Using c:set in Facelets

By Roger Keays, 7 June 2008

This blog is about a common mistake people make with the Facelets c:set tag. The thing about c:set is that the Facelets version isn't like the JSP version. In fact, the Facelets version is just a convenient hack. In JSP, if you write:

```
<c:set scope="request" name="foo" value="${bar.property}"/>
```

the JSP TagHandler is responsible for evaluating the EL and putting the result in scope when the page is executed . However, in Facelets, you write:

```
<c:set name="foo" value="${bar.property}"/>
```

The first thing you notice is that no scope is specified. That is because the Facelets TagHandler doesn't actually evaluate the EL. Instead, it creates a VariableMapping for the name foo. It is just an alias, and leads to an often overlooked consequence:

- *Every time you use ${foo}, ${bar.property} is evaluated.*

This is generally okay, except when `${bar.property}` is an expensive operation such as a database lookup. If you don't want that behaviour, here is a simple Facelets `TagHandler` which only does the evaluation once:

```
/**
 * An implementation of c:set which evaluates the value expression when
 * the variable is set, creating a facelet-scoped attribute.
 *
 * This tag can also be used as child of a ui:include tag to pass attributes
 * into the next facelet scope/context.
 */
public class SetHandler extends TagHandler {

    private final TagAttribute var;
    private final TagAttribute value;

    public SetHandler(TagConfig config) {
        super(config);
        this.value = this.getRequiredAttribute("value");
        this.var = this.getRequiredAttribute("var");
    }

    /** evaluate and set the attribute in the facelet scope */
    public void apply(FaceletContext ctx, UIComponent parent) {
        ctx.setAttribute(var.getValue(ctx), value.getObject(ctx));
    }
}
```

There is one serious side effect of using this method, and that is the expression result becomes part of the view state. If your view state is being serialized (e.g. by client side state saving), that could mean trouble if the result is a large set of objects! It is necessary so that the result is available on a postback.

Alternatively, you could write a TagHandler to use the various servlet scopes (the same as the JSP version), but you should be aware thatTagHandlers are only instantiated when the view is built, so your expression won't be available on a postback if you use the request scope.

About Roger Keays

Roger Keays is an artist, an engineer, and a student of life. He has no fixed addressand has leftfootprints on 40-something different countries around the world. Roger is addicted to surfing. His other interests are music, psychology, languages, the proper use of semicolons, and finding good food.