

MetaDoc Documentation (MDOC)

Bjørnar Grip Fjær

June 17, 2010

1 XML document

The XML document should follow the form described in the MetaDoc DTD [1].

1.1 Special attributes

The **id** attribute of elements have a special function in MetaDoc. This attribute is used to identify the object when receiving receipts from the server whether elements have been added. The attribute is *not* saved in caching to avoid duplicate **ids** when resending cached data together with new data. If you want to give elements a special identifier that should be saved, it must be called something other than **id**.

2 Extensibility

MetaDoc is made so that extending the data sent between client and server should be as easy as possible.

2.1 Client

1. Create a module in the client folder. The name of the module should be the name of the base tag for the information sent.
2. Create **definition.py** and **entries.py** inside this module.

definition.py should contain the definition of the base tag for the information. It must be a sub-class of **metaelement.MetaElement**.

It should define the class variables **xml.tag_name** and **url**, being the base tag for the information and url to send/receive information to/from, respectively.

3 MetaDoc API description

3.1 Server API

The MetaDoc server implements a REST-like API. The server defines several URLs that can be accessed from the client:

baseurl/allocations/ Retrieves a list of allocations relevant to the site

baseurl/users/ Retrieves a list of users for the site

baseurl/projects/ Retrieves a list of projects relevant to the site

baseurl/config/ Sends system configuration to server

baseurl/events/ Sends site events to the server

baseurl/software Sends system software to server

When sending information to the MetaDoc Server, only the information relevant to that URL is processed. Any XML data sent that is not relevant for that URL is discarded, e.g. event information sent to **/baseurl/config/** will be discarded by the server. No receipt will be returned for this data.

The server will return a MetaDoc XML document containing a **<receipt>** element, which will contain **<r_entry>** elements for each element received. The **<r_entry>** element should return a code from table 1 for each element. See section 4 for more information on errors.

3.1.1 Differences from REST

There are certain differences in the API compared to the REST specification. The MetaDoc Server API makes use of HTTP POST where HTTP PUT should be used in accordance with REST. This is due to limitations in standard Python libraries, as well as limitations to the server side implementation in Django.

Because the access the MetaDoc Server API gives to the client is limited, this change does not prohibit any other functionality.

4 Errors

The server returns a `<receipt>` containing an `<r_entry>` for each element parsed. The `<r_entry>` has the required attributes **id** and **code**, containing the ID of the element and the error code, respectively. It may also contain an attribute **note** with a short note explaining the error if extra information is available. The `<r_entry>` tag might also contain text with a longer message, if more information is needed about the error.

4.1 Document errors

In the special case where there are problems with the document itself, such as XML errors or the document not passing DTD verification, the `<r_entry>` **id** attribute will be set to 0 (zero), referring to the document itself.

5 Caching

The client will cache any information that is not accepted by the server, *unless* the server returns a receipt for the information that marks the information as invalid or malformed in some way, such that the information will not be accepted if resent at a later date.

References

- [1] *MetaDoc Document Type Definition*, <http://www.austad.us/metadoc/MetaDoc.dtd>

A List of errors

Table 1: Error codes recieved from server

Error code	Meaning	Extra notes
1000	No errors	
2000	Error with the XML data	
2001	Missing attribute	Missing attribute should be returned as a note.
3000	Database error	
3001	MySQL database error	Note should contain the MySQL error code, and the message the MySQL error message