Including CSS resources

Generally there are two cases when it comes to including resources such as css: Wicket managed or not. Not managed by Wicket are the references like you would do with any other web framework, where you would put the css files in your web app directory (root of your war file), and reference that in your pages like:

```
<head>
   <link rel="stylesheet" type="text/css" href="style.css"/>
</head>
```

For packaged resources to be referenced directly from markup, we have <wicket:link>. This tag can be used in different contexts (like designating links to Wicket pages without having create an actual Wicket component for them), but in the context of <link> tags, the href attribute will be replaced by the URL to the packaged resource (style.css which should then be in the same package as where this markup file is)

```
<wicket:head>
  <wicket:link>
    <link rel="stylesheet" type="text/css" href="style.css"/>
  </wicket:link>
</wicket:head>
```

The `<wicket:head>` pair can be used in Panels, Borders and Fragments, and is used for doing a 'header contribution', meaning that the contents will be written to the <head> section of the page the component is placed in. If you are authoring a normal page, you don't need <wicket:head> tags but instead put it in the page's head section directly. The exception to this is when you use markup inheritance, and one of the pages to extend wants to contribute to the header of some extending page, and the header is not part of the <wicket:extend> region. In that case, you can use <wicket:head>.

Another way of achieving the same is via Java instead of markup:

```
add(HeaderContributor.forCss(MyClass.class, "style.css"));
```

In Wicket 1.4 HeaderContributor.forCss() is deprecated, you can use the code below:

```
add(CSSPackageResource.getHeaderContribution(MyClass.class, "style.css"));
```

You can add this to any component. Header contribution of style.css relative to MyClass will automatically be taken care off and double contributions will be filtered (ignored). This is my favorite, as you don't need components that have their own markup to do this, you can do this 'inside or outside' of a component, and it is just less typing.

In Wicket 1.5, you can use the code below:

```
@Override
public void renderHead(IHeaderResponse response) {
   response.renderJavaScriptReference(new PackageResourceReference(MyClass.class, "mypackage/myscript.js"));
   response.renderCSSReference(new PackageResourceReference(MyClass.class, "mypackage/mystyle.css"));
}
```