JavaScript and CSS support

JavaScript and CSS support

(You might want to check out Adding Javascript or CSS using a Resource also.)

Overview

There are two ways of adding CSS/JavaScript support to a Wicket page. The first is by using the <wicket:head> section, while the other is by simply providing a link to a CSS/JavaScript page.

Contributing to the page head

The output of a Wicket html page is composed by assembling various things (borders, panels, page extensions, and of course pages themselves). Anything in a "normal" page is automatically contributed to the output. The same cannot be said, however, for borders, panels, or page extensions.

These components (identified respectively by <wicket:border, wicket:panel>, and <wicket:extend> tags) allow the use of a <head> section, but by default, the head is only for previewing. In other words, anything you add to the <head> in one of these types of components will NOT be contributed to the final page output unless you explicitly add it yourself.

The best way to contribute something in the head (such as JavaScript and CSS) to component markup is with the <wicket:head> tag. This will allow pages to use your component without being concerned with external dependencies.

In general terms, when you create a panel and want to contribute something to the page head, you would use this type of structure:

```
<wicket:head>panel header</wicket:head>
<wicket:panel>
    panel
</wicket:panel>
```

So, assuming that your page output looks something like this:

```
<html>
    <body>
        ...
    </body>
</html>
```

The output generated, including the contribution to head made by the panel above, will look something like:

```
<html>
    <head><wicket:head>panel header</wicket:head></head>
    <body>
        <span wicket:id="panel"><wicket:panel>
            panel
        </wicket:panel></span>
    </body>
</html>
```

Note: since the head of a "normal" page is automatically
contributed to the output, and therefore the
<wicket:head> tag would be redundant, use of this
tag on a "normal" page will throw an exception.

Using <wicket:head> to contribute CSS

If we are able to use the <wicket:head> section above to contribute to the page model, then we can certainly use it to reference a CSS resource!

First, the html:

```
<html>
    <wicket:head>
        <link wicket:id="mycss" rel="Stylesheet" type="text/css" href="styles/main.css"/>
    </wicket:head>
    <wicket:panel>
        panel
    </wicket:panel>
</html>
```

And now, the java:

```
import wicket.markup.html.WebMarkupContainer;

...

WebMarkupContainer css = new WebMarkupContainer( "mycss" );
add( css );
```

That's it!

Customizing CSS output based on some parameters

You can customize CSS output based on various parameters, such as local, browser agent, etc. The easiest way is to use an anonymous class as follows.

In your html:

```
<link wicket:id="mycss" rel="Stylesheet" type="text/css" href="styles/main.css"/>
```

In your java:

```
import wicket.markup.html.WebComponent;
import wicket.model.IModel;
import wicket.model.Model;
import wicket.AttributeModifier;
import wicket.Component;

...

WebComponent c = new WebComponent( "mycss" );
IModel model = new Model()
{
    public Object getObject( Component c )
    {
        if ( someConditionIsTrue )
            return "stylesheetx.css";
        else
            return "stylesheety.css";
    }
};
c.add( new AttributeModifier( "href", model ) );
add( c );
```

A word of caution about the <body> tag

There is one more thing to mention. The onLoad attribute of <body> is sometimes used to initialize certain functionality. E.g.

```
<wicket:head>panel header</wicket:head>
<body onload="function()">
    <wicket:panel>
        panel
    </wicket:panel>
</body>
```

Wicket will not (anymore) append the <body> onLoad attribute to the Page's onLoad attribute. To append javascript to the onLoad, have your component implement IHeaderContributor and add it there.

Linking directly to a CSS page

If, for some reason, you want to directly link to your CSS/JavaScript page, this is also possible with Wicket. The CSS/JavaScript file should be in the same source directory as the component's Java class and its HTML template.

To the CSS file(s), insert something like this in your markup:

```
<link wicket:id="pageCSS" rel="Stylesheet" type="text/css" href="css/page.css"/>
```

and this in your Page Java file for Wicket 1.1:

```
WebMarkupContainer pageCSS = new WebMarkupContainer("pageCSS");
add(pageCSS);
PackageResourceReference pageCSSResource = new PackageResourceReference(getClass(), "css/page.css");
pageCSS.add(new AttributeModifier("href", false, new   Model(urlFor(pageCSSResource.getPath()))));
```

or this for Wicket 1.2:

```
add(new StyleSheetReference("pageCSS", getClass(), "css/page.css"));
```