

CNT 4714: Enterprise Computing Spring 2009

Introduction to JavaServer Pages (JSP) – Part 2

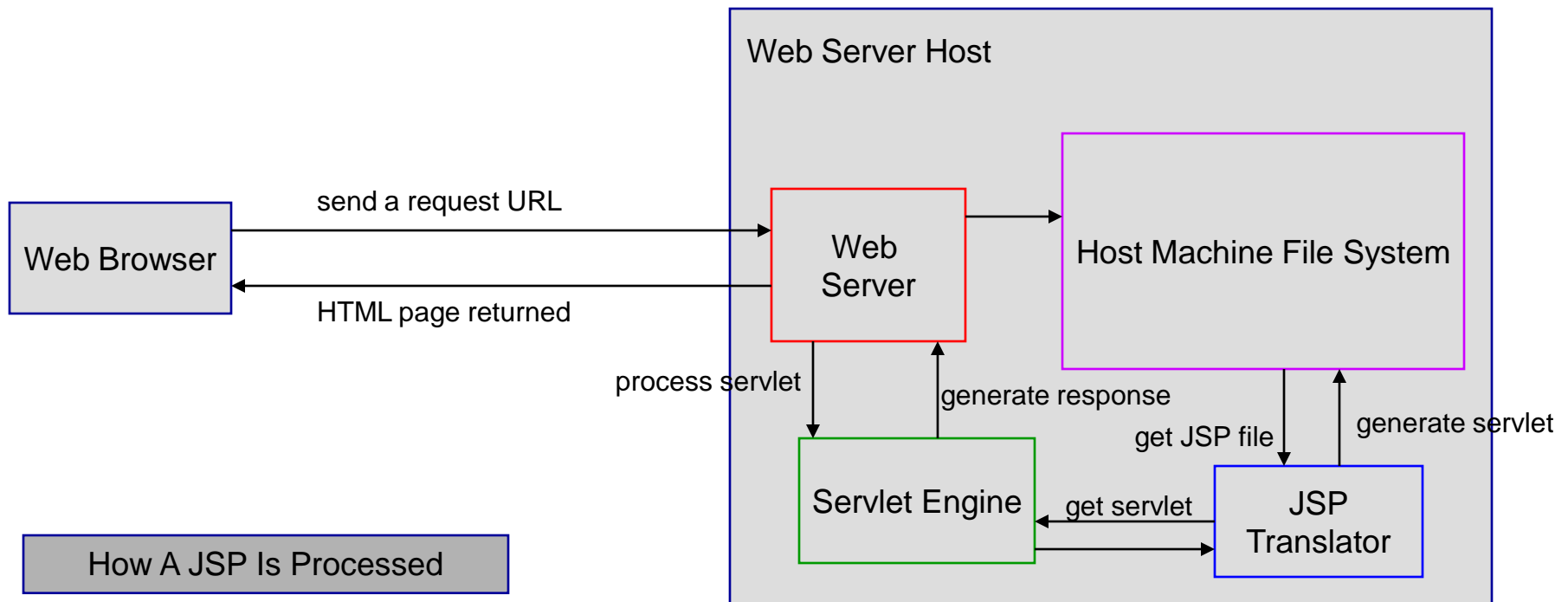
Instructor : Dr. Mark Llewellyn
 markl@cs.ucf.edu
 HEC 236, 407-823-2790
 <http://www.cs.ucf.edu/courses/cnt4714/spr2009>

School of Electrical Engineering and Computer Science
University of Central Florida



How A JSP Is Processed

- Much like a servlet, a JSP must first be processed by a web server before it can be displayed in a web browser. The web server must support JSPs and the JSP page must be stored in a file with a `.jsp` extension. The web server translates the JSP into a Java servlet, compiles the servlet, and executes it. The result of the execution is sent to the browser for display.



More On JSP Scripting Constructs

- There are three main types of JSP constructs: **scripting constructs**, **directives**, and **actions**.
- **Scripting elements** allow you to specify Java code that will become part of the resultant servlet.
- **Directives** enable you to control the overall structure of the resultant servlet.
- **Actions** enable you to control the behavior of the JSP engine.
- We'll look in more detail at all of these, starting with the scripting constructs.



Scripting Constructs

- There are three main types of JSP scripting constructs that can be used to insert Java code into a resultant servlet: **expressions**, **scriptlets** and **declarations**. Recall that there are also **comments** and **escape sequences**.
- A **JSP expression** is used to insert a Java expression directly into the output. It has the following form:

```
<%= java expression %>
```

- The expression is evaluated, converted into a string, and set to the output stream of the servlet.



Scripting Constructs

- A **JSP scriptlet** enables you to insert a Java statement into the servlet's `jspService` method which is invoked by the service method. A JSP scriptlet has the following form:

```
<% java statement %>
```

- A **JSP declaration** is for declaring methods or fields into the servlet. It has the following form:

```
<%! java declaration %>
```

- HTML comments have the form:

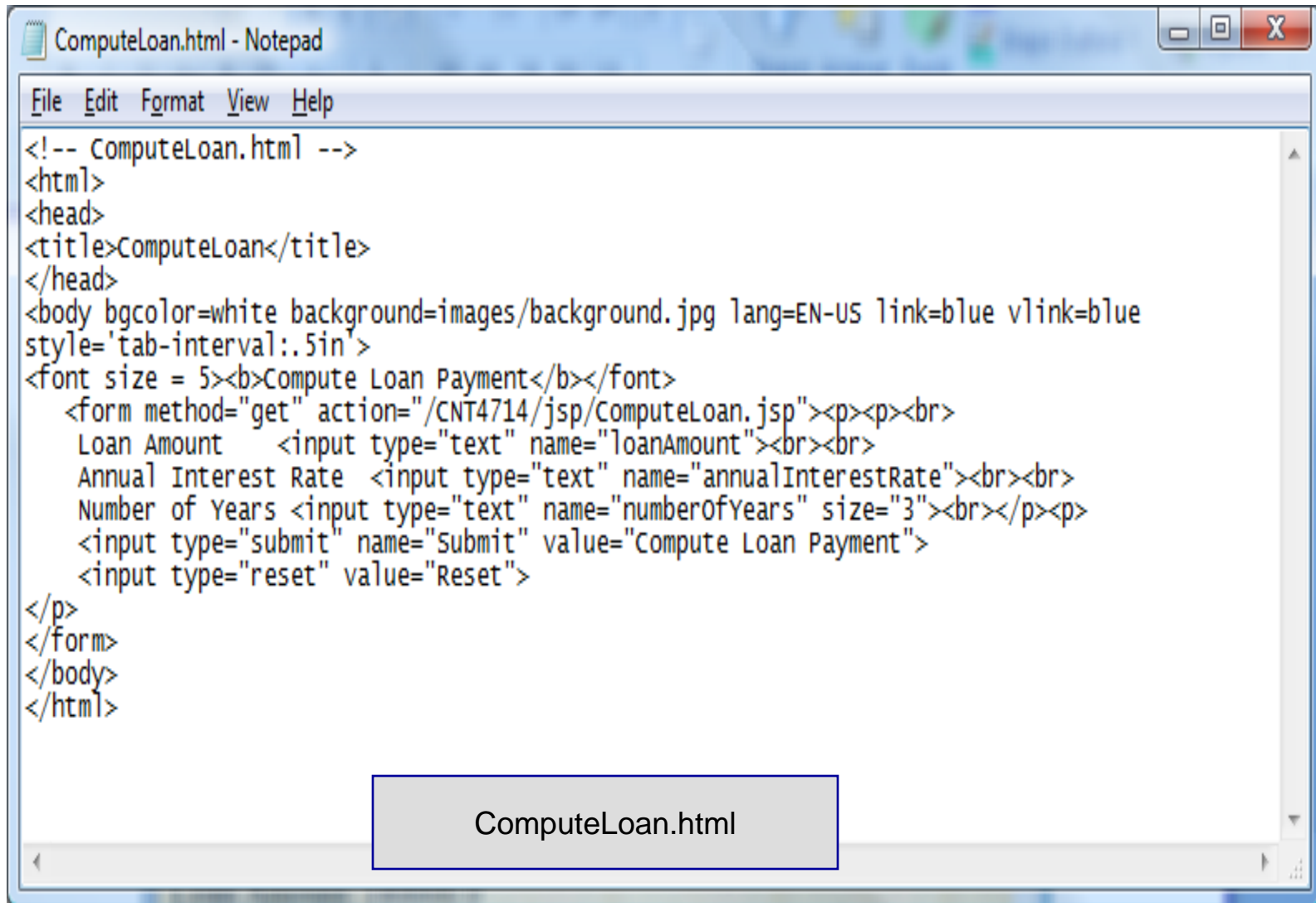
```
<!-- HTML comment -->
```

- If you don't want the comment to appear in the resultant HTML file, use a **JSP comment** which has the form:

```
<%-- JSP comment -->
```



Scripting Example



The image shows a Notepad window titled "ComputeLoan.html - Notepad". The window contains the following HTML code:

```
<!-- ComputeLoan.html -->
<html>
<head>
<title>ComputeLoan</title>
</head>
<body bgcolor=white background=images/background.jpg lang=EN-US link=blue vlink=blue
style='tab-interval:.5in'>
<font size = 5><b>Compute Loan Payment</b></font>
  <form method="get" action="/CNT4714/jsp/ComputeLoan.jsp"><p><p><br>
  Loan Amount   <input type="text" name="loanAmount"><br><br>
  Annual Interest Rate <input type="text" name="annualInterestRate"><br><br>
  Number of Years <input type="text" name="numberOfYears" size="3"><br></p><p>
  <input type="submit" name="Submit" value="Compute Loan Payment">
  <input type="reset" value="Reset">
</p>
</form>
</body>
</html>
```

At the bottom of the Notepad window, there is a button labeled "ComputeLoan.html".



Scripting Example

```
ComputeLoan.jsp - Notepad
File Edit Format View Help
<!-- ComputeLoan.jsp -->
<html>
<head>
<title>ComputeLoan</title>
</head><body bgcolor=white background=images/background.jpg lang=EN-US link=blue vlink=blue
style='tab-interval:.5in'>
<% double loanAmount = Double.parseDouble( request.getParameter("loanAmount"));
double annualInterestRate = Double.parseDouble(request.getParameter("annualInterestRate"));
double numberOfYears = Integer.parseInt(request.getParameter("numberOfYears"));
double monthlyInterestRate = annualInterestRate / 1200;
double monthlyPayment = loanAmount * monthlyInterestRate /
(1 - 1 / Math.pow(1 + monthlyInterestRate, numberOfYears * 12));
double totalPayment = monthlyPayment * numberOfYears * 12;
%>

<b><font size = 7> Loan Details </font></b><br><br>
<font size = 5>
Loan Amount:
<%= loanAmount %>
<br><br>
Annual Interest Rate:
<%= annualInterestRate %>
<br><br>
Number of Years:
<%= numberOfYears %>
<br><br>
<b>
Monthly Payment:
<%= monthlyPayment %>
<br><br>
Total Payment:
<%= totalPayment %>
<br><br>
</b>
</body>
</html>
```

Java statements

Java expression

ComputeLoan.jsp



Scripting Example

ComputeLoan - Windows Internet Explorer

http://localhost:8080/CNT4714/jsp/ComputeLo

File Edit View Favorites Tools Help

Google G Go Bookmarks 0 blocked Settings

CNN.com - Bre... ComputeLo...

Compute Loan Payment

Loan Amount

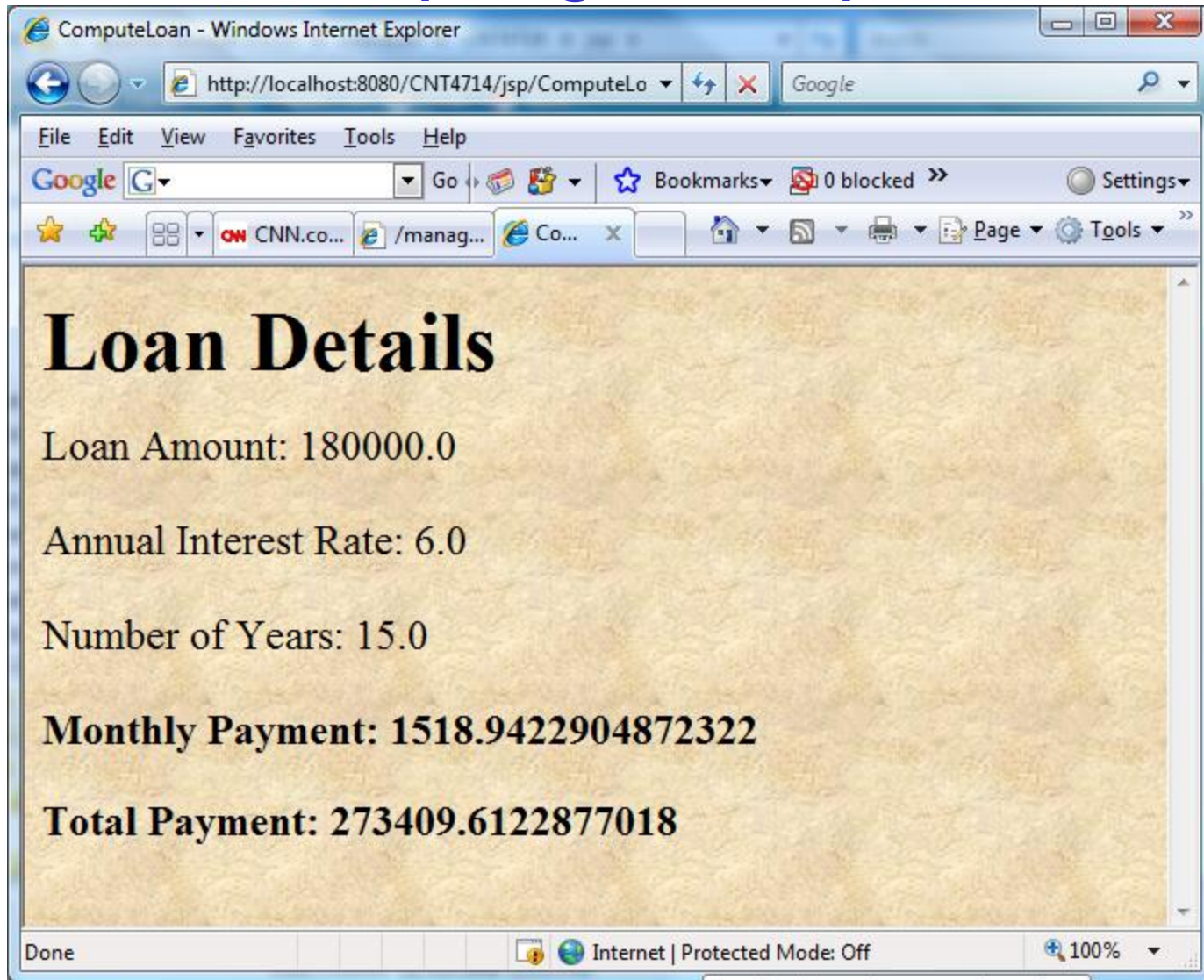
Annual Interest Rate

Number of Years

Internet | Protected Mode: Off 100%



Scripting Example



The screenshot shows a Windows Internet Explorer browser window. The address bar displays the URL `http://localhost:8080/CNT4714/jsp/ComputeLo`. The page content is displayed on a light brown, textured background and includes the following information:

- Loan Amount:** 180000.0
- Annual Interest Rate:** 6.0
- Number of Years:** 15.0
- Monthly Payment:** 1518.9422904872322
- Total Payment:** 273409.6122877018

The browser interface includes a menu bar (File, Edit, View, Favorites, Tools, Help), a search bar with the Google logo, and a status bar at the bottom showing 'Done', 'Internet | Protected Mode: Off', and a zoom level of 100%.



Scripting Example Using Directives

package code;

```
public class Loan {
    private double annualInterestRate;
    private int numOfYears;
    private double loanAmount;
    private java.util.Date loanDate;

    /** Default constructor */
    public Loan() {
        this(7.5, 30, 100000);
    }

    /** Construct a loan with specified annual interest rate,
        number of years and loan amount
        */
    public Loan(double annualInterestRate, int numOfYears,
        double loanAmount) {
        this.annualInterestRate = annualInterestRate;
        this.numOfYears = numOfYears;
        this.loanAmount = loanAmount;
        loanDate = new java.util.Date();
    }
}
```



```
/** Return annualInterestRate */  
public double getAnnualInterestRate() {  
    return annualInterestRate;  
}
```

```
/** Set a new annualInterestRate */  
public void setAnnualInterestRate(double annualInterestRate) {  
    this.annualInterestRate = annualInterestRate;  
}
```

```
/** Return numOfYears */  
public int getNumOfYears() {  
    return numOfYears;  
}
```

```
/** Set a new numOfYears */  
public void setNumOfYears(int numOfYears) {  
    this.numOfYears = numOfYears;  
}
```

```
/** Return loanAmount */  
public double getLoanAmount() {  
    return loanAmount;  
}
```



```
/** Set a newloanAmount */
public void setLoanAmount(double loanAmount) {
    this.loanAmount = loanAmount;
}

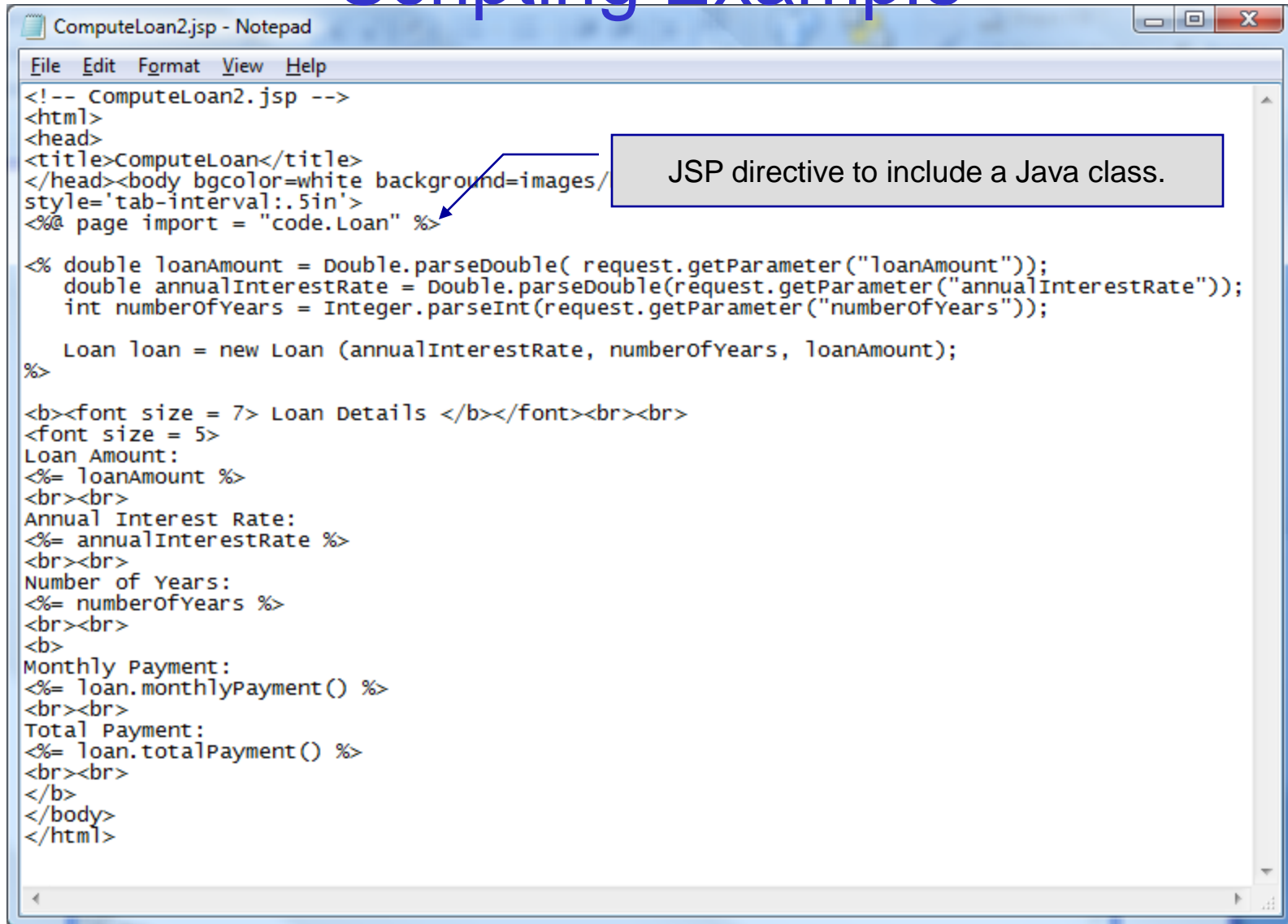
/** Find monthly payment */
public double monthlyPayment() {
    double monthlyInterestRate = annualInterestRate / 1200;
    return loanAmount * monthlyInterestRate / (1 -
        (Math.pow(1 / (1 + monthlyInterestRate), numOfYear * 12)));
}

/** Find total payment */
public double totalPayment() {
    return monthlyPayment() * numOfYear * 12;
}

/** Return loan date */
public java.util.Date getLoanDate() {
    return loanDate;
}
}
```



Scripting Example



```
ComputeLoan2.jsp - Notepad
File Edit Format View Help
<!-- ComputeLoan2.jsp -->
<html>
<head>
<title>ComputeLoan</title>
</head><body bgcolor=white background=images/
style='tab-interval:.5in'>
<%@ page import = "code.Loan" %>

<% double loanAmount = Double.parseDouble( request.getParameter("loanAmount"));
double annualInterestRate = Double.parseDouble(request.getParameter("annualInterestRate"));
int numberOfYears = Integer.parseInt(request.getParameter("numberOfYears"));

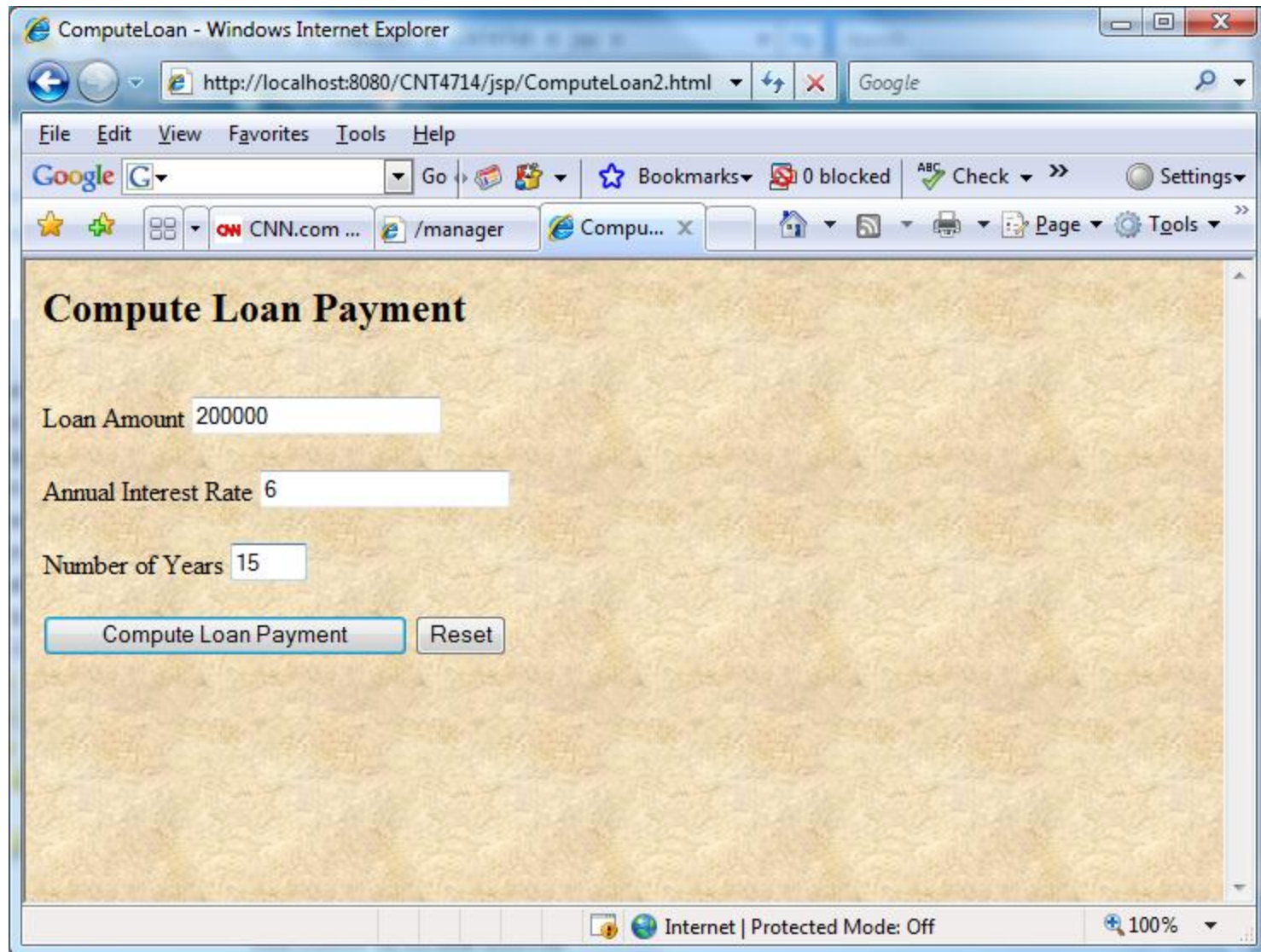
Loan loan = new Loan (annualInterestRate, numberOfYears, loanAmount);
%>

<b><font size = 7> Loan Details </font></b><br><br>
<font size = 5>
Loan Amount:
<%= loanAmount %>
<br><br>
Annual Interest Rate:
<%= annualInterestRate %>
<br><br>
Number of Years:
<%= numberOfYears %>
<br><br>
<b>
Monthly Payment:
<%= loan.monthlyPayment() %>
<br><br>
Total Payment:
<%= loan.totalPayment() %>
<br><br>
</b>
</body>
</html>
```

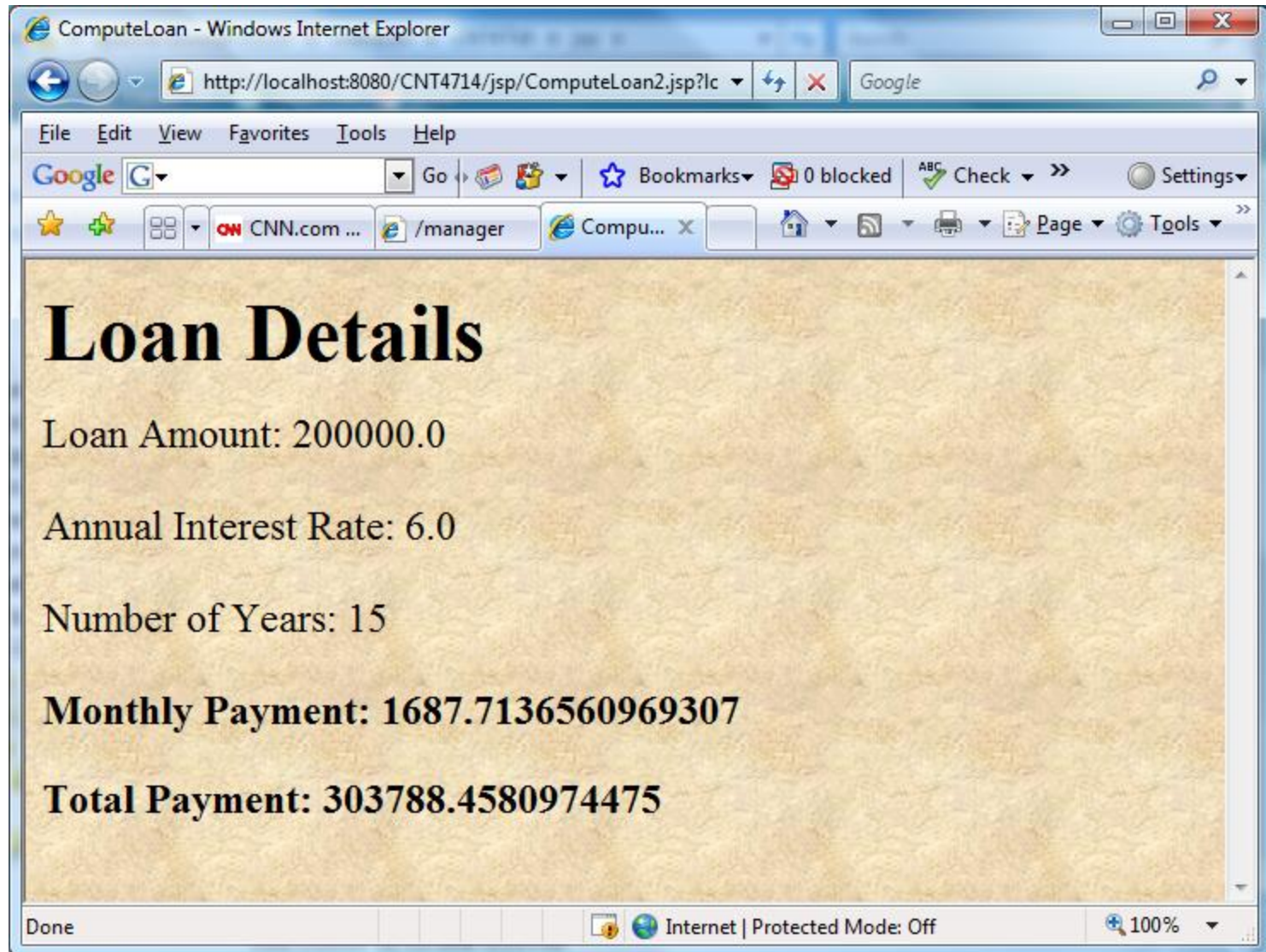
JSP directive to include a Java class.



Scripting Example Using Directives



Scripting Example Using Directives



JSP Standard Actions

- JSP standard actions provide programmers with access to several of the most common tasks performed in a JSP, such as including content from other resources, forwarding requests to other resources and interacting with JavaBean software components.
- JSP containers process actions at request time.
- Actions are delimited by `<jsp: action>` and `</jsp: action>`, where *action* is the standard action name.
 - In cases where nothing appears between the starting and ending tags, the XML empty element syntax `<jsp: action />` can be used.



JSP Standard Actions

<code><jsp: include></code>	Dynamically includes another resource in a JSP. As the JSP executes, the referenced resource is included and processed.
<code><jsp: forward></code>	Forwards request processing to another JSP, servlet or static page. This action terminates the current JSP's execution.
<code><jsp: plugin></code>	Allows a plug-in component to be added to a page in the form of a browser-specific object or embed HTML element. In the case of a Java applet, this action enables the browser to download and install the Java Plug-in, if it is not already installed on the client computer.
<code><jsp: param></code>	Used with the include, forward and plug-in actions to specify additional name-value pairs of information for use by these actions.



JSP Standard Actions

JavaBean Manipulation	
<code><jsp:useBean></code>	Specifies that the JSP uses a JavaBean instance (i.e., an object of the class that declares the JavaBean). This action specifies the scope of the object and assigns it an ID (i.e., a variable name) that scripting components can use to manipulate the bean.
<code><jsp:setProperty></code>	Sets a property in the specified JavaBean instance. A special feature of this action is automatic matching of request parameters to bean properties of the same name.
<code><jsp:getProperty></code>	Gets a property in the specified JavaBean instance and converts the result to a string for output in the response.



<jsp: include> Action

- JSPs support two include mechanisms – the `<jsp: include>` action and the `include` directive.
- Action `<jsp: include>` enables dynamic content to be included in a JSP at request time. If the included resource changes between requests, the next request to the JSP containing the `<jsp: include>` action includes the resource's new content.
- The `include` directive copies the content into the JSP once, at JSP translation time. If the included resource changes, the new content will not be reflected in the JSP that uses the `include` directive, unless the JSP is recompiled, which would normally occur only if a new version of the JSP were installed.



A JSP Using the `<jsp: include>` Action

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<!-- include.jsp -->

<html xmlns = "http://www.w3.org/1999/xhtml">
  <head>
    <title>Using jsp:include</title>
    <style type = "text/css">
      body {
        font-family: tahoma, helvetica, arial, sans-serif;
      }
      table, tr, td {
        font-size: 1.1em;
        border: 3px groove;
        padding: 5px;
        background-color: #dddddd;
      }
    </style>
  </head>
```



```
<body>
  <table>
    <tr>
      <td style = "width: 250px; text-align: center">
        <img src = "smallucf.gif"
          width = "140" height = "93"
          alt = "pegasus logo" />
      </td>
      <td>
        <!-- include banner.html in this JSP -->
        <jsp:include page = "banner.html"
          flush = "true" />
      </td>
    </tr>
    <tr>
      <td style = "width: 250px">
        <!-- include toc.html in this JSP -->
        <jsp:include page = "toc.html" flush = "true" />
      </td>
      <td style = "vertical-align: top">
        <!-- include clock2.jsp in this JSP -->
        <jsp:include page = "clock2.jsp"
          flush = "true" />
      </td>
    </tr>
  </table>
</body>
</html>
```



Banner.html

```
<!-- banner.html          -->
<!-- banner to include in another document -->
<div style = "width: 800px">
  <p>
    CNT 4714 - Enterprise Computing
    <br />
    Spring 2009 Semester - University of Central Florida
  </p>
  <p>
    <a href = "mailto:markl@cs.ucf.edu">markl@cs.ucf.edu</a>
  </p>
</div>
```



Table of Contents (toc.html)

```
<!-- toc.html -->
<!-- contents to include in another document -->
<p><a href = "http://www.cs.ucf.edu/courses/cnt4714/fall2008">
    CNT 4714 Course Webpage
</a></p>
<p><a href = "http://www.cs.ucf.edu/faculty/markl.html">
    Instructor's Webpage
</a></p>
<p><a href =
"http://www.cs.ucf.edu/courses/cnt4714/spr2009/code.html">
    Code Download Page
</a></p>
<p><a href =
"http://www.cs.ucf.edu/courses/cnt4714/spr2009/homework.html">
    Programming Assignments Page
</a></p>
<p>Send questions or comments about this site to
    <a href = "mailto:markl@cs.ucf.edu">
        markl@cs.ucf.edu
    </a><br />
</p>
```



Clock2.jsp

```
<!-- clock2.jsp -->
<!-- date and time to include in another document via redirection -->
<table>
  <tr>
    <td style = "background-color: black;">
      <p class = "big" style = "color: cyan; font-size: 3em;
        font-weight: bold;">
        <%-- script to determine client local and --%>
        <%-- format date accordingly --%>
        <%
          // get client locale
          java.util.Locale locale = request.getLocale();

          // get DateFormat for client's Locale
          java.text.DateFormat dateFormat =
            java.text.DateFormat.getDateTimeInstance(
              java.text.DateFormat.LONG,
              java.text.DateFormat.LONG, locale );
        %> <%-- end script --%>
        <%-- output date --%>
        <%= dateFormat.format( new java.util.Date() ) %>
      </p>
    </td>
  </tr>
</table>
```





CNT 4714 - Enterprise Computing
Spring 2009 Semester - University of Central Florida

markl@cs.ucf.edu

[CNT 4714 Course Webpage](#)

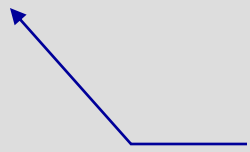
[Instructor's Webpage](#)

[Code Download Page](#)

[Programming Assignments Page](#)

Send questions or comments about this site to markl@cs.ucf.edu

March 23, 2009 3:15:57 PM EDT



Execution of include.jsp



<jsp: forward> Action

- JSP action <jsp: forward> enables a JSP to forward request processing to a different resource, such as an error page.
- Request processing by the original JSP terminates as soon as the JSP forwards the request.
- In the next example, this action is illustrated by forwarding a welcome request to another welcome page. JSP forward1.jsp forwards the request to JSP forward2.jsp. The forwarding action requests a date and time at which the original request was received that is forwarded.



Initial Forward JSP (forward1.jsp)

```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!-- forward1.jsp -->
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
    <title>Forward request to another JSP</title>
</head>
<body>
    <% // begin scriptlet
        String name = request.getParameter( "firstName" );
        if ( name != null )
        {
    %> <%-- end scriptlet to insert fixed template data --%>

        <jsp:forward page = "forward2.jsp">
            <jsp:param name = "date"
                value = "<%= new java.util.Date() %>" />
        </jsp:forward>

    <% // continue scriptlet
        } // end if
```



Initial Forward JSP (forward1.jsp) (cont.)

```
else
{
%> <%-- end scriptlet to insert fixed template data --%>

    <form action = "forward1.jsp" method = "get">
        <p>Type your first name and press Submit</p>

        <p><input type = "text" name = "firstName" />
            <input type = "submit" value = "Submit" />
        </p>
    </form>

    <% // continue scriptlet

        } // end else

    %> <%-- end scriptlet --%>
</body>
</html> <!-- end XHTML document -->
```



Forward2 JSP (forward2.jsp)

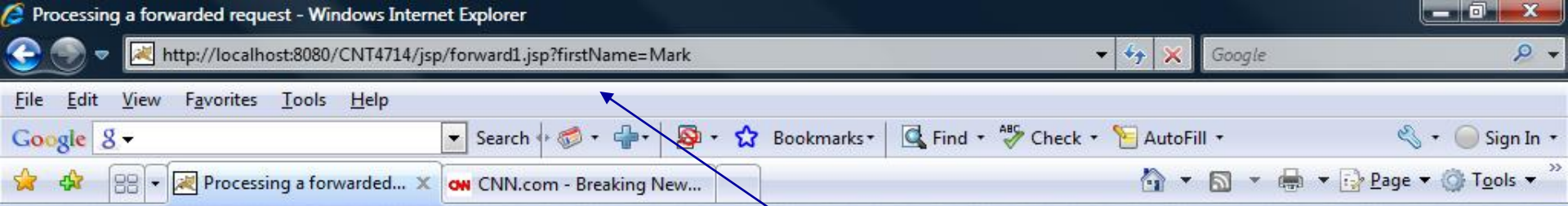
```
<?xml version = "1.0"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<!-- forward2.jsp -->
<html xmlns = "http://www.w3.org/1999/xhtml">
<head>
    <title>Processing a forwarded request</title>
    <style type = "text/css">
        .big
        {
            font-family: tahoma, helvetica, arial, sans-serif;
            font-weight: bold;
            font-size: 2em;
        }
    </style>
</head>
<body>
    <p class = "big">
        Hello <%= request.getParameter( "firstName" ) %>, <br />
        Your redirection request was received <br /> and
        forwarded at
```



Forward2 JSP (forward2.jsp) (cont.)

```
</p>
<table style = "border: 6px outset;">
  <tr>
    <td style = "background-color: black;">
      <p class = "big" style = "color: cyan;">
        <%= request.getParameter( "date" ) %>
      </p>
    </td>
  </tr>
</table>
</body>
</html>
```

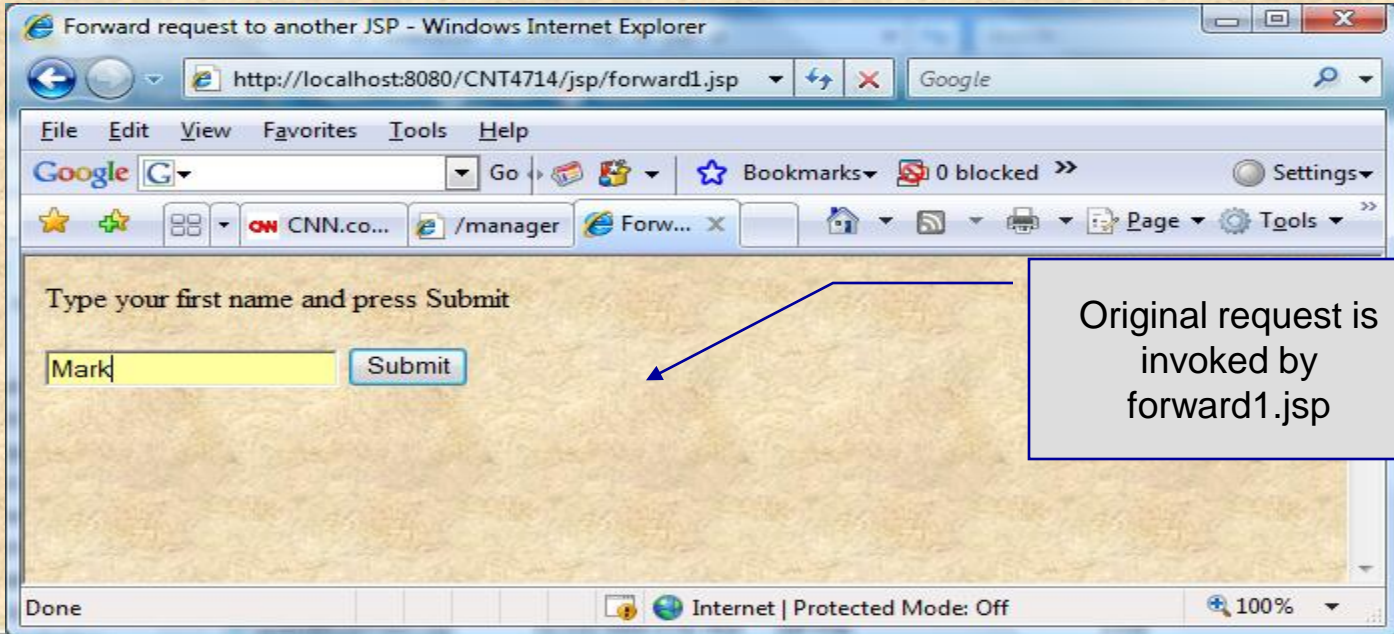




**Hello Mark,
Your redirection request was received
and forwarded at**

Mon Mar 23 15:17:08 EDT 2009

Forward2.jsp receives forwarded request for service with firstname passed as a parameter



Original request is invoked by forward1.jsp

