Dynamic Struts Configuration

Dipl.-Inf. Manfred Wolff
mail@manfred-wolff.de
January 2004

abstract

In the current version struts 1.1 it is not possible to configure the framework "on the fly". But with a little modification it is no problem, to add actions, forwards etc. at runtime without starting the server or manipulate paths of forwards at runtime. This paper describes the steps.

problem description

The struts framework has a static module-configuration that will be freeze after the servlet will finished his configuration. If you cancel this freezing process you have whole access of all configuration issues of the application. There are situations where you will change the mapping information of an application without restarting the web container and without restarting you application.

But it is dangerous to change more than one attribute of the configuration. In a multi-threading environment many threads use the same configuration. So it it better to change whole configuration issues (like forward-config) regardless there is a timeslot, where no configuration exists.

solution

Step 1: Be care, that the configuration will not be freeze.

Create a own `ActionServlet`-class and overwrite the `init()`-method. Uncomment the both `freeze()`-methods. The new class I call `DynamicActionServlet`.

```
    /**
     * Initialize this servlet.  Most of the processing has been factored into
     * support methods so that you can override particular functionality at a
     * fairly granular level.
     *
     * @exception ServletException if we cannot configure ourselves correctly
     */
    public void init() throws ServletException {

        initInternal();
        initOther();
        initServlet();

        // Initialize modules as needed
        getServletContext().setAttribute(Globals.ACTION_SERVLET_KEY, this);
        ModuleConfig moduleConfig = initModuleConfig("", config);
        initModuleMessageResources(moduleConfig);
        initModuleDataSources(moduleConfig);
        initModulePlugIns(moduleConfig);
        //moduleConfig.freeze();
        Enumeration names = getServletConfig().getInitParameterNames();
        while (names.hasMoreElements()) {
            String name = (String) names.nextElement();
            if (!name.startsWith("config/")) {
                continue;
            }
            String prefix = name.substring(6);
            moduleConfig = initModuleConfig
                    (prefix, getServletConfig().getInitParameter(name));
            initModuleMessageResources(moduleConfig);
            initModuleDataSources(moduleConfig);
            initModulePlugIns(moduleConfig);
            //moduleConfig.freeze();
        }
        destroyConfigDigester();
    }
```

The `ActionServlet` does not offer a method to return his module-configuration. So you have make our own method to get the actually module-configuration (better have a method in the request-processor).

```
    /**
     * Gets the actually module-config out with the request information
     * @param request the HttpServletRequest
     * @return A <code>ModulConfg</code> instance.
     */
    public synchronized ModuleConfig getActuallyModuleConfig(
            HttpServletRequest request) {
        RequestUtils.selectModule(request, getServletContext());
        return getModuleConfig(request);
    }
```

You have to configure your struts application, so that this new servlet is loaded:

web.xml

```
<servlet>
  <servlet-name>action</servlet-name>
  <servlet-class>org.mwolff.struts.action.DynamicActionServlet</servlet-class>
    <init-param>
      <param-name>config</param-name>
      <param-value>/WEB-INF/struts-config.xml, </param-value>
    </init-param>
  <init-param>
    <param-name>debug</param-name>
    <param-value>2</param-value>
  </init-param>
  <init-param>
    <param-name>detail</param-name>
    <param-value>4</param-value>
  </init-param>
  <load-on-startup>4</load-on-startup>
</servlet>
```

Step 2: Manipulate your module-configuration

Now you have full access to your configuration in every action:

```
  public ActionForward execute(
          ActionMapping actionMapping,
          ActionForm actionForm,
          HttpServletRequest httpServletRequest,
          HttpServletResponse httpServletResponse)
          throws Exception {

      // Getting the module configuration
      DynamicActionServlet servlet = (DynamicActionServlet) getServlet();
      ModuleConfig config = servlet.getActuallyModuleConfig(
              httpServletRequest);

      // We want to manipulate the jsp path of this action
      ActionConfig actionConfig = config.findActionConfig("/Demo");
      ForwardConfig forwardConfig =
              actionConfig.findForwardConfig("hasSuceeded");
      forwardConfig.setPath("/pages/Demo2.jsp");

      // Everything is good
      return actionMapping.findForward("hasSuceeded");
  }
```

Optional

It is also possible to configure the action with a plug-in. At the end my plug-in and the `struts-config.xml`. You will see, the whole configuration is made in code.

```
/**
 * <p>A PlugIn for doing struts configuration at runtime.</p>
 *
 * @author Manfred Wolff
 * @version 1.0
 */
public class dynamicConfigPlugIn implements PlugIn {

    /**
     * <p>Receive notification that our owning module is being
     * shut down.</p>
     */
    public void destroy() {
        ; // No action required
    }


    /**
     * <p>Receive notification that the specified module is being
     * started up.</p>
     *
     * @param servlet ActionServlet that is managing all the
     *  modules in this web application
     * @param config ModuleConfig for the module with which
     *  this plug-in is associated
     *
     * @exception ServletException if this <code>PlugIn</code> cannot
     *  be successfully initialized
     */
    public void init(ActionServlet servlet, ModuleConfig config)
            throws ServletException {

        // Creating an ActionConfig Instance
        ActionConfig actionConfig = new ActionMapping();

        // Configure the instance
        actionConfig.setPath("/Demo");
        actionConfig.setType("org.mwolff.struts.action.DemoAction");
        actionConfig.setScope("session");

        // Creating an ForwardConfig Instance
        ForwardConfig forwardConfig = new ActionForward(
                "hasSuceeded", "/pages/Demo.jsp", false);

        // Adding the ForwardConfig to the ActionConfig
        actionConfig.addForwardConfig(forwardConfig);

        // Adding the ActionConfig to the ForwordConfig
        config.addActionConfig(actionConfig);
    }
}
```

```
<struts-config>

    <controller processorClass="org.apache.struts.action.RequestProcessor"/>
    <message-resources
        parameter="resources.application"/>

    <plug-in className="org.mwolff.struts.plugin.dynamicConfigPlugIn"/>

</struts-config>
```

Problems

The configuration of struts is stored in unsynchronized maps. It is not possible to change the configuration in a transaction. So if you have more than one change at one time the data may be not consistence for some threads. So you can remove a forward and add a forward with the same name, but there is a timeslot where other threads have no forward at all.

Looking to the future

Better were some struts modifications. There are good arguments to freeze the configuration, but it might be good, that the developer can choose if he will have those feature.

```
    <controller processorClass="org.apache.struts.action.RequestProcessor"
                freeze="false"
    />
```

Quite better is the way to store such information in a database. Then you have the possibility to write own `ActionMapping` Objects to do the mapping.