Getting Code Assistance in Eclipse when Creating ValidateThis XML Files

Posted At : July 19, 2010 3:56 PM | Posted By : Bob Silverberg
Related Categories: CFBuilder, Eclipse, ValidateThis

Although ValidateThis, my validation framework for ColdFusion objects, can accept metadata in a number of formats, my preferred method is via an XML file. This file defines all of the validation rules for a particular object, and can, obviously, be created using any text editor. If, like me, you are using Eclipse as your IDE (which includes users of ColdFusion Builder), you can enable code assist thanks to the XML Schema Definition (XSD) that I created for ValidateThis. This post will describe how to enable that feature.

What is an XSD?

XSD stands for XML Schema Definition. It is the successor to the Document Type Definition (DTD), providing a description of the required structure of a particular type of XML document. XSDs are themselves written in XML, which makes them both machine- and human-readable. They are therefore an excellent source of documentation about the type of XML document that they describe, and they can also be used to validate an XML document and to enable code assist when editing an XML document.

Enabling Code Assist in Eclipse

Many modern IDEs support code assist via XSDs when editing an XML file. Certain Eclipse XML plugins provide this feature, and it is enabled automatically simply by including the path to the XSD in the XML file itself. Here's an example from a ValidateThis rules definition file:

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

<validateThis xsi:noNamespaceSchemaLocation="validateThis.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<!-- rules defined in here -->

</validateThis>
```

The value of the *xsi:noNamespaceSchemaLocation* attribute is what tells the IDE where to find the XSD. In this example the XSD is in the same folder as the XML file, which is why the value of the attribute is simply the name of the XSD. You can also point to an XSD that lives online. In fact, there's an XSD available at http://www.ValidateThis.org/validateThis.xsd, so your xml file could include:

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

<validateThis xsi:noNamespaceSchemaLocation="http://www.ValidateThis.org/validateThis.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">

<!-- rules defined in here -->

</validateThis>
```

and you should be able to validate your XML file and get code assistance from your XML plugin.

Validating the XML File

Once you have a valid link to an XSD in your xsi:noNamespaceSchemaLocation attribute, you should be able to validate your document against the schema by either right-clicking inside the document, or right-clicking on the file name in the navigator, and choosing *Validate*. Note that you will also have inline validation, so if you change some code in your XML document so that it is no longer valid, that code should appear with a squiggly red line underneath it - an indicator that there is a problem.

The Aptana XML Editor Does Not Provide Code Assist

Unfortunately, the Aptana XML Editor, which is the one that comes bundled with ColdFusion Builder, does not provide code assist. This is really too bad, as, in my opinion, there isn't much point to having an XML plugin that does not provide this feature. You can still get code assist when using CFBuilder as long as you use a different XML editor.

Right-click on an xml file in the Navigator and choose *Open With*. You will see an option for *CF Builder XML Editor*, which is probably the default for your install. Do not choose that one as you will not get code assist. If you have CFBuilder installed as a plugin to Eclipse you should also see an option for *XML Editor*. If you choose that one you *should* get code assist. If you have CFBuilder installed in standalone mode, you may not see any other XML editors listed, in which case you'll need to install a new XML editor as a plugin in order to get code assist. But don't worry, this is a relatively simple task.

Installing a New XML Editor into Eclipse

As mentioned above, CFBuilder *is* Eclipse, so even though it doesn't come bundled with any other XML editors when installed in standalone mode, you can add a new one. The editor that is normally available to folks running CFBuilder as an Eclipse plugin is the one that is part of the *Web Tools Platform (WTP)*, so one of the easiest ways of getting a more useful XML editor is to install the WTP XML editor into your standalone CFBuilder. To do that, follow these steps:

1. From the *Help* menu, choose *Install New Software...*
2. If you haven't already added the Galileo Update Site to your available sites do so by clicking the *Add...* button, entering "Galileo Update Site" for the *Name* and "http://download.eclipse.org/releases/galileo" for the *Location*, and clicking *OK*. Otherwise, simply select the Galileo Update Site from your list of available sites.
3. After a moment you should see a list of available packages. The XML editor is part of the *Web, XML, and Java EE Development* package. You can place a check mark beside that entire package, to install all of the plugins, or you can expand the package using the little triangle, and just place a check mark beside the *Eclipse XML Editors and Tools* item and click *Next*.
4. You'll see a dialog telling you what is about to be installed. Click *Next*.
5. You'll see a dialog asking you to accept the licences for the plugins. Choose *I accept the terms of the licence agreements* and click *Finish*.
6. CFBuilder will download and install the plugins, after which you'll see a dialog suggesting you restart CFBuilder. This is a good idea, so click *Yes*.
7. CFBuilder will restart. You should now see an option for *XML Editor* in the Open With dialog when pointing to an XML file in the navigator, and choosing that option should enable code assist when editing your XML file.

Enabling code assist when editing ValidateThis Rules Definition files should make editing them easier and faster. I also plan on adding some Eclipse snippets to the VT distribution (an idea I'm stealing from MXUnit), for folks who prefer to work with snippets.