macrofocus GmbH

Macrofocus TreeMap v. 2.0

Applet Deployment Guide

Version 04

Version	Date	Authors	Comment
01	February 28, 2011	L. Girardin	Initial Version
02	March 8, 2011	L. Girardin	Additional parameters and improved API
03	March 15, 2011	L. Girardin	Support for setting TableModel
04	August 17, 2011	L. Girardin	TreeMap v. 2.0

Content

1 Introduction

The server (applet) version is a good solution if you want to give access to TreeMap to a large audience. It simplifies the deployment because the code (Java applet) and the data are both delivered to the client computers from a centralized server. The main restriction is that a recent version of Java be installed on all client computers. All it takes is to put the JAR (Java ARchive) file containing the applet on your web server and to place the appropriate <applet> tag in some of your web pages, e.g.:

```
<applet code="com.macrofocus.application.treemap.v2.TreeMapApplet.class"
   archive="treemap-applet.jar"
   width="100%"
   height="100%">
   <param name="fileURL" value="http://www.company.com/FileOrScript.csv">
   <param name="format" value="CommaDelimited">
   <param name="size" value="Column 2">
   <param name="color" value="Column 3">
   <param name="label" value="Column 1">
   <param name="username" value="www.company.com">
   <param name="key" value="XXXX-XXXX-XXXX-XXXX-XXXX-XXXX">
   <param name="codebase_lookup" value="false">
</applet>
```

You can pass parameters to the applet to specify which URL the data should be retrieved from (can be either a script that will generate data on the fly, or a static file). Additional parameters can be used to specify all the default settings used for the visualization.

One constraint to keep in mind is that the data should come from the same server as the applet. This is inherent to the Java applet security scheme (when unsigned applet are used) and because of our licensing model.

1.1 Prerequisites

A web server capable of serving files using the HTTP protocol is necessary to transmit the code and the data from the server to the browser. Current versions of TreeMap require that your web browser support Java 5 or higher.

1.2 Supported browsers

The following browsers are currently supported:

Firefox v. 3.0 and higher

Internet Explorer v. 7.0 and higher

Safari v. 4.0 and higher

Google Chrome

2 Basic deployment

TreeMap can be deployed as an applet in 5 easy steps:

2.1 Place treemap.jar on your webserver

The latest version of the TreeMap applet can be downloaded from:

http://www.macrofocus.com/treemap/treemap-applet.jar

and should be placed so that it is accessible through your web server.

2.2 Create the HTML page serving the applet

It is usually recommended to deploy applets using the Java Deployment Toolkit (deployJava.js)[1]. To do so, create a file named TreeMap.html and place it at the same location as the treemap-applet.jar file:

```
<html>
<head>
 <style type="text/css">
  BODY { margin: 0em }
 </style>
 <meta http-equiv="Content-Type" content="text/html; charset=windows-1252"><title>TreeMap</title>
 <link rel="shortcut icon" href="favicon.ico">
</head>
<body>
<script type="text/javascript" src="deployJava.js">
</script>
<script type="text/javascript">
  var attributes = {codebase:'.',
              code:'com.macrofocus.application.treemap.v2.TreeMapApplet.class',
              archive:'treemap-applet.jar',
              width:'100%', height:'100%', hspace:'0', vspace:'0', align:'middle', name:'TreeMap'} ;
  var parameters = {progressbar:'true',
              boxmessage:'TreeMap is being loaded. This may take a while',
              username:'macrofocus.com',
              key:'WBUHJ-29ZUY-94S4R-KQ4T4-XGFZP-FEJ9S'
             } ;
  var version = '1.5.0' ;
  if (!deployJava.versionCheck(version+'+')) {
    if (confirm('To use TreeMap, you will need to install a newer version of Java. Install Java now?')) {
      deployJava.runApplet(attributes, parameters, version);
    } else {
      deployJava.writeAppletTag(attributes, parameters, version);
    }
  } else {
    deployJava.runApplet(attributes, parameters, version);
  }
</script>
<noscript>
Your browser does not support JavaScript!
</noscript>
</body>
</html>
```

2.3 Set the username and the license key

Replace the values of the username and key parameters with the license key information that has been supplied to you.

[1] Available from http://www.java.com/js/*deployJava.js*

2.4 Make your data available

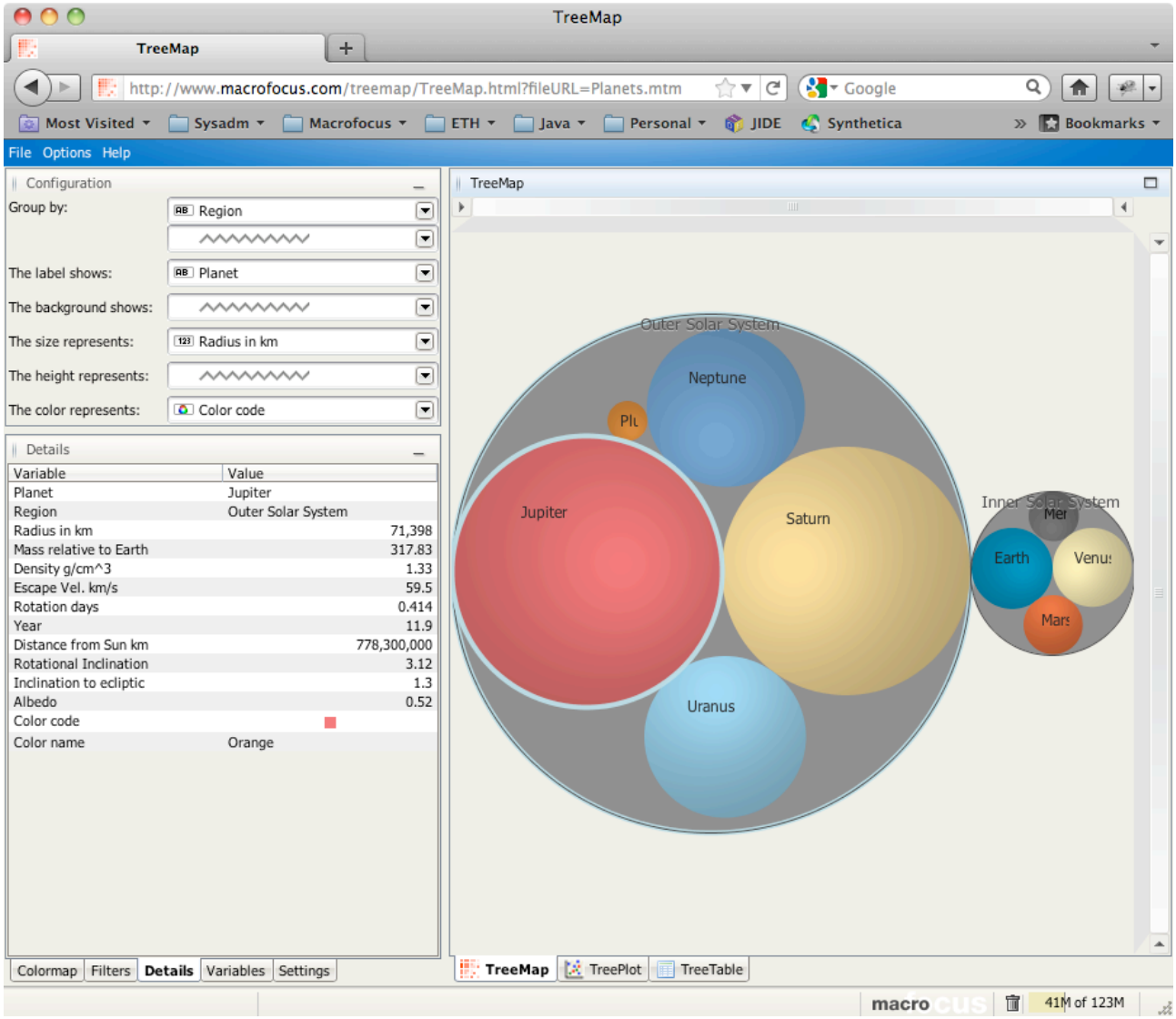
Place your data files, either in the native .mtm format or in one of the supported import file format so that they are accessible through your webserver.

2.5 Open your web browser

Point your web browser to the HTML document you have created in step 3 and supply it with the name of one of the dataset file as parameter:

http://www.company.com/TreeMap.html?fileURL=MyDataset.mtm

After Java has been loaded, the corresponding data set should open within TreeMap, e.g.:



3 Parameterizing

Here is the current list of parameters that can be specified (parameters in **bold** are mandatory):

- `fileURL`: URL of the data
- **`username`**: host name
- **`key`**: license key
- `format`: file format of the specified data source. Supported formats are `TabDelimited`, `CommaDelimited`, `Excel`, `TreeMap`, `HCIL`, `treeml`, `Zip`, `Jar`, `Shapefile`. Default is `TreeMap`.
- `language`: user interface default language. Supported languages `en`, `de`, `es`, `fr`, `it`, `ja`, `pt_BR`, `zh_CN`, `zh_TW`. Default is `en` (English).
- `expert`: true to enable expert functions
- menuBar: false to disable the display of the menu bar.
- statusBar: false to disable the display of the status bar.
- titleBar: false to disable the display of the title bar.
- panels: comma-separated list of panels to display.
- `listener`: name of a JavaScript function that should be called once the applet has been loaded.
- `groupBy`: list of fields to be use for grouping
- `size`: field used to represent the size
- `color`: field used to represent the color
- `height`: field used to represent the height
- `label`: list of fields to be use for labeling
- `background`: field used to display the background label
- `colormap`: color gradient to use the draw the treemap. Typical values for color gradients going from green to red are "posneg" and "inverseComposite", but you will possibly find that other colormaps (e.g. "bty") give more appealing visualization. The easiest way to find out the names of the various colormaps (as well as other parameter values) is to use the desktop version of Macrofocus TreeMap, select a color scheme, and save it using File->Export Settings. You can open the resulting file using any text editor.
- `algorithm`: algorithm to use to lay out the treemap
- `aggregation`: aggregation scheme to use to aggregate values
- `scale`: scaling scheme to use to project values
- `nesting`: nesting to use to lay out the treemap
- `ordering`: ordering to use to lay out the treemap
- `depth`: depth to use to lay out the treemap
- `labeling`: labeling scheme to use to draw the treemap
- `rendering`: rendering scheme to use to draw the treemap. Possible values are “`Cushion`”, "`Flat`", and "`Flat (No Border)`"

Parameters can either be specified using the <param name="" value=""> tag or by encoding them as part of the URL, for example:

http://www.company.com/TreeMap.html?fileURL=FileOrScript.csv

4 Calling Macrofocus TreeMap Applet from JavaScript

Macrofocus TreeMap Applet has an API that allows JavaScript code to interact with the Macrofocus TreeMap Applet. To do so, the applet should be given a name and register a JavaScript callback method that will be triggered once the applet is operational:

```
<applet name="TreeMapApplet"
   code="com.macrofocus.application.treemap.v2.TreeMapApplet.class" … MAYSCRIPT>
   <param name="listener" value="treeMapListener"/>
   …
</applet>
```

The specified callback function can then be used to perform some customizations:

```
   <script language="JavaScript">
      <!--
      function treeMapListener(state) {
         // Hide some panels
         document.TreeMapApplet.setPanelVisible("Variables", false);
         document.TreeMapApplet.setPanelVisible("Layout", false);

         // Tune the appearance
         document.TreeMapApplet.setHeaderFont("Arial-ITALIC-32");
         document.TreeMapApplet.setHeaderBackgroundColor("#80145a");

         // Remove some filters
         document.TreeMapApplet.setFilterVisible(0, false);
         document.TreeMapApplet.setFilterVisible(1, false);
         document.TreeMapApplet.setFilterVisible(2, false);
         document.TreeMapApplet.setFilterVisible(3, false);
         document.TreeMapApplet.setFilterVisible(14, false);
      }
      // -->
   </script>
```

4.1 Available methods

The following methods are currently exposed:

getDockingManager

```
public DockingManager getDockingManager()
```

Gets the docking manager in use.

Returns

the docking manager

---

setTableModel

```
public void setTableModel(TableModel tableModel)
```

Change the underlying TableModel.

Parameters

tableModel - the new TableModel

---

load

```
public javax.swing.table.TableModel load(String url, String format)
```

Load the data at the specified url in the given format

Parameters

- - **Returns**

---

addProbingListener

```
public void addProbingListener(String function)
```

Register a JavaScript method for receiving events about probing. The specified method will receive an Object as parameter that can then be used to find information about the node properties and values.

Parameters

function - the name of the JavaScript function

---

addSelectionListener

```
public void addSelectionListener(String function)
```

Register a JavaScript method for receiving events about selection. The specified method will receive an array of Object as parameter that can then be used to find information about the node properties and values.

Parameters

function - the name of the JavaScript function

---

addContextMenuItem

```
public void addContextMenuItem(String name, String function)
```

Adds an entry to the context menu and register the provided JavaScript method as a callback. The specified method will receive an array of Object as parameter that can then be used to find information about the node properties and values.

Parameters

name - the name of the context menu entry

function - the name of the JavaScript function

---

removeContextMenuItem

```
public void removeContextMenuItem(String name)
```

Removes an entry from the context menu. Standard menu entry names arezoomIn,zoomOut,drillDown, anddrillUp.

Parameters

- _____

setSelectOnPopupTrigger

```
public void setSelectOnPopupTrigger(boolean value)
```

Defines whether selection will occur prior to the display of the context menu.

Parameters

value - true if selection should occur, false otherwise

---

getValueAt

```
public java.lang.Object getValueAt(int rowIndex, int columnIndex)
```

Returns the value for the cell atcolumnIndexand rowIndex.

Parameters

rowIndex - the row whose value is to be queried

columnIndex - the column whose value is to be queried

Returns

the value Object at the specified cell

---

getValueAt

```
public java.lang.Object getValueAt(Object node, int columnIndex)
```

Returns the value for the node atcolumnIndex.

Parameters

node - the node whose value is to be queried

columnIndex - the column whose value is to be queried

Returns

the value Object at the specified cell

---

getParent

```
public java.lang.Object getParent(Object node)
```

Gets the parent node of the given node.

Parameters

node - the node

Returns

its parent

---

getRow

```
public int getRow(Object node)
```

Gets the row in the underlying TableModel, -1 if it doesn't relate to a leaf node.

Returns

the row in the TableModel.

---

setPanelVisible

```
public void setPanelVisible(String panelName, boolean visible)
```

Sets the visibility of a given panel.

Parameters

panelName - the name of the panel

visible - true to make it visible, false otherwise.

---

setShowTitleBar

```
public void setShowTitleBar(boolean showtitleBar)
```

Sets the visibility of the title bar.

Parameters

showtitleBar - true to make it visible, false otherwise.

---

maximizePanel

```
public void maximizePanel(String panelName)
```

Maximize a given panel.

Parameters

panelName - the name of the panel

---

setMenuBarVisible

```
public void setMenuBarVisible(boolean visible)
```

Sets the visibility of the title bar.

Parameters

visible - true to make it visible, false otherwise.

---

setStatusBarVisible

```
public void setStatusBarVisible(boolean visible)
```

Sets the visibility of the title bar.

Parameters

visible - true to make it visible, false otherwise.

---

setFilterVisible

```
public void setFilterVisible(int column, boolean visible)
```

Sets the visibility of a given filter.

Parameters

column - the column associated with the filter

visible - true to make it visible, false otherwise.

---

setCategoricalFilter

```
public void setCategoricalFilter(int column, String[] values)
```

Filter out data with the specified values.

Parameters

column - the column containing the data

values - the values to filter out

---

setNumericalFilter

```
public void setNumericalFilter(int column, double min, double max)
```

Filter out data with the specified values.

Parameters

column - the column containing the data

min - the lower value

max - the upper value

---

setHideFiltered

```
public void setHideFiltered(boolean hideFiltered)
```

Hide the nodes that are filtered out.

Parameters

hideFiltered - true to hide the filtered nodes, false otherwise

---

setHideSearched

```
public void setHideSearched(boolean hideSearched)
```

Hide the nodes that do not match the search criteria.

Parameters

hideSearched - true to hide the searched nodes, false otherwise

---

setSearch

```
public void setSearch(String value)
```

Search out data with the specified value.

Parameters

value - the value to search for

---

setGroupBy

```
public void setGroupBy(int[] columns)
```

Defines the variables to be use for grouping.

Parameters

columns - the indexes of the columns to be used for grouping

---

setGroupByByNames

```
public void setGroupByByNames(String[] columnNames)
```

Defines the variables to be use for grouping.

Parameters

columnNames - the names of the columns to be used for grouping

---

setLabels

```
public void setLabels(int[] columns)
```

Defines the variables to be use for labeling.

Parameters

columns - the indexes of the columns to be used for labeling

---

setLabelsByNames

```
public void setLabelsByNames(String[] columnNames)
```

Defines the variables to be use for labeling.

Parameters

columnNames - the names of the columns to be used for labeling

---

setBackground

```
public void setBackground(int column)
```

Defines the variable to use for background labeling.

Parameters

column - the index of the column to be used for background labeling

---

setBackgroundByName

```
public void setBackgroundByName(String columnName)
```

Defines the variable to use for background labeling.

Parameters

columnName - the name of the column to be used for background labeling

---

setSize

```
public void setSize(int column)
```

Defines the variable to use to represent the size.

Parameters

column - the index of the column to be used for representing the size

---

setSizeByName

```
public void setSizeByName(String columnName)
```

Defines the variable to use to represent the size.

Parameters

columnName - the name of the column to be used for representing the size

---

setColor

```
public void setColor(int column)
```

Defines the variable to use for coloring.

Parameters

column - the index of the column to be used for coloring

---

setColorByName

```
public void setColorByName(String columnName)
```

Defines the variable to use for coloring.

Parameters

columnName - the name of the column to be used for coloring

---

setColorMap

```
public void setColorMap(String colorMap)
```

Defines the colormap to use for coloring.

Parameters

colorMap - the name of the colormap

---

setColorRange

```
public void setColorRange(int column, double start, double end)
```

Defines the range to use to map the numerical values to colors.

Parameters

column - the index of the column

start - lowest value

end - highest value

---

setHeight

```
public void setHeight(int column)
```

Defines the variable to use for mapping the height.

Parameters

column - the index of the column to be used for mapping the height

---

setHeightByName

```
public void setHeightByName(String columnName)
```

Defines the variable to use for mapping the height.

Parameters

columnName - the name of the column to be used for mapping the height

---

setAlgorithm

```
public void setAlgorithm(String algorithm)
```

Defines the algorithm to use to lay out the treemap.

Parameters

algorithm - the algorithm to use

---

setAggregation

```
public void setAggregation(String aggregation)
```

Defines the aggregation scheme to use to aggregate values of the treemap.

Parameters

aggregation - the aggregation scheme to use

---

setScale

```
public void setScale(String scale)
```

Defines the scaling scheme to use to project values the treemap.

Parameters

scale - the scale scheme to use

---

setNesting

```
public void setNesting(String nesting)
```

Defines the nesting to use to lay out the treemap.

Parameters

nesting - the nesting scheme to use

---

setOrdering

```
public void setOrdering(String ordering)
```

Defines the ordering to use to lay out the treemap.

Parameters

ordering - the ordering to use

---

setDepth

```
public void setDepth(String depth)
```

Defines the depth to use to lay out the treemap.

Parameters

depth - the depth to use

---

setLabeling

```
public void setLabeling(String labeling)
```

Defines the labeling scheme to use to draw the treemap.

Parameters

labeling - the labeling schem to use

---

setRendering

```
public void setRendering(String rendering)
```

Defines the rendering scheme to use to draw the treemap.

Parameters

rendering - the rendering scheme to use

---

setLightSourceHeight

```
public void setLightSourceHeight(double value)
```

Sets the light source height used for drawing the cushions.

Parameters

value - the height

---

setLightSourceAmbient

```
public void setLightSourceAmbient(double value)
```

Sets the light source ambient intensity used for drawing the cushions.

Parameters

value - the ambient intensity

---

setLightSourceX

```
public void setLightSourceX(double value)
```

Sets the light source X position used for drawing the cushions.

Parameters

value - the X position

---

setLightSourceY

```
public void setLightSourceY(double value)
```

Sets the light source Y position used for drawing the cushions.

Parameters

value - the Y position

---

setLightSourceZ

```
public void setLightSourceZ(double value)
```

Sets the light source Z position used for drawing the cushions.

Parameters

value - the Z position

---

setLabelingFont

```
public void setLabelingFont(String font)
```

Sets the font used for labeling. Naming follows the fontname-style-pointsize convention of Font.decode().

Parameters

font - the font to be used

---

setLabelingForegroundColor

```
public void setLabelingForegroundColor(String color)
```

Sets the foreground color to use for drawing the labels.

Parameters

color - the color to be used

---

setLabelingEffectColor

```
public void setLabelingEffectColor(String color)
```

Sets the background color to use for drawing the labels.

Parameters

color - the color to be used

---

setHeaderFont

```
public void setHeaderFont(String font)
```

Sets the font used for labeling the headings. Naming follows the fontname-style-pointsize convention of Font.decode()

Parameters

font - the font to be used

---

setHeaderForegroundColor

```
public void setHeaderForegroundColor(String color)
```

Sets the foreground color to use for drawing the headers.

Parameters

color - the color to be used

---

setHeaderBackgroundColor

```
public void setHeaderBackgroundColor(String color)
```

Sets the background color to use for drawing the headers.

Parameters

color - the color to be used

---

setHeaderEffectColor

```
public void setHeaderEffectColor(String color)
```

Sets the effect color to use for drawing the headers.

Parameters

color - the color to be used

---

setFormatPattern

```
public void setFormatPattern(int column, String format)
```

Defines the format to use for displaying numerical values of a given variable.

Parameters

column - the index of the variable

format - the formatting pattern

See Also

java.text.DecimalFormat#DecimalFormat(String)

5 Receiving events from Macrofocus TreeMap Applet

It is also possible to register callback functions to receive interesting events from the Macrofocus TreeMap Applet. This can for example be used to add a context menu entry to each treemap node and have the specified listener be called every time a user select that entry:

```
<script language="JavaScript">
  <!--
  function treeMapListener(state) {
    // Add a menu entry to the context menu
    document.TreeMapApplet.setSelectOnPopupTrigger(true);
    document.TreeMapApplet.addContextMenuItem("Open URL", "treeMapContextMenuItemListener");
  }

  function treeMapContextMenuItemListener(selected) {
    if (selected.length > 0) {
      var url = document.TreeMapApplet.getValueAt(selected[0], 15);
      window.open(url, "sideWindow");
    }
  }
  // -->
</script>
```