

Chapter 16: Introduction to XML and DTD Files

The configuration files for the **CRL** desktop as well as for the inquiry and forms features are provided in **XML** format. In this chapter we provide the information you need in order to understand **CRL**'s **XML** configuration files. This chapter is not intended as a general **XML** reference.

16.1 XML Compared to HTML

XML is most easily described by starting with **HTML**. **HTML** markup consists of *tags* that define text structures, formatting, and so on, for documents (e.g., `<H1> . . . </H1>` for top level header), and *attributes* that set properties for those tags (e.g., `<H1 ALIGN="CENTER">`). **HTML** is limited to documentation. It is not case-sensitive.



XML is a configurable, extensible language with syntax similar to **HTML**. **XML** is case-sensitive. **XML** can be used for documentation, but it is more versatile than just that. In particular, the **CRL** code, written in **Java**, is designed to read in a configuration file supplied in **XML** format. *Elements* in **XML** are analogous to tags in **HTML**; they represent structures or desired behavior. *Attributes* define properties of elements. Elements are used with attributes in **XML** the same way as tags are in **HTML**:

```
<ELEMENT ATTRIBUTE="value">data</ELEMENT>
```

The configuration files for **CRL** are provided in *valid XML* (as opposed to *well formed XML*)¹. A file written in valid **XML** is based upon a document type definition (DTD) file. As of V1_7_06 the DTD is embedded in the **XML** files. When you install **CRL**, the following **XML** files for **CRL** itself and for its inquiry feature (described in section 8.2 *Inquiries*) are stored under the `LogBook_admin` directory:

- `LogBookConfig.xml`
- `LogBookInquiryConfig.xml`

1. There are many references on the Web which discuss valid and well formed XML. In writing this section, the author referenced http://pdbeam.uwaterloo.ca/~rlander/XML_Tutorial/.

The **XML** files (with embedded DTDs) for forms¹ are located under a separate directory, set by the `Logbook.file_location.forms_directory` in the properties file.

The DTD file defines the *element types* and the *attribute lists* that can be used in the corresponding **XML** file as elements and attributes, respectively, in markup declarations.

In order to edit an **XML** configuration file effectively, you need to understand both it and the DTD so that you know what elements/attributes you're allowed to include, and what they mean. The DTD and the **CRL** code have been developed in an integrated fashion and are highly dependent upon each other.

Do not edit the DTD portion of the files! It will cause CRL to malfunction or become inoperable!

The **XML** configuration files can be edited at will as long as the changes conform to the DTDs, and as long as information corresponding to hard-coded items/functions in the **CRL** code are not changed. Use a good **XML** authoring tool rather than a straight text editor in order to ensure that your edits are valid.

16.2 Let's Get a Feel For It...

As we said, **XML** can be thought of as **HTML** with user-configurable elements and attributes. So, before we get too deep, here's a sample scenario intended to give you a feel for how these entities are defined and used.

In the DTD file, we have the following element type declaration for `EntryInputPage`, which corresponds to a page on the **CRL** desktop.

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
<!ATTLIST EntryInputPage
    class CDATA #REQUIRED
    title CDATA #REQUIRED
    tooltip CDATA #REQUIRED>
```

Don't worry about all the syntactic details yet, just notice that it defines `EntryInputPage`, that `Keyword`, `Menu` and `ToolBar` appear in the declaration (it turns out they are also element types, defined in the same DTD), and that `EntryInputPage` has a list of attributes (`class`, `title` and `tooltip`).

1. The forms definition files are not written in "valid XML"; they do not strictly follow a DTD file grammar.

In the **XML** configuration file, this `EntryInputPage` element type can be used to define a page on the desktop that allows data entry. The `EntryInputPage` element shown below (taken from the default **CRL** configuration file and edited for brevity) includes the listed attributes. It also includes `Keyword`, `Menu` and `ToolBar` elements.

```
<EntryInputPage class="logbook.Page_EntryInput" title="Tutorial" tooltip="Tutorial">

  <Keyword type="page">
    Tutorial,
  </Keyword>

  <Menu name="Tutorial" >
    <Topic class="logbook.MyInternalTopicFrame" name="General Log"
      double-click="logbook.logentry.LogEntryText" size="90%,90%"
      offset="5%,5%" command="add_to_Menu_Page">
      <Keyword type="category">
        General_Log
      </Keyword>
    </Topic>
    ...

  <ToolBar>
    <ToolButton class="logbook.logentry.LogEntryImage" tip="Drag and Drop
      MiniBooNE Detector Button to Selected Container"
      image="images/entryinputpages/boone.gif"
      parml="http://www-boone.fnal.gov/images/illustration.gif" command1="" />
    <ToolButton class="logbook.logentry.LogEntryText" tip="Drag and Drop Text
      Button to Selected Container" image="images/entryinputpages/Text.gif"
      command1="" />
    ...
  </ToolBar>

</EntryInputPage>
```

16.3 Element Types and Attributes in the DTD File

16.3.1 Element Type Declarations

In **CRL**, elements typically define items on the desktop, e.g., a desktop page, a menu, a toolbar, and so on. Each element type is defined in the (embedded) DTD file in the following form:

```
<!ELEMENT NAME (CONTENT1-plus-OPERATOR,CONTENT2-plus-OPERATOR)>
```

In the **CRL** DTD file, the content area consists of either a sequence of other element types (these element types must also be defined in the DTD file), or the string `#PCDATA`, which indicates text. An operator (as in `CONTENT-plus-OPERATOR`) indicates whether the content is optional or required, and repeatable or not. The operators used in the **CRL** DTD file include:

*	optional and repeatable
?	optional, not repeatable
+	required and repeatable
no operator	required, not repeatable

Multiple content elements are separated by a comma and must be used in the order listed. Here is an example element type declaration from the **CRL** DTD file:

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,ToolBar?)>
```

This tells us that when used in the **XML** file, the element type `Page` can include the three element types `Keyword`, `Menu` and `ToolBar` as content (see example below). The operator symbols with them indicate that the element `Page`:

- *can* include one and only one occurrence of the element `Keyword`; if included, it must come before any occurrences of `Menu`
- *must* include at least one occurrence of `Menu`
- *can* include one and only one occurrence of the element `ToolBar`; if included, it must come after occurrences of `Menu`

Every element in the XML configuration file must be specifically closed at the end. E.g., every `<ElementName>` has a corresponding `</ElementName>`. If an element has no subelements, it may be constructed as `<ElementName ... />`, where the slash-angle bracket (`/>`) closes it.

16.3.2 Attributes and Attribute Lists

Each element type in the DTD file may have an associated *attribute list* defined. This list determines set of possible attributes that element can take in the **XML** file. In **CRL**, attributes typically define titles, roll-over text, file locations, and so on, and it is straightforward to assign values to them. A few attributes are more exacting; for example, values for the `class` attribute are restricted to those in the **CRL** class library, listed in section 17.3 *The CRL Java Class Libraries*.

The **CRL** DTD file includes only CDATA type attributes¹. This type of attribute is also called a *string attribute* since in **XML** markup it takes a text string as a value. A CDATA attribute list takes the form:

```
<!ATTLIST ELEMENT-TYPE
    ATTRIBUTE1 CDATA DEFAULT
    ATTRIBUTE2 CDATA DEFAULT
    ...>
```

1. There are several different kinds of attributes; defining them is beyond the scope of this document.

where (in the **CRL** DTD file) the `DEFAULT` value is one of the following:

- `#REQUIRED` (required)
- `#IMPLIED` (optional)

16.3.3 Example of Element Type Declaration plus Attribute List

Here is an example from the **CRL** DTD file of a declaration for the element type `EntryInputPage` followed by its corresponding attribute list:

```
<!ELEMENT EntryInputPage (Keyword?,Menu+,Toolbar?)>
<!ATTLIST EntryInputPage
    class CDATA #REQUIRED
    title CDATA #REQUIRED
    tooltip CDATA #REQUIRED>
```

This tells us that in the **XML** file, whenever the element `EntryInputPage` is used, it must provide values for all three listed attributes, e.g.:

```
<EntryInputPage
    class="logbook.Page_EntryInput"
    title="Tutorial"
    tooltip="Tutorial">
    ...
</EntryInputPage>
```

What these particular attributes do is discussed in Chapter 17: *The CRL Desktop Configuration File*.

