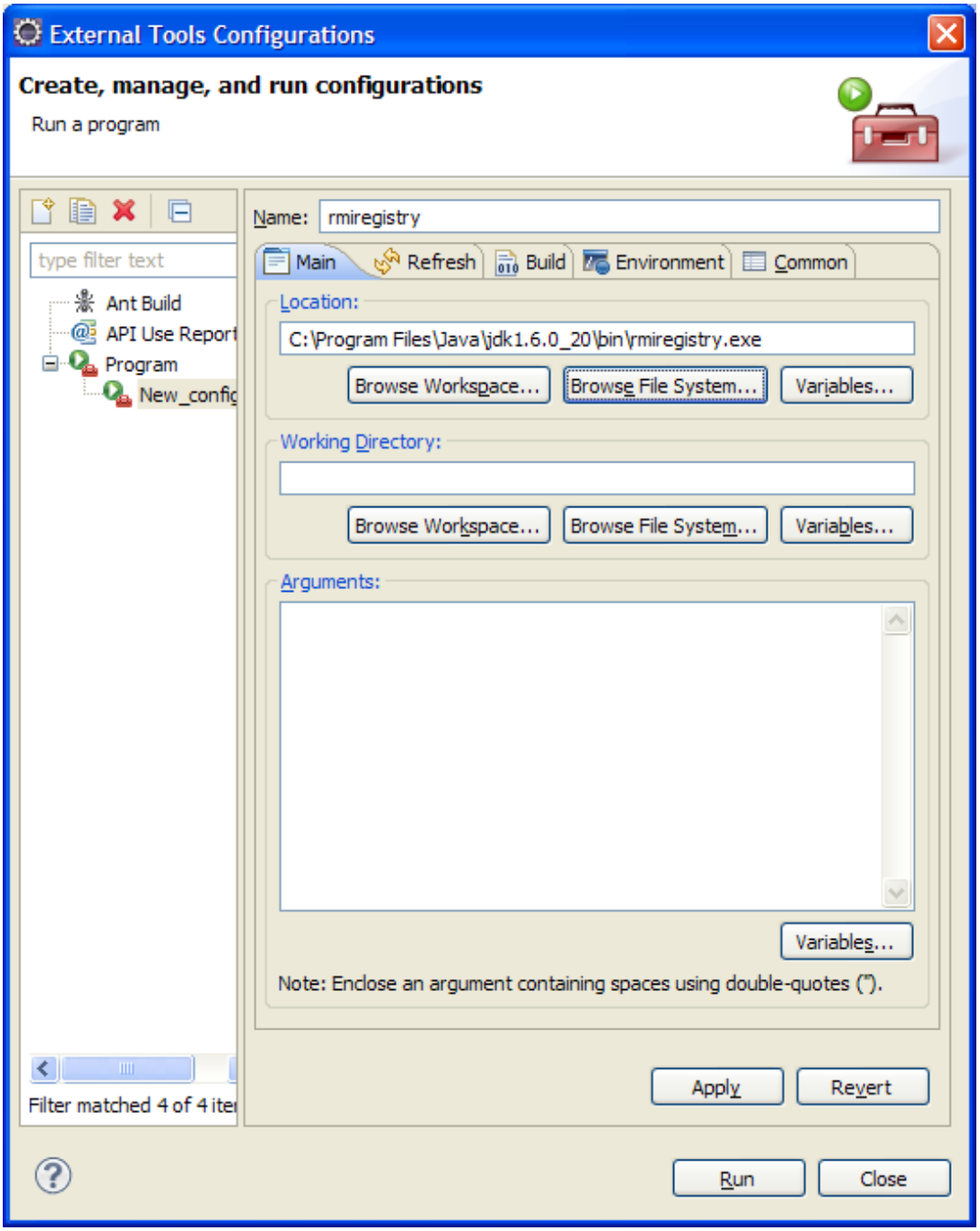
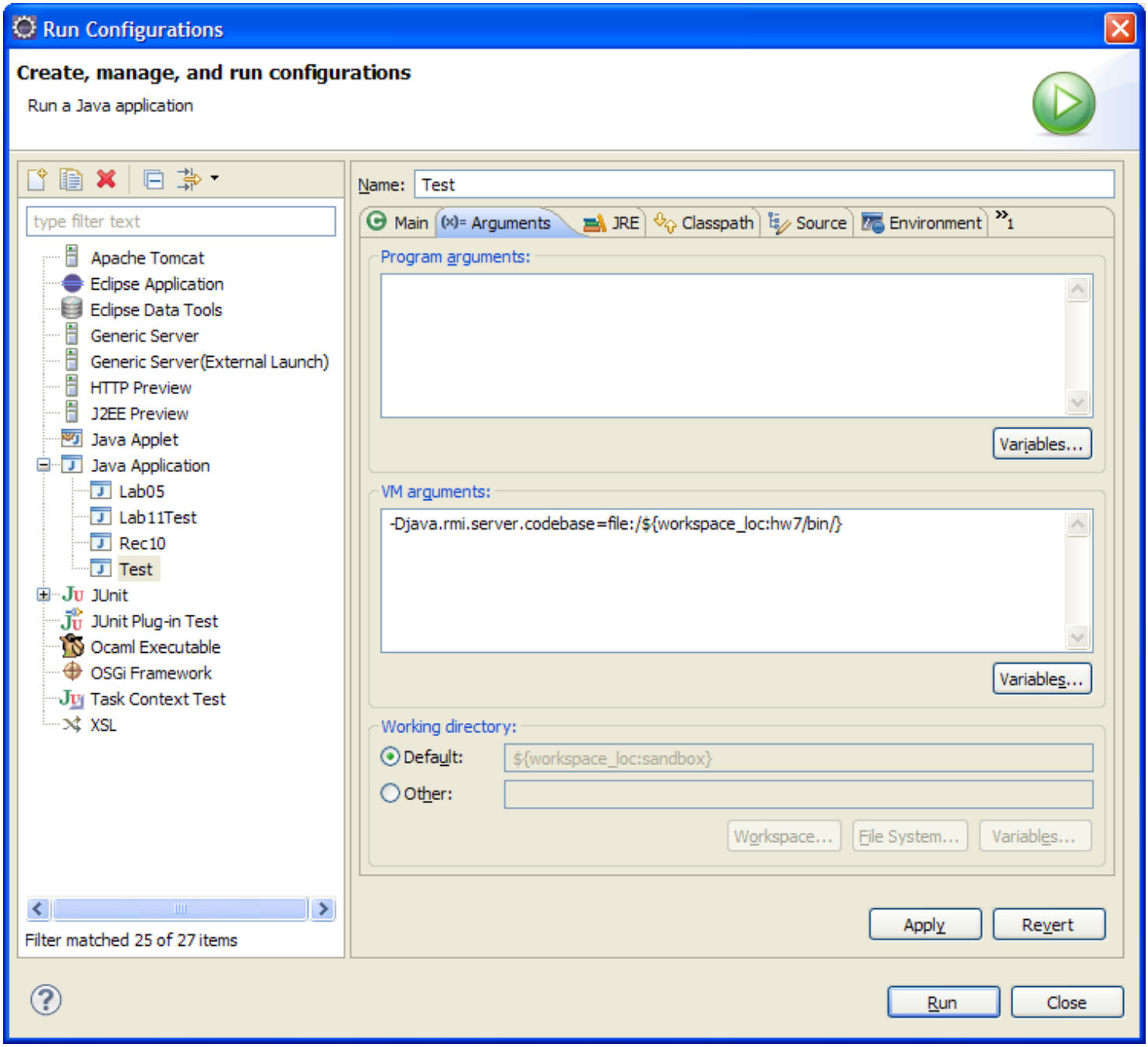
Running RMI in Eclipse

To successfully set up RMI communications, you need three components: `rmiregistry`, a server, and a client. Once `rmiregistry` and the server are running, the client should be able to connect to the server successfully. See generic RMI server-client implementations on how to connect a client to a server. This tutorial helps you set up such necessary components in Eclipse.

First, run `rmiregistry` as an external program in Eclipse. To do this, go to *Run* → *External Tools* → *External Tools Configurations...* from the menu. Create a new Program configuration, which will look like the figure below. *Browse File System...* to locate your `rmiregistry`, which is usually in the `bin` directory of the Java distribution. Save and run this external tool. `rmiregistry` can keep running in the background, but be sure to terminate it before you close Eclipse.
Caveat: If you have Java 6 Update 29, you will need to update it to a more recent version (e.g., Update 32) because its `rmiregistry` disallows reading class files in the codebase (see below) for an erroneous security reason.

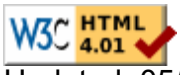


Next, go to the `main` method which spawns the server. You will need to provide a correct codebase for the server to run properly. To do this, go to *Run → Run Configurations...* from the menu. Create a new Java Application configuration, which will look like the figure below. (If you have run this `main` method before, go to its configuration instead.) You will need to provide an argument to the Java virtual machine as in the figure, where `${workspace_loc:hw7/bin/}` refers to the directory containing `.class` files of your server implementation. For instance, if your server implementation `ServerImpl` is in package `myname`, the desired directory should contain a subdirectory `myname` containing `ServerImpl.class`. The Eclipse's `workspace_loc` variable refers to the workspace directory.
Caveat: If your path contains a space, this might not work. Consider renaming your directory to eliminate spaces.



With `rmiregistry` running, you should now be able to run the server. In contrast with `rmiregistry`, you should kill your server even if it fails to run, because the `main` method will not exit.

Now you should be able to run your client applet successfully.



Updated: 05/03/2012