Visual Mining, Inc.

Using NetCharts™ with Java Server Pages and Flat Files

A Programmers Guide to Scripting with JSP and NetCharts

Table of Contents

1. Scope

This document provides web page designers with detailed information on the capabilities of NetCharts when used with Java Server Pages. A companion document, *The Visual Mining CDL Reference Guide*, provides additional useful information on designing chart templates to be used with NetCharts.

Note to our customers:

Thank you for evaluating and/or purchasing NetCharts. We sincerely believe that the charts produced by NetCharts are among the most robust online charts available.

Please direct any questions or comments on this product to support@visualmining.com.

—The Visual Mining Team

2. Introduction

The purpose of this document is to provide web designers familiar with Sun's Java Server Pages technology an example of using NetCharts with JSP. An example will be given that: runs the sample JSP script that is provided, and explains the script and how NetCharts and JSP interact with each other.

3. NetCharts JSP Example

Using NetCharts with JSP

This example uses JSP scripting commands to extract data from a flat file data source (`regionalsales.dat`), populate variables with the data, and build an HTML page that contains a NetCharts applet that uses this data. This paper provides a brief overview of how to use JSP to interact with flat files, how to construct an HTML page containing a NetCharts applet, and how to populate that applet using NetCharts `<param>` tags and JSP variables.

Setting up a Flat File Data Source

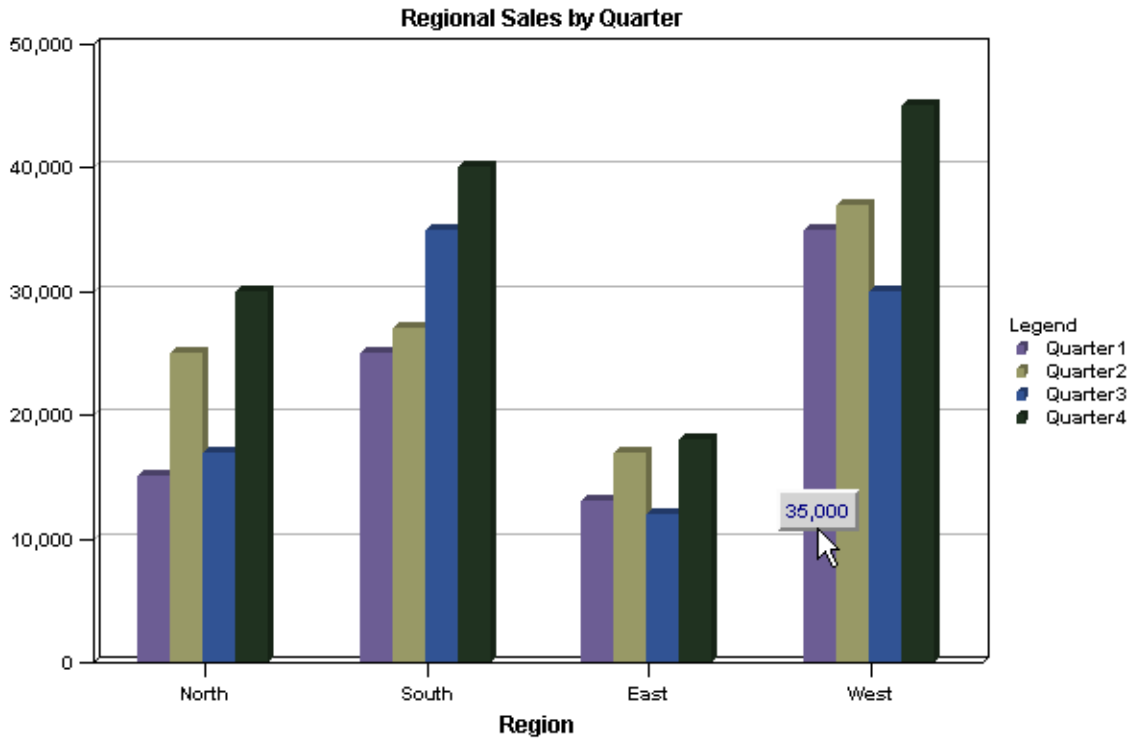
The installation of NetCharts 4.0 should have added a small flat file to the host machine. It can be found under the installation directory of NetCharts. The default installation is c:\program files\visual mining\netcharts 4.0\netcharts. `Regionalsales.dat` can be found in the \examples directory.

Running the Example

The example can be configured and run using the following steps:

1. Copy the `getSalesDataChartFF.jsp` file and regionalsales.dat to any subdirectory under the JSP-enabled Web Application Server. For instance, when using JRun 3.0, it could be in the `c:\program files\allaire\jrun\servers\default\default-app` directory or any subdirectory the \default-app directory (assuming you are running this Java Server Page under the default application directory).
2. Make copies of the NetCharts 4.0 `netcharts.jar` and `NFLicense.dat` files under this same directory. These files can be found in the \classes directory under the NetCharts 4.0 installation directory.
3. Start a web browser and load the JSP page. For example, if the JSP page was installed as in steps 1 and 2 above, the URL would look like `http://localhost:8100/getSalesDataChart.jsp`. This URL points to the root directory of the JRun installation, which is listening on port 8100 of the host machine.

When you load the page, you should see something like the following:



Looking at the Code

The basic sequence of events in using JSP to populate a NetCharts applet is as follows:

- Define the variables
- Open the file
- Read file line by line, and parse the data into Vectors
- Convert the Vectors into their respective variables
- Close the connection to the file
- Instantiate the applet and pass the variables in through the `NFParamScript` parameter.

The following code fragments demonstrate how this is accomplished.

The first section of code defines several variables. These variables are used to open the flat file in a BufferedReader, hold the data in a vector, and then prepare the data in string format to be passed to the NetCharts applet:

```
// variables for connecting to the flat file data source
// Note:  this is set up to look in the same directory as the jsp file.
String fileName = "regionalsales.dat";
BufferedReader breader = null;

// variables for holding the data from the data source
Vector row = new Vector();
String ds1 = "25,50,75,100";
String ds2 = "100,200,300,400";
String ds3 = "100,125,150,200";
String ds4 = "100,75,50,25";
String labels = "\"North\", \"South\", \"East\", \"West\"";
```

The next section of code opens the file in a `BufferedReader`, and prepares the StringTokenizer and variables for parsing the data that is extracted from `regionalsales.dat`.

```
try {
    // open a bufferedreader to read the file
    // get the current path of the jsp under the application server.
    // 'application' is a freebie from the parent servlet.
    String inputPath = application.getRealPath("/") + File.separatorChar;
    breader = new BufferedReader (new FileReader (inputPath + fileName));
    String buffer;
    StringTokenizer token;
    boolean firstLine = true;
    int rowNum = 0;
    ds1 = ds2 = ds3 = ds4 = labels = "";
```

Next, the code loops through each line of the file. The first line of the file is read and ignored. Each line is parsed with a `StringTokenizer`. The resulting data is placed in a Vector. The results are then placed in strings. These strings contain the data in a comma-separated format, which NetCharts can understand. Ultimately, the data will look something like "5,10,15,20". After all of the data is extracted, the file is closed.

```
    // loop through the bufferedreader one line at a time
    while ((buffer = breader.readLine()) != null)
    {
        token = new StringTokenizer (buffer,",");

        // if it's the first line of the file, ignore
        //   else, place into tokenizer and parse
        if (!firstLine)
        {
            // place string tokens into row Vector
            while (token.hasMoreTokens())
             {
                  row.addElement (token.nextToken());
             }

              // if it's after the first line of data, then tack
              //     on a comma.
              if (rowNum > 0)
              {
                     ds1 += ",";
                     ds2 += ",";
                     ds3 += ",";
                     ds4 += ",";
                     labels += ",";
              }

              // copy strings from Vector into data strings
              labels += (String) row.elementAt(0);
              ds1 += (String) row.elementAt(1);
              ds2 += (String) row.elementAt(2);
              ds3 += (String) row.elementAt(3);
              ds4 += (String) row.elementAt(4);

              row.removeAllElements();

             rowNum++;
        }
        else
        {
              firstLine = false;
        }}
```

The applet tag can now be constructed. NetCharts uses an applet parameter called `NFParamScript` to pass in chart definition strings. A simple *name=value* format is used to construct complete chart definitions that can be processed by the NetCharts applet.

```
<applet name=Quarterly Sales
        code=NFBarchartApp.class
        codebase=/classes
        width=600 height=400>
<param name=NFParamScript value='
#Populate the chart with all of the static template information;
ChartName             = "Basic Grouped Barchart";
DebugSet                = ALL;
ChartWidth            = 600;
ChartHeight           = 400;
Background            = (white,NONE,3,null,TILE,black);
...
```

The relevant parameters for this chart are `DataSet1, DataSet2, DataSet3, DataSet4,` and `BarLabels:`

```
#Now populate the chart with the dynamic data extracted from the flat file;
DataSet1                = <%=ds1%>;
DataSet2                = <%=ds2%>;
DataSet3                = <%=ds3%>;
DataSet4                = <%=ds4%>;
BarLabels               = <%=labels%>;
```

When this JSP page runs, it will convert the variables into the strings created earlier. These strings will then be passed in to the applet, and the chart will be created.

For a complete code listing, go to `/examples/getSalesDataChartFF.jsp`.

Visual Mining, Inc.
15825 Shady Grove Road
Suite 20
Rockville, MD 20850

Phone:
800-308-0731 in US
+1-301-795-2200 outside US

E-mail:
sales@visualmining.com
support@visualmining.com

Web:
http://www.visualmining.com