Fast Development of Widgets for IBM Connections 3

By Chuck Connell

Any software development task involves many iterations of the code/build/test cycle. It is important to be able to perform this cycle efficiently – so you can quickly add or change code, build and install a new executable, test it, and then repeat this process until you get the software right.

When I started creating widgets under IBM Connections, I found the development cycle to be extremely awkward. If the documentation is followed closely, there are at least 15 steps between the moment you make a code change and the start of testing that change in an updated widget. I followed this process several times, but felt there had to be a faster way. There is. It turns out you can do this *much faster*, with only one step between a code change and testing it!

Here is the standard development cycle for Connections widgets. The steps at the start are one-time only, but items in red need to be repeated between each code change and test.

> Define a "context root" for the widget under Websphere and remember the URL to it… Install, configure and learn Eclipse IDE for Java EE… Create a WAR (web archive) project within Eclipse and load it with an iWidget shell… Edit the widget XML file within the WAR project… Export the updated WAR… Install the WAR under Websphere at its context root… Remove any previous version of the widget from the user's widget page, the administrator's list of enabled widgets and the administrator's list of disabled widgets… Add the widget to the administrator's list of widgets and then enable it… The user adds the widget… The user tests the widget…

The key ideas for a faster process are: dispense with Eclipse, since it is being used as a glorified text editor; manually place the widget in a location where the HTTP server can see it, skipping Websphere WAR installation; do not cache the widget in Connections, so you can edit it in place.

Here is the better process, which I have been using successfully for my own programming. As before, steps at the start are one-time only. Just the item in red is repeated between a code change and its test.

1. Find the "home directory" of the Connections HTTP server. This is usually c:\IBM\HTTPServer\htdocs. Create a subdirectory here named "widgets".

2. Put an iWidget XML shell into the htdocs\widgets directory. You can steal the shell from my article about Google gadgets for Connections.

3. Add the widget shell to the Connections administrator's list of widgets and enable it. The URL is http://your-connections-server/widgets/widget-name.xml. Do *not* check the installation option "cache widget descriptor".

4. The test user adds the widget to his/her home page. The widget should do nothing yet, since it is just a shell.

5. The programmer opens the installed widget XML file with a plain text editor such as Notepad, notepad++, or nano.

6. The programmer adds/changes code within the widget XML file.

7. The programmer saves the XML code change without closing the file, usually by pressing CTRL-S.

8. The user tests the widget, immediately seeing the new code.

This process requires *one step* (CTRL-S) between a code change and testing the new code.

Other supporting files (such as Javascript and HTML) that are referenced by the core XML file can be placed in the same widgets directory and edited/saved in the same way.

Chuck Connell is president of CHC-3 Consulting, which helps organizations with all the IBM Lotus products – Domino, Notes, Connections, Sametime, Traveler, LotusLive and others.