DevelopingComponents

Writing components for Cocoon

This is an introduction for Java developers who would like to write their own components for Cocoon.

There are generally two types of components that you can use in Cocoon:

- generic components, not tied to Cocoon. These could be reused in other, non-Cocoon applications.
- Cocoon-specific components. These are mainly the components used in the sitemap, such as generators, transfomers and actions.

Cocoon is based on the Jakarta Avalon component framework. Hence all components in Cocoon, and thus also your own components, will always be Avalon components. It is important that you understand Avalon, otherwise you won't understand the way your classes are handled within the system, and you won't know how to do certain things like accessing other components. Fortunaly, the basics of Avalon are actually quite simple and straightforward. There's a very good manual about Avalon titled Developing with Avalon, which you should read right now. For the impatient, I'll probably provide a short Avalon introduction at a later time.

Component containers in Cocoon

If you have read the Developing With Avalon guide, you should already now that components are managed by containers. In Cocoon, a generic component container is used, the Excalibur Component Manager (ECM for short). There are multiple of these containers:

- One main container, created in the Cocoon class
- One container for each sitemap

The main container is configured by the cocoon.roles file included with cocoon (inside the cocoon jar) and the cocoon.xconf file (usually found in your webapp's WEB-INF directory). This container manages core Cocoon components such as the "sitemap" and the "eventpipeline", the entityresolver, the datasources, and so on. You can also add your own (non-sitemap) components here.

The sitemap-specific containers manage the sitemap-specific components (such as generators, transformers, actions, ...). They are configured by the map:components section of the sitemap.

The containers are arranged hierarchically, with the parent container of the root sitemap being the main container.

The picture below also illustrates this.

Component container overview

Writing components

- WritingPipelineComponents (generators, transformers, serializers)
- WritingNonPipelineSitemapComponents (actions, matchers, selectors, readers)