NetRexx Server Pages

24th International Rexx Language Symposium

Raleigh/Durham, NC

René Vincent Jansen, May 8th, 2013

What is it

- Java2EE, also called enterprise Java
- First there was Jeeves, servlets, JSP, JSF
- Consists of Application Servers, and Web Containers that are part of those
- Pages are code compiled on the fly to servlets

The goal

- The inspiration is not unlike Rails (for Ruby)
- To program an active website in NetRexx with as less setup as possible
  - No setup would be ideal
- Use standards - we are not bound to one product

An implementation choice

- We chose a web container called Jetty for this purpose
- It is light, can be embedded and does not need a lot of configuration
- We can develop 'in place' and see results

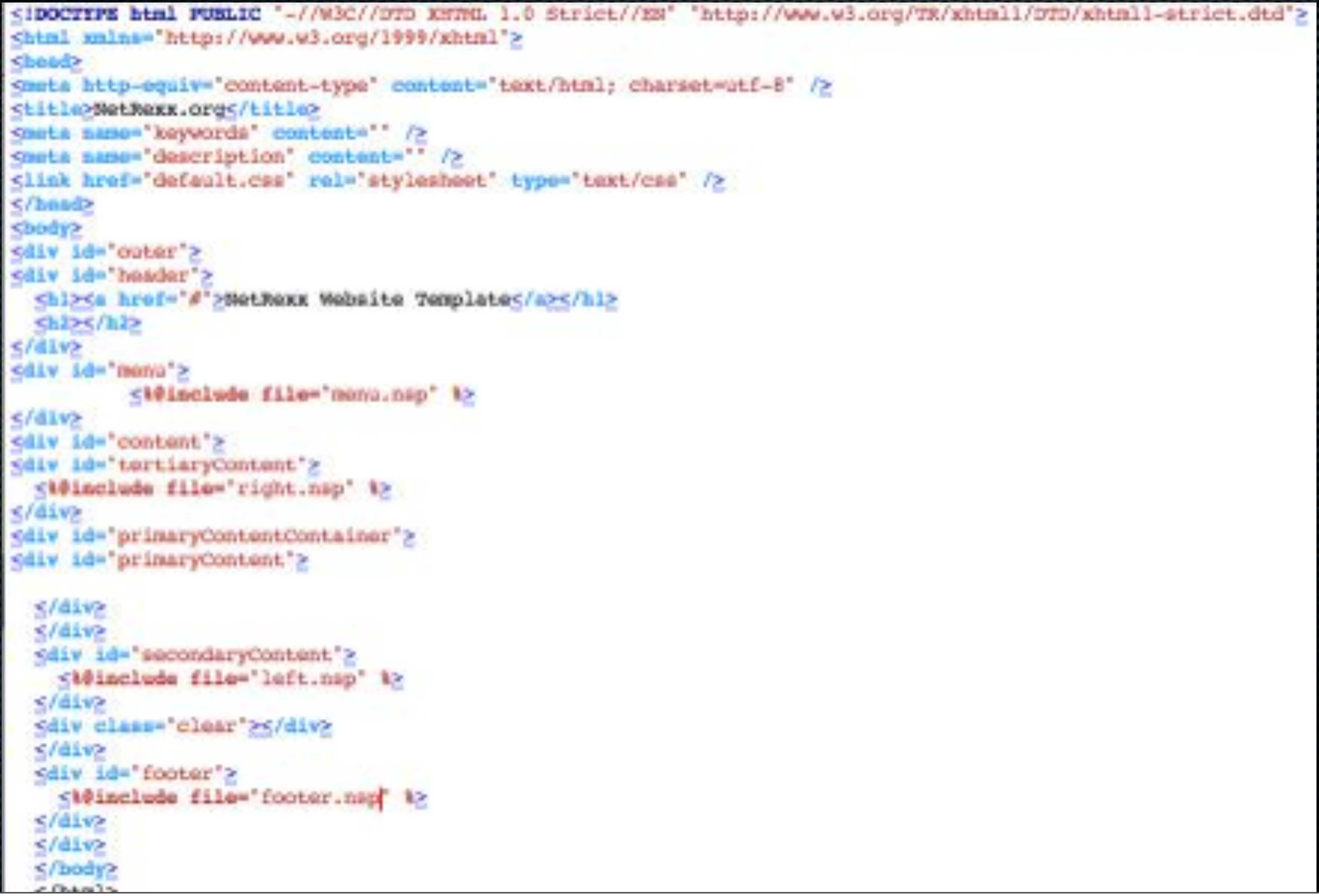
The new NetRexx.org site

- This site has been started fresh after the first version was made with a proprietary html5 product, that unfortunately never was too fast on one of the ~~most ridiculed~~ popular browsers
- Not using NetRexx for the NetRexx site was not compatible with our sense of justice
- It is slowly getting its form now; still much ideas unimplemented

Active parts of the site (currently)

- 'the Hursley time' using qtime, one of the first Rexx (and also NetRexx) programs
- the download and automatic build page
- the examples page, these are straight out of the Kenai source repository, and formatted as tables with comments read out of a file
- the left and right columns, and the page footer
- the response form

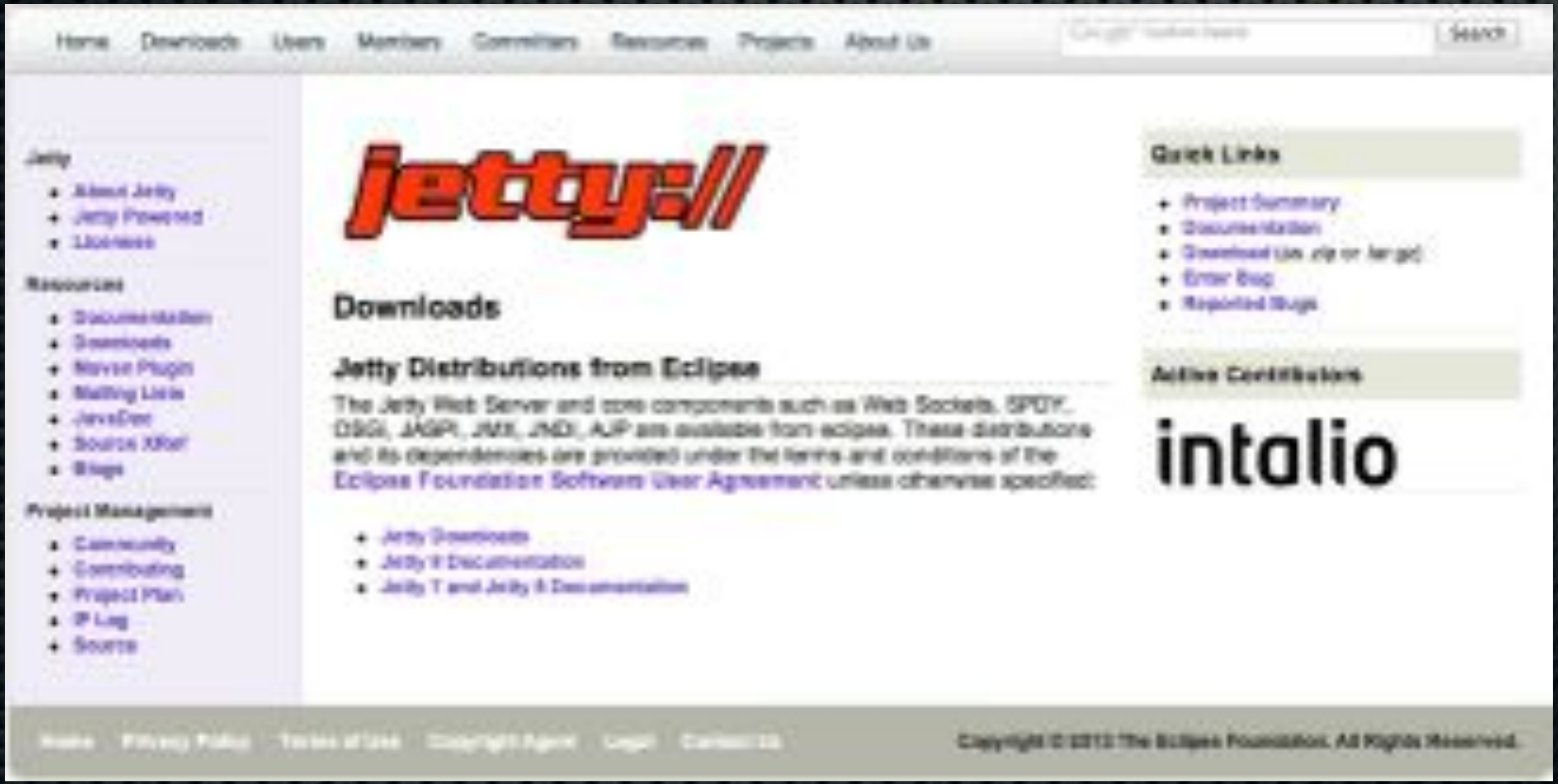
Including text



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>NetRexx.org</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link href="default.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="outer">
<div id="header">
  <h1><a href="#">NetRexx Website Template</a></h1>
  <h2></h2>
</div>
<div id="menu">
          <%@include file="menu.nsp" %>
</div>
<div id="content">
<div id="tertiaryContent">
  <%@include file="right.nsp" %>
</div>
<div id="primaryContentContainer">
<div id="primaryContent">

  </div>
  </div>
  <div id="secondaryContent">
    <%@include file="left.nsp" %>
  </div>
  <div class="clear"></div>
  </div>
  <div id="footer">
    <%@include file="footer.nsp" %>
  </div>
  </div>
  </body>
```

Downloading Jetty

- http://www.eclipse.org/jetty/downloads.php


Home Downloads Users Members Committers Resources Projects About Us
Search
jetty://
intalio

Jetty
- About Jetty
- Jetty Powered
- Licenses

Resources
- Documentation
- Downloads
- Maven Plugin
- Mailing Lists
- JavaDoc
- Source Xref
- Blogs

Project Management
- Community
- Contributing
- Project Plan
- IP Log
- Source

Downloads

Jetty Distributions from Eclipse

The Jetty Web Server and core components such as Web Sockets, SPDY, OSGi, JASPI, JMX, JNDI, AJP are available from eclipse. These distributions and its dependencies are provided under the terms and conditions of the Eclipse Foundation Software User Agreement unless otherwise specified.

- Jetty Downloads
- Jetty 9 Documentation
- Jetty 7 and Jetty 8 Documentation

Quick Links
- Project Summary
- Documentation
- Download (as zip or tar.gz)
- Enter Bug
- Reported Bugs

Active Contributors

Home Privacy Policy Terms of Use Copyright Agent Legal Contact Us

Copyright © 2013 The Eclipse Foundation. All Rights Reserved.

Setting it up

- Unzip it to any directory you want
- set the JETTY_HOME environment variable to this directory, e.g. setenv JETTY_HOME=`pwd`

Delete the whole sample set

- delete anything under JETTY_HOME/webapps
- or save them somewhere for reference

Modify the /etc

- We do want our NetRexx Server Pages to have the .nsp filename extension - call it chauvinism
- in JETTY_HOME/etc/webdefaults.xml,

Add one line to recognize .nsp as .jsp

- <servlet-mapping>
-   <servlet-name>jsp</servlet-name>
-   **<url-pattern>*.nsp</url-pattern>**
-   <url-pattern>*.jsp</url-pattern>
-   <url-pattern>*.jspf</url-pattern>
-   <url-pattern>*.jspx</url-pattern>

Add a ROOT app to webapps

- This ROOT app will be the application selected for the url of the website
- These are called domains in J2EE
- We can arrange for domains to be served by virtual servers

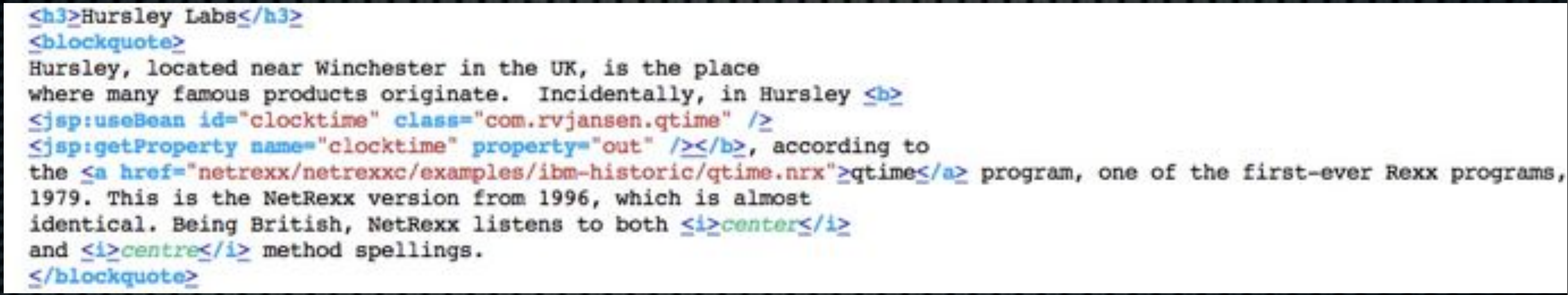
Running it

- Make sure the JETTY_HOME environment variable is set correctly
- in JETTY_HOME/bin, issue **./jetty.sh start**
- (stopping would be **./jetty.sh stop**)

Adding active content to pages

- Server side
- Edit html pages and add the tags
- e.g. for the qtime program

We use jsp tags



```
<h3>Hursley Labs</h3>
<blockquote>
Hursley, located near Winchester in the UK, is the place
where many famous products originate.  Incidentally, in Hursley <b>
<jsp:useBean id="clocktime" class="com.rvjansen.qtime" />
<jsp:getProperty name="clocktime" property="out" /></b>, according to
the <a href="netrexx/netrexxc/examples/ibm-historic/qtime.nrx">qtime</a> program, one of the first-ever Rexx programs,
1979. This is the NetRexx version from 1996, which is almost
identical. Being British, NetRexx listens to both <i>center</i>
and <i>centre</i> method spellings.
</blockquote>
```

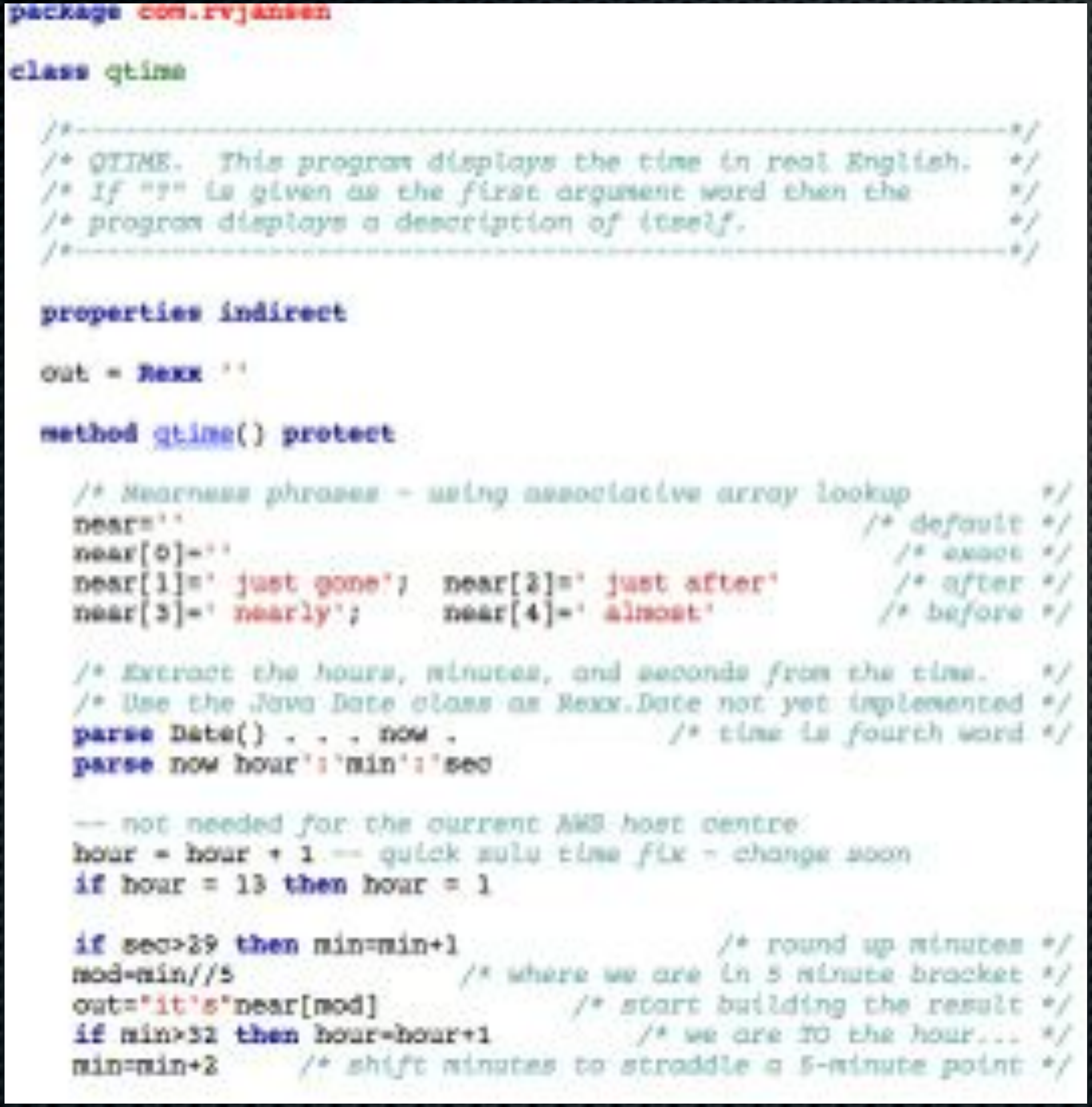
The <jsp:usebean> tag

- This instructs the J2EE processor to find a class corresponding to this on the classpath that is formed by ROOT/WEB-INF/classes
- in this case, the class is found in ROOT/WEB-INF/classes/com/rvjansen
- That is the package name I gave it

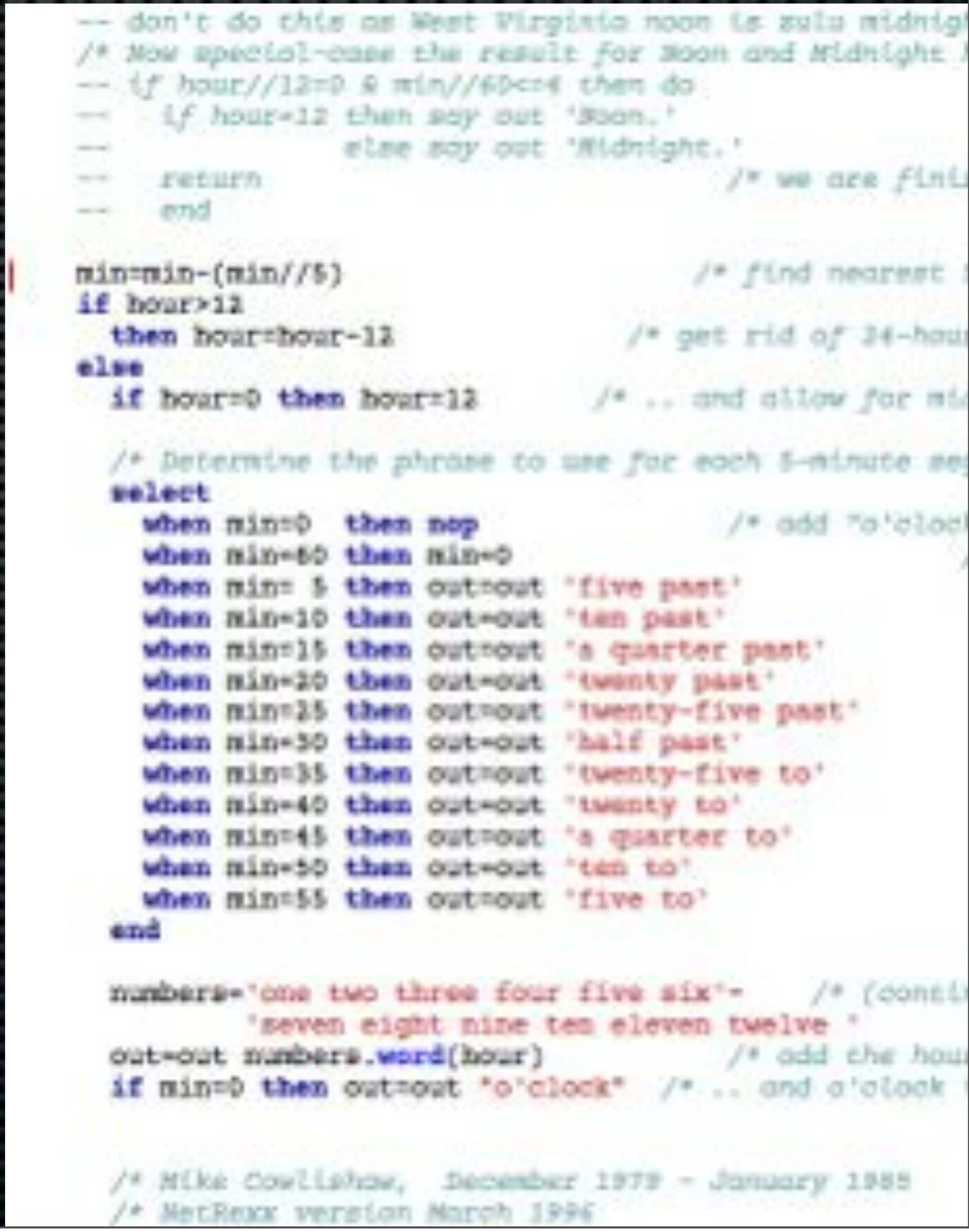
The <jsp:getProperty> tag

- this has a name= attribute which refers to the <useBean> tag and a property= attribute which refers to the name of the property
- if you understand this, the hardest part is in the past

Small mods to qtime

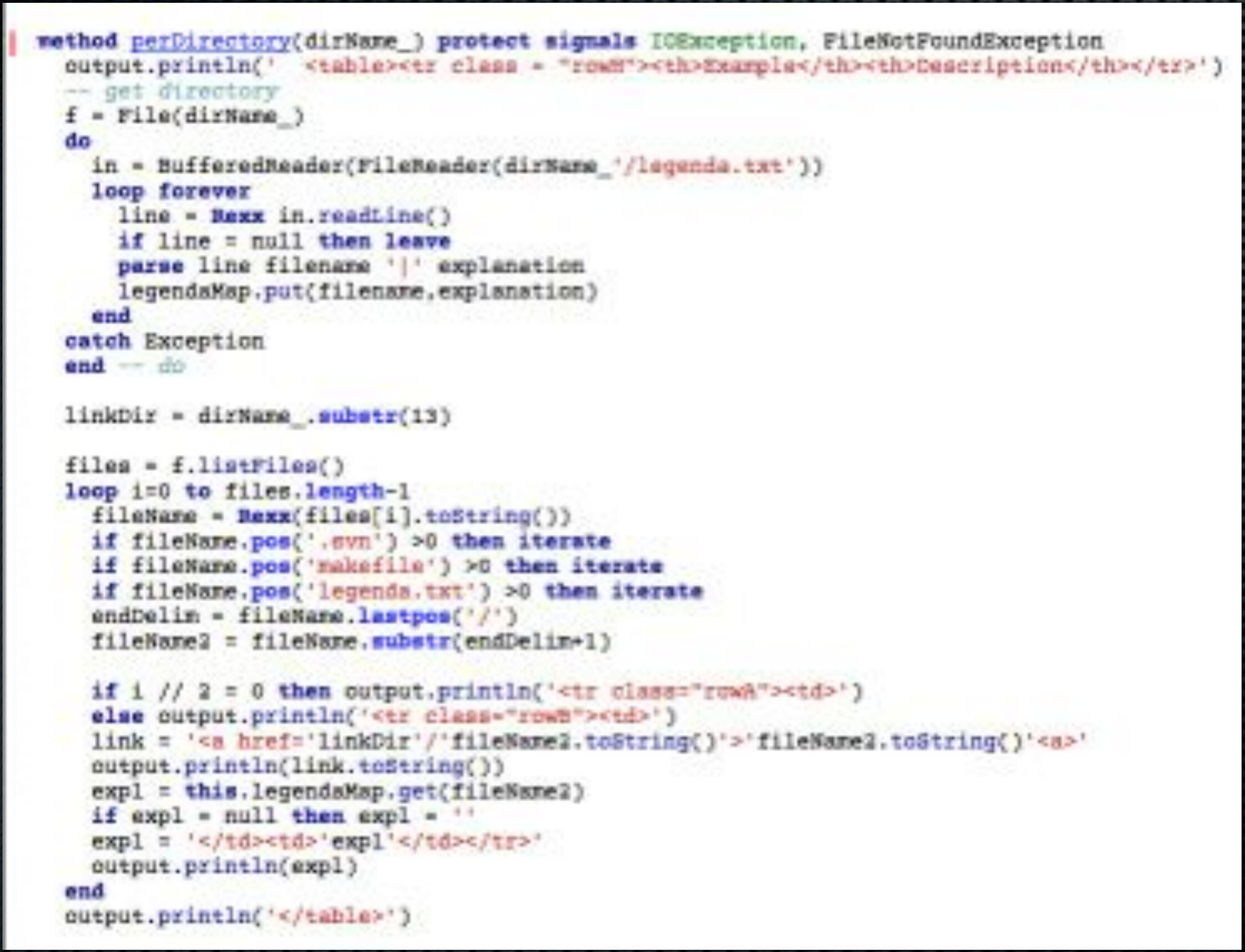


```
package com.rvjansen

class qtime

  /*----------------------------------------------------------------*/
  /* QTIME.  This program displays the time in real English.        */
  /* If "?" is given as the first argument word then the            */
  /* program displays a description of itself.                      */
  /*----------------------------------------------------------------*/

  properties indirect

  out = Rexx ''

  method qtime() protect

    /* Nearness phrases - using associative array lookup           */
    near=''                                               /* default */
    near[0]=''                                              /* exact */
    near[1]=' just gone';  near[2]=' just after'           /* after */
    near[3]=' nearly';     near[4]=' almost'              /* before */

    /* Extract the hours, minutes, and seconds from the time.       */
    /* Use the Java Date class as Rexx.Date not yet implemented */
    parse Date() . . . now .                  /* time is fourth word */
    parse now hour':'min':'sec

    -- not needed for the current AWS host centre
    hour = hour + 1 -- quick zulu time fix - change soon
    if hour = 13 then hour = 1

    if sec>29 then min=min+1                      /* round up minutes */
    mod=min//5                      /* where we are in 5 minute bracket */
    out="it's"near[mod]                  /* start building the result */
    if min>32 then hour=hour+1              /* we are TO the hour... */
    min=min+2      /* shift minutes to straddle a 5-minute point */
```



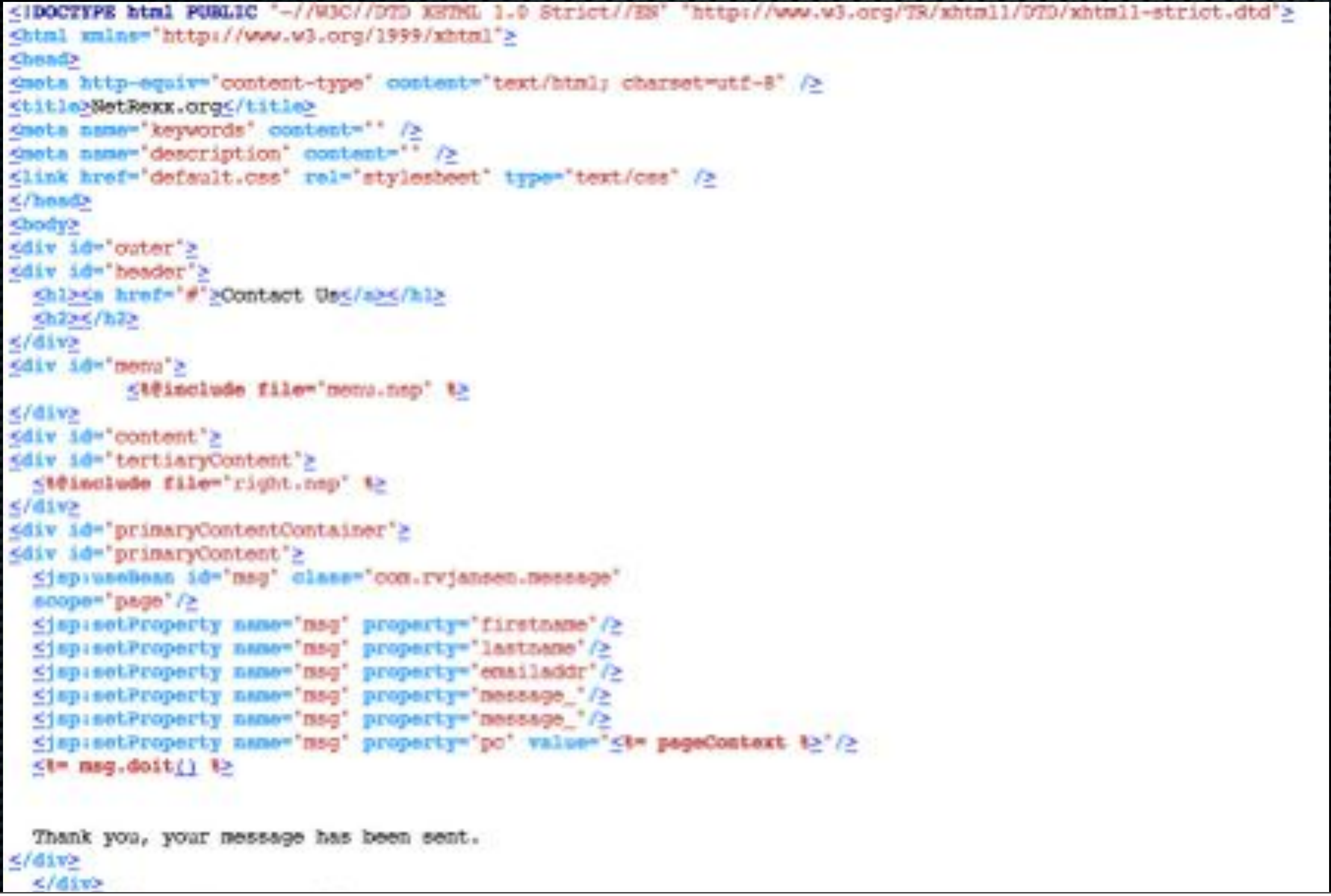
```
  -- don't do this as West Virginia noon is zulu midnig
  /* Now special-case the result for Noon and Midnight
  -- if hour//12=0 & min//60<=4 then do
  --    if hour=12 then say out 'Noon.'
  --                    else say out 'Midnight.'
  --    return                                   /* we are fini
  --    end

  min=min-(min//5)                              /* find nearest
  if hour>12
    then hour=hour-12                         /* get rid of 24-hou
  else
    if hour=0 then hour=12              /* ... and allow for mi

    /* Determine the phrase to use for each 5-minute se
    select
      when min=0  then nop                       /* add "o'cloc
      when min=60 then min=0
      when min= 5 then out=out 'five past'
      when min=10 then out=out 'ten past'
      when min=15 then out=out 'a quarter past'
      when min=20 then out=out 'twenty past'
      when min=25 then out=out 'twenty-five past'
      when min=30 then out=out 'half past'
      when min=35 then out=out 'twenty-five to'
      when min=40 then out=out 'twenty to'
      when min=45 then out=out 'a quarter to'
      when min=50 then out=out 'ten to'
      when min=55 then out=out 'five to'
    end

    numbers='one two three four five six'-      /* (conti
            'seven eight nine ten eleven twelve '
    out=out numbers.word(hour)                  /* add the hou
    if min=0 then out=out "o'clock"  /* ... and o'clock


    /* Mike Cowlishaw,  December 1979 - January 1985
    /* NetRexx version March 1996
```

The examples page



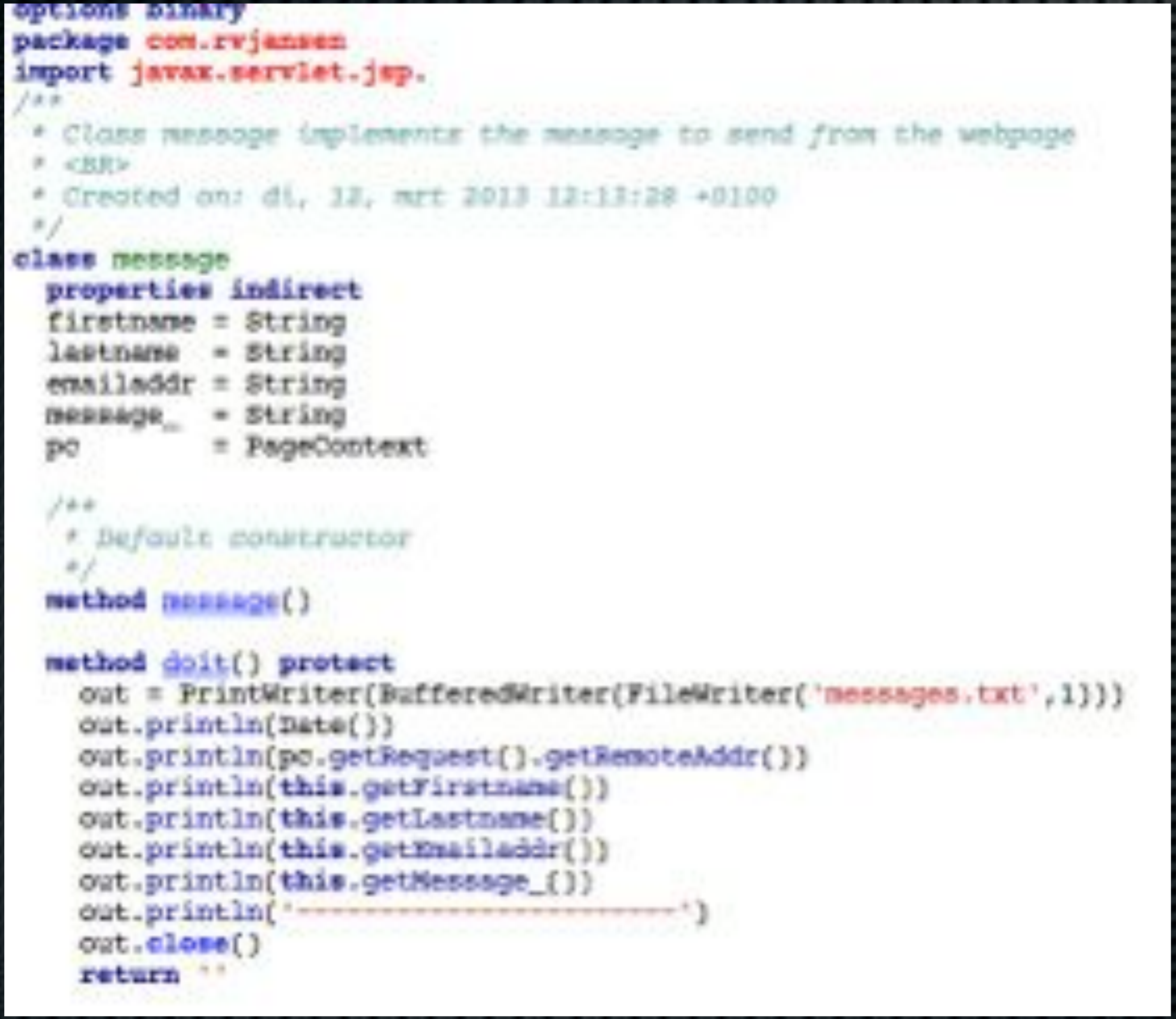
```
method perDirectory(dirName_) protect signals IOException, FileNotFoundException
  output.println('  <table><tr class = "rowH"><th>Example</th><th>Description</th></tr>')
  -- get directory
  f = File(dirName_)
  do
    in = BufferedReader(FileReader(dirName_'/legenda.txt'))
    loop forever
      line = Rexx in.readLine()
      if line = null then leave
      parse line filename '|' explanation
      legendaMap.put(filename,explanation)
    end
  catch Exception
  end -- do

  linkDir = dirName_.substr(13)

  files = f.listFiles()
  loop i=0 to files.length-1
    fileName = Rexx(files[i].toString())
    if fileName.pos('.svn') >0 then iterate
    if fileName.pos('makefile') >0 then iterate
    if fileName.pos('legenda.txt') >0 then iterate
    endDelim = fileName.lastpos('/')
    fileName2 = fileName.substr(endDelim+1)

    if i // 2 = 0 then output.println('<tr class="rowA"><td>')
    else output.println('<tr class="rowB"><td>')
    link = '<a href='linkDir'/'fileName2.toString()'>'fileName2.toString()'<a>'
    output.println(link.toString())
    expl = this.legendaMap.get(fileName2)
    if expl = null then expl = ''
    expl = '</td><td>'expl'</td></tr>'
    output.println(expl)
  end
  output.println('</table>')
```

The feedback form



```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="content-type" content="text/html; charset=utf-8" />
<title>NetRexx.org</title>
<meta name="keywords" content="" />
<meta name="description" content="" />
<link href="default.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="outer">
<div id="header">
  <h1><a href="#">Contact Us</a></h1>
  <h2></h2>
</div>
<div id="menu">
            <%@include file="menu.nsp" %>
</div>
<div id="content">
<div id="tertiaryContent">
  <%@include file="right.nsp" %>
</div>
<div id="primaryContentContainer">
<div id="primaryContent">
  <jsp:useBean id="msg" class="com.rvjansen.message"
  scope="page"/>
  <jsp:setProperty name="msg" property="firstname"/>
  <jsp:setProperty name="msg" property="lastname"/>
  <jsp:setProperty name="msg" property="emailaddr"/>
  <jsp:setProperty name="msg" property="message_"/>
  <jsp:setProperty name="msg" property="message_"/>
  <jsp:setProperty name="msg" property="pc" value="<%= pageContext %>"/>
  <%= msg.doit() %>


  Thank you, your message has been sent.
</div>
  </div>
```

The feedback code



```
options binary
package com.rvjansen
import javax.servlet.jsp.
/**
 * Class message implements the message to send from the webpage
 * <BR>
 * Created on: di, 12, mrt 2013 12:13:28 +0100
 */
class message
  properties indirect
  firstname = String
  lastname  = String
  emailaddr = String
  message_  = String
  pc        = PageContext

  /**
   * Default constructor
   */
  method message()

  method doit() protect
    out = PrintWriter(BufferedWriter(FileWriter('messages.txt',1)))
    out.println(Date())
    out.println(pc.getRequest().getRemoteAddr())
    out.println(this.getFirstname())
    out.println(this.getLastname())
    out.println(this.getEmailaddr())
    out.println(this.getMessage_())
    out.println('------------------------')
    out.close()
    return ''
```

Solving problems

- top
- kill -3 <pid>
- jstack <pid>
- trace

Potential problems

- class not found
- threading issues

Multithreading

- Active web content programs are multithreaded by nature
- Even one user can have multiple windows open and/or press the submit buttons in a high tempo

Avoiding threading issues

- no static variables
- use the *synchronized* version of JVM collection classes

Future plans

- Bridging it to ooRexx using BSF4ooRexx
- Making an integrated component for this (JavaBean)

Conclusion

- Using NetRexx, you are able to put together an active website using standard J2EE concepts and facilities
- There is only one line added to a standard config file
-