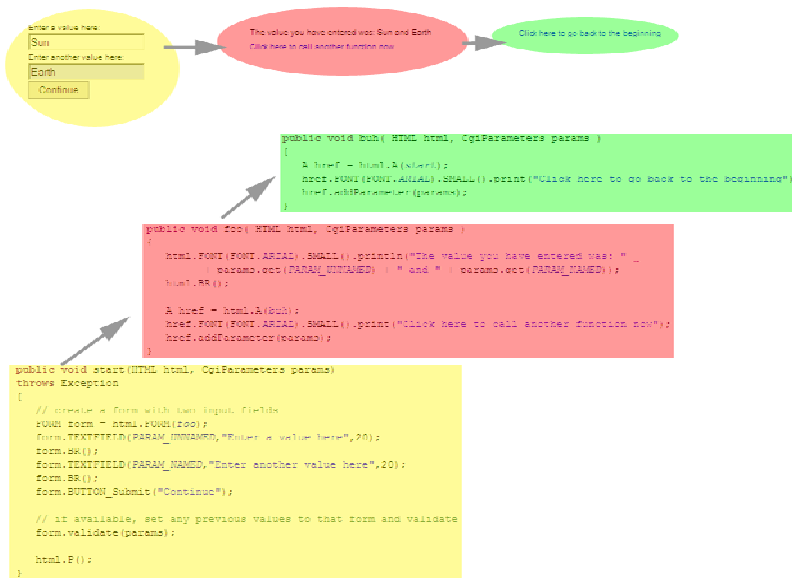


MethodReflection

MethodReflection objects are essential to the E4S framework. In a web application, there is always an interaction between content generated by an application but the application itself. This is exactly what the MethodReflection covers, and it is used in several ways within E4S. Two important issues are forms (<FORM>) and anchor tags (<A>) enabling links.

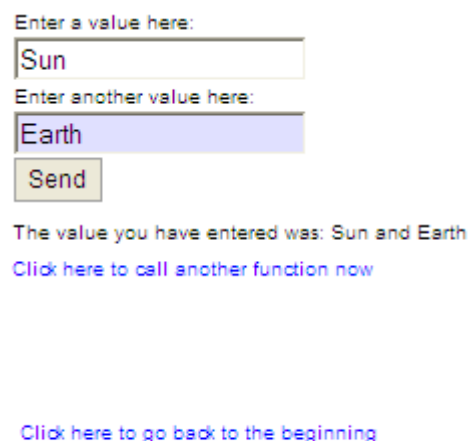


When the applications interacts on the HTML level, there must be related function in the servlet's Java code.

A form uses the CGI interface to pass values entered by the user to an URL. In the case of E4S, the URL is a Java function but the question now is how to find the right function, and beside track permissions and statistical performance information along.

This example is build up in three sections:

- 1) We build a form that enables input of two parameters, the parameters shall be passed to a function named "foo" in our Java module (it also could be in another module).
- 2) The function "foo" becomes involved after the user has clicked the "Send" button in our form. Parameters are passed in an object of type CgiParameters.
- 3) Now, we want to make an anchor tag and when the user clicks on this, another function "buh" shall become involved. Parameters shall also be passed to that function. And from "buh" back to our "start" function.



The screenshot shows a web form with two input fields: "Enter a value here:" (containing "Sun") and "Enter another value here:" (containing "Earth"). A "Send" button is below the second field. Below the form, the output is displayed: "The value you have entered was: Sun and Earth" and a blue link "Click here to call another function now". At the bottom, there is another blue link "Click here to go back to the beginning".

Code Example

```

package e4s.tutorial;

import e4s.html.*;
import e4s.html.input.extended.*;
import e4s.servlet.moduleImplementation;

/**
 * This is an example how to deal with MethodReflections.
 * MethodReflection cover the messaging of your application and each
 * MethodReflection is associated with a Java function. So we can treat
 * them as "normal" arguments in our application, for example if we
 * want an Anchor (<A>) tag to call another function. It is typically
 * for web applications, that either a form (<FORM>) or an anchor will
 * make branches to other functions. This example does not make
 * use of CSS definitions, so font attributes are included in the
 * application now.
 */
public class Example_MethodReflection extends moduleImplementation
{
    /**
     * This a placeholder for a Java function called "start". You can
     * find it below. From the tutorial framework, this is also the entry
     * into our module. Please note, that the variable is not initialized
     * (so it has a value of null) unless it will be initialized by the
     * E4S framework during servlet startup.
     */
    public static MethodReflection start = null;

    /**
     * This is another placeholder for another function.
     */
    public static MethodReflection foo = null;

    /**
     * This is another placeholder for another function.
     */
    public static MethodReflection buh = null;

    /**
     * This defines our first input field, we do not care it's name.
     */
    private final static InputFieldName PARAM_UNNAMED = InputFieldName.ANY();

    /**
     * This is another input field, we named it "ALPHA".
     */
    private final static InputFieldName PARAM_NAMED = new InputFieldName("ALPHA");

    /**
     * Our first function, becoming involved by the tutorial framework either
     * or when you click on that anchor tag generated in function buh.
     *
     * @param html the HTML object where we can let E4S render output
     * @param params an object containing all CGI parameters passed to that URL
     */
    public void start(HTML html, CgiParameters params)
    throws Exception
    {
        // create a form with two input fields and function "foo" as receiver
        FORM form = html.FORM(foo);
        form.TEXTFIELD(PARAM_UNNAMED, "Enter a value here", 20);
        form.BR();
        form.TEXTFIELD(PARAM_NAMED, "Enter another value here", 20);
        form.BR();
        form.BUTTON_Submit("Send");

        // set any previous values to that form
        form.setValue(params);

        html.P();
    }

    /**
     * This function receives data from the form defined in function start above.
     * We display the parameters, and call another function called "buh" using
     * an anchor tag.
     */
}

```

Code Example

```
* @param html the HTML object where we can let E4S render output
* @param params an object containing all CGI parameters passed to that URL
*/
public void foo( HTML html, CgiParameters params )
{
    html.FONT(FONT.ARIAL).SMALL().println("The value you have entered was: "
        + params.get(PARAM_UNNAMED) + " and " + params.get(PARAM_NAMED));
    html.BR();

    A href = html.A(buh);
    href.FONT(FONT.ARIAL).SMALL().print("Click here to call another function now");
    href.addParameter(params);
}

/**
 * Offer an back button to function "start" here, pass the parameters.
 *
 * @param html the HTML object where we can let E4S render output
 * @param params an object containing all CGI parameters passed to that URL
 */
public void buh( HTML html, CgiParameters params )
{
    A href = html.A(start);
    href.FONT(FONT.ARIAL).SMALL().print("Click here to go back to the beginning");
    href.addParameter(params);
}
}
```

See <http://www.element4solution.com> for details
©2007 door2solution software gmbH