
Visual Mining, a division of Tervela, Inc.

Web Scripting with NetCharts® Pro Applets

**A Programming Language Independent
Guide to Creating Chart Applets with
NetCharts® Pro 7.2
Version 7.2
Summer 2015**

Table of Contents

1.0 SCOPE	4
2.0 A GUIDE FOR THE IMPATIENT	5
3.0 INTRODUCTION	8
THE VISUAL MINING SUITE OF CHART AND DASHBOARD GENERATION SOLUTIONS	8
<i>NetCharts Pro</i>	8
<i>NetCharts Designer</i>	8
<i>NetCharts Server</i>	8
<i>NetCharts Performance Dashboards</i>	8
CHART DEFINITION LANGUAGE	9
NETCHARTS PRO CHART TYPES	10
4.0 DEPLOYING NETCHARTS PRO APPLETS	12
SYSTEM REQUIREMENTS	12
WEB BROWSER REQUIREMENTS	12
CONFIGURING WEB SERVERS TO DELIVER NETCHARTS PRO CHARTS AS APPLETS	12
MIME TYPE CONFIGURATION	12
5.0 INSTALLING LICENSES	14
LICENSE KEYS	14
NFLICENSE.DAT	14
LICENSE ACCESS	14
<i>License File Located Via CODEBASE Attribute</i>	14
<i>License File Located Via LICENSEURL Attribute</i>	15
<i>License File Specified Via LICENSEKEYS Attribute</i>	15
UPDATING LICENSES	16
<i>Troubleshooting License issues</i>	16
6.0 NETCHARTS APPLETS INTERACTIVITY FEATURES	18
<i>Pop-up / Hover Labels</i>	18
<i>Drilldown</i>	19
<i>Zooming and Scrolling</i>	19
<i>Pie Chart Rotation</i>	20
7.0 CONFIGURING NETCHARTS PRO APPLETS	21
NFPARAMSCRIPT	22
NFPARAMURL	22
NFRUNTIMEPROPERTIES	23
<i>MaxDataSets</i>	23
<i>MaxAxes</i>	23
<i>MaxNoteSets</i>	23
<i>DateFormats</i>	23
<i>ReconfigStripChartAxes</i>	23
<i>UseOldComboLayout</i>	23
8.0 DESIGNING APPLICATIONS WITH NETCHARTS PRO APPLETS	24
CREATING CHART TEMPLATES	24
A GENERAL STRUCTURE FOR CHART-ENABLED WEB PAGES	25
9.0 DEBUGGING NETCHARTS APPLETS	29
THE NETCHARTS DEBUGGER	29
USING A WEB BROWSER'S JAVA CONSOLE	29

GENERAL INFORMATION	31
----------------------------------	-----------

1.0 Scope

This document provides detailed information on the methods, features and benefits of using NetCharts® Pro in combination with any web programming or scripting language to deliver applet-enabled web pages. NetCharts Pro is a comprehensive Java programmer-friendly chart library, allowing application developers to create award-winning data visualizations in virtually any format for interactive charting within applications, web-browsers, and internet-enabled devices without the need for a third party plug-in.

A companion document, *The Visual Mining Chart Definition Language Reference Guide*, provides additional useful information on designing chart templates to be used with NetCharts Pro. NetCharts Designer, can also be used to develop chart templates to be used with NetCharts Pro. More information about NetCharts Designer can be found at <http://www.visualmining.com/nc-designer/>.

Additional information on Java is available from <https://www.oracle.com/java/index.html>.

Note to our customers:

Thank you for evaluating and/or purchasing NetCharts Pro Version 7.2. We sincerely believe that the charts and chart images produced by NetCharts Pro 7.2 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*



2.0 A Guide for the Impatient

This section summarizes a minimum process for installing and running NetCharts Pro Charts as Java Applets. These are the minimum steps necessary to get up and running. If you are going to read nothing else in this guide, read this section!

1. Download NetCharts Pro

Unzip the distribution and this will provide you a fully functioning copy of NetCharts Pro on your local system, including the signed NetCharts Pro Applets JAR file **ncp-core.jar**, and **NFLicense.dat** license file,

2. Allow your webserver to access NetCharts Pro files

Two files, **ncp-core.jar** and **NFLicense.dat** are the only files needed and must be accessible by your web server. You can create a virtual directory in your web server pointed to the `/applets` directory of your unzipped NetCharts Pro distribution.

Or, alternatively, place these files at the top level of your web server in a directory called `netcharts`. For example, in Microsoft's IIS server you may create the directory `c:\inetpub\wwwroot\netcharts`.

NOTE: Recent browser updates are changing the way unsigned Java applets and web start applications are run. The default security level for Java applets and web start applications has been increased from "Medium" to "High" affecting the conditions under which unsigned Java web applications can run. A signed JAR file is now recommended for applet deployments.

A signed version of the NetCharts Pro core library, **ncp-core.jar**, used to serve NetCharts Pro charts in applets mode is included within the NetCharts Pro distribution in the `/applets` folder. Use this version, along with the "sandbox" permissions value for most uses.

If you are an existing user and accessing templates or CDL files on a different port or server using the *NFParamURL* feature, use the signed JAR file within the `/applets/all-permissions` folder to allow cross-domain access.

3. Create an HTML page containing a NetCharts Pro <applet> tag

Create a simple web page by copying the following into an empty html file.

```
<html>
<head>
    <title>NetCharts Pro Applets</title>
</head>

<body>
<applet      name='mychart'
              codebase='/netcharts'
              archive='ncp-core.jar'
              code='netcharts.apps.NFBarchartApp'
              width='500'
              height='300'>
<param name='permissions' value='sandbox' />
<param name='NFPParamScript' value='
    AntiAlias = "ON";
    Background = (, NONE, 4, , TILE, , SQUARE, SQUARE, SQUARE, SQUARE, );
    ColorTable = x0a9696, xfec619, x94c209, xcc7138, xd64646, x8e468e,
    xfb2f99, xb8e0ff, xcae78f, xflaab2, xa29c65, x604300, x339276, x64bcb7;
    DwellLabel = ("", black, "Arial", 10, 0, CENTER);
    DwellLabelBox = (xffffffff_229, BOX, 1, , TILE, grey, SQUARE, SQUARE,
    SQUARE, SQUARE, white);
    BottomColor = xc0c0c0_127;
    BottomMajorTicColor = white;
    BottomTics = ("", x000000_178, "Arial Plain", 10, 0, CENTER, , CENTER);
    LeftColor = xc0c0c0_127;
    LeftMajorTicColor = white;
    LeftTics = ("", x000000_178, "Arial Plain", 10, 0, CENTER, , CENTER);
    LeftFormat = (DECIMAL, "###,###.###", null, null);
    GridAxis = ("BOTTOM", "LEFT");
    Grid = (xefefef_127, , xefefef_127, , TILE);
    GridLine = (BOTH, SOLID, 1);
    GraphLayout = VERTICAL;
    BarSymbol = (BAR, );
    BarWidth = 70;
    Bar3DDepth = 0;
    BarBorder = (NONE, 1, );
    BarDropShadow = (x000000_110, 0.01, 0.0090, 0.02);
    BarHighlights = (GRADIENTHORIZONTAL, xxxfff_80, xxxfff_0, 0.0010,
    0.0010, 0.0010, 0.0010, -1, -1, "SQUARE_0.0", "SQUARE_0.0", "SQUARE_0.0",
    "SQUARE_0.0"), (GRADIENTHORIZONTAL, xxxfff_0, x000000_80, 0.0010, -0.09,
    0.0010, -0.09, -0.09, -1, "SQUARE_0.0", "SQUARE_0.0", "SQUARE_0.0",
    "SQUARE_0.0");
    BarCorners = ("ROUND_0.1", "ROUND_0.1", "ROUND_0.1", "ROUND_0.1");
    DataSets = ("DataSet1", , BAR, 4, FILLED);
    DataSet1 = 10, 20, 12, 15, 5, 10;
    BarLabels = "Label1", "Label2", "Label3", "Label4", "Label5",
    "Label6";'>
</applet>
</body>
</html>
```

Figure 1 - Sample applet

Save this file as `mychart.html` in the root directory of the web server. Make sure the web server is started, and able to receive requests from a browser.

4. Request the page from a browser.

Start up a browser, and browse to <http://localhost/mychart.html> (replace *localhost* if the page is deployed to a different system). The following chart should appear:

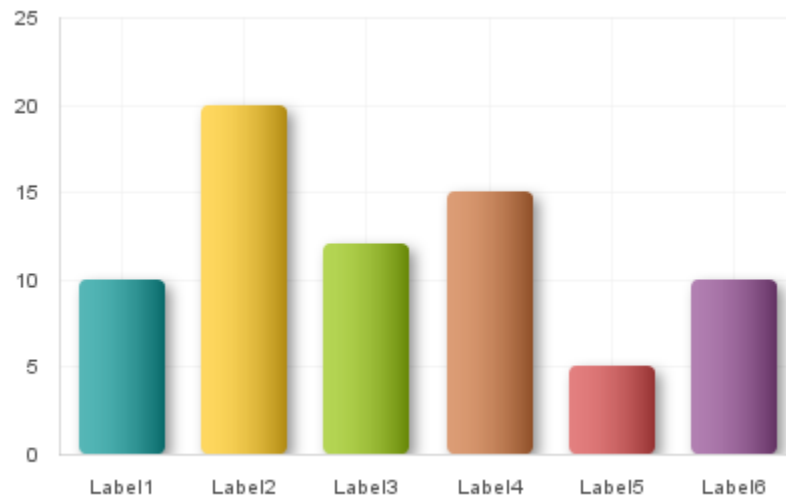


Figure 2 - Chart from Sample applet

3.0 Introduction

NetCharts Pro as Java Applets is one member of a suite of charting solutions offered by Visual Mining. All of the products are written entirely in Java and use some form of Visual Mining's *Chart Definition Language* (CDL) as a basis for defining and generating charts. Because of this common base rendering engine, all NetCharts solutions are capable of generating the same charts. The products differ primarily in the way in which each is integrated into the software infrastructure of a user's application.

The Visual Mining Suite of Chart and Dashboard Generation Solutions

NetCharts Pro

NetCharts Pro (<http://www.visualmining.com/nc-pro/>) is a Java programmer-friendly chart generation solution. The NetCharts Pro API allows it to be used in Integrated Development Environments (IDE) such as Eclipse and IBM's WebSphere Studio. NetCharts Pro can create images of charts in standard web formats such as PNG, SVG and JPG, making it ideally suited for server-side use to chart enable applications using EJBs, servlets or JSP pages.

NetCharts Designer

NetCharts Designer (<http://www.visualmining.com/nc-designer/>) provides a comprehensive desktop Integrated Development Environments (IDE) for creating and managing charts, graphs, tables, interactive dashboards, and scorecards that can then be used in web-based applications. NetCharts Designer streamlines data analytics and development with one simple interactive dashboard solution. NetCharts Designer can be used to create chart templates for use by NetCharts Pro and complete dashboard applications for deployment within NetCharts Server.

NetCharts Server

NetCharts Server (<http://www.visualmining.com/nc-server/>) is a platform that can present complete dashboards or charts that developers define and publish using NetCharts Designer, the solution IDE. Together they are a traditional dashboard development solution allowing users complete control over the content, styling and interactivity included in the dashboards to implement very complex and rich dashboards. The NetCharts Server platform can also be used in conjunction with an entire range of web infrastructures from the simplest CGI scripts, to the most sophisticated Enterprise Application Servers. Its simple HTTP based interface and pre-built toolkits can be used by nearly any server-side web programming language (e.g. .NET, JSP, Java, CFML, PERL and C) to dynamically create chart enabled web pages.

NetCharts Performance Dashboards

NetCharts Performance Dashboards (<http://www.visualmining.com/ncpd/>) is a complete end-user enabling, agile, dashboarding solution. NetCharts Performance Dashboards allows users or end-users to view, create and customize dashboards. No coding or programming is required and there are no languages to learn to connect to your data and produce and present dashboards. It's all done automatically based on selections, allowing for rapid, agile dashboard development with rich client-side interactivity.

Chart Definition Language

Visual Mining's *Chart Definition Language (CDL)* is a simple ASCII scripting language. There are 20 basic chart types defined in CDL, each of which has hundreds of configurable attributes, allowing for the creation of thousands of different charts. See the *Visual Mining Chart Definition Language Reference Guide* for a complete description of CDL.

```

ChartType = LINECHART;
ChartName = "Basic Line Chart";
ChartSize = (400,250);
Background = (, BOX, 1, , TILE, silver, SQUARE, SQUARE, SQUARE, SQUARE, );
Header = ("Linechart", grey, "Arial", 16, 0, CENTER, CENTER, "OFF");
GridLine = (BOTH, SOLID, 1);
Grid = (xc0c0c0_89, white, white, , TILE);
LineSets = ("LineSet1", );
LineSet1 = 100, 125, 245.78, 147, 67;
LineDropShadow = (x000000_110, 0.01, 0.0090, 0.02);
LineSymbolSpotlights = (xffffffff_150, xxxxxxxx_0, CENTER, 0, 0, -0.5, -0.25, 0.9999);
LineSymbol = (CIRCLE, 9, FILLED, , 1, , blue, 0);
LineStyle = (SOLID, 1, blue, , NORMAL, );
LeftMajorTicColor = x000000_0;
LeftColor = x000000_0;
LeftScale = (0, 300, 50);
LeftTicks = ("", grey, "Arial Plain", 11, 0, CENTER, , CENTER);
BottomLabels = "January", "February", "March", "April", "May";
BottomMargins = (8, 13);
BottomTicks = ("", grey, "Arial Plain", 11, 0, CENTER, , CENTER);
BottomMajorTicColor = x000000_0;
BottomColor = x000000_0;

```

Figure 3-CDL for a Linechart

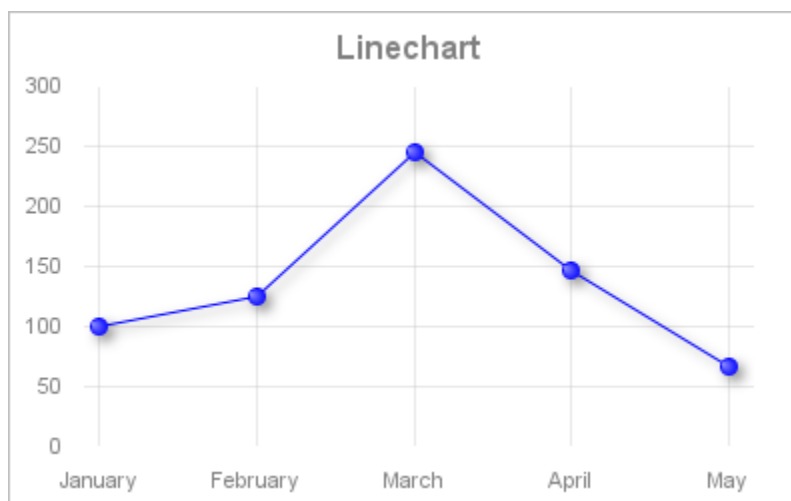
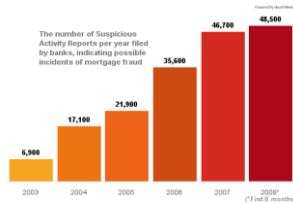


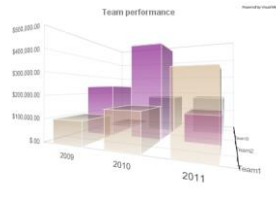
Figure 4 - Linechart created by CDL in Figure 3

NetCharts Pro Chart Types

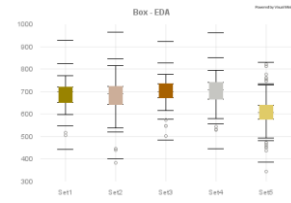
All of Visual Mining's charting solutions use the Chart Definition Language (CDL) to create and manipulate charts. This common use of CDL makes it easy to preserve chart definitions when moving from one product to another. The following shows the 20 chart types that are supported as Java Applets with NetCharts Pro:



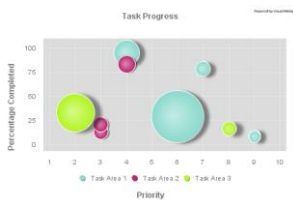
Bar



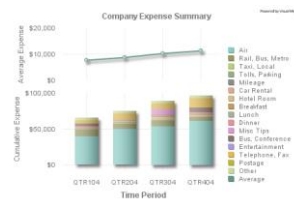
3d bar



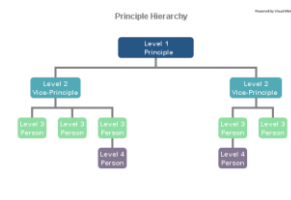
Box



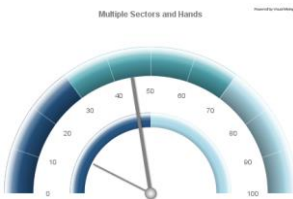
Bubble



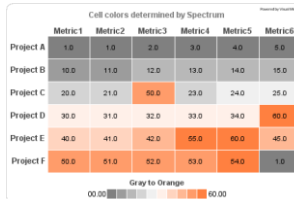
Combo



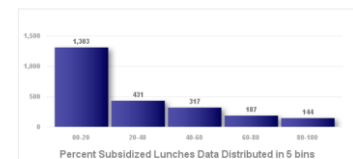
Diagram



Dial



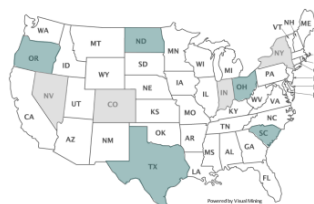
Heat map



Histogram



Line



Map

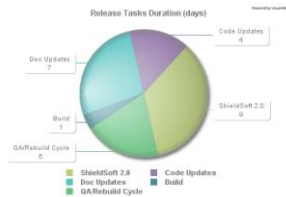


Multi-pie

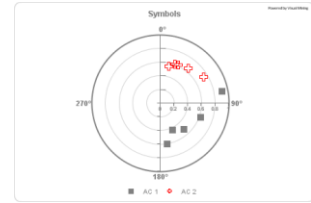
3.0 Introduction



Pareto



Pie



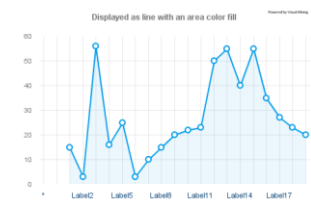
Polar



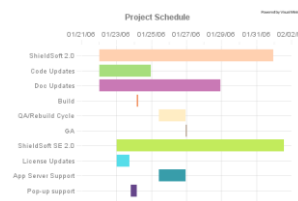
Radar



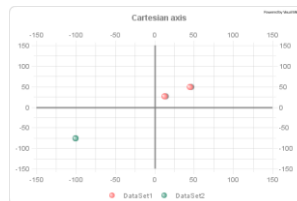
Stock



Strip



Time



XY plot

4.0 Deploying NetCharts Pro Applets

System Requirements

Rendering NetCharts Pro as Java Applets will occur client-side. So there are no minimum requirements for the hosting server or system. The minimum NetCharts Pro files require 1MB of disk space.

Web Browser Requirements

A minimum Java Plugin 1.6 or greater is required to run NetCharts Pro Applets.

Recent browser updates are changing the way unsigned Java applets are run. The default security level for Java applets has been increased from "Medium" to "High". This affects the conditions under which unsigned Java web applications can run. With the "High" setting the user is always warned before any unsigned application is run, and may not allow unsigned applications to run at all, to prevent silent exploitation. At the "High" setting the JRE will now require that users indicate acceptance for the Java runtime to run an unsigned or self-signed applet within a web page or may not run the application at all.

A signed version of the NetCharts Pro core library, **ncp-core.jar**, used to serve NetCharts Pro charts in applets mode is available within the NetCharts Pro distribution in the *applets* folder.

Configuring Web Servers to deliver NetCharts Pro charts as Applets

In order to distribute NetCharts Pro charts as Applets via a web server, it is necessary to either create a virtual directory under the web server to link to the NetCharts Pro Applet files, or copy **ncp-core.jar** and **NFLicense.dat** files to one or more directories within the web server hierarchy. For example, in Microsoft's IIS server you may create the directory `c:\inetpub\wwwroot\applets`.

For a virtual directory, it is recommended that an */netcharts* alias be created at the top of the web server hierarchy and linked to the */applets* directory in the original NetCharts Pro distribution containing **ncp-core.jar** and **NFLicense.dat**. This will enable any HTML file in the web server hierarchy to easily reference the applets using the following applet tag:

```
<applet codebase='/netcharts' archive='ncp-core.jar'  
code='netcharts.apps.NFPiechartApp' ...>
```

MIME Type Configuration

When using NetCharts Applets in Microsoft IIS server, the default configuration for IIS will prevent the validation of the license for NetCharts Applets. This results in a screen starting that NetCharts Applets is not properly licensed.

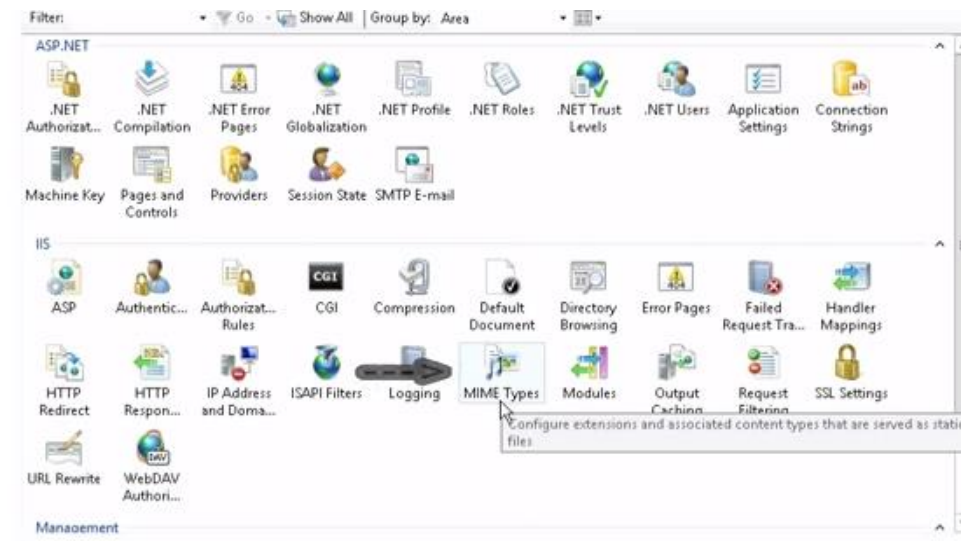
The first step to correct this issue is to verify that the license file required for NetCharts Applets, **NFLicense.dat**, is in the correct location. In the `<applet>` tag is an attribute named *codebase*. This attribute points to the location where the NetCharts Applets library, **ncp-core.jar**, can be found. Additionally, the **NFLicense.dat** file containing the license must be located in the same directory. You can verify the existence of the file by entering `http://server<codebase location>/NFLicense.dat` in a browser. For instance, if the codebase in the applet tag is `"/classes"` on the local system, then entering `http://localhost/classes/NFLicense.dat` should return the license file.

If you receive a 404 Status Error when trying to verify the **NFLicense.dat** file, your IIS is not configured to serve *.dat* files. This will prevent NetCharts Applets from verifying the license. Most IIS servers need

to have the *.dat* MIME type configured to allow access to the license file. To do so follow these instructions:

1. Open the *Internet Information Services Management Console*.
2. Select the site or server.
3. Click *MIME Types* (it might be under the IIS section).
4. Click *Add* button.
5. In the *Extension* field, type in *.dat*.
6. In the *Content Type* or *MIME Type* field, type *text/plain*.
7. Apply the new settings.

Your IIS instance might have slightly different instructions but essentially find the *MIME Types* control area and add the *.dat* MIME Type as *text/plain*.



5.0 Installing Licenses

All NetCharts Pro Applets are licensed to operate off of one or more servers. Once a server is licensed, any number of clients can consume NetCharts Pro Applets charts without restriction.

License Keys

NetCharts Pro Applets license keys are generated based on the:

- version of NetCharts Pro being used
- unique customer or project identifier
- expiration date if applicable

License keys are formatted as follows:

```
NetCharts7.2 BANNER=NO EXPIRATION=31-JUL-2016 Zm9v=Z29v
KEY=NNZXMBYDFYMZZKKLSROCKSEMGBHHNFDKCCJE
```

NFLicense.dat

License keys are stored in the **NFLicense.dat** file and are normally located on the Java Applet *classpath* in the same directory as the **ncp-core.jar** file. This is the most convenient way to access and maintain license keys, and has been successfully employed by most NetCharts users.

The **NFLicense.dat** file should have the same file permissions set as the NetCharts jar file. For most web servers, the files must be readable by "everyone" to be retrieved using a browser. Write permission is NOT required for anyone except the Web/Java administrator if applicable to the hosting platform.

License Access

NetCharts Pro Applets must be able to locate a license key at runtime. The following methods are available to allow users to configure their application's NetCharts license.

License File Located Via CODEBASE Attribute

By default, all NetCharts Pro Applets attempt to read the **NFLicense.dat** file from the directory specified in the applet's `codebase` attribute. The `codebase` attribute specifies the top-level directory containing the NetCharts **ncp-core.jar** file.

For example, if the **ncp-core.jar** and **NFLicense.dat** files are located at the top of your Web hierarchy in the */netcharts* directory, the following applet tag would be correct:

```
<applet codebase='/netcharts'
        archive='ncp-core.jar'
        code='netcharts.apps.NFPiechartApp'
        width='400' height='400'>
```

On the other hand, if the NetCharts files are located in a directory that is relative to the current document directory, then the following applet tag could be used:

```
<applet codebase='../..//netcharts'
        archive='ncp-core.jar'
        code='netcharts.apps.NFPiechartApp'
        width='400' height='400'>
```

In either case, if the **nep-core.jar** and **NFLicense.dat** files are located in the directory specified by the `codebase` attribute then the license validation will succeed.

Most Web browsers are capable of viewing HTML documents in the local file system by specifying a file path URL (e.g. `file:///C:/charts/mychart.html`). In such cases, if relative addressing is used for the `codebase` attribute then the license file processing will succeed. However, if absolute addressing is specified in the `codebase` attribute (i.e. the `codebase` value begins with a protocol like `http://` or a relative path from the server root like `/`) then the NetCharts files will not be located properly and charts will fail to load.

License File Located Via `LICENSEURL` Attribute

In situations where it is not convenient or possible to place the **NFLicense.dat** file in the `CODEBASE` directory, it may be placed anywhere that is accessible via a Web browser. This is accomplished by specifying the *LicenseURL* parameter within the `<applet>` tag that defines a NetCharts Pro applet.

For example, the following applet tag defines a simple piechart, that uses a *LicenseURL* that is different from the `codebase` path:

```
<applet codebase='/netcharts'
        archive='nep-core.jar'
        code='netcharts.apps.NFPiechartApp'
        width='200' height='100'>
<param name='permissions' value='all-permissions' />
<param name=NFPParamScript value ='
        LicenseURL = "/admin/NFLicense.dat";
        SliceData = 12,34,56,78;
'>
</applet>
```

Any URL can be used, including a CGI command, provided the resulting data stream is a text/plain file containing a valid license key for the IP address of the Web Server. (The data stream can contain any number of license keys, with each one being checked.)

NOTE: Browser security measures prevent applets from accessing a Web server other than the one the applet classes were loaded from. This makes it impossible to locate your License file on another Web server if you plan on running NetCharts from within a browser. **Note:** The *LicenseURL* uses an absolute address to specify the location of the license file, while the `codebase` can a relative address to locate the NetCharts class files.

NOTE: With recent browser updates, the additional *permissions* parameter set to *all-permissions* will need to be set to allow for access to the remote license. Running your applets with *all-permissions* will limit the warning dialogs to a single dialog when the first NetCharts Pro Applet is presented. You must also use the signed JAR file within the **/applets/all-permissions** folder of the NetCharts Pro distribution.

NOTE: The *LicenseURL* method only works for HTML documents that are viewed from a Web server. Web browsers restrict URL access whenever an HTML document is viewed directly from the file system.

License File Specified Via `LICENSEKEYS` Attribute

One or more license keys can be specified directly in the HTML document using the *LicenseKeys* parameter. However, this method requires the entries to be made in every HTML document using a NetCharts Pro Applet can be an administrative and maintenance burden.

The *LicenseKeys* parameter defines one or more quoted strings, separated by commas, each of which specifies a license entry. For example, the following parameter definition specifies keys for three different servers, thereby enabling this document to run off of those servers:

```
<applet codebase='/netcharts'
        archive='ncp-core.jar'
        code='netcharts.apps.NFPiechartApp'
        width='400' height='400'>
<param name='permissions' value='sandbox' />
<param name=NFPParamScript value='
    LicenseKeys = "NetCharts7.2 KEY=XX...";
    SliceData = 12,34,56,78;
'>
</applet>
```

Updating Licenses

The evaluation copy of NetCharts Pro Applets comes with a temporary 30 day license. When the evaluation period has expired, NetCharts Pro Applets will display the following image.



Figure 5 - NetCharts Evaluation Expiration Splash Screen

If you have a current agreement, or once you purchase a license for NetCharts Pro, Visual Mining Technical Support will send you a license file that will enable NetCharts Pro operation. This file should replace the temporary license file that accompanied the NetCharts Pro distribution.

Troubleshooting License issues

If you encounter difficulty configuring your applets to find their license, consider the following.

1. Place the **NFLicense.dat** file (spelled properly with upper and lower case) in the same directly you have chosen to place **ncp-core.jar**.
2. If the evaluation banner is not displayed, then the applet itself is probably not running. This is usually the result of a Web browser or server misconfiguration. Check the Java plugin console for any error messages. This console is typically accessed from the system tray on a windows machine.
3. All web browsers provide some form of file caching to increase performance and reduce network load. Try clearing this browser cache or your Java plug-in cache, exit and restart your browser. That will ensure that the old class files are no longer in the cache and the new ones will be reloaded, along with any new license file.

4. If the evaluation or invalid license banner is still displayed, but you believe that you have a valid license file, you can add the following Parameter to your NetCharts Pro Applet to print out debug information on the Java Console:

```
DebugSet = LICENSE;
```

5. If the debug output shows that the license file is not being located properly, review the options described above and re-check any `CODEBASE` or `CLASSPATH` variables. If the NetCharts applets are located on the `CLASSPATH` of the browser AND the `CODEBASE` points to someplace where the license file isn't, then the applets will execute, but the license file will not be found.

Developers sometimes use the `CLASSPATH` variable to speed up access to Java applets, but that practice leads to confusion when Java applets are accessed by normal users via the Web server. Also, security restrictions imposed by most browsers prevent an applet that is loaded from the local file system (via the `CLASSPATH`) from accessing files via a URL (which is how the license file is usually accessed.) In such situations, an alternative license scheme should be used, such as the *LicenseKeys* parameter. (Note, the *LicenseURL* method will NOT work either, because of the same security restriction.)

6. If you still can't solve the license problem, contact [Visual Mining Technical Support](#) for assistance. For faster assistance, please be prepared to provide the following information:
 - o Your Name
 - o Company Name
 - o Phone Number (for return calls only)
 - o Web Server O/S Version
 - o Web Browser Version
 - o Java Console Output

6.0 NetCharts Applets Interactivity Features

There are several different types of interactivity available when presenting applets using NetCharts Pro. These include interactive pop-up/hover labels, drilldown, zooming/scrolling, and pie chart rotation and explosion.

Pop-up / Hover Labels

Popup labels are labels that appear over any activated region. By default NetCharts Pro will generate standard popup labels for data elements such as bars, line point, pie slices, etc. But nearly every item on a chart, including titles, axes, legends and individual data points has an associated `ActiveLabel` to allow for direct control over the labels. For instance, consider the following pie chart CDL:

```
DwellLabel    = ("",black,"Arial Plain",16,0);
DwellLabelBox = (wheat,BOX,2,null,TILE,grey);
SliceData     = 15, 25, 35, 25,5;
ActiveLabels  = ("vmi"), ("yahoo"), ("msn"), ("amazon"), ("google");
```

Figure 6 – CDL for Pop-up labels

If the mouse were held over the wedge with the value of 14, it would appear something like:

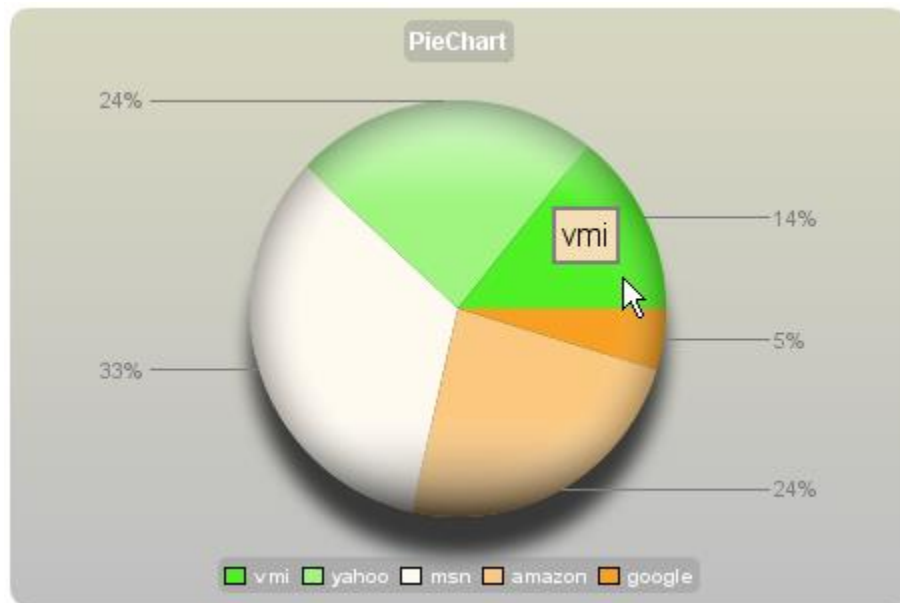


Figure 7 – Active Label from CDL in Figure 7

Popup labels are useful when displaying additional information related to the data that isn't necessarily a static part of the chart.

See *The Visual Mining Chart Definition Language Reference Guide*, Chapter 6, for more details on how `ActiveLabels` work.

Drilldown

Drilldown allows the user to click on any active region in the chart and ‘drilldown’ to new information—be it in a new browser window that pops up, a different frame, etc. Drilldown is enabled through the various `ActiveLabels` CDL parameters. For example, if a pie chart had the following CDL:

```
SliceData      = 15, 25, 35, 25;
ActiveLabels   = ("vmi", "http://www.visualmining.com"),
                  ("yahoo", "http://www.yahoo.com"),
                  ("msn", "http://www.msn.com"),
                  ("amazon", "http://www.amazon.com");
```

Figure 8 - ActiveLabels for Drilldown

Clicking on the respective wedges would then take the user to the particular website that is indicated in the `ActiveLabels` parameter value for the pie slice. For instance, if the wedge with the value of 15, the first pie slice, were clicked on, the Visual Mining website would appear.

The `ActiveLabels` CDL parameter also allows for targeting the drilldown. This means that the chart designer can force the drilldown to occur in a particular target. For instance, the above CDL can be modified so that a target window was specified:

```
SliceData      = 15, 25, 35, 25;
ActiveLabels   = ("vmi", http://www.visualmining.com, "_new"),
                  ("yahoo", http://www.yahoo.com, "_new"),
                  ("msn", http://www.msn.com, "_new"),
                  ("amazon", http://www.amazon.com, "_new");
```

Figure 9 – Specifying drilldown targets in Active Labels

If an end-user were to click on any one of the active regions (the pie slices or wedges), the users would be redirected to a specific window or tab (if one existed by the name in the third attribute), or a new window or tab when using “_new”.

See *The Visual Mining Chart Definition Language Reference Guide*, Chapter 6, for more details on how `ActiveLabels` work.

Zooming and Scrolling

Rectangular charts that use axes (e.g. bar, line, x-y/quadrant charts, etc.) have the ability to scroll and zoom. This is done by specifying several CDL parameters. For instance, consider the following CDL:

```

DataSets      = ("BarSet1",null,BAR,4,FILLED);
DataSet1      = 100,125,245.78,147,167,120,60;
BarLabels     = "Mon","Tue","Wed","Thu","Fri","Sat","Sun";

Header        = ("Title",black,"SansSerif",12,0);
HeaderBox     = (null,NONE,5,null,TILE,black);
BarWidth      = 61;
Bar3DDepth    = 5;

Grid          = (lightgray,null,black,null,TILE);
GridAxis      = (BOTTOM,LEFT);
GridLine      = (BOTH,SOLID,1);

ColorTable    = x6c5d94,x999966,x315394,x213321,x00566f,x690931,x515f23;

BottomTicks   = ("ON",black,"SansSerif",10,0,null);

BottomScale   = (1,5,1);
BottomScroll  = (0,6);

```

The key parameters are `BottomScale` and `BottomScroll`. `BottomScroll` indicates the total range of data values the chart is capable of displaying. Notice on line 2 that `DataSet1` has 7 values. `BottomScroll` corresponds to this by allowing a range from 0 to 6 (NetCharts uses 0 as the base for these calculations). `BottomScale` indicates what is currently being displayed. In this case, a bar chart with 5 bars (out of a possible 7) is being displayed:

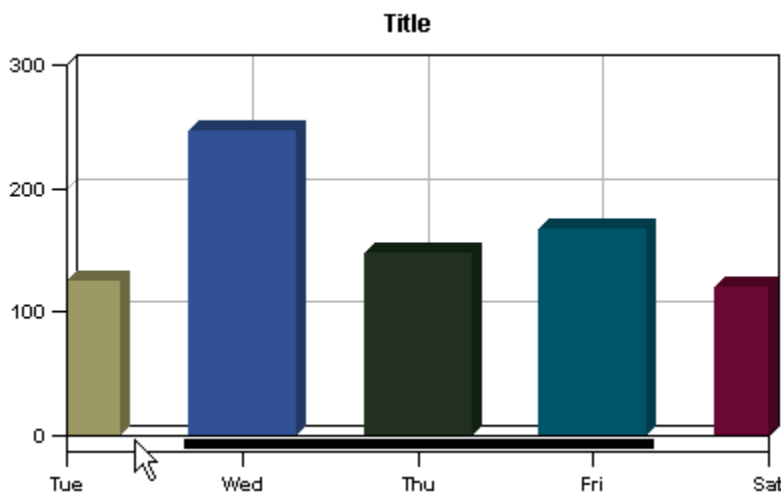


Figure 11 - Scrolling and Zooming with a Bar Chart

Clicking on the scrollbar in the applet will scroll the bars back and forth. Right-clicking in the grid region will zoom out. Clicking and dragging in the grid region will zoom in.

All axes have scrolling and zooming capabilities. See Chapter 4 of *The Visual Mining Chart Definition Language Reference Guide* for more details on Axis parameters.

Pie Chart Rotation

Pie Chart applets can be rotated, and the pie slices/wedges can be interactively exploded out from the center of the pie. Clicking on a pie chart and moving the mouse around the center will rotate the pie. Right-clicking on a wedge and 'pulling' the wedge out from the center will explode the pie.

7.0 Configuring NetCharts Pro Applets

As stated previously, NetCharts Pro Applets are controlled via Visual Mining's Chart Definition Language (CDL). CDL is a *name=value* parameter language that is used to completely manage the look and feel of the charts. From the tic marks on an axis to the width of the lines in a line chart to the background to the text in the legend—all aspects of a chart are managed via CDL. The following figure shows some sample CDL.

```
DataSets      = ("BarSet1",null,BAR,4,FILLED);
DataSet1      = 100,125,245.78,147,167,120,60;
BarLabels     = "Mon","Tue","Wed","Thu","Fri","Sat","Sun";

Header        = ("Title",black,"SansSerif",12,0);
HeaderBox     = (null,NONE,5,null,TILE,black);
BarWidth      = 61;
Bar3DDepth    = 5;

Grid          = (lightgray,null,black,null,TILE);
GridAxis      = (BOTTOM,LEFT);
GridLine      = (BOTH,SOLID,1);

ColorTable    = x6c5d94,x999966,x315394,x213321,x00566f,x690931,x515f23;

BottomTics    = ("ON",black,"SansSerif",10,0,null);

BottomScale   = (1,5,1);
BottomScroll  = (0,6);
```

Figure 13 - Sample CDL

Individual CDL parameters can be passed into a NetCharts applet via the *<param>* tag that is a part of the *<applet>* tag specification. For example, a bar chart might appear in an HTML file like:

```
<applet name=mychart
        code= netcharts.apps.NFBarchartApp
        codebase=/netcharts
        archive=ncp-core.jar
        width=400
        height=250>
<param name='permissions' value='sandbox' />
<param name='DataSets' value=' ("BarSet1",null,BAR,4,FILLED)'>
<param name='DataSet1' value=' 100,125,245.78,147,167,120,60'>
<param name='BarLabels' value=' "Mon","Tue","Wed","Thu","Fri","Sat","Sun"'>
</applet>
```

Figure 14 – Individual param tags using CDL

Multiple parameters are passed in via the *<param>* tag. This can, however, get tedious, especially when many parameters are being passed in, so NetCharts allows for multiple ways of passing parameters into a chart.

NOTE: With recent browser updates, it is recommended to use the additional *permissions* parameter and run the applet in *sandbox* mode. Running your applets in *sandbox* mode, it is possible to limit the warning dialogs to a single dialog when the first NetCharts Pro Applet is presented.

NFParamScript

NetCharts has a parameter called *NFParamScript*. This parameter can be used to combine the CDL parameters into a single ‘script’. For example, the above applet code could be rewritten as:

```
<applet name=mychart
  code=netcharts.apps.NFBarchartApp
  codebase=/netcharts
  archive=ncp-core.jar
  width=400
  height=250>
<param name='permissions' value='sandbox' />
<param name='NFParamScript' value='
  DataSets=("BarSet1",null,BAR,4,FILLED);
  DataSet1=100,125,245.78,147,167,120,60;
  BarLabels="Mon","Tue","Wed","Thu","Fri","Sat","Sun";
'>
</applet>
```

Figure 15 - Sample *NFParamScript*

See Chapter 2 of the *The Visual Mining Chart Definition Language Reference Guide* for more details on how *NFParamScript* works.

NFParamURL

Instead of placing all of the parameter definitions within an HTML file, you can use a URL access to retrieve the parameter definitions. For example:

```
<applet name=mychart
  code=netcharts.apps.NFBarchartApp
  codebase=/netcharts
  archive=ncp-core.jar
  width=400
  height=250>
<param name='permissions' value='all-permissions' />
<param name='NFParamURL' value='barchart.dat'>
</applet>
```

Figure 16 - Sample *NFParamURL*

Now, *barchart.dat* can be a static file located within the codebase of the applet, or it could actually be pointing to a file generated dynamically by a server-side script, or it could even be a script itself, generating data on-the-fly. Regardless, the resulting content-type needs to be `text/plain`. Note, if the URL references a file outside of the codebase folder, the default applet security features in most web browsers will prevent the applet from accessing the file.

NOTE: With recent browser updates, the additional *permissions* parameter set to *all-permissions* will need to be set to allow for cross-domain access to the remote CDL. Running your applets with *all-permissions* will limit the warning dialogs to a single dialog when the first NetCharts Pro Applet is presented. You must also use the signed JAR file within the **/applets/all-permissions** folder of the NetCharts Pro distribution.

See Chapter 2 of the *The Visual Mining Chart Definition Language Reference Guide* for more details on how *NFParamURL* works.

NFRunTimeProperties

NetCharts has a parameter called *NFRunTimeProperties* that provides general directives to a NetCharts Pro Applet. These parameters override default processing when needed and are loaded when the Java Virtual Machine is created, and before any NetCharts objects are constructed.

MaxDataSets

MaxDataSets allows users to control how many data sets a NetCharts applet can have. The default is **50**.

MaxAxes

MaxAxes defines the maximum number of axes a chart can have in any one dimension. The default is **10**.

MaxNoteSets

MaxNoteSets defines the maximum number of note sets a chart can have. The default is **20**.

DateFormats

DateFormats can be used to define legal date strings formats in a chart. The default is the standard Java SimpleDate format.

ReconfigStripChartAxes

ReconfigStripChartAxes controls axis drawing on dynamic strip charts. When this property is set to true, NFStripChart will continually recalculate the layout of the axis labels. This allows the chart to handle widely variable axis labels. This has performance implications and should not be used unless labels are clipped. For example, there is no need to reconfigure the axis layout if every axis label will be of the form *hh:mm*. The default value is **false**.

UseOldComboLayout

Starting in version 6, when a Combochart is drawn with *GraphLayout=HORIZONTAL* the X and Y coordinates of data points in each line series are transposed. This is considered correct behavior. Previous versions did not transpose these coordinates, which required users to manually transpose to achieve correct results. The original behavior can be maintained by using this runtime property. Legal values are **true** and **false**. Default value is **false**.

The following applet tag shows *NFRunTimeProperties* set up to increase the number of data sets in a chart to 75 and to activate the Netscape printing fix.

```
<applet name=mychart
  code=netcharts.apps.NFBarchartApp
  codebase=/netcharts
  width=400
  height=250>
<param name='permissions' value='all-permissions' />
<param name='NFParamURL' value='http://anywhere.com/barchart.dat'>
<param name='NFRunTimeProperties' value='MaxDataSets=75 NetscapeprintFix=true'>
</applet>
```

Figure 17—Setting *NFRunTime* properties

8.0 Designing Applications with NetCharts Pro Applets

NetCharts Applets can be used to chart-enable a variety of Java programs including desktop applications, and applets. This chapter presents general guidelines for adding applets to web pages using NetCharts Pro 7.2.

Creating Chart Templates

It can be useful to consider using *chart templates* when designing applications that use NetCharts Applets.

Chart templates are files or strings of CDL that define a chart or portions of a chart. Chart templates support the notion of separating a chart definition into static and dynamic components. The static component of a chart definition contains all those attributes that do not change between invocations. The dynamic component of a chart definition contains those attributes that are unique to a particular chart request.

Consider an application that contains a bar chart. Much of the definition of that bar chart will likely be static. For instance, its background may always be white; it may always use 12-point Helvetica fonts, and contain a title, a legend and two axes. The dynamic component of the chart might be limited to the actual bar values, the axis labels and the text of the title.

The static components of charts can be created in advance and stored in chart templates. At runtime, the application could create an instance of a NetCharts applet, initialize it with a chart template, add any additional dynamic data using the `set()` API, and then generate the chart image.

Creation of chart templates can be done inside the application code, or externally with [NetCharts Designer](#). NetCharts Designer is a desktop design tool that allows chart authors to visually design chart templates. NetCharts Designer provides graphical editors to allow quick logical access to every chart attribute. The resulting chart definition can be exported as a file of Resolved CDL (RCDL) and used in the construction or initialization of a NetCharts Pro object. Resolved CDL has had all references to styles, and external data resolved into CDL parameters. The following figure shows NetCharts Designer being used to create a chart template.

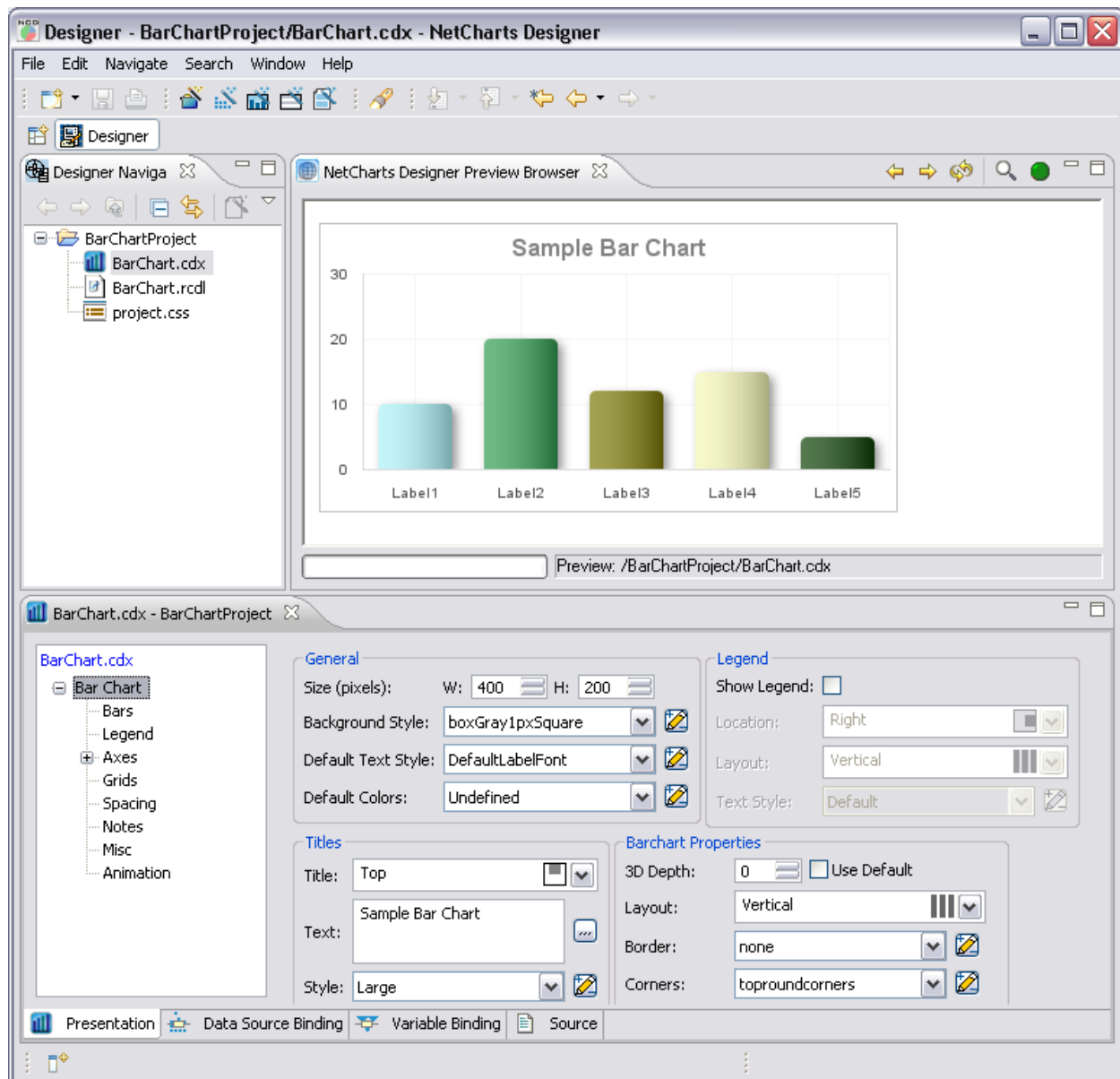


Figure 18 – Creating a chart template with NetCharts Designer

Creating and using chart templates also simplifies the process of migrating a NetCharts application to one of Visual Mining's other charting solutions. A chart template created for use with NetCharts can also be used by NetCharts Pro or NetCharts Server. This allows programmers to adapt their programs to changing application infrastructures.

A General Structure for Chart-Enabled Web Pages

NetCharts can be used in tandem with server-side scripting languages to provide a complete raw-data-to-finished-chart solution. For example, Java Server Pages (JSP) or ASP.NET WebForms can be used to extract raw data from a database, format that data to comply with NetCharts CDL, and use that CDL to create an applet definition.

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

- Define the variables
- Connect to the database and pass in the SQL statement
- Open the database and extract the data
- Populate the variables with the data from the database
- Close the connection to the database
- 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 connect to the database, pass in the SQL statement, and hold the values in string format to be passed to the NetCharts applet:

```
// variables for connecting to the ODBC data source
String driverName = "sun.jdbc.odbc.JdbcOdbcDriver";
String serverURLBase = "jdbc:odbc:";
String dbName = "regionalsales";
String sql = "Select region, quarter1, quarter2, quarter3, quarter4 from
Sales";

// variables for holding the data from the data source
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\"";

Connection connection = null;
Statement statement;
boolean isdata;
ResultSet rs;
int updateCount = 0;
```

Figure 19 – Defining JSP variables

The next section of code loads the JDBC driver, opens a connection to the database, and prepares and executes the SQL statement.

```
try {
    Class.forName(driverName);
} catch (Exception e){
}

try {
    // Open connection to db
    connection = DriverManager.getConnection(serverURLBase+dbName);

    // create statement to handle SQL
    statement = connection.createStatement();
    isdata = statement.execute(sql);
}
```

Figure 20 – Connecting to a JDBC data source

Next, the code loops through the ResultSets extracted from the database. The resulting data is placed into 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 each row of data is extracted, the ResultSet is closed. After all of the data is extracted, the Statement and Connection are closed.

```
int loopCount=0;
// This loop handles multiple sql statements or stored procedures
// that return multiple result sets.
do {
    loopCount++;
    if (loopCount > 50){
        // This prevents errors, like with the Symantec
        // driver, always returning an update count of 0
        break;
    }

    rs = statement.getResultSet();
    ResultSetMetaData rsmd = rs.getMetaData();
    int numColumns = rsmd.getColumnCount();

    boolean isdatainrow;
    ds1 = ds2 = ds3 = ds4 = labels = "";
    for (int rowNum=0; rs.next(); rowNum++){
        Vector row = new Vector();

        if (rowNum > 0)
        {
            ds1 += ",";
            ds2 += ",";
            ds3 += ",";
            ds4 += ",";
            labels += ",";
        }

        Object o = null;
        for (int i=1; i <= numColumns; i++){
            // we would return the actual object, but
            // the JDBC-ODBC bridge crashes. :-(
            o = rs.getString(i);
            row.addElement(o);
        }

        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);
    }
    rs.close();
    System.gc();
} while (statement.getMoreResults() == true);

// close the statement to clean up cursors.
statement.close();
if (!connection.isClosed())
    connection.close();
```

Figure 21 – Extracting data from the data source

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='netcharts.apps.NFBarchartApp'
        codebase='/netcharts'
        archive='ncp-core.jar'
        width='600' height='400'>
<param name='permissions' value='sandbox' />
<param name='NFPParamScript' 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);
...

```

Figure 22 - constructing an applet in JSP

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

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

```

Figure 23 - populating a chart with JSP variables

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.

9.0 Debugging NetCharts Applets

The NetCharts Debugger

There are several useful debug mechanisms built into NetCharts.

The utility class *NFDebug* can be used to turn on various levels of debug output. See the API documentation for details on each option. Adding the following line to a NetCharts Pro server-side Java program will print detailed diagnostics as the chart starts and runs.

```
netcharts.util.NFDebug.set("ALL");
```

While debugging it can also be useful to extract and examine the complete chart definition that the chart is using. This can be done with the following server-side Java command:

```
System.out.println(chart.getCDL());
```

Using a Web Browser's Java Console

Web browsers also have a Java Console available for debugging applets. It is possible, therefore, to get NetCharts debugging output via the Java Console. If the `DebugSet` CDL parameter is used, output specific to the particular setting will be printed. To get all NetCharts debugging output, use `DebugSet = ALL`. Other NetCharts-specific `DebugSet` attributes are:

- ACTION
- AXIS
- BEANS
- CACHE
- DWELL
- FILE
- GRAPH
- IMAGE
- JDBC
- LEGEND
- LICENSE
- NOTES
- PARAM
- REMOTE
- SECURITY
- SYMBOL
- THREAD
- WINDOW

These attributes will cause NetCharts to output context-specific information, so that any problems with the chart can be narrowed down to specifics.

```
<applet      name='mychart'
              codebase='/netcharts'
              archive='ncp-core.jar'
              code='netcharts.apps.NFBarchartApp'
              width='408' height='256'>
<param name='permissions' value='all-permissions' />
<param name=NFPParamScript value='
    DebugSet                = ALL;
    GraphType                = GROUP;
    Background                = (white,BOX,1,null,TILE,black);
    BarLabels                 = "January","February","March","April","May";
    BottomTics                = ("ON",black,"TimesRoman",12,0,null);
    DataSet1                  = 100,125,245.78,147,67;
    DataSets                  = ("BarSet1",dodgerblue,BAR,4,FILLED);
    DwellLabel                = ("",black,"Courier",12,0);
    Header                    = ("      Title      ",black,"TimesRoman",18,0);
    LeftTics                  = ("ON",null,"TimesRoman",12,0,null);
    BarWidth                  = 61;
    Bar3DDepth                = -1;
    DwellLabelBox              = (yellow,RAISED,3,null,TILE,black);
    BottomTicLayout            = (AUTO,0,1);
'>
</applet>
```

Figure 24 - An example of an applet that includes *DebugSet*

General Information

NetCharts, NetCharts Pro, NetCharts Server, NetCharts Designer, NetCharts Performance Dashboards, Chart Definition Language and Visual Mining are trademarks of Visual Mining, a division of Tervela, Inc.

Other product names used in this document are trademarks of their respective owners.

© 1996-2015 Visual Mining, a division of Tervela, Inc. All rights reserved.

Visual Mining, a division of Tervela, Inc.

2301 Research Blvd.
Suite 201
Rockville, MD 20850

Inquiries

General Phone	800.308.0731
International	+1.301.795.2200
Fax	301.947.8293
General Inquiries	info@visualmining.com
Customer Support	support@visualmining.com
Sales	sales@visualmining.com
Press and Media Inquiries	marketing@visualmining.com

Web

<http://www.visualmining.com>