FlowHelloWorld

"Hello World" Simple Flow Sample

- TARGET-AUDIENCE: ***beginner*** advanced expert
- COCOON-RELEASES: 2.1.3, 2.1.4
- DOCUMENT-STATUS: ***draft*** reviewed released

What you will get from this page

An example of the basics required to work with flow and JXTemplates.

Your basic skills

- You have basic knowledge about XML
- You have setup Cocoon (and maybe looked at samples)
- You know how to setup a new Cocoon application - see BeginnerSimpleWebappOrganisation

Technical prerequisites

- You need a cleanly installed version of Cocoon. Please refer to BeginnerInstallation for further information.

Setting Up

You should create a new subdirectory under the Cocoon installation directory called, for example `HelloFlow`. Under this create a `flow` subdirectory and d`ocuments` subdirectory.

The Sitemap

The following code should be saved in a `sitemap.xmap` file in the `HelloFlow` directory.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- "Hello World" - a simple flow and jxt sample -->
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">
<map:components>
  <map:generators default="file">

    <!-- use JXTemplateGenerator to insert flow variables into page content -->
    <map:generator label="content,data" logger="sitemap.generator.jxt"
                   name="jxt" src="org.apache.cocoon.generation.JXTemplateGenerator"/>

  </map:generators>

  <!-- handle the processing of any javascript -->
  <map:flow-interpreters default="JavaScript"/>

  <map:transformers default="xslt"/>

  <map:serializers default="html"/>
  <map:matchers default="wildcard"/>
  <map:selectors default="browser"/>

  <map:actions/>
  <map:pipes default="caching"/>
</map:components>

<map:views/>
<map:resources/>
<map:action-sets/>

<map:flow language="javascript">
  <!-- Hello World will use the javascript functions defined in flow/hello.js -->
  <map:script src="flow/hello.js"/>
</map:flow>

<map:pipelines>
  <map:component-configurations>
    <global-variables/>
  </map:component-configurations>

  <map:pipeline>

    <!-- call the function hello() in flow/hello.js -->
    <map:match pattern="">
      <map:call function="hello"/>
    </map:match>
    <map:match pattern="hello">
      <map:call function="hello"/>
    </map:match>

    <!-- use JXtemplate to generate page content -->
    <map:match pattern="*.jxt">
      <map:generate type="jxt" src="documents/{1}.jxt"/>
      <map:serialize type="xhtml"/>
    </map:match>

  </map:pipeline>
</map:pipelines>
</map:sitemap>
```

The Flow Script

The following code should be saved in a `hello.js` file in the `flow` directory.

```
function hello() {
  var name =  "World";
  cocoon.sendPage("hello.jxt", {"name" : name} );
}
```

The Generated Page

The following code should be saved in a `hello.jxt` file in the `documents` directory.

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<head>
  <title>Cocoon Flow Hello World</title>
</head>
<body>
  <h1>Hello ${name}!</h1>
</body>
</html>
```

What did we achieve?

- Introduced the necessary components into the sitemap to allow for flow handling; these have been clearly indicated and can be added to existing or new applications that require flow.
- Demonstrated how flow can create variable data and pass this to a template generator.
- Showed how simple it is to include flow-generated data in a web page.

Links to other information sources

- WhatIsFlow?
- GettingStartedWithFlow

---

page metadata

- AUTHOR: DerekH
- REVIEWED-BY: (none yet)
- REVIEWER-CONTACT:BR