notation to a context
                                        parameter which should be replaced by the
                                        variable. Rules are addressed by their title,
                                        engines by the engine name. Example: "rules.
                                        <ruleTitle>.then.similarityEngines.
                                        <simEngineName>.global.contribution".
    }
  ]
}

or

{
  "type": "subscriberProperty",       required
  "path": <string>,                   required if "type"=="subscriberProperty", path to
                                        the subscriber property containing the variable
                                        value
  "default": string,                  optional, from R5.4, specifies a default value for
                                        the variable, to be used if the variable cannot
                                        be resolved
  "patches": [                        optional, from R5.1, defines where the variable
                                        will be used in the context
    {
      "operation": <enum>,            optional, allowed value = default value "replace"
      "path": <string>                required, pointer in dot notation to a context
                                        parameter which should be replaced by the
                                        variable. Rules are addressed by their title,
                                        engines by the engine name. Example: "rules.
```

```
                                    <ruleTitle>.then.similarityEngines.
                                    <simEngineName>.global.contribution".
    }
  ]
}

or

{
  "type": "seedProperty",           required, intended for similarity engines to limit
                                      the recommendations to a property (e.g. genre) of
                                      the seed asset.
  "seed": string,                   required, refers to the query parameter specifying
                                      the seed asset Id. Note that the context may use
                                      multiple similarity engines with multiple seeds,
                                      hence this has to be specified explicitly
  "path": string,                   required, path to the metadata attribute of the
                                      seed asset, e.g. "persons.actor.0.canonical".
                                      Note that in contrast to standard dot notation
                                      individual elements in a list are addressable via
                                      their index here.
  "default": string,                optional, a default value of the variable, to be
                                      used if the seed property cannot be resolved
  "patches": [                      optional, from R5.1, defines where the variable
                                      will be used in the context
    {
      "operation": <enum>,          optional, allowed value = default value "replace"
      "path": <string>              required, pointer in dot notation to a context
                                      parameter which should be replaced by the
                                      variable. Rules are addressed by their title,
                                      engines by the engine name. Example: "rules.
                                      <ruleTitle>.then.similarityEngines.
                                      <simEngineName>.global.contribution".
    }
  ]
}


@rule =
{"title": string,                   optional
 "if": @predicate,                  optional, @predicate object
 "then": @properties,               required, @properties object
 "else": @properties                optional, @properties object
}

@predicate =
{"operator": enum,                  required, one of "and", "or"
 "operands" : [@predicate]          required, any number of @predicate
}

or

@predicate =
{"variable": enum,                  required, see table below
 "operator": enum,                  required, see table below
 "value": string                    required, see table below
}
```

| variable | allowed operators | value definition |
|---|---|---|

| | | |
|---|---|---|
| timeOfDay | atLeast, atMost | time of day in XSD time format (see Section 3.5) |
| subscriberGroups | contains | string |
| dateTime | atLeast, atMost | date / time in XSD datetime format (see Section 3.5) |
| From R4.4 also: <queryParamName> | equals, notEquals | The value of the corresponding query parameter of the recommendation request. |
| from R5.6 also: subscriber.<path> | equals, notEquals | the value of the attribute of the subscriber, addressed in dot notation. E.g. subscriber.customProperties.property1 or subscriber.subscriberGroups |

```
@properties =
{"similarityEngines": [@simEngProps,...],                        optional
 "preferenceEngines": [@prefEngProps,...],                       optional
 "serendipityPreferenceEngines": [@prefSerendipityEngProps],     optional
 "sharedProfilePreferenceEngines": [@sharedProfilePrefEngProps], optional
 "socialEngines": [@socEngProps,...],                            optional
 "postingEngines": [@postingEngProps,...],                       optional
 "collaborativeEngines": [@collEngProps,...],                    optional
 "statisticalEngines": [@statEngProps,...],                      optional
 "moderatedEngines": [@modEngProps,...],                         optional
 "searchEngines": [@searchEngProps,...],                         optional
 "blendingPreferenceEngines":[@blndPrefEngProps]                 optional
 "blendingFilterPreferenceEngines": [@blndFiltPrefEngProps]      optional
 "collaborativeSimilarityEngines": [@collSimEngProps],           optional
 "customScoreEngines": [@customScoreEngProps],                   optional, from R5.6
 "adHocPreferenceEngines": [@adHocPrefEngProps],                 optional, from R5.6
 "adTargeting": @adTargetingProps,                               optional
 "global": @globalProps                                          optional
}

@simEngProps =
{"assetTypes": [string],              optional, defines the asset types to be used by the
                                        engine. If there is an assetTypes definition in
                                        the global context, it is overruled by this
                                        definition for the given engine. Note that for
                                        compatibility to earlier releases the asset type
                                        "ad" is reserved for advertising assets and is
                                        never used by recommendation engines. If
                                        "assetTypes" is not specified, all asset types
                                        with the exception of "ad" are used.
 "name": string,                      required, unique name of the engine
 "inputs": [string,...],              optional, list of sourceGroupIds used as input,
                                        empty list or omitted entitlement means all
                                        sourceGroups will be used
 "assetStores": [<storeid>],          optional, if defined, all engines for which no
                                        separate assetStores configuration is provided
                                        will use the given asset store(s) instead of the
                                        active asset store(s). If the list is empty, no
                                        asset store will be used (this is useful e.g. to
                                        provide recommendations from playlists only).
 "playlists": [<string>],             optional, list of playlist IDs. The assets on the
                                        given playlists will be used as recommendation
                                        candidates (exclusively if assetStores=[], in
                                        addition to the assets in the active/given
                                        assetStores if assetStores is missing or non-
                                        empty). Overwrites global assetStores/playlists
                                        configuration for the specific engine.
 "subscriberPlaylists": <boolean>,    optional, if true, all assets in all playlists of
```

```
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
"bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"contribution": decimal,            optional, range[0, +inf[
"minScore: decimal,                 optional, range [0, 1]
"minBusinessScore: decimal,         optional, range [0, +inf], supported from R5.1
"minAge": int,                      optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,        optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"weightFunctions":[@weightFunction,...], optional
"seed": string,                     optional, <reference> from the recommendation
                                    request URL identifying the seed asset to be used
                                    The default value of "<engineName>.seed" is used
                                    if the seed reference is not specified.
"seeds": string,                    optional, supported from R5.0, <reference> query
                                    parameter of the recommendation request
                                    identifying the list of seed assets to be used by
                                    the similarity engine. The default value
                                    <engineName>.seeds is usedif the seed reference
                                    is not specified. If seed and seeds are both
                                    specified, all the asset IDs are used.
"excludeSeed": enum,                optional, one of "none", "asset", "program",
                                    "series", default: "asset", defines if only the
                                    seed asset(s), all assets with the same program
                                    IDs as the seed(s), or all assets with the same
                                    series IDs as the seed(s) shall be excluded from
                                    the recommendations
"personalization": <enum>,          optional, from R4.5, one of "none" or "subscriber",
                                    if set to "subscriber", the results are scored in
                                    all profiles of the subscriber. The
                                    subscriber ID has to be present in the
                                    recommendation request in that case. Default
                                    value "none".
"personalizationWeight": decimal,   optional, from R4.5 relative contribution of the
                                    preference score wrt the similarity score, range
                                    [0:1], default value 1.0.
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
```

| | |
|---|---|
| | "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different behaviors for already followed series and for new discoveries can be configured. |
| "windowOfAvailabilityType": enum, | optional, from R5.0, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Sets the window of availability type or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityPriority": <enum>, | optional, from R5.0, allowed values "relative", "tightest", default value "relative". Sets the window of availability priority or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. Note that engine level window of availability always overrides global window of availability, the priority attribute only decides between absolute and relative windows specified at the same level. |
| "windowOfAvailabilityStart": timestamp, | optional, from R5.0, default –infinity. Sets the window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityEnd": timestamp, | optional, from R5.0, default +infinity. Sets the window of availability end or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |

```
"relativeWindowOfAvailabilityStart":   optional, from R5.0, defines the relative time
                                         window in which recommendations shall be
                                         available. Sets the reletive window of
                                         availability start or overrides the global
                                         setting for the particular engine only. See
                                         @globalProps for a complete description of the
                                         attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                         "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, from R5.0, defines the relative time
                                         window in which recommendations shall be
                                         available. Sets the reletive window of
                                         availability start or overrides the global
                                         setting for the particular engine only. See
                                         @globalProps for a complete description of the
                                         attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                         "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, from R5.0, overrules the generic window
                                         of availability definition for selected source
                                         IDs for the particular engine only.
      <sourceId>: {                      the sourceId the special window is applicable to
      "start": <timestamp>,              optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
        "offset": <duration>             required, offset in XSD duration format
        }
      "relativeEnd": {                   optional, relative end of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
        "offset": <duration>             required, offset in XSD duration format
        }
      },
      ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                         generic window of availability definition for
                                         selected source group IDs.
      <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                         to (sourceGroupId and sourceGroupIds are both
                                         used)
      "start": timestamp,                optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": enum,                  required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": duration               required, offset in XSD duration format
        }
      "relativeEnd": {                   optional, relative end of the window
        "origin": enum,                  required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": duration               required, offset in XSD duration format
        }
```

```
      },
      ...
 },
 "availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                         "availabilityStartEndTime", default value
                                         ["availabilityStartEndTime"]. Defines which asset
                                         attribute pairs determine the asset's
                                         availability. By default only
                                         availabilityStartTime / availabilityEndTime are
                                         used. This can be replaced or extended by
                                         originalStartTime / originalEndTime. Overwrites
                                         the global setting for one engine only. Supported
                                         from R5.1.
 "frequencyCapping": {                 optional, from R5.4, limits how often Ncanto is
                                         allowed to recommend the same asset to a
                                         subscriber in a time period. Overrides for the
                                         particular engine the global frequencyCapping
                                         configuration.
  "timeWindow": <duration>,            required, the observation period
  "cap": <integer>                     required, the maximum number of occurrences of the
                                         same assetId in recommendations of the same
                                         subscriber.
 },
 "customExplanations": {               optional, from R5.0, custom explanations applied to
                                         all recommendations coming from this engine
  <type>: {                           mandatory, explanation type (string)
   <langcode>: <explanationText>       mandatory, language / explanation text pairs. The
                                         explanationText may include references to context
                                         variables in the syntax
                                         ${<variableName!"<defaultValue>"}.
                                         Variable names which cannot be resolved are
                                         replaced by the <defaultValue> if one is
                                         specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {           optional, from R5.0, if present, Ncanto will
                                         automatically fine tune the contribution of the
                                         engine between contribution and contribution *
                                         minFactor
  "minFactor": decimal                 optional, decimal value in range [0.0:1.0[
 }
}
```

**@prefEngProps = @prefSerendipityEngProps =**

```
{"assetTypes": [string],               optional, defines the asset types to be used by the
                                         engine. If there is an assetTypes definition in
                                         the global context, it is overruled by this
                                         definition for the given engine. Note that for
                                         compatibility to earlier releases the asset type
                                         "ad" is reserved for advertising assets and is
                                         never used by recommendation engines. If
                                         "assetTypes" is not specified, all asset types
                                         with the exception of "ad" are used.
 "name": string,                       required, unique name of the engine
 "inputs": [string,...],               optional, list of sourceGroupIds used as input,
                                         empty list or omitted inputs mean all
                                         sourceGroups will be used
 "assetStores": [<storeid>],           optional, if defined, all engines for which no
                                         separate assetStores configuration is provided
```

```
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
                                    asset store will be used (this is useful e.g. to
                                    provide recommendations from playlists only).
"playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                    given playlists will be used as recommendation
                                    candidates (exclusively if assetStores=[], in
                                    addition to the assets in the active/given
                                    assetStores if assetStores is missing or non-
                                    empty). Overwrites global assetStores/playlists
                                    configuration for the specific engine.
"subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
"bookmarkListFilter": [<string>],  optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"contribution": decimal,           optional, range [0, +inf[
"minScore: dcimal,              optional, range [0, 1]. Preference engines ignore
                                    the minimum score requirement until the profile
                                    becomes active.
"minBusinessScore: decimal,        optional, range [0, 1], supported from R5.1
"minAge": int,                     optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,       optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"profile": string,                 optional, <reference> from the recommendation
                                    request URL corresponding to one profile ID
                                    The default value of "<engineName>.profile" is
                                    used if the profile reference is not specified.
"profileName": string,             optional, <reference> from the recommendation
                                    request URL corresponding to one profileName,
                                    which, in connection with the subscriberId
                                    uniquely identifies a profile. If the query
                                    parameter <engineName>.profileName is used as
                                    <reference>, the specification in the context is
                                    not necessary. Note that if "profile" is
                                    specified in the context or the
                                    <engineName>.profile query parameter is present
                                    in the recommendation request, profileName and
                                    <engineName>.profileName are ignored.
"weightFunctions":[@weightFunction,...],     optional
```

```
"profileBlacklisting": boolean,        optional, default = false
"recordingSuggestion": boolean         optional, default = false, if true, recommendation
                                         requests will return a flag indicating if the
                                         recommendation is suggested to be automatically
                                         scheduled by recording devices.
"skipProfileDislikeThreshold": int,    optional, if present, preference engines having no
                                         positively rated asset but at least the given
                                         number of negatively rated assets in the profile
                                         will be skipped by setting their contribution to
                                         0, unless all preference engines would be
                                         skipped, in which case the attribute is ignored.
"excludeOwnRatedItems": enum,          optional, R4.4, one of "none", "asset", "program",
                                         "series", excludes assets/programs/series which
                                         have already been rated in the profile from
                                         recommendations. Default "none". If set to
                                         program, program and asset IDs are excluded. If
                                         set to series, series, program, and asset IDs are
                                         excluded.
"contributionAdjustment": {            optional, from R4.6, allows to reduce the
                                         contribution of the preference engine if the
                                         strength of the profile is low.
  "zeroContributionProfileStrengthLimit": deprecated from R5.0, renamed to
                                         minContributionProfileStrenghLimit
  "minContributionProfileStrengthLimit": decimal, required, range [0:1]. If the
                                           profile strength is below this limit, the
                                           contribution of the preference engine is set to
                                           contribution * minFactor. Negative values are
                                           deprecated.
  "fullContributionProfileStrengthLimit": decimal, required, range ]
                                         minContributionProfileStrengthLimit, 1]. If the
                                         profile strength is above that limit, the
                                         contribution is not adjusted. If the profile
                                         strength is between the two limit, the
                                         contribution is linearly adjusted between zero
                                         and the full contribution.
  "minFactor": decimal                 optional, from R5.0, decimal value in range
                                         [0.0:1.0[, default value 0.0. Defines the minimum
                                         contribution of the engine as minContribution =
                                         contribution * minFactor.
},
"randomizationFactor": decimal>=1.0,   optional, if provided, the engine does not return
                                         the requested number of assets with highest
                                         scores, but a random subset of the requested
                                         number * randomizationFactor highest score
                                         assets. Supported from R4.7.
"groupingPriority": enum,              optional, only episode & episodeForced supported
                                         before R5.6, allowed values
                                         "businessScore", "firstUnwatchedEpisode",
                                         "episode" (deprecated, same as
                                         firstUnwatchedEpisode),
                                         "firstUnwatchedEpisodeForced", "episodeForced"
                                         (deprecated, same as
                                         firstUnwatchedEpisodeForced), "firstEpisode",
                                        "lastEpisode", "latestArrival", "earliestArrival",
                                         default "businessScore". Determines which
                                         episodes are first selected to populate each
                                         series group. By default the assets are selected
                                         randomly if the business scores of the episodes
                                         are identical (groupingPriority=businessSCore).
```

| | |
|---|---|
| | If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different behaviors for already followed series and for new discoveries can be configured. |
| "windowOfAvailabilityType": enum, | optional, from R5.0, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Sets the window of availability type or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityPriority": <enum>, | optional, from R5.0, allowed values "relative", "tightest", default value "relative". Sets the window of availability priority or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. Note that engine level window of availability always overrides global window of availability, the priority attribute only decides between absolute and relative windows specified at the same level. |
| "windowOfAvailabilityStart": timestamp, | optional, from R5.0, default –infinity. Sets the window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityEnd": timestamp, | optional, from R5.0, default +infinity. Sets the window of availability end or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "relativeWindowOfAvailabilityStart": | optional, from R5.0, defines the relative time window in which recommendations shall be available. Sets the reletive window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| {"origin": <enum>, | required, one of "currentTime", "startOfDay", "startOfWeek" (R5.0) |
| "offset": <duration> | required, offset in XSD duration format |
| }, | |

```
"relativeWindowOfAvailabilityEnd":     optional, from R5.0, defines the relative time
                                         window in which recommendations shall be
                                         available. Sets the reletive window of
                                         availability start or overrides the global
                                         setting for the particular engine only. See
                                         @globalProps for a complete description of the
                                         attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                         "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, from R5.0, overrules the generic window
                                         of availability definition for selected source
                                         IDs for the particular engine only.
     <sourceId>: {                       the sourceId the special window is applicable to
     "start": <timestamp>,               optional, absolute start of the window
     "end": <timestamp>,                 optional, absolute end of the window
     "relativeStart": {                  optional, relative start of the window
       "origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
       "offset": <duration>              required, offset in XSD duration format
       }
     "relativeEnd": {                    optional, relative end of the window
       "origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
       "offset": <duration>              required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                         generic window of availability definition for
                                         selected source group IDs.
     <sourceGroupId>: {                the sourceGroupId the special window is applicable
                                         to (sourceGroupId and sourceGroupIds are both
                                         used)
     "start": timestamp,                 optional, absolute start of the window
     "end": <timestamp>,                 optional, absolute end of the window
     "relativeStart": {                  optional, relative start of the window
       "origin": enum,                   required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
       "offset": duration                required, offset in XSD duration format
       }
     "relativeEnd": {                    optional, relative end of the window
       "origin": enum,                   required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
       "offset": duration                required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],       optional, allowed values: "originalStartEndTime" or
                                         "availabilityStartEndTime", default value
                                         ["availabilityStartEndTime"]. Defines which asset
                                         attribute pairs determine the asset's
                                         availability. By default only
                                         availabilityStartTime / availabilityEndTime are
                                         used. This can be replaced or extended by
                                         originalStartTime / originalEndTime. Overwrites
```

```
                                    the global setting for one engine only. Supported
                                    from R5.1.
"frequencyCapping": {               optional, from R5.4, limits how often Ncanto is
                                    allowed to recommend the same asset to a
                                    subscriber in a time period. Overrides for the
                                    particular engine the global frequencyCapping
                                    configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                    same assetId in recommendations of the same
                                    subscriber.
 },
 "customExplanations": {            optional, from R5.0, custom explanations applied to
                                    all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                    explanationText may include references to context
                                    variables in the syntax
                                    ${<variableName!"<defaultValue>"}.
                                    Variable names which cannot be resolved are
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 }
}
```

**@sharedProfilePrefEngProps =**

```
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                    engine. If there is an assetTypes definition in
                                    the global context, it is overruled by this
                                    definition for the given engine. Note that for
                                    compatibility to earlier releases the asset type
                                    "ad" is reserved for advertising assets and is
                                    never used by recommendation engines. If
                                    "assetTypes" is not specified, all asset types
                                    with the exception of "ad" are used.
 "name": string,                    required, unique name of the engine
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                    empty list or omitted inputs mean all
                                    sourceGroups will be used
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
                                    asset store will be used (this is useful e.g. to
                                    provide recommendations from playlists only).
 "playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                    given playlists will be used as recommendation
                                    candidates (exclusively if assetStores=[], in
                                    addition to the assets in the active/given
                                    assetStores if assetStores is missing or non-
                                    empty). Overwrites global assetStores/playlists
                                    configuration for the specific engine.
 "subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
```

```
                                        Supported from R5.6.
"bookmarkListFilter": [<string>],     optional, list of bookmark list IDs. The assets in
                                        The given bookmark lists will be exclusively used
                                        as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                        the subscriber will be used as recommendation
                                        candidates. After resolving into a list of
                                        bookmark list IDs this will be merged with the
                                        directly specified IDs in "bookmarkListFilter".
                                        Supported from R5.6.
"contribution": decimal,              optional, range [0, +inf[
"minScore: dcimal,              optional, range [0, 1] . Preference engines ignore
                                        the minimum score requirement until the profile
                                        becomes active.
"minBusinessScore: decimal,           optional, range [0, 1], supported from R5.1
"minAge": int,                        optional, if defined, the engine specific minimum
                                        age will overrule the global minimum age. If
                                        "country" is specified in the global section of
                                        the context, the corresponding age rating will be
                                        used. If "country" is not specified, the highest
                                        minAge of all countries available will be used.
"filter": @genericFilterVar,          optional, allows to filter recommendations to any
                                        asset attribute value. If a filter is present in
                                        the global section of the context too, the two
                                        filters will be intersected. Supports variables
                                        from R5.0.
"profile": string,                    optional, <reference> from the recommendation
                                        request URL corresponding to one profile ID
                                        The default value of "<engineName>.profile" is
                                        used if the profile reference is not specified.
"profileName": string,                optional, <reference> from the recommendation
                                        request URL corresponding to one profileName,
                                        which, in connection with the subscriberId
                                        uniquely identifies a profile. If the query
                                        parameter <engineName>.profileName is used as
                                        <reference>, the specification in the context is
                                        not necessary. Note that if "profile" is
                                        specified in the context or the
                                        <engineName>.profile query parameter is present
                                        in the recommendation request, profileName and
                                        <engineName>.profileName are ignored.
                                        If the recommendation request contains a
                                        "sharedSubscriberIds" query parameter, profiles
                                        matching profileName of the corresponding
                                        subscribers are used to blend recommendations
                                        from, as if their IDs were specified in
                                        sharedProfileIds.
"sharedProfileIds": string,           optional, <reference> from the recommendation
                                        request URL corresponding to a list of profile
                                        IDs, e.g. sharedProfiles=["profId1","profId2"].
                                        "<engineName>.sharedProfileIds" is used as
                                        default if the profile reference is not
                                        specified.
"sharedProfilesWeight": <decimal>,    required, range: [0:1], this defines the weight of
                                        all shared profiles together within the
                                        contribution of the engine (i.e. if
                                        sharedProfilesWeight=0.3, 30% of recommendations
                                        will come from the shared profiles, 70% from the
                                        main profile used)
```

```
"weightFunctions":[@weightFunction,...],     optional
"profileBlacklisting": boolean,        optional, default = false
"recordingSuggestion": boolean         optional, default = false, if true, recommendation
                                         requests will return a flag indicating if the
                                         recommendation is suggested to be automatically
                                         scheduled by recording devices.
"skipProfileDislikeThreshold": int,    optional, if present, preference engines having no
                                         positively rated asset but at least the given
                                        number of negatively rated assets in the profile
                                         will be skipped by setting their contribution to
                                         0, unless all preference engines would be
                                         skipped, in which case the attribute is ignored.
"contributionAdjustment": {            optional, from R4.6, allows to reduce the
                                         contribution of the preference engine if the
                                         strength of the profile is low. The adjustment is
                                         done individually for each main/shared profile.
  "zeroContributionProfileStrengthLimit": deprecated from R5.0, renamed to
                                         minContributionProfileStrenghLimit
  "minContributionProfileStrengthLimit": decimal, required, range [0:1]. If the
                                           profile strength is below this limit, the
                                           contribution of the preference engine is set to
                                           contribution * minFactor. Negative values are
                                           deprecated.
  "fullContributionProfileStrengthLimit": decimal, required, range ]
                                         minContributionProfileStrengthLimit, 1]. If the
                                         profile strength is above that limit, the
                                         contribution is not adjusted. If the profile
                                         strength is between the two limit, the
                                         contribution is linearly adjusted between zero
                                         and the full contribution.
  "minFactor": decimal                 optional, from R5.0, decimal value in range
                                         [0.0:1.0[, default value 0.0. Defines the minimum
                                         contribution of the engine as minContribution =
                                         contribution * minFactor.
},
"randomizationFactor": decimal>=1.0,   optional, if provided, the engine does not return
                                         the requested number of assets with highest
                                         scores, but a random subset of the requested
                                         number * randomizationFactor highest score
                                         assets. Supported from R4.7.
"groupingPriority": enum,              optional, only episode & episodeForced supported
                                         before R5.6, allowed values
                                         "businessScore", "firstUnwatchedEpisode",
                                         "episode" (deprecated, same as
                                         firstUnwatchedEpisode),
                                         "firstUnwatchedEpisodeForced", "episodeForced"
                                         (deprecated, same as
                                         firstUnwatchedEpisodeForced), "firstEpisode",
                                        "lastEpisode", "latestArrival", "earliestArrival",
                                         default "businessScore". Determines which
                                         episodes are first selected to populate each
                                         series group. By default the assets are selected
                                         randomly if the business scores of the episodes
                                         are identical (groupingPriority=businessSCore).
                                         If firstUnwatchedEpisode priority is selected,
                                         the first not yet watched episode of each series
                                         are used. If firstUnwatchedEpisodeForced is
                                         specified, the first not yet watched episode is
                                         selected even if there are other episodes with
```

```
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,            optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,   optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
                                    availability, the priority attribute only decides
                                    between absolute and relative windows specified
                                    at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                    window of availability start or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"windowOfAvailabilityEnd": timestamp, optional, from R5.0, default +infinity. Sets the
                                    window of availability end or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"relativeWindowOfAvailabilityStart":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
```

```
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {  optional, from R5.0, overrules the generic window
                                    of availability definition for selected source
                                    IDs for the particular engine only.
     <sourceId>: {                  the sourceId the special window is applicable to
     "start": <timestamp>,          optional, absolute start of the window
     "end": <timestamp>,            optional, absolute end of the window
     "relativeStart": {             optional, relative start of the window
       "origin": <enum>,            required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
       "offset": <duration>         required, offset in XSD duration format
       }
     "relativeEnd": {               optional, relative end of the window
       "origin": <enum>,            required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
       "offset": <duration>         required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                    generic window of availability definition for
                                    selected source group IDs.
     <sourceGroupId>: {             the sourceGroupId the special window is applicable
                                    to (sourceGroupId and sourceGroupIds are both
                                    used)
     "start": timestamp,            optional, absolute start of the window
     "end": <timestamp>,            optional, absolute end of the window
     "relativeStart": {             optional, relative start of the window
       "origin": enum,              required, one of "currentTime", "startOfDay",
                                    "startOfWeek"
       "offset": duration           required, offset in XSD duration format
       }
     "relativeEnd": {               optional, relative end of the window
       "origin": enum,              required, one of "currentTime", "startOfDay",
                                    "startOfWeek"
       "offset": duration           required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],    optional, allowed values: "originalStartEndTime" or
                                    "availabilityStartEndTime", default value
                                    ["availabilityStartEndTime"]. Defines which asset
                                    attribute pairs determine the asset's
                                    availability. By default only
                                    availabilityStartTime / availabilityEndTime are
                                    used. This can be replaced or extended by
                                    originalStartTime / originalEndTime. Overwrites
                                    the global setting for one engine only. Supported
                                    from R5.1.
"frequencyCapping": {               optional, from R5.4, limits how often Ncanto is
                                    allowed to recommend the same asset to a
                                    subscriber in a time period. Overrides for the
```

```
                                    particular engine the global frequencyCapping
                                    configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                    same assetId in recommendations of the same
                                    subscriber.
 },
 "customExplanations": {            optional, from R5.0, custom explanations applied to
                                    all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                    explanationText may include references to context
                                    variables in the syntax
                                    ${<variableName!"<defaultValue>"}.
                                    Variable names which cannot be resolved are
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 }
}

@socEngProps =
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                    engine. If there is an assetTypes definition in
                                    the global context, it is overruled by this
                                    definition for the given engine. Note that for
                                    compatibility to earlier releases the asset type
                                    "ad" is reserved for advertising assets and is
                                    never used by recommendation engines. If
                                    "assetTypes" is not specified, all asset types
                                    with the exception of "ad" are used.
 "name": string,                    required, unique name of the engine
 "owners": [string,...],            optional, list of topic owners the social engine
                                    shall use to provide recommendations. If both
                                    "owners" and "topics" are defined, all topics
                                    matching either criteria are used
 "topics": [string,...],            optional, list of topics the social engine
                                    shall use to provide recommendations. If both
                                    "owners" and "topics" are defined, all topics
                                    matching either criteria are used
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                    empty list or omitted inputs mean all
                                    sourceGroups will be used.
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
                                    asset store will be used (this is useful e.g. to
                                    provide recommendations from playlists only).
 "playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                    given playlists will be used as recommendation
                                    candidates (exclusively if assetStores=[], in
                                    addition to the assets in the active/given
                                    assetStores if assetStores is missing or non-
                                    empty). Overwrites global assetStores/playlists
                                    configuration for the specific engine.
 "subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
```

```
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
"bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"contribution": decimal,            optional, range [0,+inf[
"minScore: deimal,              optional, range [0, 1]
"minBusinessScore: decimal,         optional, range [0, 1], supported from R5.1
"minAge": int,                      optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,        optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"weightFunctions":[@weightFunction,...],    optional
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
                                    firstUnwatchedEpisode),
                                    "firstUnwatchedEpisodeForced", "episodeForced"
                                    (deprecated, same as
                                    firstUnwatchedEpisodeForced), "firstEpisode",
                                   "lastEpisode", "latestArrival", "earliestArrival",
                                    default "businessScore". Determines which
                                    episodes are first selected to populate each
                                    series group. By default the assets are selected
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,            optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
```

```
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,   optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
                                    availability, the priority attribute only decides
                                    between absolute and relative windows specified
                                    at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                    window of availability start or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"windowOfAvailabilityEnd": timestamp, optional, from R5.0, default +infinity. Sets the
                                    window of availability end or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"relativeWindowOfAvailabilityStart":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {  optional, from R5.0, overrules the generic window
                                    of availability definition for selected source
                                    IDs for the particular engine only.
     <sourceId>: {                  the sourceId the special window is applicable to
     "start": <timestamp>,          optional, absolute start of the window
     "end": <timestamp>,            optional, absolute end of the window
```

```
      "relativeStart": {                optional, relative start of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
        "offset": <duration>            required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
        "offset": <duration>            required, offset in XSD duration format
        }
      },
      ...
 },
 "windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                          generic window of availability definition for
                                          selected source group IDs.
      <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                          to (sourceGroupId and sourceGroupIds are both
                                          used)
      "start": timestamp,               optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
 },
 "availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                          "availabilityStartEndTime", default value
                                          ["availabilityStartEndTime"]. Defines which asset
                                          attribute pairs determine the asset's
                                          availability. By default only
                                          availabilityStartTime / availabilityEndTime are
                                          used. This can be replaced or extended by
                                          originalStartTime / originalEndTime. Overwrites
                                          the global setting for one engine only. Supported
                                          from R5.1.
"frequencyCapping": {                  optional, from R5.4, limits how often Ncanto is
                                          allowed to recommend the same asset to a
                                          subscriber in a time period. Overrides for the
                                          particular engine the global frequencyCapping
                                          configuration.
  "timeWindow": <duration>,            required, the observation period
  "cap": <integer>                     required, the maximum number of occurrences of the
                                          same assetId in recommendations of the same
                                          subscriber.
 },
 "customExplanations": {               optional, from R5.0, custom explanations applied to
                                          all recommendations coming from this engine
  <type>: {                            mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>       mandatory, language / explanation text pairs. The
                                          explanationText may include references to context
```

```
                                    variables in the syntax
                                    ${<variableName!"<defaultValue>"}.
                                    Variable names which cannot be resolved are
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {        optional, from R5.0, if present, Ncanto will
                                      automatically fine tune the contribution of the
                                      engine between contribution and contribution *
                                      minFactor
  "minFactor": decimal              optional, decimal value in range [0.0:1.0[
 }
}
```

**@postingEngProps =**

```
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                      engine. If there is an assetTypes definition in
                                      the global context, it is overruled by this
                                      definition for the given engine. Note that for
                                      compatibility to earlier releases the asset type
                                      "ad" is reserved for advertising assets and is
                                      never used by recommendation engines. If
                                      "assetTypes" is not specified, all asset types
                                      with the exception of "ad" are used
 "name": string,                    required, unique name of the engine
 "postingType": string,             required, supported values: "twitter"
 "postings": [string,...],          required, list of postings (tweets)
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                      empty list or omitted inputs mean all
                                      sourceGroups will be used.
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                      separate assetStores configuration is provided
                                      will use the given asset store(s) instead of the
                                      active asset store(s). If the list is empty, no
                                      asset store will be used (this is useful e.g. to
                                      provide recommendations from playlists only).
 "playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                      given playlists will be used as recommendation
                                      candidates (exclusively if assetStores=[], in
                                      addition to the assets in the active/given
                                      assetStores if assetStores is missing or non-
                                      empty). Overwrites global assetStores/playlists
                                      configuration for the specific engine.
 "subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
                                      the subscriber will be used as recommendation
                                      candidates too. After resolving into a list of
                                      playlist IDs this will be merged with the
                                      directly specified IDs in "playlists". Overwrites
                                      the global configuration for the specific engine.
                                      Supported from R5.6.
 "bookmarkListFilter": [<string>],  optional, list of bookmark list IDs. The assets in
                                      The given bookmark lists will be exclusively used
                                      as recommendation candidates. From R5.1.
 "subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                      the subscriber will be used as recommendation
                                      candidates. After resolving into a list of
                                      bookmark list IDs this will be merged with the
                                      directly specified IDs in "bookmarkListFilter".
```

```
                                    Supported from R5.6.
"contribution": decimal,           optional, range [0,+inf[
"minScore: deimal,           optional, range [0, 1]
"minBusinessScore: decimal,        optional, range [0, 1], supported from R5.1
"minAge": int,                     optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,       optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"weightFunctions":[@weightFunction,...],     optional
"groupingPriority": enum,          optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
                                    firstUnwatchedEpisode),
                                    "firstUnwatchedEpisodeForced", "episodeForced"
                                    (deprecated, same as
                                    firstUnwatchedEpisodeForced), "firstEpisode",
                                   "lastEpisode", "latestArrival", "earliestArrival",
                                    default "businessScore". Determines which
                                    episodes are first selected to populate each
                                    series group. By default the assets are selected
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,           optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,  optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
```

```
                                        "tightest", default value "relative". Sets the
                                        window of availability priority or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute. Note that engine level window of
                                        availability always overrides global window of
                                        availability, the priority attribute only decides
                                        between absolute and relative windows specified
                                        at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                        window of availability start or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"windowOfAvailabilityEnd": timestamp,  optional, from R5.0, default +infinity. Sets the
                                        window of availability end or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"relativeWindowOfAvailabilityStart":   optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, from R5.0, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
     <sourceId>: {                      the sourceId the special window is applicable to
     "start": <timestamp>,              optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     },
     ...
```

```
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
      <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
                                        used)
      "start": timestamp,               optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
},
"availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                        "availabilityStartEndTime", default value
                                        ["availabilityStartEndTime"]. Defines which asset
                                        attribute pairs determine the asset's
                                        availability. By default only
                                        availabilityStartTime / availabilityEndTime are
                                        used. This can be replaced or extended by
                                        originalStartTime / originalEndTime. Overwrites
                                        the global setting for one engine only. Supported
                                        from R5.1.
"frequencyCapping": {                 optional, from R5.4, limits how often Ncanto is
                                        allowed to recommend the same asset to a
                                        subscriber in a time period. Overrides for the
                                        particular engine the global frequencyCapping
                                        configuration.
 "timeWindow": <duration>,            required, the observation period
 "cap": <integer>                     required, the maximum number of occurrences of the
                                        same assetId in recommendations of the same
                                        subscriber.
},
"customExplanations": {               optional, from R5.0, custom explanations applied to
                                        all recommendations coming from this engine
 <type>: {                            mandatory, explanation type (nonEmptyString)
  <langcode>: <explanationText>       mandatory, language / explanation text pairs. The
                                        explanationText may include references to context
                                        variables in the syntax
                                        ${<variableName!"<defaultValue>"}.
                                        Variable names which cannot be resolved are
                                        replaced by the <defaultValue> if one is
                                        specified, or by "" if no default is given.
 }
},
"contributionAdjustment": {           optional, from R5.0, if present, Ncanto will
                                        automatically fine tune the contribution of the
                                        engine between contribution and contribution *
                                        minFactor
 "minFactor": decimal                 optional, decimal value in range [0.0:1.0[
```

```
 }
}
```

**@collEngProps =**

| | |
|---|---|
| {"assetTypes": [string], | optional, defines the asset types to be used by the engine. If there is an assetTypes definition in the global context, it is overruled by this definition for the given engine. Note that for compatibility to earlier releases the asset type "ad" is reserved for advertising assets and is never used by recommendation engines. If "assetTypes" is not specified, all asset types with the exception of "ad" are used. |
| "name": string, | required, unique name of the engine |
| "inputs": [string,...], | optional, list of sourceGroupIds used as input, empty list or omitted inputs mean all sourceGroups will be used |
| "assetStores": [<storeid>], | optional, if defined, all engines for which no separate assetStores configuration is provided will use the given asset store(s) instead of the active asset store(s). If the list is empty, no asset store will be used (this is useful e.g. to provide recommendations from playlists only). |
| "playlists": [<string>], | optional, list of playlist IDs. The assets on the given playlists will be used as recommendation candidates (exclusively if assetStores=[], in addition to the assets in the active/given assetStores if assetStores is missing or non-empty). Overwrites global assetStores/playlists configuration for the specific engine. |
| "subscriberPlaylists": <boolean>, | optional, if true, all assets in all playlists of the subscriber will be used as recommendation candidates too. After resolving into a list of playlist IDs this will be merged with the directly specified IDs in "playlists". Overwrites the global configuration for the specific engine. Supported from R5.6. |
| "bookmarkListFilter": [<string>], | optional, list of bookmark list IDs. The assets in The given bookmark lists will be exclusively used as recommendation candidates. From R5.1. |
| "subscriberBookmarkListFilter": bool, | optional, only the assets in all bookmark lists of the subscriber will be used as recommendation candidates. After resolving into a list of bookmark list IDs this will be merged with the directly specified IDs in "bookmarkListFilter". Supported from R5.6. |
| "distinction": decimal, | optional, range 0...1, default value 0.5. If set to a low value, the engine will recommend assets which are popular among users similar to the subscriber. If set to a high value, the engine will prefer assets which are more popular among users similar to the subscriber than with other users. Supported from R4.8. |
| "collaborativeLearningLevel": enum | optional, supported from R5.0, one of "asset", "program", "series", default: "asset". This parameter defines if the collaborative engine shall work at individual asset level, at program level (not distinguishing between assets with the same program ID), or at series level (not |

| | |
|---|---|
| | distinguishing between assets with the same seriesId). |
| "minScore: decimal, | optional, range [0, 1] |
| "minBusinessScore: decimal, | optional, range [0, 1], supported from R5.1 |
| "minAge": int, | optional, if defined, the engine specific minimum age will overrule the global minimum age. If "country" is specified in the global section of the context, the corresponding age rating will be used. If "country" is not specified, the highest minAge of all countries available will be used. |
| "filter": @genericFilterVar, | optional, allows to filter recommendations to any asset attribute value. If a filter is present in the global section of the context too, the two filters will be intersected. Supports variables from R5.0. |
| "excludeOwnRatedItems": enum, | optional, one of "none", "asset", from R5.0 also "program" and "series. default="none". If "asset", the collaborative engine will not recommend assets which have been rated by the subscriber itself. If set to program, program and asset IDs are excluded. If set to series, series, program, and asset IDs are excluded. |
| "contribution": decimal, | optional, range [0, +inf[ |
| "weightFunctions":[@weightFunction,...], | optional |
| "groupingPriority": enum, | optional, only episode & episodeForced supported before R5.6, allowed values "businessScore", "firstUnwatchedEpisode", "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different |

```
                                        behaviors for already followed series and for new
                                        discoveries can be configured.
"windowOfAvailabilityType": enum,       optional, from R5.0, one of "overlap",
                                        "strictStart", "strictEnd", "strict", default:
                                        "overlap". Sets the window of availability type
                                        or overrides the global setting for the
                                        particular engine only. See @globalProps for a
                                        complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                        "tightest", default value "relative". Sets the
                                        window of availability priority or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute. Note that engine level window of
                                        availability always overrides global window of
                                        availability, the priority attribute only decides
                                        between absolute and relative windows specified
                                        at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                        window of availability start or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"windowOfAvailabilityEnd": timestamp,   optional, from R5.0, default +infinity. Sets the
                                        window of availability end or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"relativeWindowOfAvailabilityStart":    optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                   required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>                required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":      optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                   required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>                required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {      optional, from R5.0, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
     <sourceId>: {                      the sourceId the special window is applicable to
     "start": <timestamp>,              optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
```

```
        "offset": <duration>            required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
        "offset": <duration>            required, offset in XSD duration format
        }
      },
      ...
 },
 "windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                          generic window of availability definition for
                                          selected source group IDs.
      <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                          to (sourceGroupId and sourceGroupIds are both
                                          used)
      "start": timestamp,               optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
 },
 "availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                          "availabilityStartEndTime", default value
                                          ["availabilityStartEndTime"]. Defines which asset
                                          attribute pairs determine the asset's
                                          availability. By default only
                                          availabilityStartTime / availabilityEndTime are
                                          used. This can be replaced or extended by
                                          originalStartTime / originalEndTime. Overwrites
                                          the global setting for one engine only. Supported
                                          from R5.1.
"frequencyCapping": {                  optional, from R5.4, limits how often Ncanto is
                                          allowed to recommend the same asset to a
                                          subscriber in a time period. Overrides for the
                                          particular engine the global frequencyCapping
                                          configuration.
  "timeWindow": <duration>,            required, the observation period
  "cap": <integer>                     required, the maximum number of occurrences of the
                                          same assetId in recommendations of the same
                                          subscriber.
 },
 "customExplanations": {                optional, from R5.0, custom explanations applied to
                                          all recommendations coming from this engine
  <type>: {                            mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>       mandatory, language / explanation text pairs. The
                                          explanationText may include references to context
                                          variables in the syntax
                                          ${<variableName!"<defaultValue>"}.
                                          Variable names which cannot be resolved are
```

```
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {        optional, from R5.0, if present, Ncanto will
                                    automatically fine tune the contribution of the
                                    engine between contribution and contribution *
                                    minFactor
  "minFactor": decimal              optional, decimal value in range [0.0:1.0[
 }
}

@statEngProps =
{
 "statisticalList": enum,           optional, ID of the statistical list, see
                                    Table 6., deprecated from R5.3. If the list ID is
                                    specified, list type, timeWindow, and
                                    programTypes, and assetFamilies are not allowed.
 "type": enum,                      optional, from R5.3, type of the statistical list,
                                    allowed values: see Table 7. If type is
                                specified,
                                    timeWindow has to be specified, too.
 "timeWindow": xsd_duration         optional, from R5.3, time window of the statistical
                                    list, mandatory if type/timeWindow are used to
                                    specify the statistical list.
 "statisticalListLevel": enum,      optional, one of "asset", "program", or "series",
                                    default "asset", defines if the statistical asset
                                    or program or series lists have to be used.
                                    Supported from R5.4.
 "assetTypes": [string],            optional, defines the asset types to be included in
                                    the statistical list. If there is an assetTypes
                                    definition in the global context, it is overruled
                                    by this definition for the given engine. Note
                                    that for compatibility to earlier releases the
                                    asset type "ad" is reserved for advertising
                                    assets and is never used by recommendation
                                    engines. If "assetTypes" is not specified, all
                                    asset types with the exception of "ad" are used.
 "programTypes": [string],          optional, from R5.3, defines the program types to
                                    be included in the statistical list. Allowed only
                                    if type/timeWindow are used to specify the
                                    statistical list.
 "assetFamilies": [string],         optional, from R5.4, defines the asset families to
                                    be included in the statistical list. Allowed only
                                    if type/timeWindow are used to specify the
                                    statistical list.
 "subscriberGroupFilter": bool,     optional, from R5.6. Ignored if the list type is
                                    "latestArrivals". For other list types only the
                                    interactions of the subscriber groups the
                                    subscriber belongs to will be taken into account.
 "name": string,                    required, unique name of the engine
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                    empty list or omitted inputs mean all
                                    sourceGroups will be used
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
                                    asset store will be used (this is useful e.g. to
```

| | |
|---|---|
| | provide recommendations from playlists only). |
| "playlists": [<string>], | optional, list of playlist IDs. The assets on the given playlists will be used as recommendation candidates (exclusively if assetStores=[], in addition to the assets in the active/given assetStores if assetStores is missing or non-empty). Overwrites global assetStores/playlists configuration for the specific engine. |
| "subscriberPlaylists": <boolean>, | optional, if true, all assets in all playlists of the subscriber will be used as recommendation candidates too. After resolving into a list of playlist IDs this will be merged with the directly specified IDs in "playlists". Overwrites the global configuration for the specific engine. Supported from R5.6. |
| "bookmarkListFilter": [<string>], | optional, list of bookmark list IDs. The assets in The given bookmark lists will be exclusively used as recommendation candidates. From R5.1. |
| "subscriberBookmarkListFilter": bool, | optional, only the assets in all bookmark lists of the subscriber will be used as recommendation candidates. After resolving into a list of bookmark list IDs this will be merged with the directly specified IDs in "bookmarkListFilter". Supported from R5.6. |
| "minScore: decimal, | optional, range [0, 1] |
| "minBusinessScore: dcimal, | optional, range [0, 1], supported from R5.1 |
| "minAge": int, | optional, if defined, the engine specific minimum age will overrule the global minimum age. If "country" is specified in the global section of the context, the corresponding age rating will be used. If "country" is not specified, the highest minAge of all countries available will be used. |
| "filter": @genericFilterVar, | optional, allows to filter recommendations to any asset attribute value. If a filter is present in the global section of the context too, the two filters will be intersected. Supports variables from R5.0 |
| "contribution": decimal, | optional, range [0, +inf[ |
| "weightFunctions":[@weightFunction,...], | optional |
| "personalization": <enum>, | optional, from R4.5, one of "none" or "subscriber", if set to "subscriber", the results are scored in all profiles of the subscriber. The subscriber ID has to be present in the recommendation request in that case. Default value "none". |
| "personalizationWeight": decimal, | optional, from R4.5 relative contribution of the preference score wrt the statistical score, range [0:1], default value 1.0. |
| "groupingPriority": enum, | optional, only episode & episodeForced supported before R5.6, allowed values "businessScore", "firstUnwatchedEpisode", "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each |

```
                                    series group. By default the assets are selected
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,            optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,   optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
                                    availability, the priority attribute only decides
                                    between absolute and relative windows specified
                                    at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                    window of availability start or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"windowOfAvailabilityEnd": timestamp, optional, from R5.0, default +infinity. Sets the
                                    window of availability end or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"relativeWindowOfAvailabilityStart":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
```

```
                                        "startOfWeek" (R5.0)
    "offset": <duration>                required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":      optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                   required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>                required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {      optional, from R5.0, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
     <sourceId>: {                      the sourceId the special window is applicable to
     "start": <timestamp>,              optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
     <sourceGroupId>: {                 the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
                                        used)
     "start": timestamp,                optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
       "offset": duration               required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
       "offset": duration               required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],        optional, allowed values: "originalStartEndTime" or
                                        "availabilityStartEndTime", default value
                                        ["availabilityStartEndTime"]. Defines which asset
                                        attribute pairs determine the asset's
                                        availability. By default only
```

```
                                    availabilityStartTime / availabilityEndTime are
                                    used. This can be replaced or extended by
                                    originalStartTime / originalEndTime. Overwrites
                                    the global setting for one engine only. Supported
                                    from R5.1.
"frequencyCapping": {               optional, from R5.4, limits how often Ncanto is
                                    allowed to recommend the same asset to a
                                    subscriber in a time period. Overrides for the
                                    particular engine the global frequencyCapping
                                    configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                    same assetId in recommendations of the same
                                    subscriber.
 },
 "customExplanations": {            optional, from R5.0, custom explanations applied to
                                    all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                    explanationText may include references to context
                                    variables in the syntax
                                    ${<variableName!"<defaultValue>"}.
                                    Variable names which cannot be resolved are
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {        optional, from R5.0, if present, Ncanto will
                                    automatically fine tune the contribution of the
                                    engine between contribution and contribution *
                                    minFactor
  "minFactor": decimal              optional, decimal value in range [0.0:1.0
 },
 "randomizationFactor": decimal>=1.0,  optional, if provided, the engine does not return
                                    the requested number of assets with highest
                                    scores, but a random subset of the requested
                                    number * randomizationFactor highest score
                                    assets. Supported from R5.0.

}

@modEngProps =
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                    engine. If there is an assetTypes definition in
                                    the global context, it is overruled by this
                                    definition for the given engine. Note that for
                                    compatibility to earlier releases the asset type
                                    "ad" is reserved for advertising assets and is
                                    never used by recommendation engines. If
                                    "assetTypes" is not specified, all asset types
                                    with the exception of "ad" are used.
 "name": string,                    required, unique name of the engine
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                    empty list or omitted inputs mean all
                                    sourceGroups will be used
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
```

```
                                    asset store will be used (this is useful e.g. to
                                    provide recommendations from playlists only).
"playlists": [<string>],            optional, list of playlist IDs. The assets on the
                                    given playlists will be used as recommendation
                                    candidates (exclusively if assetStores=[], in
                                    addition to the assets in the active/given
                                    assetStores if assetStores is missing or non-
                                    empty). Overwrites global assetStores/playlists
                                    configuration for the specific engine.
"subscriberPlaylists": <boolean>,   optional, if true, all assets in all playlists of
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
"bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"minScore": decimal,                optional, range [0, 1]
"minBusinessScore: ecimal,          optional, range [0, 1], supported from R5.1
"minAge": int,                      optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,        optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"contribution": decimal,            optional, range [0, +inf[
"weightFunctions":[@weightFunction,...], optional
"moderatedList": string,            optional, ID of the moderated list
“moderatedLists”: [string],         optional, from R5.0, list of IDs of moderated lists
                                    to be used by the engine. All assets of all lists
                                    are concatenated into one list. If moderatedList
                                    is specified, too, all list IDs in moderatedList
                                    and moderatedLists are used together.
"personalization": <enum>,          optional, from R4.5, one of "none" or "subscriber",
                                    if set to "subscriber", the results are scored in
                                    all profiles of the subscriber. The
                                    subscriber ID has to be present in the
                                    recommendation request in that case. Default
                                    value "none".
"personalizationWeight": decimal,   optional, from R4.5 relative contribution of the
                                    preference score wrt the moderated score, range
                                    [0:1], default value 1.0.
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
```

| | |
|---|---|
| | firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different behaviors for already followed series and for new discoveries can be configured. |
| "windowOfAvailabilityType": enum, | optional, from R5.0, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Sets the window of availability type or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityPriority": <enum>, | optional, from R5.0, allowed values "relative", "tightest", default value "relative". Sets the window of availability priority or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. Note that engine level window of availability always overrides global window of availability, the priority attribute only decides between absolute and relative windows specified at the same level. |
| "windowOfAvailabilityStart": timestamp, | optional, from R5.0, default –infinity. Sets the window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityEnd": timestamp, | optional, from R5.0, default +infinity. Sets the window of availability end or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "relativeWindowOfAvailabilityStart": | optional, from R5.0, defines the relative time |

```
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {  optional, from R5.0, overrules the generic window
                                    of availability definition for selected source
                                    IDs for the particular engine only.
      <sourceId>: {                 the sourceId the special window is applicable to
      "start": <timestamp>,         optional, absolute start of the window
      "end": <timestamp>,           optional, absolute end of the window
      "relativeStart": {            optional, relative start of the window
        "origin": <enum>,           required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
        "offset": <duration>        required, offset in XSD duration format
        }
      "relativeEnd": {              optional, relative end of the window
        "origin": <enum>,           required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
        "offset": <duration>        required, offset in XSD duration format
        }
      },
      ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                    generic window of availability definition for
                                    selected source group IDs.
      <sourceGroupId>: {            the sourceGroupId the special window is applicable
                                    to (sourceGroupId and sourceGroupIds are both
                                    used)
      "start": timestamp,           optional, absolute start of the window
      "end": <timestamp>,           optional, absolute end of the window
      "relativeStart": {            optional, relative start of the window
        "origin": enum,             required, one of "currentTime", "startOfDay",
                                    "startOfWeek"
        "offset": duration          required, offset in XSD duration format
        }
      "relativeEnd": {              optional, relative end of the window
        "origin": enum,             required, one of "currentTime", "startOfDay",
                                    "startOfWeek"
        "offset": duration          required, offset in XSD duration format
        }
      },
```

```
      ...
 },
 "availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                         "availabilityStartEndTime", default value
                                         ["availabilityStartEndTime"]. Defines which asset
                                         attribute pairs determine the asset's
                                         availability. By default only
                                         availabilityStartTime / availabilityEndTime are
                                         used. This can be replaced or extended by
                                         originalStartTime / originalEndTime. Overwrites
                                         the global setting for one engine only. Supported
                                         from R5.1.
"frequencyCapping": {                 optional, from R5.4, limits how often Ncanto is
                                         allowed to recommend the same asset to a
                                         subscriber in a time period. Overrides for the
                                         particular engine the global frequencyCapping
                                         configuration.
  "timeWindow": <duration>,           required, the observation period
  "cap": <integer>                    required, the maximum number of occurrences of the
                                         same assetId in recommendations of the same
                                         subscriber.
 },
 "customExplanations": {              optional, from R5.0, custom explanations applied to
                                         all recommendations coming from this engine
  <type>: {                          mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>      mandatory, language / explanation text pairs. The
                                         explanationText may include references to context
                                         variables in the syntax
                                         ${<variableName!"<defaultValue>"}.
                                         Variable names which cannot be resolved are
                                         replaced by the <defaultValue> if one is
                                         specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {          optional, from R5.0, if present, Ncanto will
                                         automatically fine tune the contribution of the
                                         engine between contribution and contribution *
                                         minFactor
  "minFactor": decimal                optional, decimal value in range [0.0:1.0[
 },
 "randomizationFactor": decimal>=1.0, optional, if provided, the engine does not return
                                         the requested number of assets with highest
                                         scores, but a random subset of the requested
                                         number * randomizationFactor highest score
                                         assets. Supported from R5.0.
}

@searchEngProps =
{"assetTypes": [string],              optional, defines the asset types to be used by the
                                         engine. If there is an assetTypes definition in
                                         the global context, it is overruled by this
                                         definition for the given engine. Note that for
                                         compatibility to earlier releases the asset type
                                         "ad" is reserved for advertising assets and is
                                         never used by recommendation engines. If
                                         "assetTypes" is not specified, all asset types
                                         with the exception of "ad" are used.
 "name": string,                      required, unique name of the engine
 "inputs": [string,...],              optional, list of sourceGroupIds used as input,
```

```
                                    empty list or omitted inputs mean all
                                    sourceGroups will be used
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
                                    asset store will be used (this is useful e.g. to
                                    provide recommendations from playlists only).
 "playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                    given playlists will be used as recommendation
                                    candidates (exclusively if assetStores=[], in
                                    addition to the assets in the active/given
                                    assetStores if assetStores is missing or non-
                                    empty). Overwrites global assetStores/playlists
                                    configuration for the specific engine.
 "subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
 "bookmarkListFilter": [<string>],  optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
 "subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
 "minScore: decimal,                optional, range [0, 1]
 "minBusinessScore: dcimal,         optional, range [0, 1], supported from R5.1
 "relativeMinScore": decimal        optional, range [0, 1], supported from R5.1. If
                                    set, only search results with score >
                                    relativeMinScore * score of the best result will
                                    be returned.
 "minAge": int,                     optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
 "filter": @genericFilterVar,       optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
 "contribution": decimal,           optional, range [0, +inf[
 "weightFunctions":[@weightFunction,...],     optional
 "personalization": enum,           optional, one of "none",
                                    "subscriber", default "none". If "subscriber",
                                    the search engine returns personalized results
                                    (candidate results are determined by the search
                                    query, but the scores are obtained using scoring
                                    in all profiles of the subscriber). The
                                    subscriber ID has to be present in the
                                    recommendation request in that case.
"personalization": <enum> ,         optional, from R4.5, one of "none" or "subscriber",
```

```
                                    if set to "subscriber", the results are scored in
                                    all profiles of the subscriber. The
                                    subscriber ID has to be present in the
                                    recommendation request in that case. Default
                                    value "none".
"personalizationWeight": decimal,   optional, from R4.5 relative contribution of the
                                    preference score wrt the similarity score, range
                                    [0:1], default value 1.0.
"highlighting": bool,               optional, if set to true, the response will contain
                                    asset fragments containing the search term
"highlightingConfig": {             optional, highlighting options
  "fragmentSize": int,              optional, default = 100, maximum size of the text
                                    fragment in characters, the fragment will be
                                    truncated to whole words
  "startTag": string,               optional, default "<em>", tag to indicate the
                                    beginning of the search term in the fragment
  "endTag": string,                 optional, default "</em>", tag to indicate the
                                    end of the search term in the fragment
},
"attributeWeights": {               optional, allows to limit the asset attributes the
                                    search is performed in and the boost factor
                                    (weight) of a hit in the given attribute.
  <attribute>: <decimal>,           at least one pair required, <attribute>: asset
  ...                               attribute in dot notation. Only the attributes
                                    which are used by default for searching are
                                    supported: titles, uiGenres, genreBroad,
                                    genreMedium, genreFine, originalTitle, keywords,
                                    categories, persons, subtitles, synopses,
                                    sourceNames. If the attribute is not a leaf of
                                    the asset object tree, a wildcard has to be used,
                                    e.g. persons.* or persons.*.perLanguage.english.
                                    The weight is a decimal number defining the score
                                    contribution of a search hit in the given
},                                  attribute.
"autocorrection": boolean,          optional, from R5.3, default true, allows to
                                    disable the automatic correction of typos based
                                    on phonetic analysis of the query
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
                                    firstUnwatchedEpisode),
                                    "firstUnwatchedEpisodeForced", "episodeForced"
                                    (deprecated, same as
                                    firstUnwatchedEpisodeForced), "firstEpisode",
                                   "lastEpisode", "latestArrival", "earliestArrival",
                                    default "businessScore". Determines which
                                    episodes are first selected to populate each
                                    series group. By default the assets are selected
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
```

| | |
|---|---|
| | have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different behaviors for already followed series and for new discoveries can be configured. |
| "windowOfAvailabilityType": enum, | optional, from R5.0, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Sets the window of availability type or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityPriority": <enum>, | optional, from R5.0, allowed values "relative", "tightest", default value "relative". Sets the window of availability priority or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. Note that engine level window of availability always overrides global window of availability, the priority attribute only decides between absolute and relative windows specified at the same level. |
| "windowOfAvailabilityStart": timestamp, | optional, from R5.0, default –infinity. Sets the window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityEnd": timestamp, | optional, from R5.0, default +infinity. Sets the window of availability end or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "relativeWindowOfAvailabilityStart": | optional, from R5.0, defines the relative time window in which recommendations shall be available. Sets the reletive window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| {"origin": <enum>, | required, one of "currentTime", "startOfDay", "startOfWeek" (R5.0) |
| "offset": <duration> | required, offset in XSD duration format |
| }, | |
| "relativeWindowOfAvailabilityEnd": | optional, from R5.0, defines the relative time window in which recommendations shall be available. Sets the reletive window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| {"origin": <enum>, | required, one of "currentTime", "startOfDay", |

```
                                        "startOfWeek" (R5.0)
    "offset": <duration>                required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {      optional, from R5.0, overrules the generic window
                                          of availability definition for selected source
                                          IDs for the particular engine only.
     <sourceId>: {                        the sourceId the special window is applicable to
     "start": <timestamp>,                optional, absolute start of the window
     "end": <timestamp>,                  optional, absolute end of the window
     "relativeStart": {                   optional, relative start of the window
       "origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                            "startOfWeek" (R5.0)
       "offset": <duration>               required, offset in XSD duration format
       }
     "relativeEnd": {                     optional, relative end of the window
       "origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                            "startOfWeek" (R5.0)
       "offset": <duration>               required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                          generic window of availability definition for
                                          selected source group IDs.
     <sourceGroupId>: {                 the sourceGroupId the special window is applicable
                                          to (sourceGroupId and sourceGroupIds are both
                                          used)
     "start": timestamp,                  optional, absolute start of the window
     "end": <timestamp>,                  optional, absolute end of the window
     "relativeStart": {                   optional, relative start of the window
       "origin": enum,                    required, one of "currentTime", "startOfDay",
                                            "startOfWeek"
       "offset": duration                 required, offset in XSD duration format
       }
     "relativeEnd": {                     optional, relative end of the window
       "origin": enum,                    required, one of "currentTime", "startOfDay",
                                            "startOfWeek"
       "offset": duration                 required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],        optional, allowed values: "originalStartEndTime" or
                                          "availabilityStartEndTime", default value
                                          ["availabilityStartEndTime"]. Defines which asset
                                          attribute pairs determine the asset's
                                          availability. By default only
                                          availabilityStartTime / availabilityEndTime are
                                          used. This can be replaced or extended by
                                          originalStartTime / originalEndTime. Overwrites
                                          the global setting for one engine only. Supported
                                          from R5.1.
"frequencyCapping": {                   optional, from R5.4, limits how often Ncanto is
                                          allowed to recommend the same asset to a
                                          subscriber in a time period. Overrides for the
                                          particular engine the global frequencyCapping
                                          configuration.
  "timeWindow": <duration>,             required, the observation period
```

```
  "cap": <integer>                    required, the maximum number of occurrences of the
                                        same assetId in recommendations of the same
                                        subscriber.
 },
 "customExplanations": {              optional, from R5.0, custom explanations applied to
                                        all recommendations coming from this engine
  <type>: {                           mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>      mandatory, language / explanation text pairs. The
                                        explanationText may include references to context
                                        variables in the syntax
                                        ${<variableName!"<defaultValue>"}.
                                        Variable names which cannot be resolved are
                                        replaced by the <defaultValue> if one is
                                        specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {          optional, from R5.0, if present, Ncanto will
                                        automatically fine tune the contribution of the
                                        engine between contribution and contribution *
                                        minFactor
  "minFactor": decimal                optional, decimal value in range [0.0:1.0[
 }
}
```

**@blndPrefEngProps =**

```
{"assetTypes": [string],             optional, defines the asset types to be used by the
                                        engine. If there is an assetTypes definition in
                                        the global context, it is overruled by this
                                        definition for the given engine. Note that for
                                        compatibility to earlier releases the asset type
                                        "ad" is reserved for advertising assets and is
                                        never used by recommendation engines. If
                                        "assetTypes" is not specified, all asset types
                                        with the exception of "ad" are used.
 "name": string,                     required, unique name of the engine
 "inputs": [string,...],             optional, list of sourceGroupIds used as input,
                                        empty list or omitted inputs mean all
                                        sourceGroups will be used
 "assetStores": [<storeid>],         optional, if defined, all engines for which no
                                        separate assetStores configuration is provided
                                        will use the given asset store(s) instead of the
                                        active asset store(s). If the list is empty, no
                                        asset store will be used (this is useful e.g. to
                                        provide recommendations from playlists only).
 "playlists": [<string>],            optional, list of playlist IDs. The assets on the
                                        given playlists will be used as recommendation
                                        candidates (exclusively if assetStores=[], in
                                        addition to the assets in the active/given
                                        assetStores if assetStores is missing or non-
                                        empty). Overwrites global assetStores/playlists
                                        configuration for the specific engine.
 "subscriberPlaylists": <boolean>,   optional, if true, all assets in all playlists of
                                        the subscriber will be used as recommendation
                                        candidates too. After resolving into a list of
                                        playlist IDs this will be merged with the
                                        directly specified IDs in "playlists". Overwrites
                                        the global configuration for the specific engine.
                                        Supported from R5.6.
 "bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
```

```
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"contribution": decimal,            optional, range [0, +inf[
"minScore: dcimal,            optional, range [0, 1] . Preference engines ignore
                                    the minimum score requirement until the profile
                                    becomes active.
"minBusinessScore: decimal,         optional, range [0, 1], supported from R5.1
"minAge": int,                      optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,        optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables
                                    from R5.0.
"weightFunctions":[@weightFunction,...],     optional
"profileBlacklisting": boolean,     optional, default = false
"recordingSuggestion": boolean      optional, default = false, if true, recommendation
                                    requests will return a flag indicating if the
                                    recommendation is suggested to be automatically
                                    scheduled by recording devices.
"skipProfileDislikeThreshold": int, optional, from if present, preference engines
                                    having no positively rated asset but at least the
                                    given number of negatively rated assets in the
                                    profile will be skipped by setting their
                                    contribution to 0, unless all preference engines
                                    would be skipped, in which case the attribute is
                                    ignored.
"skipProfileIds": [<string512>],    optional, the specified profile IDs are excluded
                                    and do not contribute to the blending preference
                                    recommendations.
"skipProfileNames":[<string>],      optional, the specified profile names are excluded
                                    and do not contribute to the blending preference
                                    recommendations.
"profileWeights": enum,             optional, allowed values "uniform",
                                    "numberOfRatings", from R4.5 also
                                    "numberOfPositiveRatings",
                                    "logENumberOfPositiveRatings", default "uniform".
                                    Determines if the contributions of the individual
                                    profiles to the list of recommendations are equal
                                    (i.e. contribution of the engine / number of not
                                    skipped profiles) or weighted based on the number
                                    of (all or only positive) ratings in the
                                    individual profiles. The logarithmic weight
                                    algorithm results in a slower than linear
                                    increase of the weight with the number of
                                    ratings.
"excludeOwnRatedItems": enum,       optional, R4.4, one of "none", "asset", "program",
                                    "series", excludes assets/programs/series which
                                    have already been rated in the profile from
```

```
                                        recommendations. Default "none".
"contributionAdjustment": {           optional, from R4.6, allows to reduce the
                                        contribution of the preference engine if the
                                        strength of the profile is low. The adjustment is
                                        done individually for each profile of the
                                        subscriber.
  "zeroContributionProfileStrengthLimit": deprecated from R5.0, renamed to
                                        minContributionProfileStrenghLimit
  "minContributionProfileStrengthLimit": decimal, required, range [0:1]. If the
                                          profile strength is below this limit, the
                                          contribution of the preference engine is set to
                                          contribution * minFactor. Negative values are
                                          deprecated.
  "fullContributionProfileStrengthLimit": decimal, required, range ]
                                        minContributionProfileStrengthLimit, 1]. If the
                                        profile strength is above that limit, the
                                        contribution is not adjusted. If the profile
                                        strength is between the two limit, the
                                        contribution is linearly adjusted between zero
                                        and the full contribution.
  "minFactor": decimal                optional, from R5.0, decimal value in range
                                        [0.0:1.0[, default value 0.0. Defines the minimum
                                        contribution of the engine as minContribution =
                                        contribution * minFactor.
},
"randomizationFactor": decimal>=1.0,  optional, if provided, the engine does not return
                                        the requested number of assets with highest
                                        scores, but a random subset of the requested
                                        number * randomizationFactor highest score
                                        assets. Supported from R4.7.
"groupingPriority": enum,             optional, only episode & episodeForced supported
                                        before R5.6, allowed values
                                        "businessScore", "firstUnwatchedEpisode",
                                        "episode" (deprecated, same as
                                        firstUnwatchedEpisode),
                                        "firstUnwatchedEpisodeForced", "episodeForced"
                                        (deprecated, same as
                                        firstUnwatchedEpisodeForced), "firstEpisode",
                                       "lastEpisode", "latestArrival", "earliestArrival",
                                        default "businessScore". Determines which
                                        episodes are first selected to populate each
                                        series group. By default the assets are selected
                                        randomly if the business scores of the episodes
                                        are identical (groupingPriority=businessSCore).
                                        If firstUnwatchedEpisode priority is selected,
                                        the first not yet watched episode of each series
                                        are used. If firstUnwatchedEpisodeForced is
                                        specified, the first not yet watched episode is
                                        selected even if there are other episodes with
                                        higher business scores. firstEpisode and
                                        lastEpisode use the lowest / highest season /
                                        episode numbers available, ignoring which of them
                                        have already been watched. latestArrival /
                                        earliestArrival use the episodes with earlierst /
                                        latest availability star time to populate the
                                        group. Only effective if type of items =
                                        groups and if seriesId is among the grouping
                                        attributes.
"filterOwnSeries": enum,              optional, allowed values "none"(R5.3), "exclude",
```

```
                                        "includeOnly". This parameter allows to limit the
                                        recommendations to seriesIds which the subscriber
                                        already follows (play events are available) or to
                                        exclude such series IDs from the recommendations.
                                        In connection with grouping priority different
                                        behaviors for already followed series and for new
                                        discoveries can be configured.
"windowOfAvailabilityType": enum,     optional, from R5.0, one of "overlap",
                                        "strictStart", "strictEnd", "strict", default:
                                        "overlap". Sets the window of availability type
                                        or overrides the global setting for the
                                        particular engine only. See @globalProps for a
                                        complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                        "tightest", default value "relative". Sets the
                                        window of availability priority or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute. Note that engine level window of
                                        availability always overrides global window of
                                        availability, the priority attribute only decides
                                        between absolute and relative windows specified
                                        at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                        window of availability start or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"windowOfAvailabilityEnd": timestamp,  optional, from R5.0, default +infinity. Sets the
                                        window of availability end or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"relativeWindowOfAvailabilityStart":   optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>              required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>              required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, from R5.0, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
     <sourceId>: {                      the sourceId the special window is applicable to
```

```
      "start": <timestamp>,              optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
        "offset": <duration>             required, offset in XSD duration format
        }
      "relativeEnd": {                   optional, relative end of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek" (R5.0)
        "offset": <duration>             required, offset in XSD duration format
        }
      },
      ...
 },
 "windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                          generic window of availability definition for
                                          selected source group IDs.
      <sourceGroupId>: {                the sourceGroupId the special window is applicable
                                          to (sourceGroupId and sourceGroupIds are both
                                          used)
      "start": timestamp,                optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": enum,                  required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": duration               required, offset in XSD duration format
        }
      "relativeEnd": {                   optional, relative end of the window
        "origin": enum,                  required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": duration               required, offset in XSD duration format
        }
      },
      ...
 },
 "availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                          "availabilityStartEndTime", default value
                                          ["availabilityStartEndTime"]. Defines which asset
                                          attribute pairs determine the asset's
                                          availability. By default only
                                          availabilityStartTime / availabilityEndTime are
                                          used. This can be replaced or extended by
                                          originalStartTime / originalEndTime. Overwrites
                                          the global setting for one engine only. Supported
                                          from R5.1.
"frequencyCapping": {                  optional, from R5.4, limits how often Ncanto is
                                          allowed to recommend the same asset to a
                                          subscriber in a time period. Overrides for the
                                          particular engine the global frequencyCapping
                                          configuration.
  "timeWindow": <duration>,            required, the observation period
  "cap": <integer>                     required, the maximum number of occurrences of the
                                          same assetId in recommendations of the same
                                          subscriber.
                                        },
 "customExplanations": {               optional, from R5.0, custom explanations applied to
                                          all recommendations coming from this engine
  <type>: {                            mandatory, explanation type (nonEmptyString)
```

```
   <langcode>: <explanationText>        mandatory, language / explanation text pairs. The
                                          explanationText may include references to context
                                          variables in the syntax
                                          ${<variableName!"<defaultValue>"}.
                                          Variable names which cannot be resolved are
                                          replaced by the <defaultValue> if one is
                                          specified, or by "" if no default is given.
  }
 }
}

@blndFiltPrefEngProps =
{
 "name": string,                      required, unique name of the engine

 "profile": string,                   optional, <reference> from the recommendation
                                          request URL corresponding to one profile ID
                                          The default value of "<engineName>.profile" is
                                          used if the profile reference is not specified.
 "profileName": string,               optional, <reference> from the recommendation
                                          request URL corresponding to one profileName,
                                          which, in connection with the subscriberId
                                          uniquely identifies a profile. If the query
                                          parameter <engineName>.profileName is used as
                                          <reference>, the specification in the context is
                                          not necessary. Note that if "profile" is
                                          specified in the context or the
                                          <engineName>.profile query parameter is present
                                          in the recommendation request, profileName and
                                          <engineName>.profileName are ignored.

 "assetTypes": [string],              optional, defines the asset types to be used by the
                                          engine. If there is an assetTypes definition in
                                          the global context, it is overruled by this
                                          definition for the given engine. Note that for
                                          compatibility to earlier releases the asset type
                                          "ad" is reserved for advertising assets and is
                                          never used by recommendation engines. If
                                          "assetTypes" is not specified, all asset types
                                          with the exception of "ad" are used.
 "inputs": [string,...],              optional, list of sourceGroupIds used as input,
                                          empty list or omitted inputs mean all
                                          sourceGroups will be used
 "assetStores": [<storeid>],          optional, if defined, all engines for which no
                                          separate assetStores configuration is provided
                                          will use the given asset store(s) instead of the
                                          active asset store(s). If the list is empty, no
                                          asset store will be used (this is useful e.g. to
                                          provide recommendations from playlists only).
 "playlists": [<string>],             optional, list of playlist IDs. The assets on the
                                          given playlists will be used as recommendation
                                          candidates (exclusively if assetStores=[], in
                                          addition to the assets in the active/given
                                          assetStores if assetStores is missing or non-
                                          empty). Overwrites global assetStores/playlists
                                          configuration for the specific engine.
 "subscriberPlaylists": <boolean>,    optional, if true, all assets in all playlists of
                                          the subscriber will be used as recommendation
                                          candidates too. After resolving into a list of
```

```
                                        playlist IDs this will be merged with the
                                        directly specified IDs in "playlists". Overwrites
                                        the global configuration for the specific engine.
                                        Supported from R5.6.
"bookmarkListFilter": [<string>],     optional, list of bookmark list IDs. The assets in
                                        The given bookmark lists will be exclusively used
                                        as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                        the subscriber will be used as recommendation
                                        candidates. After resolving into a list of
                                        bookmark list IDs this will be merged with the
                                        directly specified IDs in "bookmarkListFilter".
                                        Supported from R5.6.
"contribution": decimal,              optional, range [0, +inf[
"minScore: dcimal,              optional, range [0, 1] . Preference engines ignore
                                        the minimum score requirement until the profile
                                        becomes active.
"minBusinessScore: decimal,           optional, range [0, 1], supported from R5.1
"minAge": int,                        optional, if defined, the engine specific minimum
                                        age will overrule the global minimum age. If
                                        "country" is specified in the global section of
                                        the context, the corresponding age rating will be
                                        used. If "country" is not specified, the highest
                                        minAge of all countries available will be used.
"weightFunctions":[@weightFunction,...],     optional
"profileBlacklisting": boolean,       optional, default = false
"recordingSuggestion": boolean        optional, default = false, if true, recommendation
                                        requests will return a flag indicating if the
                                        recommendation is suggested to be automatically
                                        scheduled by recording devices.
"excludeOwnRatedItems": enum,         optional, one of "none", "asset", "program",
                                        "series", excludes assets/programs/series which
                                        have already been rated in the profile from
                                        recommendations. Default "none".
"contributionAdjustment": {           optional, allows to reduce the
                                        contribution of the preference engine if the
                                        strength of the profile is low. The adjustment is
                                        done individually for each profile of the
                                        subscriber.
  "minContributionProfileStrengthLimit": decimal, required, range [0:1]. If the
                                          profile strength is below this limit, the
                                          contribution of the preference engine is set to
                                          contribution * minFactor. Negative values are
                                          deprecated.
  "fullContributionProfileStrengthLimit": decimal, required, range ]
                                        minContributionProfileStrengthLimit, 1]. If the
                                        profile strength is above that limit, the
                                        contribution is not adjusted. If the profile
                                        strength is between the two limit, the
                                        contribution is linearly adjusted between zero
                                        and the full contribution.
  "minFactor": decimal                optional, decimal value in range
                                        [0.0:1.0[, default value 0.0. Defines the minimum
                                        contribution of the engine as minContribution =
                                        contribution * minFactor.
},
"randomizationFactor": decimal>=1.0,  optional, if provided, the engine does not return
                                        the requested number of assets with highest
                                        scores, but a random subset of the requested
```

| | |
|---|---|
| | number * randomizationFactor highest score assets. Supported from R4.7. |
| "groupingPriority": enum, | optional, only episode & episodeForced supported before R5.6, allowed values "businessScore", "firstUnwatchedEpisode", "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. |
| "filterOwnSeries": enum, | optional, allowed values "none"(R5.3), "exclude", "includeOnly". This parameter allows to limit the recommendations to seriesIds which the subscriber already follows (play events are available) or to exclude such series IDs from the recommendations. In connection with grouping priority different behaviors for already followed series and for new discoveries can be configured. |
| "windowOfAvailabilityType": enum, | optional, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Sets the window of availability type or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |
| "windowOfAvailabilityPriority": <enum>, | optional, allowed values "relative", "tightest", default value "relative". Sets the window of availability priority or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. Note that engine level window of availability always overrides global window of availability, the priority attribute only decides between absolute and relative windows specified at the same level. |
| "windowOfAvailabilityStart": timestamp, | optional, default –infinity. Sets the window of availability start or overrides the global setting for the particular engine only. See @globalProps for a complete description of the attribute. |

```
"windowOfAvailabilityEnd": timestamp,  optional, default +infinity. Sets the
                                         window of availability end or overrides the
                                         global setting for the particular engine only.
                                         See @globalProps for a complete description of
                                         the attribute.
"relativeWindowOfAvailabilityStart":   optional, defines the relative time
                                         window in which recommendations shall be
                                         available. Sets the reletive window of
                                         availability start or overrides the global
                                         setting for the particular engine only. See
                                         @globalProps for a complete description of the
                                         attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                         "startOfWeek"
    "offset": <duration>               required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, defines the relative time
                                         window in which recommendations shall be
                                         available. Sets the reletive window of
                                         availability start or overrides the global
                                         setting for the particular engine only. See
                                         @globalProps for a complete description of the
                                         attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                         "startOfWeek"
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, overrules the generic window
                                         of availability definition for selected source
                                         IDs for the particular engine only.
      <sourceId>: {                      the sourceId the special window is applicable to
      "start": <timestamp>,              optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": <duration>             required, offset in XSD duration format
        }
      "relativeEnd": {                   optional, relative end of the window
        "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": <duration>             required, offset in XSD duration format
        }
      },
      ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                         generic window of availability definition for
                                         selected source group IDs.
      <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                         to (sourceGroupId and sourceGroupIds are both
                                         used)
      "start": timestamp,                optional, absolute start of the window
      "end": <timestamp>,                optional, absolute end of the window
      "relativeStart": {                 optional, relative start of the window
        "origin": enum,                  required, one of "currentTime", "startOfDay",
                                           "startOfWeek"
        "offset": duration               required, offset in XSD duration format
        }
```

```
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
 },
 "availabilityWindows": [<enum>],       optional, allowed values: "originalStartEndTime" or
                                          "availabilityStartEndTime", default value
                                          ["availabilityStartEndTime"]. Defines which asset
                                          attribute pairs determine the asset's
                                          availability. By default only
                                          availabilityStartTime / availabilityEndTime are
                                          used. This can be replaced or extended by
                                          originalStartTime / originalEndTime. Overwrites
                                          the global setting for one engine only.
"frequencyCapping": {                   optional, from R5.4, limits how often Ncanto is
                                          allowed to recommend the same asset to a
                                          subscriber in a time period. Overrides for the
                                          particular engine the global frequencyCapping
                                          configuration.
  "timeWindow": <duration>,             required, the observation period
  "cap": <integer>                      required, the maximum number of occurrences of the
                                          same assetId in recommendations of the same
                                          subscriber.
 },
 "segments": [                          mandatory, list of segment definitions. The
                                          blending filter preference engine serves
                                          recommendations from different segments of one
                                          preference profile. The segments are primarily
                                          defined by a generic filter specifying which
                                          assets belong to the segment.
  {
   "name": <string>,                    mandatory, the name of the segment
   "filter": @genericFilterVar,         mandatory, defines the segment in generic filter
                                          syntax. If a filter is present in
                                          the global section of the context too, the two
                                          filters will be intersected. Supports variables.
   "customExplanations": {              optional, custom explanations applied to
                                          all recommendations coming from this segment
    <type>: {                           mandatory, explanation type (nonEmptyString)
    <langcode>: <explanationText>       mandatory, language / explanation text pairs. The
                                          explanationText may include references to context
                                          variables in the syntax
                                          ${<variableName!"<defaultValue>"}.
                                          Variable names which cannot be resolved are
                                          replaced by the <defaultValue> if one is
                                          specified, or by "" if no default is given.
   }
  }
 ],
 "skipSegmentDislikeThreshold": int,    optional, if present, segments
                                          having no positively rated asset but at least the
                                          given number of negatively rated assets in the
                                          profile will be skipped by setting their
                                          contribution to 0, unless all segments
                                          would be skipped, in which case the attribute is
                                          ignored.
```

```
 "segmentWeights": enum,              optional, allowed values "uniform",
                                        "numberOfRatings", "numberOfPositiveRatings",
                                        "logENumberOfPositiveRatings", default "uniform".
                                        Determines if the contributions of the individual
                                        segments to the list of recommendations are equal
                                        (i.e. contribution of the engine / number of not
                                        skipped segments) or weighted based on the number
                                        of (all or of only positive) ratings matching the
                                        segment. The logarithmic weight
                                        algorithm results in a slower than linear
                                        increase of the weight with the number of
                                        ratings. Note that the weight algorithm only
                                        works exactly if the segment filters contain
                                        programType, genreBroad, genreMedium, and
                                        genreFine criteria. In all other cases the number
                                        of ratings matching the segment will be
                                        approximated.
}

@collSimEngProps =
{"assetTypes": [string],             optional, defines the asset types to be used by the
                                        engine. If there is an assetTypes definition in
                                        the global context, it is overruled by this
                                        definition for the given engine. Note that for
                                        compatibility to earlier releases the asset type
                                        "ad" is reserved for advertising assets and is
                                        never used by recommendation engines. If
                                        "assetTypes" is not specified, all asset types
                                        with the exception of "ad" are used.
 "name": string,                     required, unique name of the engine
 "inputs": [string,...],             optional, list of sourceGroupIds used as input,
                                        empty list or omitted inputs mean all
                                        sourceGroups will be used
 "assetStores": [<storeid>],         optional, if defined, all engines for which no
                                        separate assetStores configuration is provided
                                        will use the given asset store(s) instead of the
                                        active asset store(s). If the list is empty, no
                                        asset store will be used (this is useful e.g. to
                                        provide recommendations from playlists only).
 "playlists": [<string>],            optional, list of playlist IDs. The assets on the
                                        given playlists will be used as recommendation
                                        candidates (exclusively if assetStores=[], in
                                        addition to the assets in the active/given
                                        assetStores if assetStores is missing or non-
                                        empty). Overwrites global assetStores/playlists
                                        configuration for the specific engine.
 "subscriberPlaylists": <boolean>,   optional, if true, all assets in all playlists of
                                        the subscriber will be used as recommendation
                                        candidates too. After resolving into a list of
                                        playlist IDs this will be merged with the
                                        directly specified IDs in "playlists". Overwrites
                                        the global configuration for the specific engine.
                                        Supported from R5.6.
 "bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
                                        The given bookmark lists will be exclusively used
                                        as recommendation candidates. From R5.1.
 "subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                        the subscriber will be used as recommendation
                                        candidates. After resolving into a list of
```

| | |
|---|---|
| | bookmark list IDs this will be merged with the directly specified IDs in "bookmarkListFilter". Supported from R5.6. |
| "minScore: decimal, | optional, range [0, 1] |
| "minBusinessScore: dcimal, | optional, range [0, 1], supported from R5.1 |
| "minAge": int, | optional, if defined, the engine specific minimum age will overrule the global minimum age. If "country" is specified in the global section of the context, the corresponding age rating will be used. If "country" is not specified, the highest minAge of all countries available will be used. |
| "filter": @genericFilterVar, | optional, allows to filter recommendations to any asset attribute value. If a filter is present in the global section of the context too, the two filters will be intersected. Supports variables from R5.0 |
| "contribution": decimal, | optional, range [0, +inf[ |
| "weightFunctions":[@weightFunction,...], | optional |
| "seed": string, | optional, <reference> from the recommendation request URL identifying the seed asset to be used The default value of "<engineName>.seed" is used if the seed reference is not specified. If seed and seeds are both provided, the union of both is used. |
| "seeds": string, | optional, from R5.6, <reference> from the recommendation request URL identifying the seed assets to be used. The default value of "<engineName>.seeds" is used if seeds is omitted. If seed and seeds are both specified, all the asset IDs are used. |
| "excludeSeed": enum, | optional, one of "none", "asset", "program", "series", default: "asset", defines if only the seed asset, all assets with the same program ID as the seed, or all assets with the same series ID as the seed shall be excluded from Recommendations |
| "collaborativeLearningLevel": enum | optional, supported from R5.0, one of "assetId", "programId", "seriesId", default: "assetId". This parameter defines if the collaborative engine shall work at individual asset level, at program level (not distinguishing between assets with the same program ID), or at series level (not distinguishing between assets with the same seriesId). |
| "groupingPriority": enum, | optional, only episode & episodeForced supported before R5.6, allowed values "businessScore", "firstUnwatchedEpisode", "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, |

```
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,           optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,  optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
                                    availability, the priority attribute only decides
                                    between absolute and relative windows specified
                                    at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                    window of availability start or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"windowOfAvailabilityEnd": timestamp, optional, from R5.0, default +infinity. Sets the
                                    window of availability end or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"relativeWindowOfAvailabilityStart":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,              required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
    "offset": <duration>           required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":  optional, from R5.0, defines the relative time
```

```
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,               required, one of "currentTime", "startOfDay",
                                      "startOfWeek" (R5.0)
    "offset": <duration>            required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {  optional, from R5.0, overrules the generic window
                                    of availability definition for selected source
                                    IDs for the particular engine only.
     <sourceId>: {                  the sourceId the special window is applicable to
     "start": <timestamp>,          optional, absolute start of the window
     "end": <timestamp>,            optional, absolute end of the window
     "relativeStart": {             optional, relative start of the window
       "origin": <enum>,            required, one of "currentTime", "startOfDay",
                                      "startOfWeek" (R5.0)
       "offset": <duration>         required, offset in XSD duration format
       }
     "relativeEnd": {               optional, relative end of the window
       "origin": <enum>,            required, one of "currentTime", "startOfDay",
                                      "startOfWeek" (R5.0)
       "offset": <duration>         required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                    generic window of availability definition for
                                    selected source group IDs.
     <sourceGroupId>: {             the sourceGroupId the special window is applicable
                                    to (sourceGroupId and sourceGroupIds are both
                                    used)
     "start": timestamp,            optional, absolute start of the window
     "end": <timestamp>,            optional, absolute end of the window
     "relativeStart": {             optional, relative start of the window
       "origin": enum,              required, one of "currentTime", "startOfDay",
                                      "startOfWeek"
       "offset": duration           required, offset in XSD duration format
       }
     "relativeEnd": {               optional, relative end of the window
       "origin": enum,              required, one of "currentTime", "startOfDay",
                                      "startOfWeek"
       "offset": duration           required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],    optional, allowed values: "originalStartEndTime" or
                                    "availabilityStartEndTime", default value
                                    ["availabilityStartEndTime"]. Defines which asset
                                    attribute pairs determine the asset's
                                    availability. By default only
                                    availabilityStartTime / availabilityEndTime are
                                    used. This can be replaced or extended by
                                    originalStartTime / originalEndTime. Overwrites
                                    the global setting for one engine only. Supported
```

```
                                    from R5.1.
"frequencyCapping": {               optional, from R5.4, limits how often Ncanto is
                                      allowed to recommend the same asset to a
                                      subscriber in a time period. Overrides for the
                                      particular engine the global frequencyCapping
                                      configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                      same assetId in recommendations of the same
                                      subscriber.
 },
 "customExplanations": {            optional, from R5.0, custom explanations applied to
                                      all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                      explanationText may include references to context
                                      variables in the syntax
                                      ${<variableName!"<defaultValue>"}.
                                      Variable names which cannot be resolved are
                                      replaced by the <defaultValue> if one is
                                      specified, or by "" if no default is given.
  }
 },
 "contributionAdjustment": {        optional, from R5.0, if present, Ncanto will
                                      automatically fine tune the contribution of the
                                      engine between contribution and contribution *
                                      minFactor
  "minFactor": decimal              optional, decimal value in range [0.0:1.0[
 }
}

@customScoreEngProps =
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                      engine. If there is an assetTypes definition in
                                      the global context, it is overruled by this
                                      definition for the given engine. Note that for
                                      compatibility to earlier releases the asset type
                                      "ad" is reserved for advertising assets and is
                                      never used by recommendation engines. If
                                      "assetTypes" is not specified, all asset types
                                      with the exception of "ad" are used.
 "name": string,                    required, unique name of the engine
 "inputs": [string,...],            optional, list of sourceGroupIds used as input,
                                      empty list or omitted inputs mean all
                                      sourceGroups will be used
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                      separate assetStores configuration is provided
                                      will use the given asset store(s) instead of the
                                      active asset store(s). If the list is empty, no
                                      asset store will be used (this is useful e.g. to
                                      provide recommendations from playlists only).
 "playlists": [<string>],           optional, list of playlist IDs. The assets on the
                                      given playlists will be used as recommendation
                                      candidates (exclusively if assetStores=[], in
                                      addition to the assets in the active/given
                                      assetStores if assetStores is missing or non-
                                      empty). Overwrites global assetStores/playlists
                                      configuration for the specific engine.
 "subscriberPlaylists": <boolean>,  optional, if true, all assets in all playlists of
```

```
                                    the subscriber will be used as recommendation
                                    candidates too. After resolving into a list of
                                    playlist IDs this will be merged with the
                                    directly specified IDs in "playlists". Overwrites
                                    the global configuration for the specific engine.
                                    Supported from R5.6.
"bookmarkListFilter": [<string>],   optional, list of bookmark list IDs. The assets in
                                    The given bookmark lists will be exclusively used
                                    as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool, optional, only the assets in all bookmark lists of
                                    the subscriber will be used as recommendation
                                    candidates. After resolving into a list of
                                    bookmark list IDs this will be merged with the
                                    directly specified IDs in "bookmarkListFilter".
                                    Supported from R5.6.
"contribution": decimal,            optional, range [0, +inf[
"minScore: dcimal,              optional, range [0, 1]. Preference engines ignore
                                    the minimum score requirement until the profile
                                    becomes active.
"minBusinessScore: decimal,         optional, range [0, 1], supported from R5.1
"minAge": int,                      optional, if defined, the engine specific minimum
                                    age will overrule the global minimum age. If
                                    "country" is specified in the global section of
                                    the context, the corresponding age rating will be
                                    used. If "country" is not specified, the highest
                                    minAge of all countries available will be used.
"filter": @genericFilterVar,        optional, allows to filter recommendations to any
                                    asset attribute value. If a filter is present in
                                    the global section of the context too, the two
                                    filters will be intersected. Supports variables.
"scoreFunctions": [                 required, define the custom score of the
                                    recommendations returned by the engine.
 {                                  The sum of the scores of all functions is used.
  "trigger": enum,                  required, one of "remainingAvailability",
                                    "elapsedAvailability" or any leaf attribute
                                    of the asset.
                                    Triggers have to be provided in dot notation and
                                    have to point to a leaf node of the asset. Branch
                                    nodes are silently ignored. Only
                                    one weight function per trigger is allowed
  "nodes":[
   {
    "triggerValue": custom,         required, all attributes requiring numeric trigger
                                    values are listed in the table below.
                                    triggerValue has to be unique for each node,
                                    multiple nodes with the same triggerValue are not
                                    allowed. For non-numeric triggers string and
                                    boolean values are expected. If the trigger
                                    attribute is multi-valued (e.g. genreBroad),
                                    multiple values may match a trigger value and the
                                    weights will be accumulated.
    "weight": decimal,              required, value of the weight function at the given
                                    triggerValue
   }
  ]
 }
],
"weightFunctions":[@weightFunction,...],    optional
"contributionAdjustment": {         optional,if present, Ncanto will
```

```
                                    automatically fine tune the contribution of the
                                    engine between contribution and contribution *
                                    minFactor
 "minFactor": decimal               optional, decimal value in range [0.0:1.0[
},
"randomizationFactor": decimal>=1.0, optional, if provided, the engine does not return
                                    the requested number of assets with highest
                                    scores, but a random subset of the requested
                                    number * randomizationFactor highest score
                                    assets.
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
                                    firstUnwatchedEpisode),
                                    "firstUnwatchedEpisodeForced", "episodeForced"
                                    (deprecated, same as
                                    firstUnwatchedEpisodeForced), "firstEpisode",
                                   "lastEpisode", "latestArrival", "earliestArrival",
                                    default "businessScore". Determines which
                                    episodes are first selected to populate each
                                    series group. By default the assets are selected
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,            optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,   optional, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
```

```
                                        availability, the priority attribute only decides
                                        between absolute and relative windows specified
                                        at the same level.
"windowOfAvailabilityStart": timestamp, optional, default –infinity. Sets the
                                        window of availability start or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"windowOfAvailabilityEnd": timestamp,  optional, default +infinity. Sets the
                                        window of availability end or overrides the
                                        global setting for the particular engine only.
                                        See @globalProps for a complete description of
                                        the attribute.
"relativeWindowOfAvailabilityStart":   optional, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek"
    "offset": <duration>               required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek"
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
      <sourceId>: {                     the sourceId the special window is applicable to
      "start": <timestamp>,             optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": <duration>            required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": <duration>            required, offset in XSD duration format
        }
      },
      ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
      <sourceGroupId>: {                the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
```

```
                                    used)
      "start": timestamp,           optional, absolute start of the window
      "end": <timestamp>,           optional, absolute end of the window
      "relativeStart": {            optional, relative start of the window
        "origin": enum,             required, one of "currentTime", "startOfDay",
                                      "startOfWeek"
        "offset": duration          required, offset in XSD duration format
        }
      "relativeEnd": {              optional, relative end of the window
        "origin": enum,             required, one of "currentTime", "startOfDay",
                                      "startOfWeek"
        "offset": duration          required, offset in XSD duration format
        }
      },
      ...
 },
 "availabilityWindows": [<enum>],   optional, allowed values: "originalStartEndTime" or
                                      "availabilityStartEndTime", default value
                                      ["availabilityStartEndTime"]. Defines which asset
                                      attribute pairs determine the asset's
                                      availability. By default only
                                      availabilityStartTime / availabilityEndTime are
                                      used. This can be replaced or extended by
                                      originalStartTime / originalEndTime. Overwrites
                                      the global setting for one engine only.
"frequencyCapping": {               optional, limits how often Ncanto is
                                      allowed to recommend the same asset to a
                                      subscriber in a time period. Overrides for the
                                      particular engine the global frequencyCapping
                                      configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                      same assetId in recommendations of the same
                                      subscriber.
 },
 "customExplanations": {            optional, custom explanations applied to
                                      all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                      explanationText may include references to context
                                      variables in the syntax
                                      ${<variableName!"<defaultValue>"}.
                                      Variable names which cannot be resolved are
                                      replaced by the <defaultValue> if one is
                                      specified, or by "" if no default is given.
  }
 }
}

@adHocPrefEngProps =
{"assetTypes": [string],            optional, defines the asset types to be used by the
                                      engine. If there is an assetTypes definition in
                                      the global context, it is overruled by this
                                      definition for the given engine. Note that for
                                      compatibility to earlier releases the asset type
                                      "ad" is reserved for advertising assets and is
                                      never used by recommendation engines. If
                                      "assetTypes" is not specified, all asset types
                                      with the exception of "ad" are used.
```

```
"name": string,                          required, unique name of the engine
"inputs": [string,...],                  optional, list of sourceGroupIds used as input,
                                           empty list or omitted inputs mean all
                                           sourceGroups will be used
"assetStores": [<storeid>],              optional, if defined, all engines for which no
                                           separate assetStores configuration is provided
                                           will use the given asset store(s) instead of the
                                           active asset store(s). If the list is empty, no
                                           asset store will be used (this is useful e.g. to
                                           provide recommendations from playlists only).
"playlists": [<string>],                 optional, list of playlist IDs. The assets on the
                                           given playlists will be used as recommendation
                                           candidates (exclusively if assetStores=[], in
                                           addition to the assets in the active/given
                                           assetStores if assetStores is missing or non-
                                           empty). Overwrites global assetStores/playlists
                                           configuration for the specific engine.
"subscriberPlaylists": <boolean>,        optional, if true, all assets in all playlists of
                                           the subscriber will be used as recommendation
                                           candidates too. After resolving into a list of
                                           playlist IDs this will be merged with the
                                           directly specified IDs in "playlists". Overwrites
                                           the global configuration for the specific engine.
                                           Supported from R5.6.
"bookmarkListFilter": [<string>],        optional, list of bookmark list IDs. The assets in
                                           The given bookmark lists will be exclusively used
                                           as recommendation candidates. From R5.1.
"subscriberBookmarkListFilter": bool,    optional, only the assets in all bookmark lists of
                                           the subscriber will be used as recommendation
                                           candidates. After resolving into a list of
                                           bookmark list IDs this will be merged with the
                                           directly specified IDs in "bookmarkListFilter".
                                           Supported from R5.6.
"contribution": decimal,                 optional, range [0, +inf[
"minScore: dcimal,                 optional, range [0, 1]. Preference engines ignore
                                           the minimum score requirement until the profile
                                           becomes active.
"minBusinessScore: decimal,              optional, range [0, 1], supported from R5.1
"minAge": int,                           optional, if defined, the engine specific minimum
                                           age will overrule the global minimum age. If
                                           "country" is specified in the global section of
                                           the context, the corresponding age rating will be
                                           used. If "country" is not specified, the highest
                                           minAge of all countries available will be used.
"filter": @genericFilterVar,             optional, allows to filter recommendations to any
                                           asset attribute value. If a filter is present in
                                           the global section of the context too, the two
                                           filters will be intersected. Supports variables
                                           from R5.0.
"ratedAssetIds": string,                 optional, <reference> from the recommendation
                                           request URL corresponding to the list of rated
                                           asset IDs. The default value of
                                           "<engineName>.ratedAssetIds" is used if not
                                           specified. The format of the query parameter
                                           value is [{"assetId": string, "rating":
                                           decimal}]. Note that a 400 exception is thrown if
                                           any of the asset IDs is not present in the active
                                           store, unless errorTolerance is set to true.
"errorTolerance": boolean,               optional, if true, the recommendation request will
```

```
                                    silently ignore if any of the rated asset IDs is
                                    missing from the active store.
"weightFunctions":[@weightFunction,...],     optional
"recordingSuggestion": boolean      optional, default = false, if true, recommendation
                                    requests will return a flag indicating if the
                                    recommendation is suggested to be automatically
                                    scheduled by recording devices.
"skipProfileDislikeThreshold": int, optional, if present, preference engines having no
                                    positively rated asset but at least the given
                                    number of negatively rated assets in the profile
                                    will be skipped by setting their contribution to
                                    0, unless all preference engines would be
                                    skipped, in which case the attribute is ignored.
"excludeOwnRatedItems": enum,       optional, R4.4, one of "none", "asset", "program",
                                    "series", excludes assets/programs/series which
                                    have already been rated in the profile from
                                    recommendations. Default "none". If set to
                                    program, program and asset IDs are excluded. If
                                    set to series, series, program, and asset IDs are
                                    excluded.
"contributionAdjustment": {         optional, from R4.6, allows to reduce the
                                    contribution of the preference engine if the
                                    strength of the profile is low.
  "zeroContributionProfileStrengthLimit": deprecated from R5.0, renamed to
                                    minContributionProfileStrenghLimit
  "minContributionProfileStrengthLimit": decimal, required, range [0:1]. If the
                                      profile strength is below this limit, the
                                      contribution of the preference engine is set to
                                      contribution * minFactor. Negative values are
                                      deprecated.
  "fullContributionProfileStrengthLimit": decimal, required, range ]
                                    minContributionProfileStrengthLimit, 1]. If the
                                    profile strength is above that limit, the
                                    contribution is not adjusted. If the profile
                                    strength is between the two limit, the
                                    contribution is linearly adjusted between zero
                                    and the full contribution.
  "minFactor": decimal              optional, from R5.0, decimal value in range
                                    [0.0:1.0[, default value 0.0. Defines the minimum
                                    contribution of the engine as minContribution =
                                    contribution * minFactor.
},
"randomizationFactor": decimal>=1.0, optional, if provided, the engine does not return
                                    the requested number of assets with highest
                                    scores, but a random subset of the requested
                                    number * randomizationFactor highest score
                                    assets. Supported from R4.7.
"groupingPriority": enum,           optional, only episode & episodeForced supported
                                    before R5.6, allowed values
                                    "businessScore", "firstUnwatchedEpisode",
                                    "episode" (deprecated, same as
                                    firstUnwatchedEpisode),
                                    "firstUnwatchedEpisodeForced", "episodeForced"
                                    (deprecated, same as
                                    firstUnwatchedEpisodeForced), "firstEpisode",
                                   "lastEpisode", "latestArrival", "earliestArrival",
                                    default "businessScore". Determines which
                                    episodes are first selected to populate each
                                    series group. By default the assets are selected
```

```
                                    randomly if the business scores of the episodes
                                    are identical (groupingPriority=businessSCore).
                                    If firstUnwatchedEpisode priority is selected,
                                    the first not yet watched episode of each series
                                    are used. If firstUnwatchedEpisodeForced is
                                    specified, the first not yet watched episode is
                                    selected even if there are other episodes with
                                    higher business scores. firstEpisode and
                                    lastEpisode use the lowest / highest season /
                                    episode numbers available, ignoring which of them
                                    have already been watched. latestArrival /
                                    earliestArrival use the episodes with earlierst /
                                    latest availability star time to populate the
                                    group. Only effective if type of items =
                                    groups and if seriesId is among the grouping
                                    attributes.
"filterOwnSeries": enum,           optional, allowed values "none"(R5.3), "exclude",
                                    "includeOnly". This parameter allows to limit the
                                    recommendations to seriesIds which the subscriber
                                    already follows (play events are available) or to
                                    exclude such series IDs from the recommendations.
                                    In connection with grouping priority different
                                    behaviors for already followed series and for new
                                    discoveries can be configured.
"windowOfAvailabilityType": enum,  optional, from R5.0, one of "overlap",
                                    "strictStart", "strictEnd", "strict", default:
                                    "overlap". Sets the window of availability type
                                    or overrides the global setting for the
                                    particular engine only. See @globalProps for a
                                    complete description of the attribute.
"windowOfAvailabilityPriority": <enum>, optional, from R5.0, allowed values "relative",
                                    "tightest", default value "relative". Sets the
                                    window of availability priority or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute. Note that engine level window of
                                    availability always overrides global window of
                                    availability, the priority attribute only decides
                                    between absolute and relative windows specified
                                    at the same level.
"windowOfAvailabilityStart": timestamp, optional, from R5.0, default –infinity. Sets the
                                    window of availability start or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"windowOfAvailabilityEnd": timestamp, optional, from R5.0, default +infinity. Sets the
                                    window of availability end or overrides the
                                    global setting for the particular engine only.
                                    See @globalProps for a complete description of
                                    the attribute.
"relativeWindowOfAvailabilityStart":  optional, from R5.0, defines the relative time
                                    window in which recommendations shall be
                                    available. Sets the reletive window of
                                    availability start or overrides the global
                                    setting for the particular engine only. See
                                    @globalProps for a complete description of the
                                    attribute.
   {"origin": <enum>,              required, one of "currentTime", "startOfDay",
                                    "startOfWeek" (R5.0)
```

```
    "offset": <duration>              required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":    optional, from R5.0, defines the relative time
                                        window in which recommendations shall be
                                        available. Sets the reletive window of
                                        availability start or overrides the global
                                        setting for the particular engine only. See
                                        @globalProps for a complete description of the
                                        attribute.
   {"origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>              required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {    optional, from R5.0, overrules the generic window
                                        of availability definition for selected source
                                        IDs for the particular engine only.
     <sourceId>: {                      the sourceId the special window is applicable to
     "start": <timestamp>,              optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": <enum>,                required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
       "offset": <duration>             required, offset in XSD duration format
       }
     },
     ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
     <sourceGroupId>: {               the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
                                        used)
     "start": timestamp,                optional, absolute start of the window
     "end": <timestamp>,                optional, absolute end of the window
     "relativeStart": {                 optional, relative start of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
       "offset": duration               required, offset in XSD duration format
       }
     "relativeEnd": {                   optional, relative end of the window
       "origin": enum,                  required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
       "offset": duration               required, offset in XSD duration format
       }
     },
     ...
},
"availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                        "availabilityStartEndTime", default value
                                        ["availabilityStartEndTime"]. Defines which asset
                                        attribute pairs determine the asset's
                                        availability. By default only
                                        availabilityStartTime / availabilityEndTime are
```

```
                                    used. This can be replaced or extended by
                                    originalStartTime / originalEndTime. Overwrites
                                    the global setting for one engine only. Supported
                                    from R5.1.
"frequencyCapping": {               optional, from R5.4, limits how often Ncanto is
                                    allowed to recommend the same asset to a
                                    subscriber in a time period. Overrides for the
                                    particular engine the global frequencyCapping
                                    configuration.
  "timeWindow": <duration>,         required, the observation period
  "cap": <integer>                  required, the maximum number of occurrences of the
                                    same assetId in recommendations of the same
                                    subscriber.
 },
 "customExplanations": {            optional, from R5.0, custom explanations applied to
                                    all recommendations coming from this engine
  <type>: {                         mandatory, explanation type (nonEmptyString)
   <langcode>: <explanationText>    mandatory, language / explanation text pairs. The
                                    explanationText may include references to context
                                    variables in the syntax
                                    ${<variableName!"<defaultValue>"}.
                                    Variable names which cannot be resolved are
                                    replaced by the <defaultValue> if one is
                                    specified, or by "" if no default is given.
  }
 }
}
```

**@adTargetingProps =**

```
{"algorithm": enum,                 optional, one of "similarity" or "preference" or
                                    "similarity+preference"
 "adFrequency": decimal,            optional, range [0, 1]
 "adPools": [string],               optional, list of adPool IDs
 "assetTypes": [string],            optional, defines the asset types which in addition
                                    to assetType="ad" are considered as advertising
                                    assets too. For compatibility with earlier
                                    relases assetType="ad" is always used for ad
                                    targeting.
 "adAnnotation": string             optional, annotationId to be applied to
                                    advertising. If not defined, the global
                                    annotation will be used.
}
```

**@globalProps =**

```
{"assetTypes": [string],            optional, defines the asset types to be used by all
                                    engines. The global asset type setting may be
                                    overruled for individual engines in the enigne
                                    specific section of the context. Note that for
                                    compatibility to earlier releases the asset type
                                    "ad" is reserved for advertising assets and is
                                    never used by recommendation engines. If
                                    "assetTypes" is not specified, all asset types
                                    with the exception of "ad" are used.
 "grouping": enum,                  optional, one of "none", "program", "series",
                                    default="none", deprecated from R4.6
 "groupingAttributes": [string],    optional, list of asset attribute names in dot
                                    notation. The first attribute that is present in
                                    an asset will define the asset's group. Only leaf
                                    attributes are supported, no wildcards. Use
```

| | |
|---|---|
| | "groupingAttributes":["seriesId","programId"] as replacement of the deprecated "grouping":"series" configuration. Supported from R4.6. Overrules "grouping" if both are present in the context. |
| "groupingPriority": enum, | optional, only episode & episodeForced supported before R5.6, allowed values "businessScore", "firstUnwatchedEpisode", "episode" (deprecated, same as firstUnwatchedEpisode), "firstUnwatchedEpisodeForced", "episodeForced" (deprecated, same as firstUnwatchedEpisodeForced), "firstEpisode", "lastEpisode", "latestArrival", "earliestArrival", default "businessScore". Determines which episodes are first selected to populate each series group. By default the assets are selected randomly if the business scores of the episodes are identical (groupingPriority=businessSCore). If firstUnwatchedEpisode priority is selected, the first not yet watched episode of each series are used. If firstUnwatchedEpisodeForced is specified, the first not yet watched episode is selected even if there are other episodes with higher business scores. firstEpisode and lastEpisode use the lowest / highest season / episode numbers available, ignoring which of them have already been watched. latestArrival / earliestArrival use the episodes with earlierst / latest availability star time to populate the group. Only effective if type of items = groups and if seriesId is among the grouping attributes. This setting is used for all engines which don't have their own groupingPriority setting. An engine level setting overrules the global setting for that particular engine. |
| "stackedGroups": boolean, | optional, from R5.3, if true, the assets belonging to a group are stacked into the first recommendation object of the group instead of a flat list of the recommendations. Allowed only if typeOfItems==groups. |
| "numberOfItems":int, | optional, number of items to be returned |
| "typeOfItems": enum, | optional, one of "assets", "groups", default = "assets" |
| "maxGroupSize": int, | optional, defines the max. No. of assets per group, ignored if typeOfItems = assets |
| "windowOfAvailabilityType": enum, | optional, one of "overlap", "strictStart", "strictEnd", "strict", default: "overlap". Defines the interpretation of window of availability and relative window of availability. If set to overlap, any overlap between the asset's availability and the specified window of availability will be allowed. If set to "strictStart" or "strictEnd", the availability start / end of the asset has to lie within the specified window of availability. If set to "strict", both the asset's availability start and also availability end have to lie within the specified window of availability. |
| "windowOfAvailabilityPriority": <enum>, | optional, allowed values "relative", "tightest", |

```
                                        default value "relative". Relative means that if
                                        a relative window is specified, it overrides an
                                        absolute window. Thightest means that the more
                                        stringent of the relative and the absolute limits
                                        is used. Applies to the generic as well as to the
                                        per source window of availability. Only applies
                                        to global availability windows, an absolute
                                        availability window defined at engine level will
                                        overwrite a global relative availability window
                                        even if windowOfAvailabilityPriority=relative.
"windowOfAvailabilityStart": timestamp,     optional, default -infinity, defines the
                                        absolute time window in which results shall be
                                        available.
"windowOfAvailabilityEnd": timestamp,  optional, default +infinity, defines the
                                        absolute time window in which results shall be
                                        available.
"relativeWindowOfAvailabilityStart":   optional, defines the relative time
                                        window in which recommendations shall be
                                        available.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"relativeWindowOfAvailabilityEnd":     optional, defines the relative time
                                        window in which recommendations shall be
                                        available.
   {"origin": <enum>,                  required, one of "currentTime", "startOfDay",
                                        "startOfWeek" (R5.0)
    "offset": <duration>               required, offset in XSD duration format
   },
"windowOfAvailabilityPerSource": {     optional, overrules the generic window of
                                        availability definition for selected source IDs.
      <sourceId>: {                     the sourceId the special window is applicable to
      "start": <timestamp>,             optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
        "offset": <duration>            required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": <enum>,               required, one of "currentTime", "startOfDay",
                                          "startOfWeek" (R5.0)
        "offset": <duration>            required, offset in XSD duration format
        }
      },
      ...
},
"windowOfAvailabilityPerSourceGroup": {optional, supported from R5.6, overrules the
                                        generic window of availability definition for
                                        selected source group IDs.
      <sourceGroupId>: {                the sourceGroupId the special window is applicable
                                        to (sourceGroupId and sourceGroupIds are both
                                        used)
      "start": timestamp,               optional, absolute start of the window
      "end": <timestamp>,               optional, absolute end of the window
      "relativeStart": {                optional, relative start of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
```

```
        "offset": duration              required, offset in XSD duration format
        }
      "relativeEnd": {                  optional, relative end of the window
        "origin": enum,                 required, one of "currentTime", "startOfDay",
                                          "startOfWeek"
        "offset": duration              required, offset in XSD duration format
        }
      },
      ...
},
"availabilityWindows": [<enum>],      optional, allowed values: "originalStartEndTime" or
                                        "availabilityStartEndTime", default value
                                        ["availabilityStartEndTime"]. Defines which asset
                                        attribute pairs determine the asset's
                                        availability. By default only
                                        availabilityStartTime / availabilityEndTime are
                                        used. This can be replaced or extended by
                                        originalStartTime / originalEndTime. Global
                                        setting applicable to all engines without engine
                                        specific availabilityWindows. Supported from
                                        R5.1.
"frequencyCapping": {                 optional, from R5.4, limits how often Ncanto is
                                        allowed to recommend the same asset to a
                                        subscriber in a time period. Global configuration
                                        applying to all engines which don't have their
                                        own frequency cap.
 "timeWindow": <duration>,            required, the observation period
 "cap": <integer>                     required, the maximum number of occurrences of the
                                        same assetId in recommendations of the same
                                        subscriber.
},
"country": string,                    optional, used to select the desired age
                                        rating
"language": langCode                  optional, determines the language of annotations
                                        and explanations. If language is present,
                                        "languages" and "fallbackLanguages" are ignored
                                        and the legacy behavior applies: all language
                                        maps are filtered to the single language desired,
                                        if that is not available an empty object is
                                        returned and no fallbacks apply.
"languages": {                        optional, from R4.5, ignored if "language" is given
  "sequence": [langCode],             required, the first available <count> languages
                                        from the list are included in the response. If
                                        none of the languages is available in the
                                        metadata, the "fallbackLanguages" are used.
  "count": int                        required,
 },
"fallbackLanguages": {                optional, from R4.5, ignored if "language" is given
  "sequence": [langCode],             required, the first available <count> languages
                                        from the list are included in the response if
                                        none of the preferred languages is available in
                                        the metadata. If none of the fallbackLanguages is
                                        available in the metadata, the defaultLanguage of
                                        the asset is used. If the asset has no
                                        defaultLanguage or if the defaultLanguage is not
                                        available, no language filtering is done at all,
                                        all available languages are returned.
  "count": int                        required,
 },
```

```
"minAge":int,                          optional, if "country" is specified, the
                                         corresponding age rating will be used. If
                                         "country" is not specified, the highest minAge of
                                         all countries available will be used. Note that
                                         the global minAge limit may be overruled by per-
                                         engine minAge values.
"filter": @genericFilterVar            optional, allows to filter recommendations to any
                                         asset attribute value. Supports variables from
                                         R5.0.
"subscriberBlacklisting": boolean,     optional, default = false. If
                                         subscriberBlacklisting is enabled, the subscriber
                                         ID becomes a required parameter of the
                                         recommendation request
"sorting": enum,                       optional default = "byScore", supported values:
                                         "byScore", "byTime", "byLineup", "byEpisode",
                                         "byEngine", "byGroupMaxScore", "byBusinessScore",
                                         "byRecordingSuggestion", "multi", or any
                                         asset attribute in dot notation. From R4.8
                                         "random" is supported too.
                                         See Section 2.3.6 for details. You can change the
                                         sorting direction using sorting=multi.
"multiSorting": [@sortingCriterion],   required if "sorting"="multi", ignored otherwise
"annotation": string,                  optional, annotationId to be applied to all
                                         asset types. Default: only assetId is returned.
"annotationAttributes":[string],       optional, list of asset attributes in dot notation,
                                         will be merged with "annotation" if both are
                                         present
"conditionalAnnotationAttributes":[@condAnn],optional, list of conditional annotation
                                         attributes, see Section 3.10.1, will be merged
                                         with "annotation" if both are present
"annotationPerAssetType": {            optional, individual <assetType>: string,
                                         annotations per asset type. If present, the
  ...                                    global annotation is selectively overruled for
 }                                       the corresponding asset types.
"entitlement":[string,...],            optional, list of entitlements, empty list means
                                         all sources will be used. If provided, an
                                         entitlement defined here overrules the default
                                         entitlement of the subscriber. The union of
                                         entitlement, entitlementSources,
                                         entitlementSourceGroups, and entitlementFilter is
                                         used.
"entitlementSources": [string],        optional, list of source IDs to be used as
                                         entitlement. If provided, an
                                         entitlement defined here overrules the default
                                         entitlement of the subscriber.  The union of
                                         entitlement, entitlementSources,
                                         entitlementSourceGroups, and entitlementFilter is
                                         used.
"entitlementSourceGroups": [string],   optional, list of sourceGroup IDs to be used as
                                         entitlement. If provided, an entitlement defined
                                         here overrules the default
                                         entitlement of the subscriber.  The union of
                                         entitlement, entitlementSources,
                                         entitlementSourceGroups, and entitlementFilter is
                                         used.
"entitlementFilter": @genericFilter    optional, from R4.6, generic filter expression
                                         defining the entitlement. If provided, an
                                         entitlement defined here overrules the default
                                         entitlement of the subscriber.  The union of
```

```
                                    entitlement, entitlementSources,
                                    entitlementSourceGroups, and entitlementFilter is
                                    used.
 "sourceLineupName": string,        optional, reference to one of the subscriber's
                                    source lineups. If byLineup is one of the sorting
                                    criteria, either sourceLineupName or sourceLineup
                                    has to be specified.
 "sourceLineup": [sourceId],        optional, list of source IDs to be used for
                                    sorting. If byLineup is one of the sorting
                                    criteria, either sourceLineupName or sourceLineup
                                    has to be specified. If both are present,
                                    sourceLineup overrules sourceLineupName.
 "lineupFilter": {                  optional, if present, the results are filtered to
                                    (a subset of) the lineup filter defined by
                                    sourceLineupName or sourceLineup. Supported from
                                    R4.7.
   "startIndex": int,               optional, default 0, specifies the index of the
                                    first sourceId in the lineup which should be
                                    included in the results
   "endIndex": int                  optional, default infinity, specifies the index of
                                    the last sourceId in the lineup which should be
                                    included in the results
 },
 "device":[string,...],             optional, list of device identifiers
 "timeZone": timezone,              optional, time zone of the client used to evaluate
                                    time of day based business rules
 "maxExplanationLevel": int,        optional, default = 1,
                                    determines the depth of explanations to be
                                    returned, deprecated from R4.6.
 "explanationConfig": {             optional, supported from R4.6, if present,
                                    maxExplanationLevel is ignored
   "language": <langcode>,          optional, the desired language of the explanation.
                                    If missing, the first metadataLanguage is used,
                                    or the default language English
   "types": [<type_enum>],          optional, any of "engine", "reference", "feature".
                                    "engine" returns basic explanations about the
                                    algorithm used to provide the recommendation.
                                    "reference" returns information about a reference
                                    object used by the engine, e.g. which profile or
                                    which seed asset was used. "feature" is only used
                                    by preference and similarity engines and returns
                                    the asset features which resulted in the
                                    recpmmendation. From R5.0 the custom explanation
                                    types defined in customExplanations are allowed
                                    values, too.
   "metadataLanguages": [<langcode>] optional, preferred language of the asset
                                    attributes used in the explanation. If missing
                                    the first available language from the global
                                    context's "language", "languages" or
                                    "fallbackLanguages" will be used. If none of the
                                    languages is available in the asset, the IDs and
                                    canonical names will be used.
 },
 "assetIds": [<string512>],         optional, if provided, the request will filter the
                                    response to the given list of asset IDs.
 "assetStores": [<storeid>],        optional, if defined, all engines for which no
                                    separate assetStores configuration is provided
                                    will use the given asset store(s) instead of the
                                    active asset store(s). If the list is empty, no
```

```
                                        asset store will be used (this is useful e.g. to
                                        provide recommendations from playlists only).
 "playlists": [<string>],               optional, list of playlist IDs. The assets on the
                                        given playlists will be used as recommendation
                                        candidates (exclusively if assetStores=[], in
                                        addition to the assets in the active/given
                                        assetStores if assetStores is missing or non-
                                        empty) for all engines for which no separate
                                        playlists/assetStores configuration is provided.
 "subscriberPlaylists": <boolean>,      optional, if true, all assets in all playlists of
                                        the subscriber will be used as recommendation
                                        candidates. After resolving into a list of
                                        playlist IDs this will be merged with the
                                        directly specified IDs in "playlists".
                                        Supported from R4.3.
 "bookmarkListFilter": [<string>],      optional, list of bookmark list IDs. The assets in
                                        The given bookmark lists will be exclusively used
                                        as recommendation candidates. Only valid for
                                        engines without their own bookmarkList
                                        specification. From R5.6.
 "subscriberBookmarkListFilter": bool,  optional, only the assets in all bookmark lists of
                                        the subscriber will be used as recommendation
                                        candidates. After resolving into a list of
                                        bookmark list IDs this will be merged with the
                                        directly specified IDs in "bookmarkListFilter".
                                        Only valid for engines without their own
                                        bookmarkList specification. Supported from R5.6.
 "setting": {                           optional, viewing setting the recommendations are
                                        requested for
  "timeOfDayBin": <string>,             optional, e.g. "morning", "evening"
  "timeOfWeekBin": <string>,            optional, e.g. "weekday", "weekend"
  "device": <string>,                   optional, the device used
  "locationBin": <string>,              optional, e.g. "home", "work"
  "moodBin": <string>,                  optional, e.g. "happy", "sad"
  "activity": <string>,                 optional, e.g. "traveling", "relaxing"
  "autoTimeOfDayBin": {                 optional, from R5.6, ignored if timeOfDayBin
                                        is set, defines the time of day bin values to be
                                        set automatically depending on current time
    <offsetFrom>: <binName>             at least one required, offset wrt beginning of the
                                        day (xsdDuration) bin name (string) pair
  },
  "autoTimeOfWeekBin: {                 optional, from R5.6, ignored if timeOfWeekBin is
                                        set, defines the time of week bin values to be
                                        set automatically depending on current time
    <offsetFrom>: <binName>             at least one required, offset wrt beginning of the
                                        week (xsdDuration) bin name (string) pair
  }
 },
  "filterFacets": {                     optional, from R4.4, returns the specified number
                                        of attribute values of the given asset
                                        attributes.
    <attribute>: <count>                leaf attribute of @asset in dot notation: number
  }                                     of values to be returned (0=unlimited)
                                        Only leaf attribute names without wildcards are
                                        supported, branch attributes and attributes
                                        including wildcards are ignored.
}
```

**@weightFunction =**

```
{"trigger": enum,                       required, one of "remainingAvailability",
                                          "elapsedAvailability", "profitability",
                                          "priceBin", "communityRating". Starting from R4.5
                                          also "communityViewCount", "communityShareCount",
                                          "communityTrendScore","productionYearFirst",
                                          "productionYearLast", "durationSeconds",
                                          "leaseDurationMinutes", "viewCount", "lightness",
                                          "pace".
                                          Starting from R4.5 any non-numeric leaf attribute
                                          is supported as "discrete weight function" to
                                          boost individual values of the trigger attribute.
                                          Triggers have to be provided in dot notation and
                                          have to point to a leaf node of the asset. Branch
                                          nodes are silently ignored. Only
                                          one weight function per trigger is allowed
 "type": enum,                          required, one of "multiplicative", "additive".
                                          "additive" is deprecated from R4.7. Because score
                                          normalization in similarity, search, social, and
                                          posting engines takes place after business
                                          scoring, the actually applied weights may be
                                          smaller than specified for those engines.
 "nodes":[
          {"triggerValue": custom,      required, all attributes requiring numeric trigger
                                          values are listed in the table below.
                                          triggerValue has to be unique for each node,
                                          multiple nodes with the same triggerValue are not
                                          allowed. For non-numeric triggers string and
                                          boolean values are expected. If the trigger
                                          attribute is multi-valued (e.g. genreBroad),
                                          multiple trigger values may be matched and the
                                          weights will be accumulated.
           "weight": decimal,           required, value of the weight function at the given
                                          triggerValue
          },...
         ]
}
```

| trigger | trigger value definition and unit |
|---|---|
| remainingAvailability | time until availabilityEndTime in seconds. (note that multiple availability windows are not considered) |
| elapsedAvailability | time since availabilityStartTime in seconds (note that multiple availability windows are not considered) |
| profitability | customer defined, same as used in @asset |
| priceBin | Discrete price bin values 0="free", 1="cheap", 2="inexpensive", 3="not too expensive", 4="premium", 5="expensive" |
| communityRating | Average rating of the whole user base, decimal value in range[0:1] |
| communityViewCount | Total number of "play" interactions registered for the asset. Integer value in range [0, infinity[ |
| communityShareCount | Total number of "share" interactions registered for the asset. Integer value in range [0, infinity[ |
| communityTrendScore | Trend score of the asset, decimal value in range [0:1] |
| productionYearFirst | Integer |
| productionYearLast | Integer |
| durationSeconds | Non-negative integer |
| leaseDurationMinutes | Non-negative integer |
| viewCount | Non-negative integer |
| lightness | Decimal in range [0:1] |
| pace | Decimal in range [0:1] |

**@genericFilterVar** =

| | |
|---|---|
| { "term": string, | optional, name of the asset attribute, e.g. "uiGenre.<type>.<lang>". All asset attributes are supported. If term is specified, the filter consists of one single criterion and a value has to be specified. |
| "value": string, | required if "term" is defined and operator is not "in", "exists", or "notExists", value of the asset attribute, not allowed if operator is "in", "exists", "notExists". Used as fallback in case a "variable" is defined but returns null. |
| "variable": string, | optional, from R5.0, only supported by recommendation requests, name of the variable the value of which shouls be substituted by Ncanto. |
| "values": [string], | required if "term" is defined and "operator"="in", not allowed for other operators, supported from R4.0". Used as fallback in case "variables" is defined but returns a list of all nulls. |
| "variables": [string], | optional, from R5.0, list of variables the values of which should be substituted by Ncanto. Only supported by recommendation requests. |
| "operator": enum, | optional, one of "and", "nestedAnd", "or", "equals", "notEquals", "contains", "notContains", "atMost", "atLeast", "smaller", "larger", "in", "exists", "notExists". If operator is one of "and", "nestedAnd" (R4.6), "or": "criteria" have to be specified. If "operator" is one of "equals", "notEquals", "contains", "notContains", "atMost", "atLeast", "smaller", "larger": "term" and "value" have to be specified. If operator is "in": "term" and "values" have to be specified. If operator is "exists", "notExists": "term" has to be specified. If "operator" is missing, the default value "equals" is assumed. The allowed combinations of "term" and "operator" are listed in Table 1. Note that "notEquals" returns true if the given metadata attribute is not present at all. |
| "criteria": [@genericFilter] | required if "operator" equals "and", "nestedAnd", or "or". |
| } | |

**@sortingCriterion** =

| | |
|---|---|
| {"criterion":enum, | required, one of "byScore", "byTime", "byLineup", "byEpisode", "byEngine", "byGroupMaxScore", "byBusinessScore". Any asset attribute in dot notation is supported too. See Section 2.3.6 for details. From R4.8 "random" sorting is supported. |
| "condition": @genericFilter, | optional, from R4.6. If the criterion points to a list of objects with nested mapping the condition may be used to select which element of the list should be used for sorting. Please note that this only applies to data structures inside customProperties and clientCustomProperties, and that nested mapping has to be configured by XroadMedia personnel. |
| "direction": enum, | optional, one of "ascending", "descending", default value of criterion used if missing, see Section |

```
                                        2.3.6 for details.
 "sequence": [string]                 optional, supported from R4.5, Values in the
                                        sorting sequence are always sorted to the
                                        beginning of the result list, in the order of
                                        their appearance in the sequence. The remaining
                                        values are sorted according to the specified
                                        direction. Note that the sorting sequence only
                                        works for single value leaf attributes.
}

@condAnn =
{<attribute>:<@genericFilter,         at least one attribute:criterion pair required
                                        <attribute>: asset attribute name in dot notation
                                        @genericFilter: condition which has to be met for
                                        the attribute to be returned, syntax see above.
                                        Note that the "and" operator does not make sense
                                        in an anootation condition, it is interpreted as
                                        an equivalent of "nestedAnd" in annotation
                                        conditions, i.e. the condition is applied to each
                                        element of a list individually.
}
```

### 3.16.2 Retrieve one or multiple Context(s)

| **GET** | `/context/<contextId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | contextId not found |
| ***response*** | `@context` | |

| **GET** | `/context?count=<no_of_contexts>&startAfter=<contextId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@context]`<br>Contexts are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

To retrieve context IDs only (instead of full contexts) the `idsOnly` parameter has to be added to the request:

| **GET** | `/context?idsOnly=true&count=<no_of_contexts>&startAfter=<contextId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[<contextId>]`<br>Context IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.16.3 Delete a Context

| **DELETE** | `/context/<contextId>` | |
|---|---|---|
| ***result*** | **204 OK** | Request successful |

### 3.16.4 Context Example

The example below shows a typical context of Ncanto.

```
{
    "contextId": "default",
    "rules": [
        {
            "title": "basic engine config",
            "then": {
                "global": {
                    "grouping": "none",
                    "numberOfItems": 20,
                    "typeOfItems": "assets",
                    "windowOfAvailabilityStart": "",
                    "windowOfAvailabilityEnd": "",
                    "sorting": "byScore",
                    "country": "deu",
                    "minAge": 16,
                    "subscriberBlacklisting": "true",
                    "annotation": "smartphone_short",
                    "entitlement": [
                        "BBC_TV",
                        "VoD_library_17"
                    ],
                    "device": [
                        "smartphone"
                    ]
                },
                "preferenceEngines": [
                    {
                        "name": "prefTv",
                        "contribution": 10,
                        "inputs": [
                            "SrcGrp_TV1",
                            "SrcGrp_TV2"
                        ],
                        "profileBlacklisting": "true",
                        "profile": "prof1"
                    },
                    {
                        "name": "prefVod",
                        "contribution": 10,
                        "inputs": [
                            "SrcGrp_VoD"
                        ],
                        "profileBlacklisting": "true",
                        "profile": "prof2",
                        "weightFunctions": [
                            {
                                "trigger": "remainingAvailability",
                                "type": "multiplicative",
                                "nodes": [
                                    {
                                        "triggerValue": "86400",
                                        "weight": "2.0"
                                    },
                                    {
                                        "triggerValue": "172800",
                                        "weight": 1.5
                                    },
                                    {
                                        "triggerValue": "432000",
```

```
                                    "weight": 1.2
                                },
                                {
                                    "triggerValue": "864000",
                                    "weight": 1
                                }
                            ]
                        }
                    ]
                }
            ],
            "socialEngines": [
                {
                    "name": "social",
                    "contribution": 10,
                    "inputs": [
                        "SrcGrp_TV1",
                        "SrcGrp_TV2",
                        "SrcGrp_VoD"
                    ]
                }
            ],
            "moderatedEngines": [
                {
                    "name": "editorial",
                    "contribution": 10,
                    "inputs": [
                        "srcGrp_VoD"
                    ],
                    "moderatedList": "moderatedListId17"
                }
            ],
            "statisticalEngines": [
                {
                    "name": "hottest",
                    "contribution": 10,
                    "inputs": [
                        "srcGrp_TV1",
                        "srcGrp_TV2",
                        "srcGrp_VoD"
                    ],
                    "statisticalList": "hottest"
                }
            ]
        }
    }
  ]
}
```


## *3.17 Recommendation Management*

### 3.17.1 **Retrieve recommendations for a specific context**

This is the main API call of Ncanto actually returning recommendations. All parameters are encoded into the parameter section of the request.
The pre-defined and ad-hoc contexts are combined to one final context at run time.

Note that the contexts do not specify subscriber, profile or asset IDs directly but always refer to such IDs via ID references. The actual IDs to be used have to be provided in the recommendation request.
For troubleshooting purposes an optional trace information block is available. In contrast to the explanation, which is intended to outline why a specific asset is recommended, the trace shall not be sent to the client devices.
Subscriber ID, although not required by some engine types, shall always be submitted by the client. If the subscriber ID is not available, Ncanto is not able to populate the collaborative matrix and to perform quality monitoring. Starting from R4.5 a fallback to a default subscriber ID is attempted if the subscriber ID provided is not available in the database.
Please note that Ncanto supports multiple recommendation engines per context and hence multi-engine recommendation requests. In request based business models a single recommendation request involving N engines is counted as N requests.
The recommendation API supports pagination of results via the paginationOffset and paginationLimit query parameters. Note that the numberOfItems specification of the context is still observed, but only a subset of the result is returned.
The preference profiles can be addressed by the combination of a subscriber ID and a profile name instead of a unique profile ID.
Starting from R4.5 arbitrary additional query parameters may be added to the recommendation request. Those are available as criteria for conditional context rules. E.g. a query parameter "deviceType"="tablet" couls be used to define context rules only applicable to tablets.

| **GET** | `/recommendations?context=<contextId>&subscriber=<subId>&<reference>=<profileId>&<reference>=<profileName>&<reference>=<assetId>&...&trace=true\|false&adHocContext=@context&query=<searchQuery>&paginationLimit=<nonNegInteger>&paginationOffset=<nonNegInteger>&<conditionParameter>=<string>&defaultSubscriber=<defaultSubId>&track=<boolean>&autocloneSubscriber=<string>` | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | `context` | Optional, type string. ID of the pre-defined context to be used. If missing, an adHocContext is mandatory. |
| | `subscriber` | Mandatory only if sorting by lineup or lineupFilter are used by the context, of if a subscriber based business model is in place. Type string. ID of the subscriber. If present, the entitlement of the subscriber will be applied. |
| | `defaultSubscriber` | Optional, type string. If present, the request will not throw a 404 exception if the subscriber Id is missing but will use the defaultSubscriber instead. |
| | `<reference>` | Optional, type string. Parameter names are defined in the context, e.g. profile IDs of preference engines of seed asset IDs of similarity engines. |
| | `trace` | Optional, type boolean. If true, the response will contain a trace message useful for debugging. |
| | `adHocContext` | Optional, type @context. If missing, a pre-defined context ID is mandatory. If both are present, the adHocContext will be merged with the pre-defined context. |
| | `query` | Mandatory if a search engine is present in the context, type string. |
| | `paginationLimit` | Optional, type positive integer. If present, only the requested number of recommendations starting from paginationOffset are returned. |
| | `paginationOffset` | Optional, type non-negative integer, default value 0. If present, the first paginationOffset recommendations will be skipped in the response. |
| | `<conditionParameter>` | Optional, type string. Additional parameters which may be used as rule conditions in the context. |
| | `track` | Optional, type boolean. Supported from R4.8.6. If present, the string specified by parameter "query" will be tracked. See Section 13.23 for details. |

| | | |
|---|---|---|
| | autocloneSubscriber | Optional, supported from R5.6. type string (subscriber ID). If present Ncanto won't throw a 404 exception if the subscriber to be used (query parameter "subscriber" is not present, but will clone the subscriber using the autocloneSubscriber ID as template. See Section 3.8.37 for more details. autocloneSubscriber is not compatible with defaultSubscriber, a 400 exception is thrown if both are provided. |
| ***body*** | - | |
| ***result*** | **200 OK** | |
| | **400 BAD REQUEST** | Request is syntactically incorrect |
| | **404 NOT FOUND** | No active asset store present, one of the referenced IDs or names not found. Note that if a non-existing adpoolId is specified, the request will still return recommendations without ads. The response code will be 200 in such case.. Further details are given in the response. |
| | **409 CONFLICT** | One of the profileNames given cannot be resolved into a unique profileId |
| ***response*** | `{"recommendations" : [@rec], "trace": @trace, "attributeValues": [@attributeValues], "contextNames": @contextNames }` | |

Because of a length limitation of request URLs to 7000 characters, an equivalent POST request is offered, where the ad-hoc context (which is typically the longest part of the parameters section) is moved into the request body.

| **POST** | `/recommendations?context=<contextId>&subscriber=<subId>&<reference>=<profileId>&<reference>=<profileName>&<reference>=<assetId>&...&trace=true\|false&query=<searchQuery>&paginationLimit=<nonNegInteger>&paginationOffset=<nonNegInteger>&<conditionParameter>=<string>&&defaultSubscriber=<defaultSubId>&track=<boolean>` | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | context | Optional, type string. ID of the pre-defined context to be used. If missing, an adHocContext is mandatory. |
| | subscriber | Mandatory only if sorting by lineup or lineupFilter are used by the context, of if a subscriber based business model is in place. Type string. ID of the subscriber. If present, the entitlement of the subscriber will be applied. |
| | defaultSubscriber | Optional, type string. If present, the request will not throw a 404 exception if the subscriber Id is missing but will use the defaultSubscriber instead. |
| | <reference> | Optional, type string. Parameter names are defined in the context, e.g. profile IDs of preference engines of seed asset IDs of similarity engines. |
| | trace | Optional, type boolean. If true, the response will contain a trace message useful for debugging. |
| | query | Mandatory if a search engine is present in the context, type string. |
| | paginationLimit | Optional, type positive integer. If present, only the requested number of recommendations starting from paginationOffset are returned. |
| | paginationOffset | Optional, type non-negative integer, default value 0. If present, the first paginationOffset recommendations will be skipped in the response. |
| | <conditionParameter> | Optional, type string. Additional parameters which may be used as rule conditions in the context. |
| | track | Optional, type boolean. Supported from R4.8.6. If present, the string specified by parameter "query" will be tracked. See Section 13.23 for details. |
| ***body*** | **@context** mandatory if no pre-defined context is used, the adHocContext to be used. | |
| ***result*** | **200 OK** | |
| | **400 BAD REQUEST** | Request is syntactically incorrect |
| | **404 NOT FOUND** | No active store present, one of the referenced IDs or names not found. Note that if a non-existing adpoolId is specified, the request will still |

| | | |
|---|---|---|
| | | return recommendations without ads. The response code will be 200 in such case. Further details are given in the response. |
| | **409 CONFLICT** | One of the profileNames given cannot be resolved into a unique profileId |
| ***response*** | `{"recommendations":[@rec], "trace": @trace, "attributeValues": [@attributeValues], "contextNames": @contextNames}` | |

```
@rec =
  {"asset": @asset  ,                  assetId and annotation of the recommended asset,
                                         based on the annotationId defined in the context.
                                         Starting from R 3.3 availabilityStartTime and
                                         availabilityEndTime are converted to the
                                         subscriber's time zone (if known)
   "groupId": string,                  ID of the recommendation group the asset belongs
                                         to. groupIds are consecutive integer numbers 0,
                                         1, 2, ... generated by Ncanto to
                                         indicate assets which represent multiple
                                         occurrences of the same program or episodes of
                                         the same series
   "groupSize": int,                   number of recommendations in the group, i.e. having
                                         the same groupId as this recommendation.
                                         Supported from R5.3.
   "score": decimal,                   recommendation score of the asset
   "businessScore": decimal,           business score of the asset
   "rating": decimal,                  previous rating of the asset, only used by assets
                                         recommended by preference engines
   "recordingSuggestion": boolean      flag indicating that Ncanto suggests to
                                         automatically record the asset
   "explanations": [@explanation,...], list of explanations why the asset has been
                                         recommended
   "adPlacements": [@adPlacement],     list of targeted ads
   "highlights": {                     supported from R4.1 for search results only
     <attributeName>: [                  name of the asset attribute the search term was
                                         found in dot  notation
       <fragment>,                       asset attribute value containing the search term
       ...                               additional occurrences of the search term in the
                                         same attribute
     ],
     ...                                 additional occurrences of the search term in
                                         other attributes
   },
   "stackedRecommendations": [@rec]    optional, additional recommendations of the same
                                         group if the global context parameter
                                         stackedGroups==true.
  }


@trace =
{"trace": json,                       verbose debugging information in JSON format
}

@attributeValues =
{"attribute": <attribute_name>,       name of the attribute in dot notation
 "values":[
    {"value": <value>,                value of the attribute
     "count": integer                 number of assets carrying the attribute value
    }
```

```
  ]
}


@explanation =
{"originator": string,                identifies the source of the explanation (e.g.
                                        engine type, engine name, producer,...)
 "level": int,                        indicates the level of detail of the explanation. 0
                                        stands for the most generic, highest level
                                        explanation (e.g. "Recommended by Social
                                        Engine", increasing level numbers indicate more
                                        detailed information. Deprecated from R4.6.
 "type": enum,                        replaces the level information from R4.6, one of
                                        "engine", "reference", "feature". "engine"
                                        corresponds to level 0. "reference" corresponds #
                                        to level 2 and contains a reference to an object
                                        used by the engine, e.g. the name of the profile
                                        used by the preference engine or the title of the
                                        seed asset used by the similarity engine.
                                        "feature" explanations are only returned by
                                        preference and similarity engines and contain
                                        detailed information about the asset features
                                        which led to the explanation.
 "message": string                    the explanation itself, e.g. "Recommended because
                                        it belongs to the most watched assets"
}

@adPlacement =
{"asset": @asset,                     metadata of the placed ad asset, filtered according
                                        to the annotation template used
 "location": enum,                    placement location, one of "pre-roll", "post-roll",
                                        "interstitial", "generic", currently only
                                        "generic" supported
 "score": decimal,                    score of the ad
 "businessScore": decimal,            not supported in the current release
 "explanations": [@explanation]       explanation why the ad has been placed, not
                                        supported in the current release
}

@contextNames =
{
 <langCode>: <contextName>            The context names as defined in the context, with
                                        resolved variables and filtered to the language
                                        specified in explanationConfig (global context).
                                        If explanationConfig does not specify a language,
                                        the first of the metadata languages of the
                                        explanationConfig is used. If the specified
                                        language is not available in contextNames, an
                                        empty object is returned. If explanationConfig is
                                        not specified, all languages are returned.
}
```

## *3.18 Ad Targeting API*

This request is used to place ads into one or a list of assets. The response is equivalent to the response of a recommendations request. Adding the subscriber ID to the request is optional, if not provided, Ncanto is not able to perform quality monitoring. Note that ad interleaving is intended to be used for video assets only.

| POST | recommendations/adInterleaver?subscriber=<subId> | |
|---|---|---|
| *body* | {"interleavingParameters": @interleavingParameters,<br>"assetIds": [<assetId>,...]<br>} | required<br>required |
| *result* | **200 OK** | |
| | **400 BAD REQUEST** | Request is syntactically incorrect, |
| | **404 NOT FOUND** | No active store present, one of the referenced IDs not found or context evaluation resulted in incomplete configuration. Further details are given in the response. |
| *response* | {"recommendations":[@rec], "trace": @trace} | |

**@interleavingParameters =**

```
{"algorithm": enum,                    required, one of "similarity" or "preference"
 "profileId": string512,               optional, profileId to be used for "preference"
 "adFrequency": decimal,               required, range [0, 1]
 "adPools": [string],                  required, list of adPool IDs
 "assetTypes": [string],               optional, defines the asset types which, in additon
                                         to "ad" are considered as advertising and may be
                                         interleaved. For compatibility to earlier
                                         releases, assetType="ad" is always used by the ad
                                         engines. Supported from R4.0
 "windowOfAvailabilityStart": timestamp,     optional, default –infinity, defines the
                                         absolute time window in which results shall be
                                         available. The absolute window is overruled by
                                         relativeWindowOfAvailabilityStart if both are
                                         present.
 "windowOfAvailabilityEnd": timestamp, optional, default +infinity, defines the
                                         absolute time window in which results shall be
                                         available. The absolute window is overruled by
                                         relativeWindowOfAvailabilityEnd if both are
                                         present.
 "relativeWindowOfAvailabilityStart":  optional, from R4.0, defines the relative time
                                         window in which recommendations shall be
                                         available. Overrules the absolute time window if
                                         both are specified.
    {"origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                         "startOfWeek" (R5.0)
     "offset": <duration>              required, offset in XSD duration format
    },
"relativeWindowOfAvailabilityEnd":     optional, from R4.0, defines the relative time
                                         window in which recommendations shall be
                                         available. Overrules the absolute time window if
                                         both are specified.
    {"origin": <enum>,                 required, one of "currentTime", "startOfDay",
                                         "startOfWeek" (R5.0)
     "offset": <duration>              required, offset in XSD duration format
    },
 "country": string,                    optional, used to select the desired age
                                         rating
 "language": langCode,                 optional, determines the language of annotations
                                         and explanations
 "annotation": string,                 optional, annotationId to be used for
                                         recommendations (all asset types)
 "annotationPerAssetType": {           optional, supported from R4.0, individual
   <assetType>: string,                  annotations for recommendations per asset type.
   ...                                   If present, the global annotation is selectively
  }                                      overruled for the corresponding asset types.
 "adAnnotation": string,               optional, annotationId to be used for ads
```

```
 "entitlement":[string,...],          optional, list of entitlements, not yet
                                        supported
 "device":[string,...],               optional, list of device identifiers, not yet
                                        supported
}
```

## *3.19 Panel Management*

See Section 2.3.13 for a description of the panel concept. Panels are used to specify a group of recommendation containers, corresponding to a group of contexts, which have to be populated by recommendations with the help of a single API call.

### 3.19.1 Create or Update a Panel

| **PUT** | /panel/<panelId> | |
|---|---|---|
| ***body*** | @panel | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |

**@panel =**

```
{
 "panelId": string,                   optional, unique ID of the panel, ignored by PUT
                                        requests
 "customProperties": noDotsJson,      optional, arbitrary json object of max. 2kB size
 "presentationProperties": noDotsJson, optional, supported from R5.6. If present, the
                                        contents are copied into the recommendation
                                        response. Intended to communicate presentation
                                        and layout parameters to the client.
 "containers": [                      at least one element required, lists the containers
                                        of the panel. The order of the containers defines
                                        the duplicate removal priority, the
                                        recommendations of a container are excluded from
                                        following containers in the list.
  {
   "id": string,                      required, unique ID of the container in the panel
   "size": integer,                   required, positive integer, number of
                                        recommendations in the container
   "contextId": string,               required, ID of the context used to populate
                                        the container
   "customProperties": noDotsJson,    optional, arbitrary JSON object of max 2kB size.
   "presentationProperties": noDotsJson,optional, supported from R5.6. If present, the
                                        contents are copied into the recommendation
                                        response. Intended to communicate presentation
                                        and layout parameters to the client.
   "slots": [                         optional, the slots of the container. If empty or
                                        missing the whole container will be populated by
                                        all engines of the context.
    {
     "size": integer,                 required, positive integer, the number of
                                        recommendation positions occupied by the slot. If
                                        stackedGroups==true in the context used, each
                                        recommendation position will be populated by one
                                        stack. Otherwise a recommendation position will
                                        be populated by a single recommended asset.
     "engineNames": [string]          required, at least one element required, list of
                                        preferred engine + fallback engines used to
```

```
                                    populate the slot. Note that blending preference,
                                    shared profile preference, blending filter
                                    preference, and multi-seed similarity engines are
                                    not allowed.
    }
   ]
  }
 ]
}
```

### 3.19.1 Retrieve one or multiple Panel(s)

| **GET** | `/panel/<panelId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **400 BAD REQUEST** | Request body is syntactically incorrect |
| | **404 NOT FOUND** | panelId not found |
| ***response*** | `@panel` | |

| **GET** | `/panel?count=<no_of_panels>&startAfter=<panelId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[@panel]`<br>Panels are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

To retrieve panel IDs only (instead of full panels) the `idsOnly` parameter has to be added to the request:

| **GET** | `/panel?idsOnly=true&count=<no_of_panels>&startAfter=<panelId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | `[<panelId>]`<br>Panels IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.19.2 Delete a Panel

| **DELETE** | `/panel/<panelId>` | |
|---|---|---|
| ***result*** | **204 OK** | Request successful |

## *3.20 Multi-context (panel) recommendations*

The panel recommendations API is used to retrieve recommendations for a whole panel in a single call. In contrast to the recommendations API only a GET request is supported and the contexts used by the panel have to be pre-defined (ad-hoc contexts are not supported).

The panel recommendation service internally calls the recommendation service for each container of the panel.
In case of containers without slot specification, the standard recommendation service as described in Section 3.17 is called. In case of containers with slot specification the number of items and the engine contributions specified in the context of the container are ignored and the amount of recommendations required from each engine is determined from the slot sizes and the engine names. In addition, if slots are defined, the sorting criteria defined in the context are applied on a per engine basis separately.

| **GET** | /panelRecommendations?panelId=<panelId>&subscriber=<subId>&<referenceKey>=<referenceValue>&query=<searchQuery>&defaultSubscriber=<defaultSubId>&track=<boolean> | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | panelId | Required, type string. ID of the panel to be used. |
| | subscriber | Mandatory only if sorting by lineup or lineupFilter are used by the context, of if a subscriber based business model is in place. Type string. ID of the subscriber. If present, the entitlement of the subscriber will be applied. |
| | defaultSubscriber | Optional, type string. If present, the request will not throw a 404 exception if the subscriber Id is missing but will use the defaultSubscriber instead. |
| | <referenceKey> | Optional, type string. Parameter names are defined in the context, e.g. profile IDs of preference engines of seed asset IDs of similarity engines. |
| | query | Mandatory if a search engine is present in the context, type string. |
| | track | Optional, type boolean. Supported from R4.8.6. If present, the string specified by parameter "query" will be tracked. See Section 13.23 for details. |
| ***body*** | - | |
| ***result*** | **200 OK** | |
| | **400 BAD REQUEST** | Request is syntactically incorrect |
| | **404 NOT FOUND** | No active asset store present, one of the referenced IDs or names not found. |
| | **409 CONFLICT** | One of the profileNames given cannot be resolved into a unique profileId |
| ***response*** | {"panelRecommendations": [@panelRecommendations], "presentationProperties": noDotsJson} | |

with @panelRecommendations being the relevant subset of the recommendation response (cf. Section 3.17) and containing the recommendations of one container of the panel:

```
@panelRecommendations =
{
 "contextNames": {
  <langCode>: <string>
 },
 "presentationProperties": noDotsJson,        contains the presentationProperties of the
                                                container
 "recommendations": [@rec]
}
```

See Section 3.17 for the definition of @rec.

## *3.21 Operations API*

### 3.21.1 **Retrieve product version**

This request returns information about the currently deployed Ncanto version.

| **GET** | /version | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | {"product": string, "version": string, "revision": string, "revisionDate": string} | |

### 3.21.2 **Log Subscriber Interaction**

This request is used to communicate subscriber interactions to Ncanto. Any type of interaction is supported, but there is a list of standard interactions which are evaluated by Ncanto to improve profiles and to populate e.g. the matrix of the collaborative engine. Customer specific non-standard interactions can be included in the calculation of profile and collaborative data on request.
Ncanto takes into account subscriber interactions according to the following algorithm:

- all interactions with a rating<>NULL
- all interactions listed in Table 8

Please note that Ncanto supports the transfer of multiple interactions in one single request. In request based business models a log request containing N interactions is counted as N requests.

| **POST** | `/log` | |
|---|---|---|
| ***body*** | `[@interaction,...]` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD REQUEST** | Malformed request |
| ***response*** | | |

```
@interaction =
{"timestamp": timestamp,                optional, automatically generated if not
                                          present in the request
 "subscriber": string,                  optional, unique ID of the subscriber
                                          performing the interaction
 "profile": string,                     optional, unique ID of the profile the
                                          interaction is associated with
 "assetId": string512,                  optional, unique ID of the asset the
                                          interaction is associated with
 "positionSeconds": int,                optional, position of the interaction in
                                          seconds from the beginning of the asset
 "interaction": string,                 required, description of the interaction
 "contextId": <string>,                 optional, if provided, the given contextId
                                          will be logged in the interaction log.
"comment": string,                      optional
 "ratingEquivalent": decimal,           optional, range [-1:+1], the implicit
                                          rating the interaction corresponds to
 "setting": {                           optional, from R4.2, viewing setting of the rating
  "timeOfDayBin": <string>,             optional, e.g. "morning", "evening"
  "timeOfWeekBin": <string>,            optional, e.g. "weekday", "weekend"
  "device": <string>,                   optional, the device used
  "locationBin": <string>,              optional, e.g. "home", "work"
  "moodBin": <string>,                  optional, e.g. "happy", "sad"
  "activity": <string>,                 optional, e.g. "traveling", "relaxing"
  "autoTimeOfDayBin": {                 optional, from R5.6, ignored if timeOfDayBin
                                          is set, defines the time of day bin values to be
                                          set automatically depending on current time
    <offsetFrom>: <binName>             at least one required, offset wrt beginning of the
                                          day (xsdDuration) bin name (string) pair
  },
  "autoTimeOfWeekBin: {                 optional, from R5.6, ignored if timeOfWeekBin is
                                          set, defines the time of week bin values to be
                                          set automatically depending on current time
    <offsetFrom>: <binName>             at least one required, offset wrt beginning of the
                                          week (xsdDuration) bin name (string) pair
  }
 },
 "revenue": <decimal>,                      optional, revenue of the interaction, from
                                              R5.6
```

```
 "currency": <string>                          optional, currency of the revenue of the
                                                 Interaction, supported from R5.6
}
```

The following interaction types are used for implicit ratings:

| Interaction | Description |
|---|---|
| play | The asset has been played starting at <position> |
| stop | Playing the asset has been stopped at <position> |
| select | The asset has been selected by the subscriber (e.g. purchased, rented, scheduled for recording, added to favorites, etc.) |
| deselect | The asset has been deselected by the subscriber (e.g. removed from favorites, deleted, purchase canceled, etc.) |

Table 8: List of reserved interaction types

Note that "select" and "deselect" interactions as well as all interactions with "ratingEquivalent" set are being used by the collaborative engine and for the "bestRated" lists of the statistical engine.

The following interaction types are reserved for internal use:

| Interaction | Description |
|---|---|
| rate | This interaction type is used for log entries automatically generated by the "Rate an Asset" API request. |
| requestRecommendations | This interaction type is used for log entries automatically generated by the Recommendation API request. |
| requestProgramGuide | This interaction type is used for log entries automatically generated by a retrieve assets by filter request. |
| calibrateProfile | This interaction type is used for log entries automatically generated when a preference profile is calibrated by Ncanto. |
| requestQueries | This interaction type is used for log entries automatically generated by query builder requests. |

Table 9: List of interaction types for internal use

The following interaction types are used to populate statistical lists and to calculate community asset attributes:

| Interaction | Description |
|---|---|
| select/deselect | Used to calculate asset attributes communityRating, communityTrendScore and to populate bestRated, hottest/trending, and mostPopular statistical lists. |
| Any interaction type with ratingEquivalent set | Used to calculate asset attributes communityRating, communityTrendScore and to populate bestRated, hottest/trending, and mostPopular statistical lists. |
| share | Used to calculate communityShareCount and to populate mostShared statistical lists (R4.4). |

Table 10: List of interaction types used to calculate community asset attributes and to populate statistical lists

The following interaction types are used for frequency capping:

| Interaction | Description |
|---|---|

| | |
|---|---|
| recommendationImpression | **Informs Ncanto that the assetId was actually shown to the subscriber as recommendation. Used by the frequency capping feature to avoid repeating the same recommendation to the same subscriber several times.** |

Table 11: List of interaction types used to calculate community asset attributes and to populate statistical lists

### 3.21.3 Retrieve basic analytics data

This request returns the number of subscribers and the number of preference profiles stored in the system. Note that the counts are not calculated in real time but once per day in the background only. The timestamp in the response indicates when the numbers have been obtained.

| **GET** | /analytics | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | {"numberOfSubscribers": int,<br>"numberOfProfiles": int,<br>"numberOfAssetStores": int, from R4.0<br>"totalNumberOfAssetsInStores": int, from R4.0<br>"timestamp": xsddatetime} | |

## *3.22 Query Builder API*

In order to find related content in public video sources, Ncanto provides queries to be sent to the APIs of such sources. Initially YouTube queries are supported. In compliance with the terms of use of YouTube Ncanto suggests YouTube API queries which have to be sent to the YouTube API by the client. The query builder accepts asset IDs and social topic IDs as input. Note that the query builder API is intended to be used with video seed assets only.

| **GET** | /queries?target=<targetSystem>&targetVersion=<targetSystemVersion>&seed=<assetId1>&seed=<assetid2>&topic=<topic1>&topic=<topic2>&subscriber=<subId>&language=<langCode><br><br><targetSystem>: required, currently only "YouTube" supported<br><targetSystemVersion>: required, for YouTube "2.0" or "3.0" are supported<br><assetId>: optional, ID of the asset to find similar content to<br><topicId>: optional, ID of the topic to find similar content to<br><subId>: optional, subscriber ID, used only for logging and reporting purposes<br><langCode>: optional, supported from R4.3, if present the metadata in the specified language will be used to generate the queries | |
|---|---|---|
| ***result*** | **200 OK** | |
| | **400 BAD REQUEST** | Request is syntactically incorrect |
| | **404 NOT FOUND** | One of the referenced IDs not found |
| ***response*** | [@externalQuery] | |

```
@externalQuery =
{"httpMethod": enum,                          optional, one of GET, POST, default value
                                                GET
 "httpHeaders": [@keyValuePair],              optional, list of http headers to be used
 "queryParameters": [@keyValuePair],          optional, list of query parameters to be
                                                used
 "requestBody": string,                       optional, body of the request
 "broadness": integer,                        optional, if multiple external queries are
```

```
                                          provided, the client shall first try the
                                          narrowest query (the one with lowest
                                          broadness value) and, if the number of
                                          results is not sufficient, continue with
                                          broader queries.
 "explanation": string                   optional, explanation for which asset/topic
                                          the query was built for
}

@keyValuePair =
{"key": string,                          required, query parameter or http header
                                          name
 "value": string                         required, query parameter or http header
                                          value
}
```

## *3.23 Frequently used search terms*

From R4.8.6 Ncanto supports the tracking of search terms and the retrieval of frequently used search terms for a specified time interval.
The tracking of search terms is available for recommendation requests (if a search engine is part of the context) as well as for retrieve assets by filter requests (e.g. for use cases where the client combines the search phrase suggestions feature with a filter request).

### 3.23.1 **Create or update search terms blacklist**

Undesired terms may be excluded from search terms tracking via the blacklisting feature. Search terms containing any of the blacklisted words are not tracked, unless they include a whitelisted phrase. Black- and whitelisted terms and phrases require an exact match to be effective.

| **PUT** | `/search-terms/config` | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | - | |
| ***body*** | `{`<br>`  "blacklist": [<string>],`<br>`  "whitelist": [<string>]`<br>`}` | |
| ***result*** | **204 NO RESPONSE** | Request successful |
| | **400 BAD_REQUEST** | Request nody is syntactically incorrect |
| ***response*** | - | |

### 3.23.2 **Retrieve search terms blacklist**

| **GET** | `/search-terms/config` | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | - | |
| ***body*** | - | |
| ***result*** | **200 NO RESPONSE** | Request successful |
| | **404 NOT FOUND** | No search terms configuration was found |
| ***response*** | `{`<br>`  "blacklist": [<string>],` | |

| | |
|---|---|
| | <pre>  "whitelist": [<string>]<br>}</pre> |

### 3.23.3 Retrieve frequently used search terms

| **GET** | /search-terms?limit=<positiveInteger>&timeWindow=<duration>&actualCounts=<boolean>&startsWith=<string> | |
|---|---|---|
| ***path parameters*** | - | |
| ***query parameters*** | limit | Type: positive integer, defines the number of most frequent search terms to be returned |
| | timeWindow | Type: positive duration, defines the time window from which the popular search terms should be collected |
| | actualCounts | Type: boolean. If true, Ncanto checks how many assets matching the search term are available in the active store at the time of querying. |
| ***body*** | - | |
| ***result*** | **200 OK** | Request successful |
| | **400 BAD_REQUEST** | Request parameters are syntactically incorrect |
| ***response*** | <pre>{<br>"totalCount": <integer>,  the total count of tracked search<br>  terms for the time window specified<br>"searchTerms": [<br>{<br>"searchTerm":<string>,  the search term (lowercased)<br>"count": <integer>,  the number of occurrences of the<br>  search term within the time window<br>"actualCount": <integer>  the number of assets in the active<br>  store matching the search term<br>}<br>]<br>}</pre> Note that actualCounts is omitted from the response if the query parameter is missing or set to false. | |

## *3.24 Dump*

Ncanto provides a generic storage facility for JSON objects called "dump". The dump is typically used by clients to store global configuration data. Ncanto is not using the dump internally.

### 3.24.1 Create or Update an dump object

This request is used to create a new or to update an existing dump entry.

| **PUT** | /dump/<objectId> | |
|---|---|---|
| ***body*** | JSON object | |
| ***result*** | **204 NO_RESPONSE** | Request successful |
| | **400 BAD_REQUEST** | Request body is syntactically incorrect |

### 3.24.2 Retrieve dump object

This request is used to retrieve a dump entry.

| **GET** | /dump/<objectId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | objectId does not exist |

| | |
|---|---|
| ***response*** | `JSON object` |

### 3.24.3 Retrieve all dump object IDs

This request is used to retrieve the IDs of all dump objects. Pagination is supported via the count and startAfter query parameters . Supported from R4.1

| **GET** | `/dump?count=<no_of_objects>&startAfter=<objectId>` | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | objectId of startAfter parameter not found |
| ***response*** | `[<objectId>]` | |

### 3.24.4 Delete a dump object

This request is used to delete one dump entry.

| **DELETE** | `/dump/<objectId>` | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | entId successfully deleted or not found |

## *3.25 Dynamic Subscriber Groups*

Ncanto supports conditional context rules based on the subscriberGroups property of the subscribers. Such conditional rules are typically used for A/B testing of two configurations. Which groups a subscriber belongs to can either be hard coded into the subscriber object ("subscriberGroups" property) or assigned dynamically at the time the subscriber uses Ncanto via the dynamic subscriber groups.

There are three different criteria which define a dynamic subscriber group, all three have to be matched:

1. subscriberId is one of the IDs in the list. This feature is typically used to confine the subscriber group to a special subset of the whole subscriber base, e.g. to the most active subscribers.
2. The subscriber matches a generic filter, e.g. customProperties.property1=value1
3. The hashed subscriber ID divided by the modulus results in one of the specified remainders. The modulo filter is used to split the subscriber base into a number of random groups.

For context evaluation and logging the dynamic subscriber groups are used as if they were static subscriber groups. There can be multiple dynamic subscriber group definitions stored in Ncanto, but maximum one of them may be active at any time.

### 3.25.1 Update Dynamic Subscriber Group Definitions

| **PUT** | `/dynamicSubscriberGroups/<dynamicSubscriberGroupsId>` | |
|---|---|---|
| ***body*** | `@dynamicSubscriberGroups` | |
| ***result*** | **204 NO_RESPONSE** | Request successful |
| | **400 BAD_REQUEST** | Request body is syntactically incorrect |

```
@dynamicSubscriberGroups =
{
 "dynamicSubscriberGroupId": string, unique ID of the group definition rule set
 "active": boolean,              specifies if the groups are active or not
```

```
"groups": [
 <groupName>: {
  "subscriberIds": [<subId>],     for candidate subscriberId lists coming from
                                   external sources, e.g. from Yaveo
  "filter": @genericFilter,       additional filtering for subscriber
                                   attributes
  "moduloFilter": {
   "modulus": int,               number of equal segments the subscribers
                                   should be split to
   "remainders": [int]           list of remainders of the modulo calculation
                                   which belong to the group
  }
 }
]
}
```

### 3.25.2 Retrieve Dynamic Subscriber Group Definitions

| **GET** | /dynamicSubscriberGroups/<dynamicSubscriberGroupsId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| | **404 NOT FOUND** | dynamicSubscriberGroups definition does not exist |
| ***response*** | @dynamicSubscriberGroups | |

| **GET** | /dynamicSubscriberGroups?count=<no_of_results>&startAfter=<dynamicSubscriberGroupsId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | [@dynamicSubscriberGroups]<br>Results are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

To retrieve dynamicSubscriberGroupIds only (instead of full objects) the `idsOnly` parameter has to be added to the request:

| **GET** | /dynamicSubscriberGroups?idsOnly=true&count=<no_of_results>&startAfter=<dynamicSubscriberGroupsId> | |
|---|---|---|
| ***result*** | **200 OK** | Request successful |
| ***response*** | [<dynamicSubscriberGroupsId>]<br>IDs are returned in a deterministic but unsorted way. Note that for performance reasons count is limited to 1000, even if count is not specified, maximum 1000 are returned. | |

### 3.25.3 Delete Dynamic Subscriber Group Definitions

| **DELETE** | /dynamicSubscriberGroups/<dynamicSubscriberGroupsId> | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | subscriber group definition successfully deleted or not found |

### 3.25.1 Activate and Deactivate Dynamuc Subscriber Group Definitions

| **POST** | /dynamicSubscriberGroups/<dynamicSubscriberGroupsId>/activate | |
|---|---|---|
| ***result*** | **204 NO RESPONSE** | subscriber group definition successfully activated, the previously active set of groups deactivated |
| | **404 NOT FOUND** | <dynamicSubscriberGroupsId> not found |

| POST | /dynamicSubscriberGroups/<dynamicSubscriberGroupsId>/deactivate | |
|---|---|---|
| *result* | **204 NO RESPONSE** | subscriber group definition successfully deactivated |
| | **404 NOT FOUND** | <dynamicSubscriberGroupsId> not found |

# 4 Appendix: Metadata Attributes

The following table describes the metadata attributes supported by Ncanto in detail. Mandatory attributes are indicated by grey background.

| Attribute | Description |
|---|---|
| "assetId": string512 | The asset ID is the unique ID identifying an asset in Ncanto's database. |
| "assetGroups": [string] | The assetGroups attribute allows to provide search phrase suggestions from a subset of the assets only. Note that search phrase suggestions don't support the usual filtering features. |
| "sourceId": string | The source ID defines which source an asset belongs to. Source IDs may denote different broadcast TV channels (BBC1, BBC2, ...) or VoD libraries (VoD lib 1, VoD lib 2,...), they represent the first hierarchical level to organize and group assets. |
| "sourceGroupId": string | Source groups represent the second hierarchical level to group assets. They may e.g. be used to distinguish between different content providers (BBC = BBC1, BBC2, ...; DisneyLibs = VoDlib1, Vodlib2). The use of Source Groups is optional, but the input of a recommendation engine, e.g. the candidate assets which are considered by the engine, can only be defined at source group level. The usage of sourceGroupId is deprecated from R4.1, the attribute is replaced by sourceGroupIds. |
| "sourceGroupIds": [string] | List of source groups the asset belongs to. Example ["BBC_TV_channels", "English_language_channels"] |
| "sourceLogoUrl": string | URL of the logo of the source, e.g. of the TV station. Only used as annotation. |
| "availabilityStartTime": timestamp | Beginning of the availability window of an asset. For TV assets this will typically correspond to the start time of the broadcast. For VoD assets, it will correspond to the beginning of the license window. It is preferred that the start time is available including time zone information. |
| "availabilityEndTime": timestamp | End of the availability window of an asset. For TV assets this will typically correspond to the end time of the broadcast. For VoD assets, it will correspond to the end of the license window. It is preferred that the end time is available including time zone information. |
| "durationSeconds": int | Duration of the asset in seconds. |
| "titles":{<lang>: <title>,<br><lang>: <title><br>} | The title of the asset in different languages. Ncanto does not dictate the use of standardized language representations (like ISO 639), any |

| | |
|---|---|
| | classification which can be mapped on a string is allowed. At least one title is required. |
| "programType": string | Program type is a high level (typically broader than genreBroad) classification of assets, e.g. into types movie / documentary / show / sports / news. If the program type classification uses IDs, the mapping to human readable program types shall be available in uiProgramType. |
| "assetType": string | Defines the type of the asset, e.g. TV, VoD, OTT, advertising. By default, recommendation engines use assets with assetType!="ad", the Ad Engine and the Ad Interleaver only work with assets of type="ad". The used asset types are configurable by engine. |
| "assetFamilies": [string] | Supported from R5.4, assetFamilies are used in addition to assetType where a multi-valued attribute is required (e.g. for an asset that is available in catch-up as well as in VoD). |
| "programId": string | Assets with the same programId represent a single program, i.e. multiple airings of the same TV show or different versions (SD, HD, ...) of the same on demand asset. programId is used for blacklisting. |
| "genreFine": list of strings | Fine granularity genre classification. Multiple fine genres per asset are allowed but the order is not taken into account. If genreFine is provided, genreMedium and genreBroad are mandatory, too. Example: genreFine = 1.3.4, genreMedium = 1.3, genreBroad = 1. If the genre classification uses IDs, the mapping to human-readable genre names shall be available in uiGenres. |
| "genreMedium": list of strings | Medium granularity genre classification. Multiple medium genres per asset are allowed but the order is not taken into account. Using a rich gerneMedium classification is highly recommended as it has a big impact on recommendation quality and consistency. If genreMedium is provided, genreBroad is mandatory too. If the genre classification uses IDs, the mapping to human-readable genre names shall be available in uiGenres. |
| "genreBroad": list of strings | Broad granularity genre classification. Multiple broad genres per asset are allowed but the order is not taken into account. Using a rich gerneBroad classification is highly recommended as it has a big impact on recommendation quality and consistency. If the genre classification uses IDs, the mapping to human-readable genre names shall be available in uiGenres. |
| "originalStartTime": timestamp | This parameter is used as annotation and represents the original start time of a broadcast TV asset. Note that startBins are calculated based on originalStartTime, hence it is essential that originalStartTime includes correct time zone information. |
| "originalEndTime": timestamp | This parameter is only used as annotation and represents the original end time of a broadcast TV asset. |
| "publicationTime": <timestamp> | Date/time of publication of the asset |

| | |
|---|---|
| "lastUpdateTime": <timestamp> | **Date/time of last modification of the asset, if missing, it is set automatically by update and ingest asset requests.** |
| "first": boolean | **Used to indicate first broadcasting of a program.** |
| "last": boolean | **Used to indicate the last broadcasting of a program.** |
| "live": string | **Used to indicate if an asset is live, live (recorded), or not live.** |
| "pay": boolean | **Indicates if an asset is a pay per view asset or not.** |
| "onDemand": boolean | **Indicates if an asset is an on demand asset or not.** |
| "seriesId": string | **All episodes of the same series are indicated by the same seriesId** |
| "season": int | **Indicates which season the episode of a series belongs to.** |
| "episode": int | **The episode number of an asset within a series.** |
| "ageCountry": map of <country>: <age> pairs | **Country specific age limits and parental ratings. <age> has to be specified as integer value.** |
| "parentalGuidance": noDotsJson | **Arbitrary information in JSON format to describe the parental guidance. May contain e.g. explanations in different languages.** |
| "tip": list of strings | **List of editorial tips of the asset, e.g. "Movie tip of the day". Deprecated from R5.2, replaced by "structTip".** |
| "structTip": [<br>{<br><string>: <int>,<br><string>: <decimal>,<br><string>: <string>,<br>...<br>},<br>...<br>], | **List of structured editorial tips of the asset, to be used as an alternative to the flat list attribute "tip" if additional information about e.g. the vendor and the type of the tip are required. Example: {"vendor": "editor1", "type":"movie tip of the day", "value": 1}. Note that integer and decimal numbers have to be provided without double quotes in order to use smaller/larger/atLeast/atMost operators.** |
| "editorialRating": list of strings | **List of editorial ratings of the asset, e.g. "humor:3 stars", "action:5 stars", "total:4 stars". Deprecated from R5.2, replaced by "structEditorialRating".** |
| "structEditorialRating": [<br>{<br><string>: <int>,<br><string>: <decimal>,<br><string>: <string>,<br>...<br>},<br>...<br>], | **List of structured editorial ratings of the asset, to be used as an alternative to the flat list attribute "editorialRating" if additional information about e.g. the vendor and the type of the tip are required. Example: {"vendor": "editor1", "type":"humor", "value": 4} Note that integer and decimal numbers have to be provided without double quotes in order to use smaller/larger/atLeast/atMost operators.** |
| "communityRating": decimal | **Average rating of the asset by the whole subscriber base. Normalized to range [0...1]** |
| "communityViewCount": integer | **Overall number of play interactions the asset received.** |
| "communityShareCount": integer | **Overall number of share interactions the asset received.** |
| "communityTrendScore": decimal | **Expresses how trending the asset is. Calculated from the ratio of the number of interactions with** |

| | |
|---|---|
| | **the asset in the last 12h period and of the previous 12h period. Normalized to range [0:1].** |
| "stereo": boolean | **Indicates if stereo audio is available.** |
| "surroundSound": boolean | **Indicates if surround audio is available.** |
| "multiChannelAudio": boolean, | Indicates the availability of multiple independent audio channels. |
| "bw": boolean | **Indicates if an asset is black & white** |
| "subtitled": boolean | **Indicates if subtitles / closed captions are available.** |
| "wideScreen": boolean | **Indicates if the aspect ratio of the video is wide (16:9, 21:9) or normal (4:3)** |
| "encoded": boolean | **Indicates if the asset requires special decoding means (smartcard, ...). Used for TV assets.** |
| "forBlind": boolean | **Indicates if the asset comes with audio commentary for visually impaired persons.** |
| "hd": boolean | **Indicates if the asset is high definition /high resolution.** |
| "interactive": boolean | **Indicates if the asset is interactive (e.g. includes a game, allows to influence the action,...)** |
| "threeD": boolean | **Indicates if the asset is 3D.** |
| "originalTitle": string | **The original title of the asset, e.g. for movies. Only used as annotation.** |
| "originalSubtitle": string | **The original subtitle of the asset, e.g. for movies or for series episodes.** |
| "seasonOriginalTitle": string | **The original title of the season of a series asset.** |
| "audioLang": list of strings | **List of audio languages supported** |
| "defaultLanguage": string | **The default metadata language of the asset. Used by language filtering as a fallback if the desired languages are not available. Supported from R4.5.** |
| "subtitleLang": list of strings | **List of subtitle (closed caption) languages supported. Available from R4.0.** |
| "themeCircle": list of strings | **Additional genre like classification of the asset. Only used as annotation.** |
| "keywords":{<lang>: [<keyword>,...],...] | **List of keywords of the asset in different languages.** |
| "persons": {<br><function>:[<br>{<br>"canonical":string,<br>"perLanguage":{<br><lang>:<name>,<br>...<br>},<br>"character": {<br><lang>:<name>,<br>...<br>},<br>] | **List of persons (cast) of the asset. The top level hierarchy inside "persons" is the function of the persons, i.e. director, actor, composer of the music, etc. For every person the canonical name (the main name of the person used by the engines or an ID) and language specific spellings are supported.**<br>**Optionally, e.g. for actors, the character the actor is playing may be added in different languages.**<br>**Example:**<br>**"persons": {**<br>**"actor": [**<br>**{**<br>**"canonical": "Clinton Eastwood",**<br>**"perLanguage": {**<br>**"eng": "Clint Eastwood",**<br>**"rus": "Клинт Иствуд"**<br>**},**<br>**"character": {**<br>**"eng": "Harry"** |

| | |
|---|---|
| | }<br>}<br>],<br>"director": [<br>{<br>"canonical": "Sergio Leone",<br>"perLanguage": {<br>"eng": "Sergio Leone",<br>"rus": "Серджио Леоне"<br>}<br>}<br>]<br>} |
| "uiFunctionNames": {<br><function_id>: {<br><lang>:<function_name>,<br>...<br>},<br>...<br>} | Used if the <function> key of the persons attribute does not contain human readable information or if function names (actor, director, ...) are required in multiple langauges. |
| "imageUrl": list of strings | List of URLs of thumbnail images of the asset. |
| "contentUrl": list of strings | List of URLs of the asset content itself, e.g. for streaming, downloading, etc. |
| "media": [<br>{<br><string>: <int>,<br><string>: <decimal>,<br><string>: <string>,<br>...<br>},<br>...<br>], | List of arbitrary media representing the assets. This generic and open attribute supports structured information about arbitrary media (images, streaming content, anything that may be linked to the asset).<br><br>The characteristics of a medium are described by a flat list of string, integer, and decimal values, e.g. "url": http://video.com, "type": "trailer", "encoding": "mp4", "width_pixels": 320, "height_pixels": 240.<br>Integer and decimal attributes support smaller/larger operators (e.g. to return all thumbnail images with a specified minimum size). |
| "leaseDurationMinutes": int | Lease duration of a pay per view asset indicating how long the subscriber can access the asset after purchasing. Unit: minutes |
| "viewCount": int | View count of a pay per view asset indicating how many times the subscriber can view the purchased asset. |
| "copyControl": list of strings | List of DRM attributes. E.g. copy control, CGMS, DTCP attributes. |
| "productionYearFirst": int | Year of production or first year of production if spread over multiple years. |
| "productionYearLast": int | Last year of production if spread over multiple years. |
| productionCountry: list of strings | List of the production countries of the asset. |
| "uiProductionCountry": {<br><lang>: [<string>],<br>...<br>} | Used if productionCountry contains IDs or if multiple languages are needed. |
| "price": map of <currency>:<price> pairs | Price of the asset in different currencies. Used as annotation only. |

| | |
|---|---|
| "priceBin": one of 0="free", 1="cheap", 2="inexpensive", 3="not too expensive", 4="premium", 5="expensive" | Used to understand preference of high or low price items. The mapping of price to priceBin is arbitrary and shall reflect local perception of price levels. In order to support weight functions the price bin is encoded in integer format. |
| "profitability": decimal | Profitability indicator used by weight functions to calculate business scores. The scale is arbitrary and defined by the service provider, the profitability parameter is not used by Ncanto except to calculate business scores. |
| "targetDemographic" list of strings | An arbitrary list of preferred demographic characteristics of subscribers the asset should be recommended to. Will be used in future release of Ncanto only. |
| "targetDevice" list of strings | List of devices the asset is suitable for. The classification of devices is up to the service provider, Ncanto provides filtering to assets which are suitable for the device used by the subscriber. |
| "subtitles": {<lang>: <subtitle>, ...} | Subtitles of the asset in different languages. Used by the recommendation engines and as annotation. |
| "seasonTitles": {<lang>: <seasonTitle>, ...} | Season titles of the asset in different languages. Used by the recommendation engines and as annotation. |
| "categories":" {<lang>: <category>,...} | Any other characterization of the asset in addition to genres and program type. |
| "synopses":{<type>: {<lang>:<synopsis>,...},...} | Textual description of the asset in different classifications (e.g. short/long synopsis) and languages. |
| "sourceNames":{<type>: {<lang>:<sourceName>,...},...} | Contains the full names of the source of the asset in different classifications and languages.<br>Example:<br>"sourceNames":<br>{<br>"short":<br>{<br>"eng":"CNN International"<br>},<br>"long":<br>{<br>"eng": "Cable News Corporation International"<br>}<br>} |
| "sourceGroupNames":[<br>{<lang>:<sourceGroupName>,<br><lang>:<sourceGroupName>,<br>...<br>},<br>...<br>] | Human readable names of the source groups. The order of the language maps shall correspond to the order of source group IDs. |
| "uiGenres":{<type>: [{"id":<genre>, <lang>:<uiGenre>,...},...],...} | Genre information for annotation and explanation purposes. In contrast to genreFine, genreMedium, and genreBroad, uiGenre supports multiple languages. Multiple uiGenre classifications are supported e.g. for hierarchical classifications. |

| | |
|---|---|
| | If types "broad", "medium", and "fine" are present, they are interpreted as the human-readable representations of genreBroad, genreMedium and genreFine, and will be used in explanations of the preference engine.<br>For each genre classification, an asset can have multiple genres.<br>Example:<br>"uiGenres":<br>{<br>"broad":<br>[<br>{<br>"id": "17",<br>"eng": "movie",<br>"deu": "Spielfilm"<br>}<br>],<br>"fine":<br>[<br>{<br>"id": "17.15",<br>"eng": "action movie",<br>"deu": "Actionfilm"<br>},<br>{<br>"id": "17.17",<br>"eng": "horror movie",<br>"deu": "Horrorfilm"<br>}<br>]<br>} |
| "uiProgramType":{<lang>: <prgT>,<br><lang>: <prgT>,<br>"id": <prgtId><br>...<br>} | Program type information for annotation and explanation purposes. If an "id" is present it will be used to translate the programType into human-readable representations.<br><br>Example:<br>"uiProgramType":{<br>"id": "2",<br>"eng": "movie",<br>"deu": "Spielfilm"<br>} |
| "adProduct": string | For advertisement assets only, description of the advertised product. |
| "adCompany": string | For advertisement assets only, manufacturer of the advertised product |
| "adTaxonomy": list of strings | For advertisement assets only, contains the taxonomy of the advertising asset. |
| "customProperties": noDotsJson | Any service provider defined properties of the asset in JSON format. Used by Ncanto to learn preference profiles. Note that if any special mapping of attributes is required (e.g. timestamp, language map, or nested), you have to ask XroadMedia personnel to configure it. |
| "clientCustomProperties": noDotsJson | Any service provider defined properties of the asset in JSON format. Not used by Ncanto. Note that if any special mapping of attributes is required (e.g. timestamp, language map, or |

| | |
|---|---|
| | **nested), you have to ask XroadMedia personnel to configure it.** |
| "lightness": decimal | **Lightness describes the mood of the asset along the axis sad-serious-lighthearted-humorous. In general the value range comprises all negative and positive decimal values. The asset importers (Funke importer etc.) map "sad" to lightness=-1, neutral/serious assets to lightness=0, and humorous assets to lightness=1.** |
| "pace": decimal | **Pace describes the mood of the asset along the axis slow-fast. In general all negative and positive values are supported. The asset importers (Funke importer etc.) map slow to 0 and fast to 1.** |
| "relatedAssets": noDotsJson | **Optional attribute allowing to link other assets from the asset store, e.g. trailers or bonus material.**<br>**Example:**<br>**"relatedAssets":{**<br>**"trailers":[<assetId>],**<br>**"bonus":[<assetId>]**<br>**}** |
| "metadataQuality": decimal | Optional attribute supported from R5.1, indicating the the richness and the quality of the metadata. Assets with metadataQuality below a certain threshold (0.1 by default, configurable by XroadMedia personnel) are masked in preference profiles. I.e. the asset will still be visible in the profile as rated asset, but the like degrees will not be influenced. |
| "awards": [<br>"name": string,<br>"uiName": {<br>optional,<br><langCode>: <string><br>},<br>"category": string,<br>"uiCategory": {<br><langCode>: <string><br>},<br>"year": int,<br>"grade": string,<br>"person": string<br>}<br>] | Optional attribute supported from R5.2. List of the awards the asset won or was nominated for. Only the award name is mandatory, all other attributes are optional.<br><br>Example:<br>"awards": [<br>{<br>"name": "Academy Awards",<br>"uiName": {<br>"eng": "Academy Award",<br>"fra": "Oscars du cinéma"<br>},<br>"category": "Best Actor in a Leading Role",<br>"uiCategory": {<br>"eng": "Best Actor in a Leading Role",<br>"fra": "Meilleur acteur"<br>},<br>"year": 2018,<br>"person": "Gary Oldman",<br>"grade": "won"<br>}<br>] |

**Table 12: Detailed explanation of asset attributes supported by Ncanto**