API Standards

This page contains some shared information about all our APIs.

Dates

We need to decide on a standardized date and datetime format. We will most likely go with ISO 8601.

 Formats

All APIs will be available in both JSON and XML format. Because XML is a more complex representation than JSON, there is not a simple 1:1 mapping. However, we will base the APIs on the JSON format and define a consistent schema for converting JSON to XML for consistency. Here are some standards that we’re using.

Objects, Types, and IDs

Root objects should always have a type and id property. The type will be used to define the XML element name of the object, and the ID will be tagged as an attribute on the root element. Here is a very simple example demonstrating this:

Sample Object JSON

```json
{ type: "myobject",
 id: "objectid5342",
 name: "An Object",
 color: "green"
}
```

Sample Object XML

```xml
<myobject id="objectid5342">
  <name>An Object</name>
  <color>green</color>
</myobject>
```

Lists

In JSON lists are allowed to be either very simple lists of a single basic value (typically a string), or lists of objects. Flat lists will be converted to XML as follows:

Sample Object with List JSON

```json
{ type: "myobject",
 id: "objectid5342",
 name: "An Object",
 colors: ["green", "blue", "purple"]
}
```

Sample Object with List XML

```xml
<myobject id="objectid5342">
  <name>An Object</name>
  <colors>
    <color>green</color>
    <color>blue</color>
    <color>purple</color>
  </colors>
</myobject>
```
The choice of element name for the internal list items (in this case "color") will be defined statically, per element, or may be guessed if there is no static mapping. Typically guessing would entail removing a trailing "s" from the list property name, as in this example.

Lists of objects expect the type property to be present on the object to choose the inner element name, although if none is present will use either a static mapping or guess. The ID attribute is optional but will be treated as described above if present. Root lists of objects should be wrapped in a one-element dictionary, the key of which represents the root element name. The following example shows a basic list of objects. If types are defined it is illegal for any object in the list to have a type different from the others.

**Sample List of Objects JSON**

```json
{objectcollection: [
  {
    type: "myobject",
    id: "1",
    name: "A clover",
    color: "green"
  },
  {
    type: "myobject",
    id: "2",
    name: "The sky",
    color: "blue"
  },
  {
    type: "myobject",
    id: "3",
    name: "Minda",
    color: "purple"
  }
]}
```

**Sample List of Objects XML**

```xml
<objectcollection>
  <myobject id="1">
    <name>A clover</name>
    <color>green</color>
  </myobject>
  <myobject id="2">
    <name>The sky</name>
    <color>blue</color>
  </myobject>
  <myobject id="3">
    <name>Minda</name>
    <color>purple</color>
  </myobject>
</objectcollection>
```