Developing the EZScript Toolkit

Writing Global Methods

1. Prerequisites
    - Download EZScript source code from:
    - http://openwonderland-modules.googlecode.com/svn/trunk/unstable
    - Create a new wonderland module using the samplemodule project or add to your favorite existing module.
2. Background
    a. The ScriptMethodSPI class is an interface, which all EZScript global script methods must implement. Classes implementing this interface should also annotate itself with the @ScriptMethod annotation. Annotating classes in this way allow for global script methods to be written and discovered in any arbitrary module that is uploaded to the Open Wonderland web server. Cells need not be rewritten to support scripting in this regard, rather a cell's module should be extended by adding an additional class. The extension of a module in this way, should not effect the non-scripting performance of the module.
    b. The ScriptMethodSPI has four methods which need to be implemented: getName, getDescription, setArguments, and run.
        i. getName() returns a string and will be the way the script library identifies and documents this method as a supplement to the JavaScript function that gets created for use in scripts.
        ii. getDescription() returns a string and will be the description associated with the name returned in getName() in the script library.
        iii. setArguments(Object args[]) is a utility method that translates the arguments passed through the JavaScript function to our java method. Traditionally, a fair amount of typecasting will occur here.
        iv. run() is where the bulk of the method's action should occur using the fields that were initialized in the setArguments() method.
    c. The ScriptMethodSPI is a great utility if you have an action that does not return a value, but what if you want to write a JavaScript function that returns something? An additional interface was created for such a purpose. The interface is also paired with an annotation in the same way that ScriptMethod is paired with ScriptMethodSPI. The sister method and annotation are: ReturnableScriptMethodSPI and ReturnableScriptMethod.

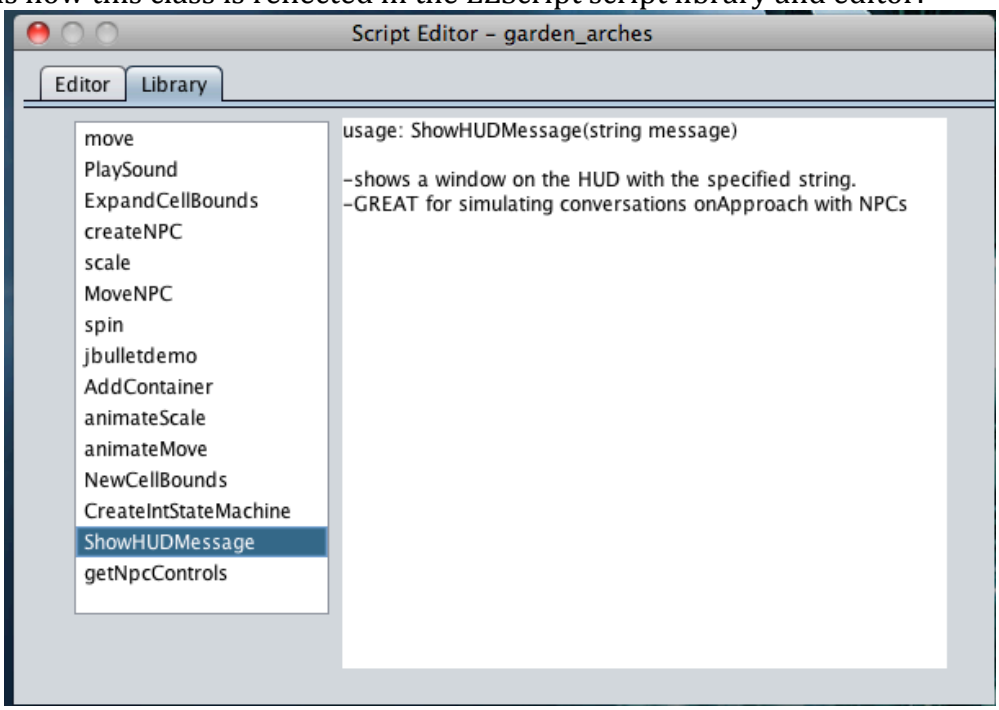
d. The ReturnableScriptMethodSPI interface has the same four arguments as in ScriptMethodSPI with the addition of one more: returns().
    i. returns() returns an Object-type based on the actions performed in run(). This is a great interface to embrace for creating global methods, which reveal access to Java objects hidden by the EZScript pipeline.
e. Notes on ReturnableScriptMethodSPI and ReturnableScriptMethod annotation
    i. Returns

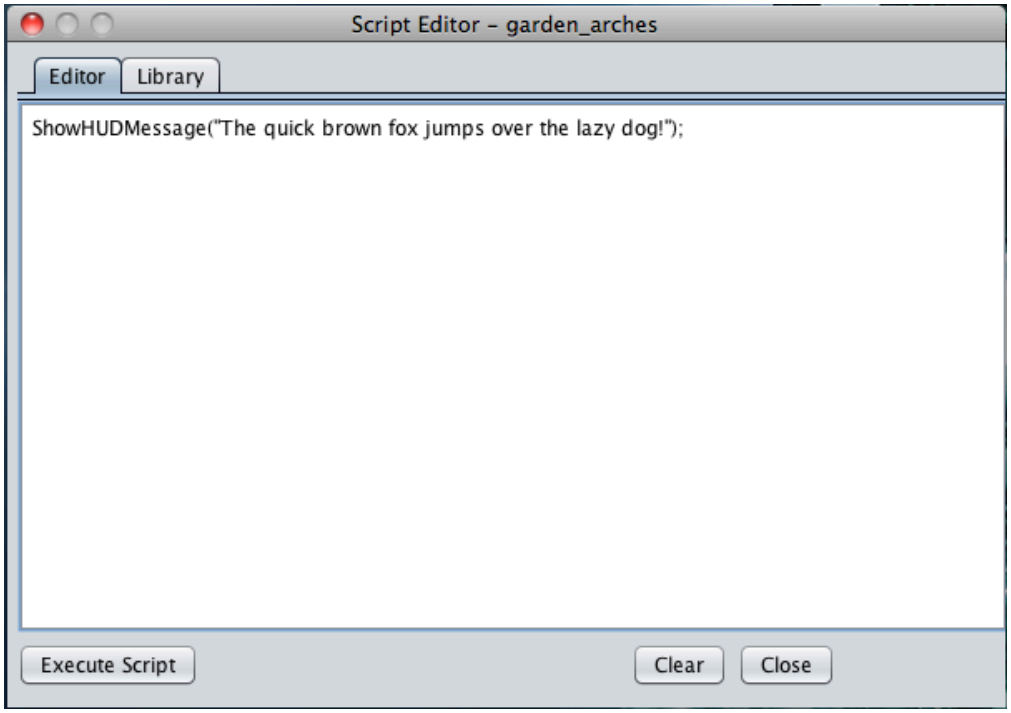
3. Examples
    a. The following example details a global method created to show a short message or notification in a window on the HUD.

```
@ReturnableScriptMethod
public class ShowHUDMessageMethod implements ReturnableScriptMethodSPI {

    HUD hud;
    HUDComponent component;
    HUDMessagePanel panel;
    String message;
    public String getFunctionName() {
        return "ShowHUDMessage";
    }

    public void setArguments(Object[] args) {
        message = (String)args[0];
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                hud = HUDManagerFactory.getHUDManager().getHUD("main");
                panel = new HUDMessagePanel(message);
                component = hud.createComponent(panel);
                panel.setComponent(component);
                component.setDecoratable(true);
                component.setPreferredLocation(Layout.CENTER);

                hud.addComponent(component);
            }
        });

    }

    public void run() {
        SwingUtilities.invokeLater(new Runnable() {
            public void run() {
                component.setVisible(true);
            }
        });
    }


    public Object returns() {
        return component;
    }
```

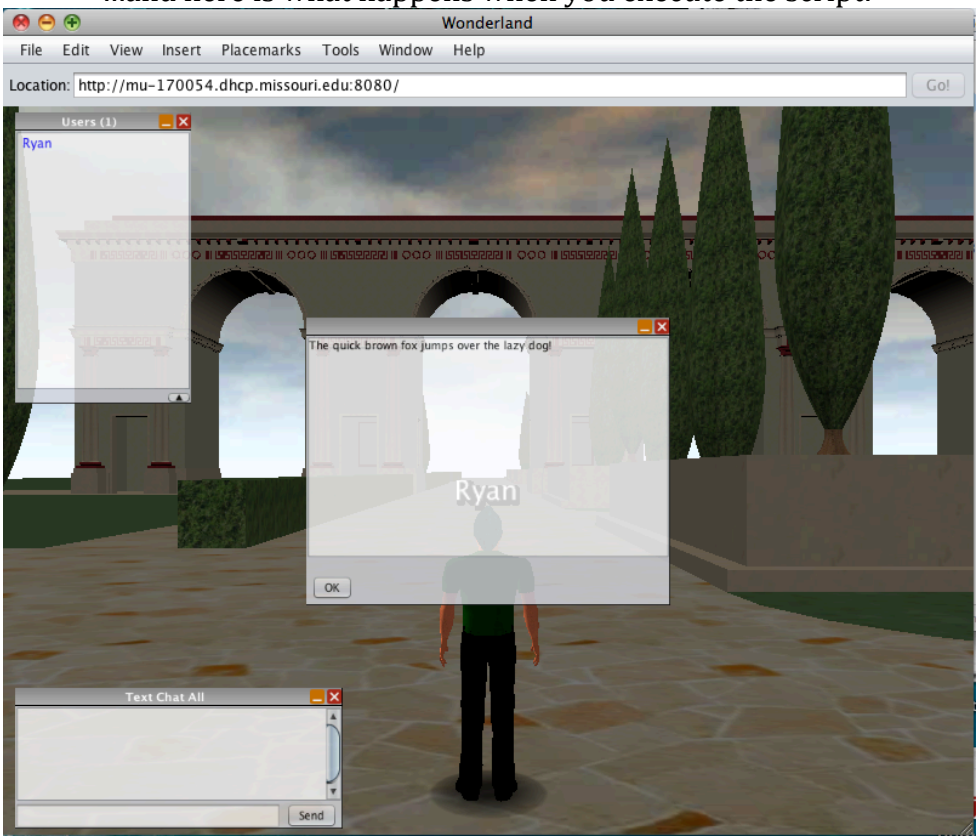
```
    public String getDescription() {
        return "usage: ShowHUDMessage(string message)\n\n"
                +"-shows a window on the HUD with the specified string.\n"
                +"-GREAT for simulating conversations onApproach with NPCs";
    }
}
```

Here is how this class is reflected in the EZScript script library and editor:





…and here is what happens when you execute the script:



Happy Scripting!