Attribute API

Create attribute

Request	POST /editor/attribute
Params	**termId** Required ID of term where attribute should be created, even if not at term-level
	level=(entry|language|term) Required Level that the attribute should be created for
	dataType Required Type of an attribute. Can be integer or string. See `filterWindow.attributes[*].(id|type)` in **Client app setup data** response
	batch=(0|1) Optional Default 0. If this param is 1, it means that batch-mode is **On** so **termId**-param can accept comma-separated term IDs
	except=(0|1) Optional Default 0. This param is only applicable when batch-mode is **On**. Before making request with **except=1** you should preliminary run term search request at least once, so the IDs of found terms except given by **termId**-param would be involved

Batch-mode response example: as you can see, value of **`inserted.id`** contains comma-separated IDs of created attributes instead of just single integer value. Also please note that if attribute of some **`dataTypeId`** should be unique at it's level but it already exists, the response will contain **`existing`**-prop, having newly created attrs IDs as keys and existing attrs IDs as values. In the below example response two Comment-attributes on termEntry-level for different termEntries were created, but one of affected termEntries does already have Comment-attribute with ID=2020358. All attributes created in batch-mode are set up with **`isDraft=1`** flag in the database, so they are drafts, and they need to be confirmed further, otherwise they will be deleted by daily cron job.

Response in case of non-batch-mode would be the same as below, except that:

1. **`inserted.id`** would contain single value instead of comma-separated values,
2. there would be no **`existing`**-property in response json
3. This is not mentioned in response, but not that **isDraft** would be **=0**,

Response

```
{
    "inserted": {
        "id": "2020477,2020478",
        "target": "",
        "value": null,
        "type": null,
        "language": null,
        "deletable": true,
        "dataTypeId": "20",
        "created": "Pavel Perminov, 09.05.2022 08:37:44",
        "updated": "Pavel Perminov, 09.05.2022 08:37:44"
    },
    "updated": "Pavel Perminov, 09.05.2022 08:37:44",
    "existing": {
        "2020478": 2020358 // "ID of draft attr": "ID of already existing attr at same level with same
dataTypeId"
    }
}
```

Update attribute

Request	PUT /editor/attribute

Params	**attrId** Required if **draft0**-param is not given. Comma separated IDs of attributes, whose values should be updated
	dropId Optional Comma-separated list of IDs of draft-attributes, that should be deleted with preliminary usage of their values to spoof values of existing attributes identified by corresponding values from **attrId**-param
	draft0 Required if **attrId**-param is not given. Comma-separated IDs of attributes, that currently are drafts, but should be un-drafted. Here can be IDs of attributes of types that can be non-unique within their level. Also, there can be IDs of attributes of types that should be unique within their levels, but that currently are unique, e.g. there were no existing attributes found of those types.
	dataIndex=(value|target) Required if ALL attrs identified by attrId-param are of type **`xGraphic`** or **`externalCrossReference`**
	value Required if it's a picklist-attribute
	target Optional Must be valid termEntryTbxId (for termEntry-level attributes) or termTbxId (for term-level attributes)
	termLang,mainLang Optional RFC5646-codes for preferred languages to return terms found by **target**-param.
	figure Required if ALL attrs identified by **attrId**-param are of type **`figure`**.

There are the following use cases each with set of request params involved:

1. Value of some certain single attribute needs to be updated
   a. It's an attribute of a type that can be non-unique at it's level (e.g. of type **`xGraphic`**, **`externalCrossReference`**, **`crossReference`** and **`figure`**)
      i. **`xGraphic`**, **`externalCrossReference`**
         1. attrId
         2. dataIndex
         3. value
      ii. **`crossReference`**
         1. attrId
         2. target
         3. termLang
         4. mainLang
      iii. **`figure`**
         1. attrId
         2. figure
   b. It's an attribute of a type that should be unique at it's level (all other attribute types, mean other than mentioned in point 1.a)
      i. It's a picklist attribute (e.g. of type **`processStatus`**, **`administrativeStatus`**, **`termType`**, **`grammaticalGender`** and others)
         1. attrId
         2. value
      ii. It's a plaintext attribute (e.g. of type **`definition`**, **`source`**, **`context`**, **`geographicalUsage`**, **`subjectField`** and others)
         1. attrId
         2. value
2. Multiple attrs can be affected
   a. Batch-update draft attributes (but attributes will be remain drafts) - request params are same as described in point 1. Such usage is involved when user updates any attributes within batch-edit window. All attributes identified by comma-separated Ids given by **attrId**-param should be of same type, otherwise error response will be returned.
   b. Batch-confirm draft attributes, so attributes won't be drafts anymore. Such usage is involved when user press Save-button in batch-edit window
      i. attrId / attrId // If during batch editing the only attrs were added that should be unique, but there
      ii. dropId / dropId // are already existing attrs of those types on their levels then both **attrId** and **dropId**
      iii. draft0 / draft0 // params ARE required. If **attrId** is given but **dropId** is not - system will assume we're in the use case described in point 2.a. If only attrs that should NOT be unique were added, and/or added attrs that should be unique and they already ARE unique even while they're drafts then **attrId** and **dropId** params ARE NOT required, but **draft0**-param IS required due to that `isDraft` should be set to 0 for those identified by **draft0**

Response [all levels]: figure


```
{
    "src": "/editor/plugins/termimage/TermPortal/tc_82/4a70516e-1ae0-4a02-96bc-139a18ff51e3.jpg",
    "updated": "Pavel Perminov, 10.05.2022 13:03:59"
}
```

Response [all levels]: externalCrossReference and xGraphic


```
{
    "updated": "Pavel Perminov, 10.05.2022 12:58:16",
    "isValidUrl": 1
}
```


Response [entry,term]: crossReference


```
{
    "updated": "Pavel Perminov, 10.05.2022 12:51:19",
    "isValidTbx": false,
    "id": "2020436",
    "collectionId": "82",
    "termEntryId": "16609",
    "language": "en",
    "termId": "222615",
    "termTbxId": "ida2526eba-c92e-4887-9137-df5838ac8f73",
    "dataTypeId": "78",
    "type": "crossReference",
    "value": null,
    "target": "asd",
    "isCreatedLocally": "1",
    "createdBy": null,
    "createdAt": "2022-04-18 21:41:05",
    "updatedBy": null,
    "updatedAt": "2022-05-10 13:51:19",
    "termEntryGuid": "08c9df82-8656-417c-888b-42f95c409b82",
    "langSetGuid": null,
    "termGuid": "791aecf8-753f-43d3-8207-c7351b659914",
    "guid": "8149f91f-98e9-4fee-a0dc-bd40cc7e5853",
    "elementName": "ref",
    "attrLang": "en",
    "isDescripGrp": "0",
    "isDraft": "0"
}
```

Response [term]: all other attr types

```
{
    "success": true,
    "updated": "Pavel Perminov, 10.05.2022 12:49:02",
    "status": {                           // Applicable on term-level only
        "status": "admittedTerm",
        "others": {
            "21": "regulatedTerm-admn-sts" // Info about which status-related attributes were changed to
which values,
        }                                 // in dataTypeId:value format. Currently those can be
administrativeStatus-
     },                                   // and normativeAuthorization-attributes. There is special logic
implemented
                                          // of how one status-related attr can affect other when changed,
so this info
                                          // is used in TermPortal GUI to reflect those changes. Note:
dataTypeId is
                                          // used as attr key due to that those attrs are unique having
their dataTypeId
                                          // at their level (term-level)
     "icons": {         // Info about which icons for which terms should be updated both in center and left
grids
        "222615": {     // Info for multiple term IDs here can be in case of batch-confirming draft attributes
            "status": "admittedTerm"
            "processStatus": "unprocessed"
        }
    }
}
```

Response [entry,language]: definition

```
{
    "success": true,
    "updated": "Pavel Perminov, 10.05.2022 13:23:04",
    "definition": {
        "value": "language-level definition12",
        "affected": [      // In left panel of TermPortal GUI there is a grid showing found terms, and each
grid row
            "222615"       // have info-icon showing definition in tooltip on hover, so here is the list of
IDs of
        ]                  // rows whose tooltips should be updated with new definition value. Logic of how
entry- and/or
     }                     // language-level definitions are replicated acrosss terms assumes that here can
be more than
 }                         // 1 ID even if definition-attr was updated not in batch-mode
```

Response [entry,language]: all other attr types

```
{
    "success": true,
    "updated": "Pavel Perminov, 10.05.2022 13:25:40"
}
```

Delete attribute

Request	DELETE /editor/attribute
Params	**attrId** Required ID of attribute to be deleted. Can be also be comma-separated list of attribute IDs

Response


```
{
    "updated": "Pavel Perminov, 09.05.2022 10:24:17"
}
```