

## Appendix D: Understanding XML & the GPI Schema

XML stands for *Extensible Markup Language*. This language is a simple way to standardize structured information—in this case, specimen metadata—so that it becomes easy to transport and use that information on the web.

More information about XML and XML schemas can be found at:

- XML Tutorial, <http://www.w3schools.com/xml/default.asp>
- O'Reilly's XML Tutorials, <http://www.xml.com>
- XML Schema Primer, <http://www.w3.org/TR/xmlschema-0/>

### General XML formatting rules

The following general XML formatting rules must be followed in constructing the GPI XML metadata file. The XML document must have the following characteristics:

- It must begin with the XML declaration (header).
- It must have one unique root element ("DataSet," for GPI).
- All of its start tags must have matching end tags.
- All tags are case-sensitive (unlike HTML).
- All elements must be closed.
- All elements must be properly nested.
- All attribute values must be quoted.
- It must use XML entities for reserved characters.

### Reserved characters and XML entities

Certain characters, referred to as *reserved characters*, are used by the XML structure and cannot be included in any metadata values (i.e., in the specific data, to be contained within an XML tab or subtab). Wherever these reserved characters appear in the data, alternate character sets, known as *XML entities*, must be substituted for them before the data may be included in the XML file to be exported. The reserved characters and corresponding XML entities are listed in the chart below.

| Reserved character |   | XML entity |
|--------------------|---|------------|
| Greater than       | > | &gt;       |
| Less than          | < | &lt;       |
| Ampersand          | & | &amp;      |

|                 |   |        |
|-----------------|---|--------|
| Quotation marks | " | &quot; |
| Apostrophe      | ' | &apos; |

The example below shows a metadata value for the Locality tag that uses reserved characters (apostrophes and quotation marks):

```
<Locality>Koopmansfontein: Agricultural Research Station; Golden Rock. Pan. 28°11'49.7"S  
24°06'17.9"E</Locality>
```

This metadata value *must* appear as follows in the exported XML file:

```
<Locality>Koopmansfontein: Agricultural Research Station; Golden Rock. Pan.  
28°11&apos;49.7&quot;S 24°06&apos;17.9&quot;E</Locality>
```

**NOTE:** If reserved characters are inadvertently included in the XML value data, the XML file will produce unpredictable results when viewed with a browser.

### Correct spelling of tags

All GPI XML tags must be spelled correctly, using the correct letter case.

| <i>Correct tag</i> | <i>Incorrect tags</i>            |
|--------------------|----------------------------------|
| DataSet            | Dataset, dataset                 |
| StoredUnderName    | Storedundername, storedundername |
| InstitutionCode    | Institutioncode                  |

### The GPI XML schema

Regardless what method you use to export data and create the XML file for your institution, the file must follow the schema adopted by GPI. This section will review the tags that make up the schema and the proper formatting of data for each.

The top level of the GPI XML schema consists of a "Dataset" tag, with five main tags:

InstitutionCode

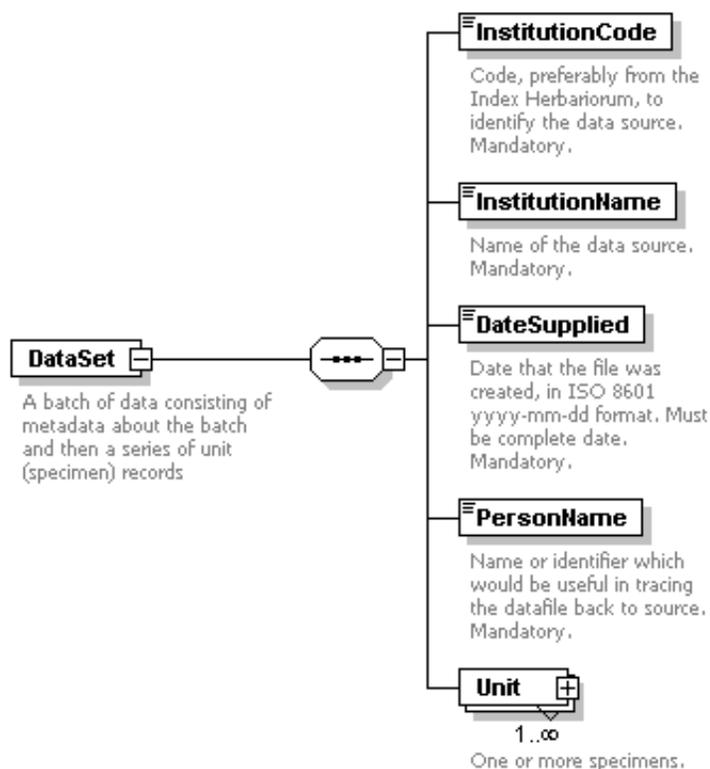
InstitutionName

DateSupplied

PersonName

Unit (this tag will be repeated, one for each specimen image file)

This is a diagram of the top-level schema structure:



### Explanation of header and top-level tags

**XML header (mandatory).** Every XML file must have a header section. The header must be the first line of the file, and must contain the following:

```
<?xml version="1.0" encoding="UTF-8" ?>
```

**DataSet tag (mandatory).** The DataSet tag follows the XML header and is required for the GPI schema. Its form is:

```
<DataSet xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://plants.jstor.org/XSD/AfricanTypesv2.xsd">
```

..

```
</DataSet>
```

This tag must be spelled *DataSet* and not *Dataset*, *dataset*, or *dataSet*.

**InstitutionCode, InstitutionName, DateSupplied, and PersonName tags (mandatory).**

Each of these four tags is mandatory.

1. The InstitutionCode value must be from Index Herbariorum (<http://sweetgum.nybg.org/ih/>). If there is no IH code for your institution, then a code will be assigned by GPI.
2. The InstitutionName is the name of your institution.
3. The DateSupplied is the date of the creation of the metadata file.
4. The PersonName is a contact at your institution for potential follow-up.

An example of the form of these tags is:

```
<InstitutionCode>K</InstitutionCode>
```

```
<InstitutionName>Royal Botanic Gardens, Kew</InstitutionName>
```

```
<DateSupplied>2004-04-01</DateSupplied>
```

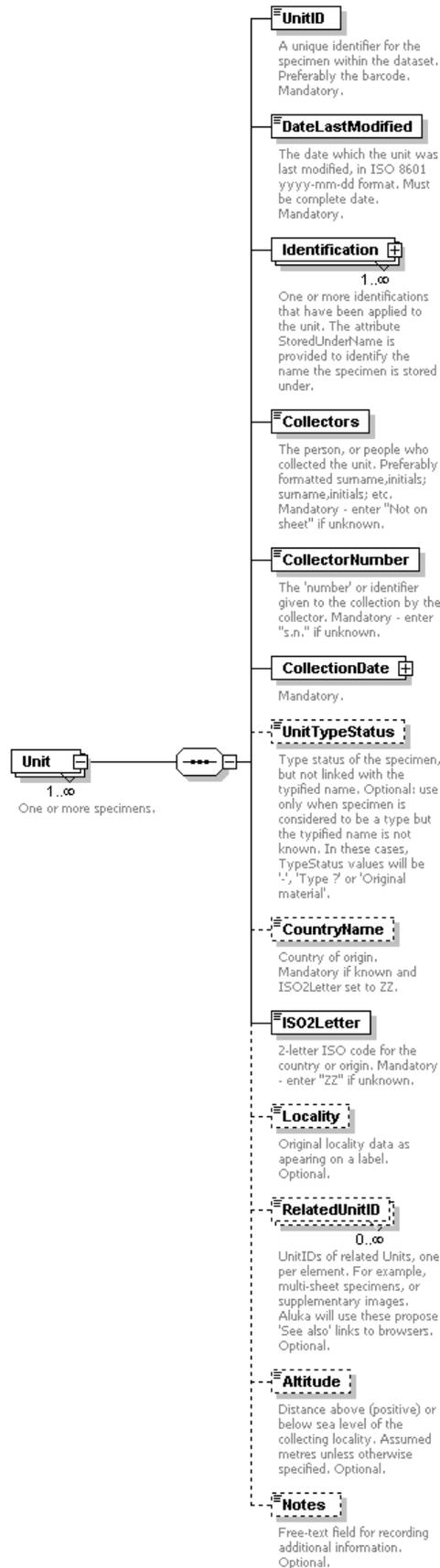
```
<PersonName>John Jones</PersonName>
```

**Unit tags.** There must be at least one Unit tag for each GPI XML file. But an unlimited number of Unit tags can be included. Each Unit tag usually represents one image file in a batch. The UnitID and the image file name must match, based on the barcode number.

*Exception.* Where multiple images have been made for a single specimen—such as additional scans of parts of the sheet (e.g., to capture details, annotations), or scans of multiple sheets used for a single specimen—there can be multiple image files for the same UnitID; but the files must be named UnitID.tif, UnitID\_a.tif, etc.

Since there are many potential metadata values associated with a specimen image, the Unit tag has many subtags available for use. Some are required (mandatory), and some are optional.

Here is a diagram of the Unit tag and its subtags:



There are seven *required* subtags for each Unit:

1. UnitID
2. DateLastModified
3. Identifications (at least one)
4. Collectors
5. CollectorNumber
6. CollectionDate
7. ISO2Letter

There are six *optional* subtags for each Unit:

1. UnitTypeStatus
2. CountryName
3. Locality
4. RelatedUnitID
5. Altitude
6. Notes

The seven required subtags for each Unit are described below. Explanations the six optional subtags for each Unit will follow.

#### **Explanation of *required* subtags of the Unit tag**

**1. UnitID (mandatory).** This is a unique identifier for the Unit or specimen image. For GPI it is *required* that the UnitID be the same as the InstitutionCode concatenated with the barcode on the specimen (this is not merely "preferable," as stated in the schema description in the above illustration). It is also required that the UnitID be the same as the name of the specimen image file for the Unit. One of the data-submission quality checks that JSTOR makes is comparing the UnitIDs in the XML dataset with the file names on the hard drive for an exact match (except for the multi-image files with *\_a*, *\_b*, etc. extensions that have been noted above.)

Examples of valid UnitIDs in XML:

```
<UnitID>K10000081</UnitID>
```

```
<UnitID>US00987634</UnitID>
```

Examples of invalid UnitIDs:

```
<UnitID>12121212</UnitID> (No Index Herbariorum acronym)
```

```
<UnitID>34567ABC</UnitID> (Index Herbariorum acronym should be at start of number)
```

```
<UnitID>L44455566</UnitID> (Where L is not the IH institution code)
```

**NOTE:** Do not include any spaces in the UnitID value.

If the barcode on the specimen does not include the institution's code as a prefix, the UnitID and the image file names can be created on output from the institution's metadata by concatenating the institution code with the barcode.

2. **DateLastModified (mandatory).** The date to be supplied here is the last time any part of the metadata for this Unit or specimen image was changed. This "last change date" is generally recorded in the primary database of the scanning institution. Note that the date format for XML is yyyy-mm-dd, including the hyphens.

Example in XML:

```
<DateLastModified>2006-06-23</DateLastModified>
```

3. **Identification (mandatory).** Each Unit can have multiple Identification subtags. The intent of this is to enable the recording of multiple names, or determinations, associated with a specimen sheet, rather than just one determination. For instance, a single specimen could have multiple determinations by different specialists, plus it could have different names for which it is a type, and it could have a name by which it is filed by an institution.

Identification has one required attribute and seven required subtags, plus six optional subtags. The attribute is described below. It will be followed by explanations of the seven required subtags and the six optional subtags.

#### **Explanation of *required* attribute of the Identification tag**

**StoredUnderName Attribute (mandatory).** The Identification tag has a mandatory attribute called StoredUnderName. This StoredUnderName attribute is either "true" or "false." One and only one Identification tag can have the value "true" for StoredUnderName, because no specimen can be stored or filed under more than one name.

In other words, the value "true" for StoredUnderName ***should occur only once*** for all Identification tags within a single Unit. All other Identifications within a single Unit must have the value "false" for StoredUnderName. This rule is checked as part of the quality control of submitted GPI XML metadata files.

**NOTE:** Do not use "0" or "1" or "Yes" or "No" or "True" or "False" for the value of StoredUnderName. The only acceptable values are either "true" or "false". Only one "true" can be used, and "false" can be used as many times as needed.

*How is StoredUnderName to be used?*

In some herbaria, specimens are placed inside folders or organized by one name, the "stored under" name, but may also have a different determined name or type name on the sheet. Using the StoredUnderName attribute allows the name the institution has chosen to store or file the specimen under to be distinguished from any other names associated with the specimen.

*What if we do not use "Stored Under" names?*

In many herbaria, no distinction is made between the determined name and the way a specimen is stored or filed. And some herbarium databases do not record multiple names for a single specimen. In either of these cases, a single name would be recorded in the specimen database for the sheet.

If only one name is recorded in the database, then the StoredUnderName attribute must be "true" for that name, since that one name is serving the purpose of a stored or filed name.

Example in XML:

```
<Identification StoredUnderName="true">
.
</Identification>
```

*Note: "True" should occur only once for any Identification tag within a single Unit.*

Or

```
<Identification StoredUnderName="false">
.
</Identification>
```

*Note: All other Identification tags within a single Unit must have a value of "false."*

#### **Explanation of *required* subtags of Identification**

a. **Family (mandatory).** Based on the scanning institution's own taxonomic decisions, or as shown on the sheet. It is recommended that the family be entered in all uppercase letters.

Example in XML:

```
<Family>ASCLEPIADACEAE</Family>
```

b. **Genus (mandatory)**. As recorded by the scanning institution. First letter should be uppercase.

Example in XML:

```
<Genus>Secamone</Genus>
```

c. **Species (mandatory)**. As recorded by the scanning institution. Should be all lowercase.

Example in XML:

```
<Species>grandiflora</Species>
```

d. **Author (mandatory)**. The author of the species name, including basionym author and ex/in authors, following standard format. Standard Author Abbreviations should be used, based on Authors of Plant Names maintained by Royal Botanic Gardens, Kew, at <http://www.ipni.org/ipni/authorsearchpage.do>

Example in XML:

```
<Author>Klack.</Author>
```

**NOTE:** If species author is missing or unknown, use the phrase *Not on sheet*.

e. **Identifier (mandatory)**. The name of the person who made the determination of this Identification, as recorded by the scanning institution.

**NOTE:** If the identifying/determining person is not known, use the phrase *Not on sheet*.

Example in XML:

```
<Identifier>Not on sheet</Identifier>
```

f. **IdentificationDate (mandatory)**. The date when the determination of this Identification was made, as recorded by the scanning institution. The date value is not entered directly into this tag. Rather, the standard Date subtags must be used.

*Date subtags*

For CollectionDate and IdentificationDate, up to seven subtags are used to specify the date value:

1. StartDay
2. StartMonth

3. StartYear
4. EndDay
5. EndMonth
6. EndYear
7. OtherText

It is mandatory to supply a date value for only one of these subtags. A value for Year can be provided without a value for Month or Day. It is expected, however, that if a Day value is recorded, there must also be a Month value and a Year value. And if a Month value is recorded, there must be a Year value. There should not be an end-date value if there is no start-date value.

**NOTE:** At least one subtag of Date must have a value.

If there are no date values on the sheet, then the phrase *Not on sheet* must be inserted into the OtherText subtag.

The Start and End tags are designed for date ranges, most often associated with the CollectionDate, not the IdentificationDate. For this reason, it is most likely that you will only use only the StartDay, StartMonth, and/or StartYear subtags for IdentificationDate.

Example:

```
<IdentificationDate>  
  <StartDay>27</StartDay>  
  <StartMonth>01</StartMonth>  
  <StartYear>1992</StartYear>  
</IdentificationDate>
```

Or, where there is no date:

```
<IdentificationDate>  
  <Other Text>Not on sheet</OtherText>  
</IdentificationDate>
```

g. **TypeStatus (mandatory)**. The type status of this Identification. Use a dash [—] if the Identification is not a type name.

Use **only one** of the following values for type status:

|                   |           |
|-------------------|-----------|
| Holotype          | Epitype   |
| Isoepitype        | Lectotype |
| Isolectotype      | Neotype   |
| Isonotype         | Paratype  |
| Isoparatype       | Syntype   |
| Isosyntype        | Isotype   |
| Type              | Type ?    |
| Original material | —         |

Any other type status recorded by the institution must be converted to one of these.

**NOTE:** If a specimen is known or thought to be a type specimen, but the name for which it is a type is unknown, then all Identifications will have the TypeStatus [—]. If you are working with nontype specimens, also use [—] for this subtag.

Example in XML:

```
<TypeStatus>Holotype</TypeStatus>
```

#### Explanation of *optional* subtags of the Identification tag

Each Identification has a further six optional subtags, described below.

a. **GenusQualifier (optional)**. Qualifier expressing doubt about the genus epithet (eg. cf)

Example in XML:

```
<GenusQualifier>cf</GenusQualifier>
```

b. **SpeciesQualifier (optional)**. Qualifier expressing doubt about the species epithet (eg. cf)

Example in XML:

```
<SpeciesQualifier>cf</SpeciesQualifier>
```

c. **Infra-specificRank (optional)**. Rank based on ICBN, and as recorded by the scanning institution. Should be all lowercase.

Example in XML:

```
<Infra-specificRank>var.</Infra-specificRank>
```

d. **Infra-specificEpithet (optional)**. As recorded by the scanning institution. Should be all lowercase.

Example in XML:

```
<Infra-specificEpithet>alba</Infra-specificEpithet>
```

e. **Infra-specificAuthor (optional)**. Follow the same guidelines as for the Author subtag.

Example in XML:

```
<Infra-specificAuthor>Wild.</Infra-specificAuthor>
```

f. **PlantNameCode (optional)**. An optional code that is meaningful to the scanning institution for the name given for this Identification. Often a tracking number. May be used to provide feedback from JSTOR to the scanning institution.

Example in XML:

```
<PlantNameCode>123ABC</PlantNameCode>
```

Four required subtags of the Unit tag remain to be explained. They are:

4. **Collectors (mandatory)**. This is simply a text string listing the collector or collecting team for this Unit. The preferred way of listing a collecting team is:

```
Surname1, Initials1; Surname2, Initials2; Surname3, Initials3
```

Note that a semicolon is used to separate the individual collectors. The senior collector should be listed first. If no collector data is available, then the value *Not on sheet* must be manually inserted in the XML. This tag cannot be left blank.

Examples:

```
<Collectors>Beentje, H. J.; Quansah, N.</Collectors>
```

If there are no collectors recorded, the value should read:

```
<Collectors>Not on sheet</Collectors>
```

5. **CollectorNumber (mandatory)**. This is generally the number assigned by the senior collector to the specimen. But it can contain letters if needed. Where there is no collector's number for the Unit, the value *s.n.* must be inserted in the XML. This tag cannot be left blank.

Examples:

```
<CollectorNumber>4559</CollectorNumber>
```

Or

```
<CollectorNumber>4559, 4560</CollectorNumber>
```

If there is no collector number, the value should read:

```
<CollectorNumber>s.n.</CollectorNumber>
```

6. **CollectionDate (mandatory)**. The date when the collection was made, as recorded by the scanning institution. But the date value is not entered directly into this tag. Rather, the standard date subtags must be used. An explanation of these subtags can be found under the section for the IdentificationDate tag.

Examples:

```
<CollectionDate>
  <StartDay>27</StartDay>
  <StartMonth>01</StartMonth>
  <StartYear>1992</StartYear>
</CollectionDate>
```

Or, if there is no collection date:

```
<CollectionDate>
  <OtherText>Not on sheet</OtherText>
</CollectionDate>
```

7. **ISO2Letter (mandatory)**. This is the 2-letter ISO 3166-1 code for the country where the specimen was collected.

The ISO 3166 master list is available at

<http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html>

When the country is missing or unknown for a specimen, the 2-letter code ZZ must be inserted into the XML.

Example in XML:

```
<ISO2Letter>ZZ</ISO2Letter>
```

**NOTE:** Where the institution has not utilized ISO codes in its data system, it will need to make a conversion from the country coding system it uses to the ISO2 system, before this information is included in the XML metadata file. The GPI XML Generator will automatically add ISO2 codes if the CountryName field is populated in the import file.

### Explanation of *optional* subtags of the Unit tag

1. **CountryName (optional).** This name is not needed if an ISO2Letter value has been provided. A value should be provided for this tag only if there is no ISO2 code for the country, and the value ZZ has been inserted for ISO2Letter. The CountryName tag allows an unusual country name, not recognized by ISO, to be assigned to the Unit or specimen image.

Example in XML:

```
<CountryName>Nova Granatensis</CountryName>
```

2. **Locality (optional).** This is the literal string of text that was recorded by an institution for "locality," describing where the specimen was collected. No other atomization of location is available in the schema. State, county, municipio, latitude and longitude, etc., must all be concatenated and added to the Locality field.

Example in XML:

```
<Locality>AMBANJA: Manongarivo Special Reserve, Bekolosi Mt.; in open montane forest.
</Locality>
```

The original data needs to be carefully examined before export to XML to replace any of the XML reserved characters, that is, <, >, &, ', and ".

3. **UnitTypeStatus (optional).** This tag is used only by RBG Kew. All other institutions will omit it.

**NOTE:** If a specimen is known or thought to be a type specimen, but the name for which it is a type is unknown, then for GPI, nothing is to be recorded in the XML metadata for UnitTypeStatus.

4. **RelatedUnitID (optional)**. This is a multiple-occurrence tag, so multiple UnitIDs for related Units can be included. The value to be inserted here, when relevant, will be the UnitID (IH code concatenated with barcode) of another, related specimen. There is no attribute to describe or classify the nature of the relation to the other specimen, just the existence of a relation.

Example in XML:

```
<RelatedUnitID> K0123456</RelatedUnitID>
```

5. **Altitude (optional)**. This is the altitude above sea level where the specimen was collected, as measured in meters or feet. Please use the word *meters* or *feet*, not just the abbreviation *m* or *f*.

Examples in XML:

```
<Altitude>100 meters </Altitude>
```

```
<Altitude>25.5 feet</Altitude>
```

6. **Notes (optional)**. This can be any text and has no other constraint. Be cautious, however, of the reserved characters and extended characters, as with other text. (See section, "Reserved characters and XML entities," concerning reserved characters).

Example in XML:

```
<Notes>This can be any text, so long as it does not use reserved characters. </Notes>
```

### Creation of the GPI XML file

To be readable by an XML viewer, all XML files must follow a standard structure. The rules of this structure are defined in an "XML Schema". The standard schema used for the GPI project is *AfricanTypesv2.xsd*, originally developed for the African Plants Initiative, and available at the JSTOR website. The schema defines the required and optional elements of the XML and constrains the content and structure of the metadata being exported. The required and optional elements of the schema used for GPI are explained in section 5.6 below.

Partners should not create individual XML files for each specimen. One XML file must be generated for all specimens contained in one batch. JSTOR will parse the XML file into individual records.

Because database software varies between Partner institutions, the method for extracting data and creating the XML file will vary. It is up to each Partner to figure out the best method for mapping the fields of their database to the fields needed for the XML schema, modifying their fields (if necessary) to fit the structure of the XML schema, and generating the XML file. It is possible that an institution's database could export directly to the proper XML structure if that institution develops an export to

match the schema fields; and certain databases (such as BRAHMS) already provide a direct export to the GPI XML schema.

It is best to discuss your options during your training session, or to contact Rafael Barron ([rafael.barron@mobot.org](mailto:rafael.barron@mobot.org)), the XML coordinator for the GPI project, when you are preparing your initial test batch.

### **GPI XML Generator**

Rafael Barron has created an Access database that will automatically create the XML necessary for delivery to JSTOR from a simple Excel spreadsheet or Access table. This GPI XML Generator removes the need for institutions with limited IT support to fully understand how XML and XML schemas work. It does require, however, that an institution is able to export data from their in-house database and reformat the data to fit the fields in the XML schema.

An Excel spreadsheet (or Access table) with the fields necessary for using the GPI XML Generator is provided with the system, and all that users must do is reformat their data to fit the spreadsheet provided. Once the data is in the correct format in the spreadsheet or table, users need only import it into the system. The GPI XML Generator will identify problems in the metadata, check taxonomic names against TROPICOS, automatically export the metadata into the proper XML format, and keep track of sent batches. More details can be obtained from Rafael Barron at [rafael.barron@mobot.org](mailto:rafael.barron@mobot.org). There is also a PowerPoint presentation on the project website that describes the use of the system.

### **Saving the GPI XML file**

Once you have used the XML Generator or another method to produce your metadata file, you must name the file according to the GPI file-naming convention, and save the file as UTF-8.

#### **File name**

The XML file for the GPI metadata must be named according to your institution's Index Herbariorum acronym (see <http://sweetgum.nybg.org/ih/>), batch number, and submission date, using the following file-name format:

institution\_batch\_YYYYMMDD.xml

Batches are numbered sequentially, starting from 0 for the first test batch, and then 1, 2, 3, etc. for each batch sent to JSTOR.

Example:

Institution:                   Missouri Botanical Garden (IH code: MO)

Batch number: 2 (the second batch sent to JSTOR)

Date: 1 November 2009

The file name would be: MO\_2\_20091101.xml

### **Saving as UTF-8**

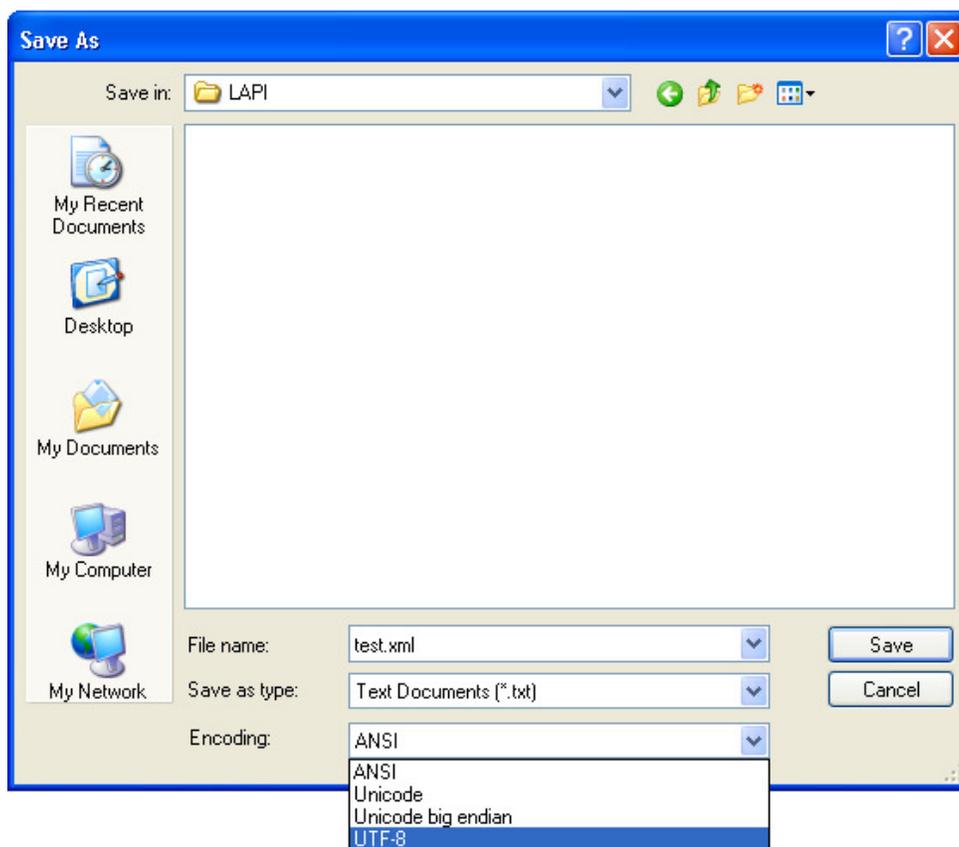
If any Windows extended characters are included in the metadata export, steps must be taken to ensure that the XML file is converted to UTF-8 encoding. This will not happen by default with Microsoft Office tools.

*UTF-8* (8-bit Unicode Transformation Format) refers to the underlying encoding method to be used for the text characters included in the XML file. UTF-8 is one way of displaying Unicode; there are others, but UTF-8 is used for GPI.<sup>1</sup>

One simple method to ensure that the file is converted to UTF-8 encoding is to use Microsoft Notepad, Version 5 or later, to open the exported XML file; select "Save As"; and from the "Encoding" drop-down menu at the bottom of the window, choose "UTF-8." The "Save As" screen with drop-down menu is shown below:

---

<sup>1</sup> The main impact of specifying UTF-8 encoding for an XML file is that it actually *excludes* a very common form of data encoding used in Western countries, namely the Microsoft "symbols," which can be inserted in the Western versions of Microsoft Word, Excel, and Access.



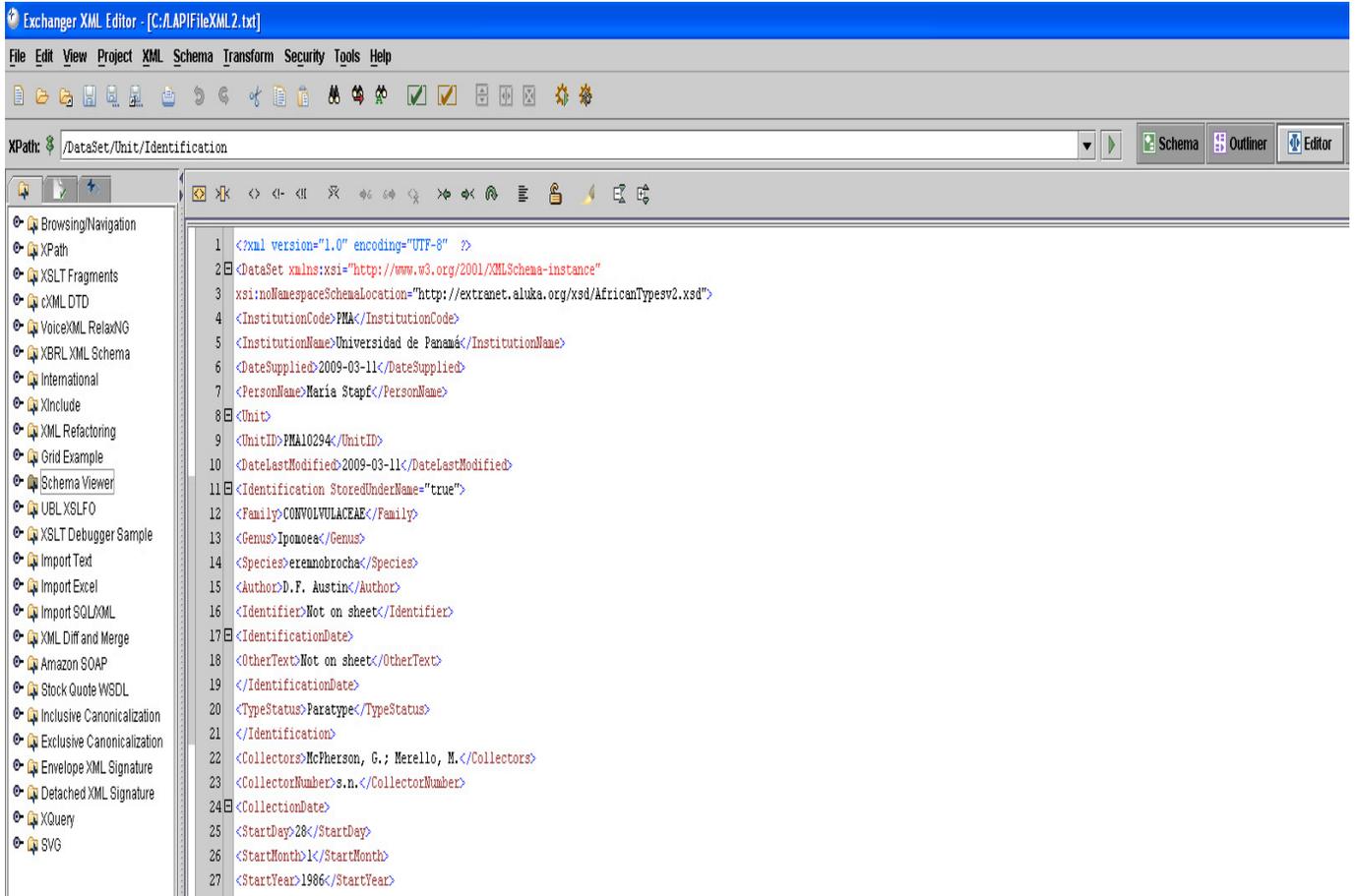
### Validating the GPI XML file

The XML file must be validated before it is sent to JSTOR. *Validated* means that the file has been checked to verify that its content and structure conform to the basic XML formatting rules, as well as to the GPI standard schema.

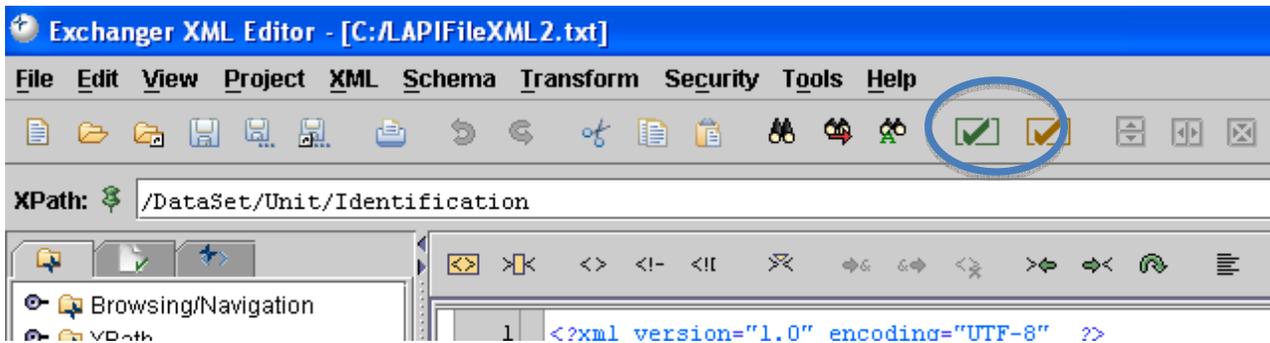
### Using Exchanger XML Lite 3.2 Editor to validate an XML file

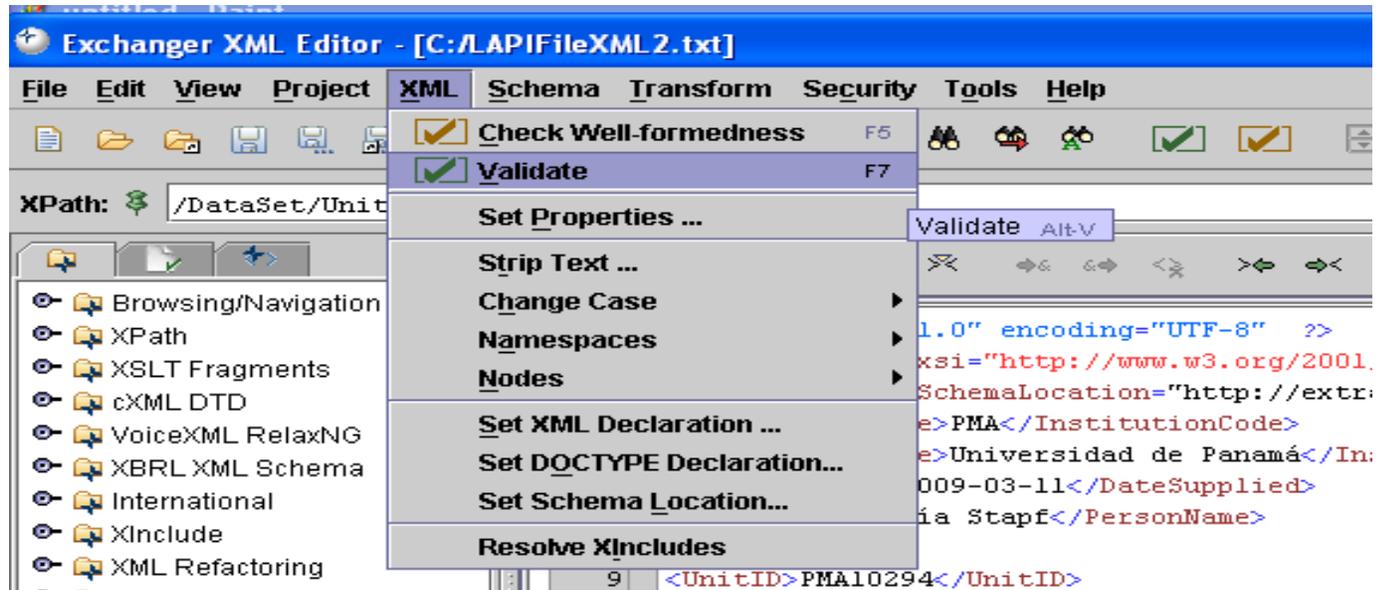
One easy way to validate an XML file is by using Exchanger XML Lite 3.2. This is a free program that can be downloaded from [www.freexmleditor.com](http://www.freexmleditor.com). The following is a step-by-step example of using Exchanger XML Lite 3.2 to validate an XML file.

1. Run Exchanger XML Lite 3.2.
2. Click on "File>Open" and select the your XML file. The contents of the XML file will appear, as shown below:

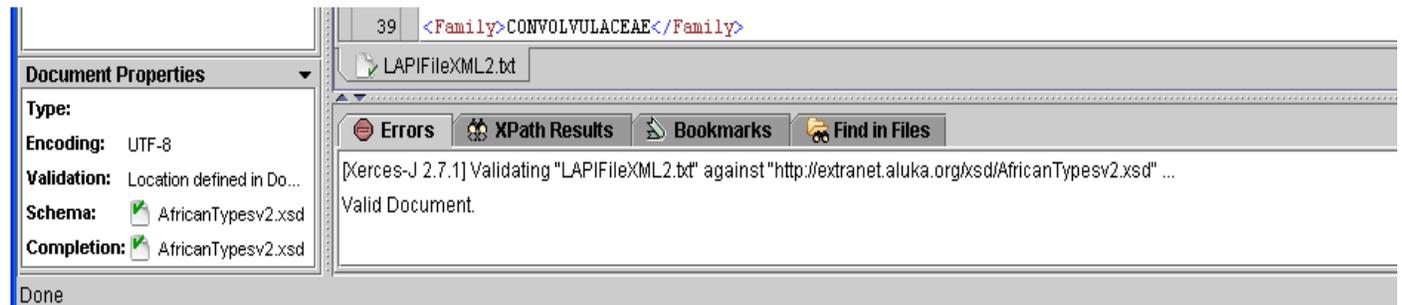


3. To start the validation process, click on the green  button in the tool bar. Or, click on "XML>Validate"; or press "F7."

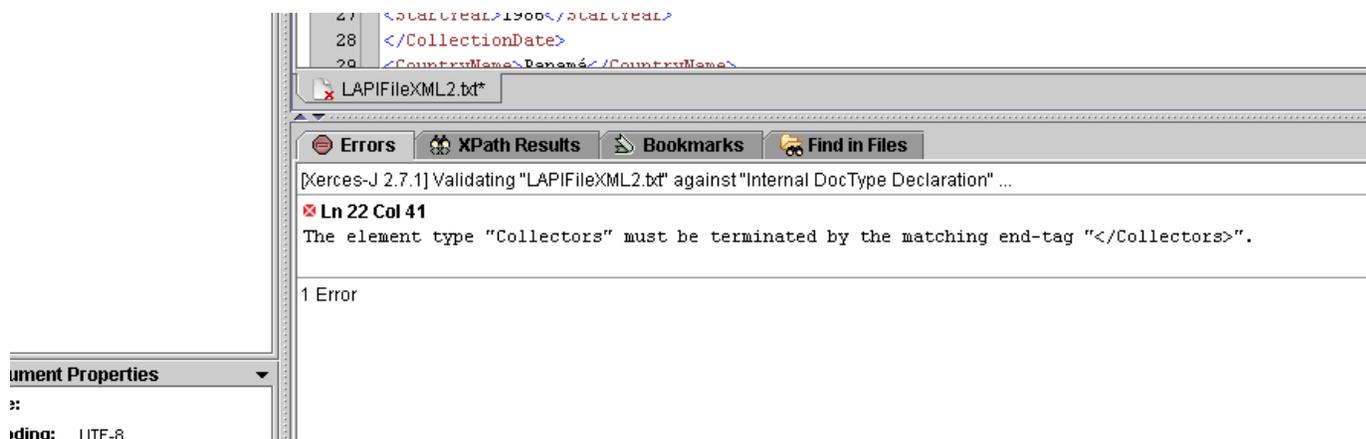




4. A message will appear at the bottom of the screen. In the example shown below, the message confirms that the file is valid:



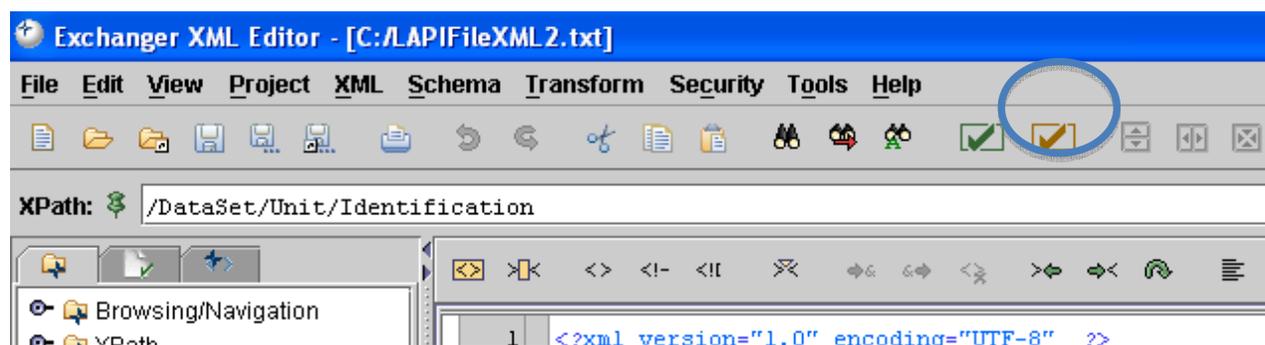
Below is an example of a message indicating that the XML is *not* valid, and identifying the specific error. In this case the problem is a mismatched end tag. All such errors must be corrected. The file must be valid before it can be sent to JSTOR.



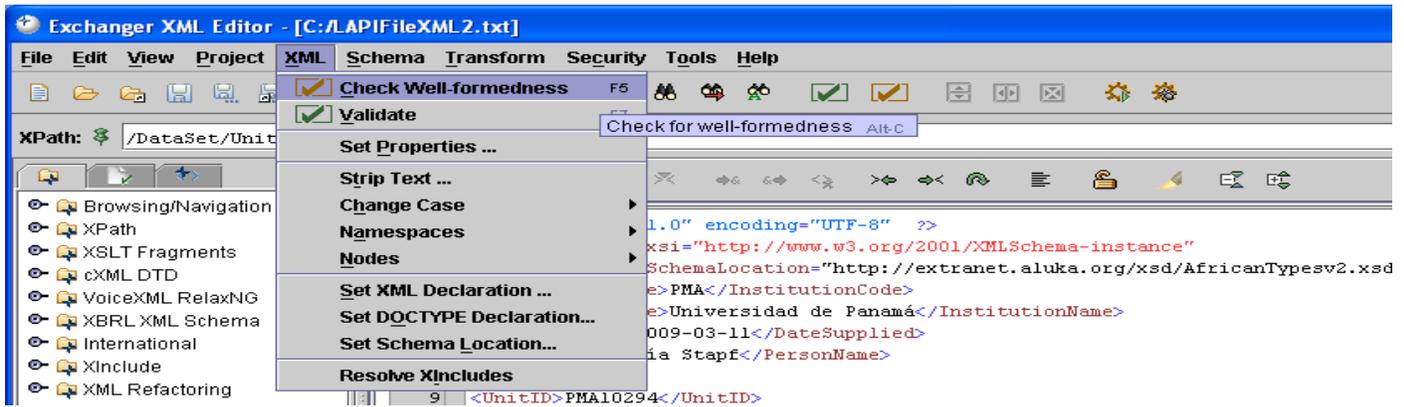
### Checking whether the XML file is well-formed

With your XML file open in Exchanger XML Lite 3.2, use the "Well-Formed" button to check whether the active document is well-formed.

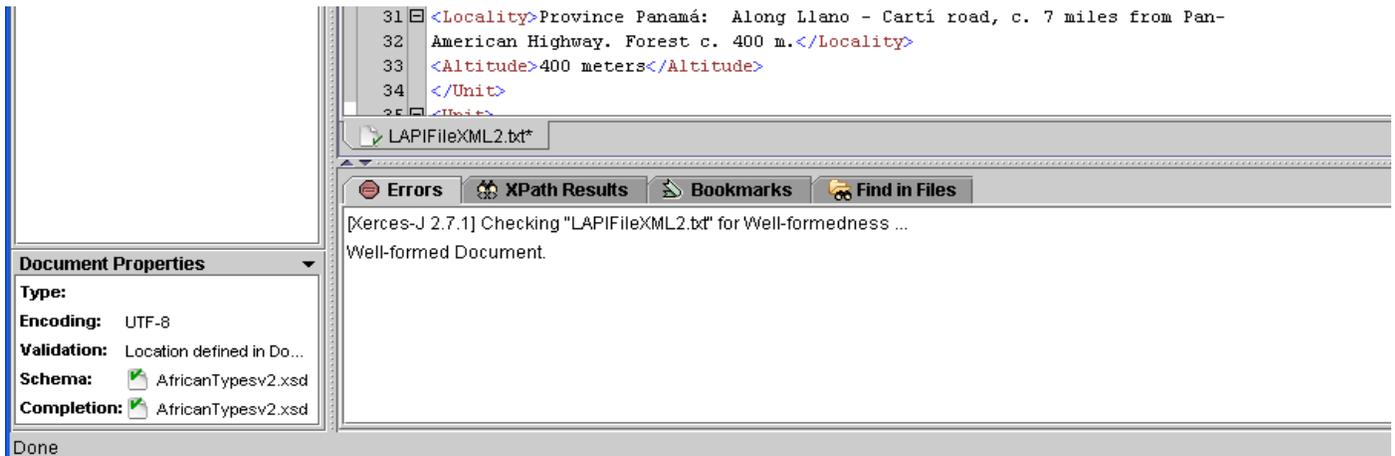
1. Click on the red  button.



Or, click on "XML>Check Well-formedness"; or press "F5."



2. A message will appear at the bottom of the screen indicating whether the file is a "Well-formed Document."



### Other XML file-validation tools/websites

There are other websites or software tools that can be used to validate the XML metadata file:

- XML Cooktop, <http://www.xmlcooktop.com/>. Free download; Windows only.
- XML Fox, <http://xmlfox.com/download.htm>. Free download; Windows only.
- XML Buddy, <http://www.xmlbuddy.com/>. Free download; Eclipse-based.
- Website <http://www.xmlvalidation.com>
- Website <http://www.validome.org/xml/>