

**CS 683 Emerging Technologies**  
**Fall Semester, 2004**  
**Doc 16 Cocoon Flow**  
**Contents**

References.....	2
Cocoon Flow .....	3
Hello World .....	4
Guessing Game Example .....	10
Flow Control Parts .....	17
Additions to Site Map.....	18
Flowscript .....	19
JXTemplate .....	20
JPath .....	21
Velocity Generator.....	22
Examples .....	23
Using Java – Hello World .....	23
Using Velocity.....	27
Look Ma No XSL .....	32

Copyright ©, All rights reserved. 2004 SDSU & Roger Whitney,  
5500 Campanile Drive, San Diego, CA 92182-7700 USA.  
OpenContent (<http://www.opencontent.org/opl.shtml>) license  
defines the copyright on this document.

## References

Cocoon Flow Documentation

<http://cocoon.apache.org/2.1/userdocs/flow/index.html>

## **Cocoon Flow**

Continuations using JavaScript

Require Cocoon 2.1

Java code can be called in flows

# Hello World

<http://bismarck.sdsu.edu:9006/cocoon/flowHello/hello.html>

## Files

- flowHello/flow/example.js
- flowHello/flow/hello.jx
- flowHello/flow/bye.jx
- flowHello/sitemap.xmap

## flowHello/flow/example.js

```
function hello() {  
  var message = "Hi Mate";  
  var now = new Date();  
  cocoon.sendPageAndWait( "hello.jx",  
    { message:message, date: now });  
  cocoon.sendPage("bye.jx");  
}
```

## flowHello/flow/hello.jx

```
<?xml version="1.0"?>  
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">  
<head>  
  <title>Hello World</title>  
</head>  
<body>  
<h1>${message}</h1>  
It is now ${date.toString()}.  
  
<form action="${cocoon.continuation.id}.kont" method="post">  
  <input type="submit" value="Done" />  
</form>  
</body>  
</html>
```

**flowHello/flow/bye.jx**

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<head>
  <title>GoodBye World</title>
</head>
<body>
<h1>So Long</h1>
<form action="{cocoon.continuation.id}.kont" method="post">
  <input type="submit" value="Now What" />
</form>
</body>
</html>
```

## flowHello/sitemap.xmap

```
<?xml version="1.0" encoding="UTF-8"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">

<map:components>
  <map:generators default="file">
    <map:generator label="content,data"
      logger="sitemap.generator.jx"
      name="jx"
      src="org.apache.cocoon.generation.JXTemplateGenerator"/>
  </map:generators>
  <map:transformers default="xslt"/>
  <map:serializers default="html"/>
  <map:matchers default="wildcard"/>
  <map:selectors default="browser">
    <map:selector name="exception"
      src="org.apache.cocoon.selection.XPathExceptionSelector">
      <exception name="invalid-continuation"
class="org.apache.cocoon.components.flow.InvalidContinuationExc
ption"/>
        <exception class="java.lang.Throwable" unroll="true"/>
      </map:selector>
    </map:selectors>
  <map:actions/>
  <map:pipes default="caching"/>
</map:components>
```

## flowHello/sitemap.xmap Continued

```
<map:views/>
```

```
<map:resources/>
```

```
<map:action-sets/>
```

```
<map:flow language="javascript">
```

```
  <map:script src="flow/example.js"/>
```

```
</map:flow>
```

```
<map:pipelines>
```

```
  <map:component-configurations>
```

```
    <global-variables/>
```

```
  </map:component-configurations>
```

```
<map:pipeline>
```

```
  <map:match pattern="hello.html">
```

```
    <map:call function="hello"/>
```

```
  </map:match>
```

```
<map:match pattern="*.jx">
```

```
  <map:generate type="jx" src="documents/{1}.jx"/>
```

```
  <map:serialize type="xhtml"/>
```

```
</map:match>
```

```
<map:match pattern="*.kont">
```

```
  <map:call continuation="{1}"/>
```

```
</map:match>
```

**flowHello/sitemap.xmap Continued**

```
<map:handle-errors>
  <map:select type="exception">
    <map:when test="invalid-continuation">
      <map:generate
        src="documents/invalidContinuation.html"/>
      <map:serialize type="xhtml"/>
    </map:when>
  </map:select>
</map:handle-errors>
</map:pipeline>

</map:pipelines>

</map:sitemap>
```

## Guessing Game Example

From <http://cocoon.apache.org/2.1/userdocs/flow/tutor.html>

<http://bismarck.sdsu.edu:9006/cocoon/game/>

### Files

- game/flow/game.js
- game/documents/guess.jx
- game/documents/success.jx
- game/sitemap.xmap

## The Flow

### game/flow/game.js

```
function main() {  
    var random = Math.round( Math.random() * 9 ) + 1;  
    var hint = "No hint for you!"  
    var guesses = 0;  
    while (true) {  
        cocoon.sendPageAndWait("guess.jx", { "random" : random,  
            "hint" : hint, "guesses" : guesses } );  
        var guess = parseInt( cocoon.request.get("guess") );  
        guesses++;  
        if (guess) {  
            if (guess > random) {  
                hint = "Nope, lower!"  
            } else if (guess < random) {  
                hint = "Nope, higher!"  
            } else {  
                break;  
            }  
        }  
    }  
    cocoon.sendPage("success.jx", { "random" : random,  
        "guess" : guess, "guesses" : guesses } );  
}
```

**game/documents/guess.jx**

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<head>
  <title>cocoon flow number guessing game</title>
</head>
<body>
  <h1>Guess the Number Between 1 and 10</h1>
  <h2>${hint}</h2>
  <h3>You've guessed ${guesses} times.</h3>
  <form method="post" action="${cocoon.continuation.id}.kont">
    <input type="text" name="guess"/>
    <input type="submit"/>
  </form>
</body>
</html>
```

**game/documents/success.jx**

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<head>
  <title>cocoon flow number guessing game</title>
</head>
<body>
  <h1>Success!</h1>
  <h2>The number was: ${random}</h2>
  <h3>It took you ${guesses} tries.</h3>
  <p><a href=".">Play again</a></p>
</body>
</html>
```

**game/sitemap.xmap**

```
<?xml version="1.0" encoding="UTF-8"?>
<map:sitemap xmlns:map="http://apache.org/cocoon/sitemap/1.0">

<map:components>
  <map:generators default="file">
    <!-- in this example we use JXTemplateGenerator to insert
         Flow variables in page content -->
    <map:generator label="content,data"
      logger="sitemap.generator.jx" name="jx"
      src="org.apache.cocoon.generation.JXTemplateGenerator"/>
  </map:generators>
  <map:transformers default="xslt"/>
  <map:serializers default="html"/>
  <map:matchers default="wildcard"/>
  <map:selectors default="browser">
    <map:selector name="exception"
      src="org.apache.cocoon.selection.XPathExceptionSelector">
      <exception name="invalid-continuation"
        class="org.apache.cocoon.components.flow.InvalidContinuationE
xception"/>
      <exception class="java.lang.Throwable" unroll="true"/>
    </map:selector>
  </map:selectors>
  <map:actions/>
  <map:pipes default="caching"/>
</map:components>
```

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

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

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

  <map:pipeline>
    <!-- no filename: call main() in game.js -->
    <map:match pattern="">
      <map:call function="main"/>
    </map:match>

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

```
<!-- .kont URLs are generated by the Flow system for
continuations -->
<map:match pattern="*.kont">
  <map:call continuation="{1}"/>
</map:match>

<!-- handle invalid continuations -->
<map:handle-errors>
  <map:select type="exception">
    <map:when test="invalid-continuation">
      <map:generate
        src="documents/invalidContinuation.html"/>
      <map:serialize type="xhtml"/>
    </map:when>
  </map:select>
</map:handle-errors>
</map:pipeline>

</map:pipelines>

</map:sitemap>
```

## Flow Control Parts

- Site map  
Several items are added to site map to indicate flows
- Flowscript  
JavaScript API that can be called in a flow
- Calling Java  
Java code can be called in a flow
- JXTemplate  
jx tags in the html page for looping extra
- Velocity  
The Velocity engine can be used in a flow

## Additions to Site Map

<http://cocoon.apache.org/2.1/userdocs/flow/sitemap.html>

### flow

```
<map:flow language="javascript">  
  <!-- Flow will use the javascript functions defined in game.js -->  
  <map:script src="flow/game.js"/>  
</map:flow>
```

Each script element give a file that will be compiled & run

### call

```
<map:match pattern="">  
  <map:call function="main"/>  
</map:match>  
  
<map:match pattern="*.kont">  
  <map:call continuation="{1}"/>  
</map:match>
```

## Flowscript

<http://cocoon.apache.org/2.1/userdocs/flow/api.html>

JavaScript API used in flow program

### Added Objects

- Cocoon
- Request
- Response
- Session
- Context
- Cookie
- Log
- WebContinuation

## JXTemplate

<http://cocoon.apache.org/2.1/userdocs/flow/jxtemplate.html>

Two languages:

- Jexl
- JXPath

Access

- Form parameter
- Java & JavaScript objects passed from Flowscript
- Cocoon Flow Object Models

Tags:

- template
- import
- set
- if
- choose
- out
- forEach
- formatNumber
- formatDate
- macro
- evalBody
- eval

## **JPath**

<http://cocoon.apache.org/2.1/userdocs/flow/jpath.html>

XSP extensions to access data from Flowscript in XSP pages

### Tags

- if
- choose
- value-of
- for-each
- continuation

## Velocity Generator

<http://cocoon.apache.org/2.1/userdocs/flow/velocity.html>

Velocity pages have access to Flowscript data

## Examples Using Java – Hello World

<http://bismarck.sdsu.edu:9006/cocoon/flowExamples/java.html>

### Files

- flowExamples/flow/example.js
- flowExamples/documents/hello.jx
- flowExamples/documents/bye.jx
- flowExamples/sitemap.xmap

### **flowExamples/flow/example.js**

```
function javaHello() {
  importPackage(java.util);
  var message = new java.lang.String("Hi there");

  var list = new ArrayList();
  list.add("Mary");
  list.add("Jack");
  list.add("Jill");

  var now = new java.util.Date();

  cocoon.sendPageAndWait( "hello.jx",
    { message:message, date: now, names: list});
  cocoon.sendPage("bye.jx", {names: list});
}
```

**flowExamples/documents/hello.jx**

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<head>
  <title>Hello World</title>
</head>
<body>
<h1>${message}</h1>
<p>It is now ${date}.</p>
<p>Hello to:</p>
<table>

<jx:forEach var="each" items="${names}" >
  <tr><td>${each}</td></tr>
</jx:forEach>

</table>
<p>A special greating to: ${names[0]}</p>
<form action="${cocoon.continuation.id}.kont" method="post">
  <input type="submit" value="Done" />
</form>
</body>
</html>
```

**flowExamples/documents/bye.jx**

```
<?xml version="1.0"?>
<html xmlns:jx="http://apache.org/cocoon/templates/jx/1.0">
<jx:macro name="tablerows">
  <jx:parameter name="list"/>
  <jx:parameter name="color"/>
  <jx:forEach var="item" items="{list}">
    <tr><td bgcolor="{color}">{item}</td></tr>
  </jx:forEach>
</jx:macro>

<head>
  <title>GoodBye World</title>
</head>
<body>
<h1>So Long</h1>
<p>Good bye to:</p>
<table>
  <tablerows list="{names}" color="lightblue"/>
</table>

</body>
</html>
```

## Changes in sitemap.xmap

```
<map:flow language="javascript">  
  <map:script src="flow/example.js"/>  
</map:flow>
```

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

```
<map:pipeline>  
  <map:match pattern="java.html">  
    <map:call function="javaHello"/>  
  </map:match>
```

## Using Velocity Evaluating JavaScript Page

<http://bismarck.sdsu.edu:9006/cocoon/flowExamples/script.html>

### Files

- flowExamples/flow/example.js
- flowExamples/documents/script.vm
- flowExamples/documents/script.xsl
- flowExamles/sitemap.xmap

### In flowExamples/flow/example.js

```
function script() {
  var source = "x = 2;\ny = 3;\nreturn x + y;";
  var result = "5.0";
  var kont = ".kont"
  while (true) {
    cocoon.sendPageAndWait( "scriptPage",
      {source: source, result: result, ending: kont});
    source = cocoon.request.get("code");
    try {
      var doIt = new Function("", source);
      result = doIt();
    } catch (exception) {
      result = exception;
    }
  }
}
```

**flowExamples/documents/script.vm**

```
<?xml version="1.0"?>
<page>
  <title>JavaScript</title>
  <content>
    <paragraph>
      Enter Some JavaScript. Your code must contain a return.
    </paragraph>
    <form action="$continuation.id$ending" method="post">
      <textarea name="code" rows="15"
        cols="40">$source</textarea>
      <input type="submit" />
    </form>
    <paragraph>
      The source
    </paragraph>
    <pre>$source</pre>
    <paragraph>
      The Result
    </paragraph>
    <pre>$result</pre>
  </content>
</page>
```

**flowExamples/documents/script.xsl**

```
<?xml version="1.0"?>
<xsl:stylesheet version="1.0"
  xmlns:xsl="http://www.w3.org/1999/XSL/Transform">

  <xsl:template match="page">
    <html>
      <head>
        <title>
          <xsl:value-of select="title"/>
        </title>

        </head>
        <body>
          <xsl:apply-templates/>
        </body>
      </html>
    </xsl:template>

    <xsl:template match="title">
      <h2>
        <xsl:apply-templates/>
      </h2>
    </xsl:template>
```

**flowExamples/documnets/script.xsl Continued**

```
<xsl:template match="paragraph">
  <p>
    <xsl:apply-templates/>
  </p>
</xsl:template>
```

```
<xsl:template match="form|pre|textarea|b|input">
  <xsl:copy>
    <xsl:copy-of select="@*" />
    <xsl:apply-templates/>
  </xsl:copy>
</xsl:template>
</xsl:stylesheet>
```

## Changes in sitemap.xmap

```
<map:flow language="javascript">
  <!-- Flow will use the javascript functions defined in game.js -->
  <map:script src="flow/example.js"/>
</map:flow>

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

  <map:pipeline>
    <map:match pattern="java.html">
      <map:call function="javaHello"/>
    </map:match>
    <map:match pattern="script.html">
      <map:call function="script"/>
    </map:match>
    <map:match pattern="scriptPage">
      <map:generate type="velocity" src="documents/script.vm" />
      <map:transform type="xslt" src="documents/script.xsl" />
      <map:serialize type="html" />
    </map:match>

    <map:match pattern="*.kont">
      <map:call continuation="{1}"/>
    </map:match>
```

## Look Ma No XSL

### JavaScript example without XSL page

<http://bismarck.sdsu.edu:9006/cocoon/flowExamples/rawScript.html>

### Files

- flowExamples/flow/example.js
- flowExamples/documents/script.vm
- flowExamles/sitemap.xmap

### In flowExamples/flow/example.js

```
function rawScript() {
  var source = "x = 2;\ny = 3;\nreturn x + y;";
  var result = "5.0";
  var kont = ".kont"
  while (true) {
    cocoon.sendPageAndWait( "rawScriptPage",
      {source: source, result: result, ending: kont});
    source = cocoon.request.get("code");
    try {
      var doIt = new Function("", source);
      result = doIt();
    } catch (exception) {
      result = exception;
    }
  }
}
```

**flowExamples/documents/rawScript.vm**

```
<?xml version="1.0"?>
<html>
  <title>JavaScript</title>
  <body>
    <p>
      Enter Some JavaScript. Your code must contain a return.
    </p>
    <form action="$continuation.id$ending" method="post">
      <textarea name="code" rows="15"
        cols="40">$source</textarea>
      <input type="submit" />
    </form>
    <p>The source</p>
    <pre>$source</pre>
    <p>The Result</p>
    <pre>$result</pre>
  </body>
</html>
```

## Changes in sitemap.xmap

```
<map:match pattern="rawScript.html">  
  <map:call function="rawScript"/>  
</map:match>  
<map:match pattern="rawScriptPage">  
  <map:generate type="velocity" src="documents/rawScript.vm" />  
  <map:serialize type="html" />  
</map:match>
```