Silverlight Navigator

Silverlight Navigator

1 Silverlight Navigator Intro

Introduction

The .netCHARTING Silverlight Navigator is a powerful and innovative new chart control which enables users to interact and explore data in real-time. Advancing significantly beyond the Ajax Zoomer and flash based solutions in use by the major financial websites, the .netCHARTING Navigator is a truly next generation real-time visualization solution. Combining the best of breed functionality from finance specific sites with major user interface enhancements, the .netCHARTING Silverlight Navigator supports a huge variety of chart types for use with your own data! Now you can leverage this powerful new real-time interface with your own companies data for sales, performance, traffic and more! The .netCHARTING Silverlight Navigator has undergone significant usability and comparison testing with offerings from the major financial web sites and has proven to be significantly more intuitive and powerful for casual and advanced users alike.

Overview (Section 1.1)
This section covers some of the basics about what the navigator is and what it can do.

- Key Benefits
- How It Works
- Supported Features

User Interface (Section 1.2)
Describes the features available to the end user.

- Navigation
  - The Navigation Bar
  - Chart Area
  - Preview Area
  - Scrollbar
- Other Features
  - Chart Area Legend
  - Mouse Tracking
  - Selection Tracking
  - Deep Zooming

Programming Interface (Section 1.3)
Details programming interface available to control the chart on the server side as well as client side.

- Server side asp.net API
- Client side JavaScript API
  - Javascript API reference.
- Performance Considerations

1.1   Overview

Sections

1. Key Benefits
2. How it works
3. Supported Features

Key Benefits

- Performance
    - Provides unmatched performance navigating large data sets.
    - Ultra-Responsive interface utilizes multi-core technologies.
    - Smooth transitions zooming in and out of data and scrolling to different points in the chart.
    - Ground-up client side design (the heavy lifting is done client side reducing server load and enhancing scalability).
    - Utilizes multi-core processor technology.

- Usability
    - Provides an innovative user interface.
    - Deep zooming from decades down to milliseconds.
    - Time, numeric, and category scale selection using quick x axis cell selection.
    - Has been put through rigorous usability testing across a range of demographics.
    - Real-time chart sizing.
    - Usable in a SlideShow presentation.

- Programmability
    - Seamlessly utilizes the .netCHARTING API to speed deployment and minimize development.
    - Provide a highly interactive Silverlight chart solution for your clients without any knowledge or editing of Silverlight required.
    - Existing clients can transform most static dotnetCHARTING charts into Silverlight Navigator versions with a single property setting.

How It Works

.netCHARTING seamlessly handles the complexity of Silverlight deployment, plug-in detection and down-level support so you don't have to. Developers are able to deploy one of the most advanced and innovative Silverlight solutions available utilizing a simple server side API which is already familiar to them. Developers with more advanced requirements are able to access the client side JavaScript API to provide further interactivity between the page and the embedded control. The .netCHARTING Silverlight Navigator is an extremely refined implementation which includes custom UI controls for range selection, automatic transfer of highly compressed data packages client to server and leverages the .netCHARTING data engine to further minimize development on the server side.

Supported Features

Many of the common features are supported; however, there are some limitations.

▶ **General Features**

Silverlight Navigator

Feature	Server Chart Support	Silverlight Chart Support
2D Chart	✓	✓
3D Charts	✓	☐
Data ShadingEffects	✓	▣

Effects 1-3 are available

Feature	Server Chart Support	Silverlight Chart Support
MicroCharts	✓	☐
Label Markup	✓	✓
Rich Tooltips	✓	✓
InfoGrids	✓	✓
Multiple and Custom LegendBoxes	✓	✓
Dynamic Chart Area Legend Entries	☐	✓
Multiple chart areas	✓	▣
Label Tokens and Expressions	✓	✓
Smart Labels	✓	✓
Annotations	✓	✓
Box Headers	✓	✓
Statistical Analysis	✓	✓
Finance Indicators	✓	✓

▶ **Chart Types**

Chart Types	Server Chart Support	Silverlight Chart Support
Combo	✓	✓
ComboHorizontal	✓	☐
ComboSideBySide	✓	☐
Pie	✓	☐
Donut	✓	☐
Radar	✓	☐
Scatter	✓	✓
Bubble	✓	✓
Financial	✓	✓
Gantt	✓	☐
Gauges	✓	☐
Map	✓	☐
Multiple	✓	☐
Organizational	✓	☐

▶ **Series Types**

Series Types	Server Chart Support	Silverlight Chart Support
Marker	✓	✓
Markers with custom Images	✓	✓
Line	✓	✓
AreaLine	✓	✓

⚠ Line and AreaLine series types support markers at each element position, however, in order to improve the performance of charts, when these series types contain more than 250 elements, the markers will be removed automatically in order to make the charts usable.

Series Types	Server Chart Support	Silverlight Chart Support
Spline	✓	☐
AreaSpline	✓	☐
Column	✓	✓

	Server Chart Support	Silverlight Chart Support
Cylinder	☑	☐ Same as Column.
Bar	☑	☑
Bubble	☑	☑
Pyramid	☑	☑
Cone	☑	☐ Same as Pyramid.
BubbleShape	☑	☐
BarSegmented	☑	☐
- Sub Values	☑	☑
- Empty Element Handling	☑	☑
- Element Hotspots (Interactivity)	☑	☑
Series Types Financial		
CandleStick	☑	☑
Bar	☑	☑
Open	☑	☑
Close	☑	☑
High	☑	☑
Low	☑	☑
HighLowArea	☑	☑

▶ **Axis Features**

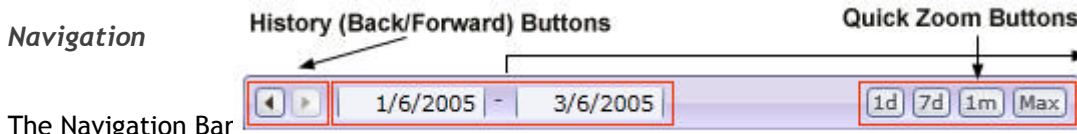| Axis Features	Server Chart Support	Silverlight Chart Support
Per Series Multiple Axes	☑	☐
Angled X Axis Labels	☑	☐
XAxis Intervals	☑	☐ X Axis intervals are always dynamic.
Scale Breaks	☑	☐ Scale breaks are available on the x axis.
Label Markup	☑	☑
Axis Orientation	☑	☐ The x axis can only be on bottom.
Range Axis Ticks	☑	☐ Available on the Y Axis
Tick Label Styling	☑	☑
Axis Markers	☑	☑
Axis Marker Labels on CA Scales	☑	☑
Normal	☑	☑
Range	☑	☑
Logarithmic	☑	☑
Time	☑	☑
Stacked	☑	☑
FullStacked	☑	☑
Logarithmic Stacked	☑	☑
Inverted (reversed)	☑	☑

1.2   Unser Interface (UI)

User Interface

The user interface is comprised of several widgets and features that facilitate navigating through different

ranges across the x axis in an intuitive manner. Different features provide similar functionality but each has its own advantages that are ideal for the different tasks one would need to accomplish. This section describes how these features are used by end users.

Navigation

History (Back/Forward) Buttons   Quick Zoom Buttons

◀ ▶   1/6/2005 - 3/6/2005   1d 7d 1m Max

The Navigation Bar

Range Selectors

Numeric Scale

∧
297
∨

Category Scale

∧
e1&40
∨

DateTime Scale

Month/Day/Year
iterator buttons

∧ ∧ ∧
7/11/2005
∨ ∨ ∨

Day/Month name
indicator

The Navigation bar contains a number of controls to facilitate range selection.
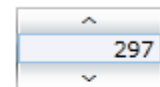
The history back and forward buttons can be used to navigate through previous views similar to a web browser. When history is available the buttons will be active, otherwise they are grayed out.

The quick zoom buttons can automatically zoom to a predefined range such as 1 day or 1 month with time scales or predefined numeric ranges.
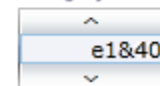
The range selectors seen on the right are an innovative new type of date, numeric, and category selector/iterator. When using it with a time scale, it enables quicker date selection than a traditional calendar while still providing day of week and month names. Values can be typed into the text box directly. The arrow buttons can be used to iterate numbers, element names, or the different parts of dates like day, month, and year individually. When using a date entry control (such as a calendar) it is often necessary to iterate the selection by one month or one year and the buttons provide quick and easy way of donig that. However, when large changes are required, the butttons also provide click and hold functionality that iterates many values quickly without using the keyboard to type them in manually. When using click-hold with the day and month buttons, the values will pause when they reach the first day of the month or first month of the year, respectively. This gives a little more control when using click-hold and also provides an easy way to reset the day or month selections back to 1.

Chart Area

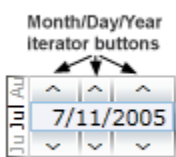The chart area can be scrolled by dragging it when the hand cursor appears and a new selection can be made when the crosshair cursor is showing. Using the scrollwheel to zoom in and out is available at any time. Dragging is not available when the chart is fully zoomed out. When both dragging and selection is available, the top part of the chart area can be used for dragging and the bottom for selection, the ratio of drag to selection area can be specified programatically.

The chart area axis also supports cell selection.

Preview Area

The preview area is used in conjunction with the scrollbar to visualize the chart area view in relation to the entire range of data. It supports cell and arbitrary selection, dragging handles *that mirror the scrollbar handles* and scrollwheel zooming. When both cell and arbitrary selection are enabled, they can be used at the same time. For example you can start by selecting a cell and end the selection at an arbitrary position and vice versa. Clicking near the top of the preview area starts the selection in cell mode while clicking near the bottom starts it at an arbitrary position. Moving the mouse pointer up or down allows switching between the modes during selection. Our studies have shown that using only cell selection is more intuitive for first-time users. Using mixed selection is more powerful however, and can be useful when the end user is is someone who uses the chart often.

Special Cursors
Different cursors are available in these sections and indicate the available functionality.

🖐 **Hand Cursor**
Indicates drag scrolling is available

✛ **Crosshair Cursor**
Indicates arbitrary selection is available

✛ **Cell Cursor**
Indicates cell selection is available

⊞ Note: When using mixed or arbitrary selection, clicking once without draging out a selection will center the scrollbar at the clicked position.

Arbitrary & Cell Selection on the preview area is demonstrated here:
Sample: **Feature Set > Navigator > NavigatorSelector**
(http://dotnetcharting.com/gallery/view.aspx?id=NavigatorSelector)

Scrollbar

The scrollbar looks very similar to a traditional scrollbar and it supports all the traditional scrollbar features, however, it also provides a number of new ways to manipulate it.

Traditional Scrollbar Features

- Drag scrolling
- Click outside the scrollbar to make it page to the mouse location.
- Click buttons at each end for smooth and controled movement.

New Features

- Drag handles of the scrollbar to zoom one side in and out.
- Double-click scrollbar handles to zoom one side out completely.
- Double-click the bar to zoom out completely. Double-Click again to zoom back to the previous position.
- Drag the bar up or down to put it into zoom mode. i.e. drag the horizontal scrollbar up to zoom in and down to zoom out.

Other Features

Chart Area Legend

Each chart area contains a legend that describes the visible data. It updates dynamically as the view changes. This legend also provides information on data at any mouse position when the mouse is hovering the chart area or preview area, and info on data while selecting a new zoom range. These features are mouse and selection tracking described below.

Mouse Tracking

The chart can track the mouse position to determine the values of data at those positions. The mouse is tracked across the chart area and preview area and the values are reported in the respective chart area legends. When hovering cells on axes, the legends will reflect calculations based on the ranges of those cells.

Selection Tracking

When selecting ranges on the axes or chart area, the part of the chart being selected shows in color while the rest is grayed out to indicate what the user is doind. Information about data within the range being selected shown in chart area legends.

Deep Zooming

When the scrollbar becomes very small, it will zoom the preview area in so it can be used more precisely and allow zooming in even farther. The zeep zoom will not always happen as soon as the scrollbar becomes as small as possible. There is a feature that prevents this while hovering the scrollbar area. The purpose of this is to prevent unintentional deep zooming in cases such as when the right handle is moved all the way to the left and you intend on moving the left handle all the way out.

Drag & Throw

When dragging the chart area with the hand cursor, the chart area can be thrown to improve the chart area scrolling usability. This is a common UI feature in many touch-screen applications. In order to achieve this, the mouse button must be let go while the mouse is in motion. The faster the mouse is moving when the button is released, the further the chart will scroll on its own.

1.3   Programming the navigator

Programming Interface (asp.net API)

The navigator is designed to function with any charts by changing single setting. (**Chart.Navigator.Enabled** = true) While all current chart types may not be supported, the goal is to support them all soon. There are two API layers in this navigator. One controls the UI and behavior and the other interacts with the chart in

Silverlight Navigator

real-time.

📰 A temporary directory setting is also required like with image charts where temporary data can be stored for the client to download.

Enabling the Navigator

The following code snippen is an example of the required code to enable the Navigator.

chart.Navigator.Enabled = true
chart.TempDirectory = "temp"
chart.LegendBox.Position = LegendBoxPosition.ChartArea

💡 Using the chart area legendbox position enabled richer functionality.

UI / Behavior API

The chart.Navigator class contains almost all the settings to configure the different navigator options. They provide a way to turn interactivity features on and off, as well as showing and removing parts of the chart. These settings are broken into sections below.

Navigation Bar

Disabling The Navigation Bar
Chart.Navigator.NavigationBar.Visible = false

Disabling History
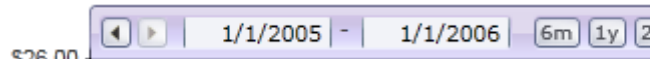Chart.Navigator.NavigationBar.EnableHistory = false

Disabling Range Selectors
Chart.Navigator.NavigationBar.EnableRangeSelectors = false

Styling the bar
Chart.Navigator.NavigationBar.Background.Color = Color.Red

Chart.Navigator.NavigationBar inherits from a Box class which offers all the styling properties such as background, padding, and corners.

🧩 Sample: **Feature Set > Navigator > NavigatorNavBarStyling (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorNavBarStyling)**

Quick Zoom Buttons (Default Setting)
Chart.Navigator.NavigationBar.QuickZoomIntervals = "Auto"

Disabling Quick Zoom Buttons
Chart.Navigator.NavigationBar.QuickZoomIntervals = ""

The Quick Zoom intervals are controled by a single string which can disable the buttons, make them determined automatically, specify what intervals to use, and optionally the label text of the buttons. The buttons support localization so users in different cultures will see the appropriate day/month abbreviations.

The string takes a comma delimited list of intervals and supports custom label specification with an equal sign. Setting it to "Auto" will let the chart determine what the best buttons are.

Syntax

"[Multiplier][TimeInterval]=[Label],..."

Examples

Setting	Result
""	No buttons
"1min,30min,5h,5d"	[1min][30min][5h][5d]
"Auto"	Automatically determined based on scale range
"1d,5d=FiveDays"	[1d][FiveDays]
"1d,1m,1y,YTD=Year To Date,Max"	[1d][1m][1y][Year To Date][Max]

Time Interv Specifier	Me
ms	mi
s	se
min	mi
h	ho
d	da
m	mc
y	ye
YTD	ye
Max	ma

 Intervals that are larger than the axis range or smaller than the zoomLimit will be omitted.

Sample: **Feature Set > Navigator > NavigatorNavBarStyling (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorNavBarStyling)**

Chart Area Legend

Enabling ChartArea Legend
Chart.LegendBox.Position = LegendBoxPosition.ChartArea

Controling the range display
Chart.LegendBox.Value = "%PercentageChange"

Sample: **Feature Set > Navigator > NavigatorFinance (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorFinance)**

Controling the display of a single element's value
DefaultElement.SmartLabel.Text = "%YValue"

Arranging the range display mode
Chart.LegendBox.Template = "%Value%Icon%Name"

Other Samples:

Feature Set > Navigator > NavigatorFinance (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorFinance)
Feature Set > Navigator > NavigatorCALegendStyling (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorCALegendStyling)
Feature Set > Navigator > NavigatorStockDB (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorStockDB)

Chart Area

Not showing the chart area
Chart.ChartArea.Visible = false

Sample: **Feature Set > Navigator > NavigatorSelector (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorSelector)**

Specifying UI Features
Chart.ChartArea.NavigatorOptions = ChartAreaNavigationOptions.DragScrolling | ChartAreaNavigationOptions.ArbitrarySelection | ChartAreaNavigationOptions.ScrollWheel;

Omiting any options will disable them.

Sample: **Feature Set > Navigator > NavigatorSelectorSeparate (http://dotnetcharting.com/gallery/view.aspx?id=NavigatorSelectorSeparate)**

Specify drag area vs. selection area
Chart.ChartArea.NavigatorDragAreaPercentage
Chart Area Axis
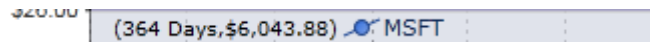
When the legend is not in the chart area it is not dyna the values for the entire dataset. When the legend is chart area however, it behaved dynamically by updati real-time. The display of each series in the chart are can be specified through the respective LegendBox.Te the mouse is on the chart area, it displays the values This value can be specified with DefaultElement.Smar all the element tokens can be used.

By setting Series.LegendEntry.Value = "" for any series not perform any calculations on the displaying info ab highlighted range.

Chart Area Legend Behavior Notes

- Legend Entries in the chart area cannot use lab
- If LegendEntry.Value is set to "%PercentageCha entry labelStyling color is not set, the color wil green for positive change and red for negative.
- When the legendEntry.LabelStyling color is not entry name will use the series color.

Silverlight Navigator

Make cells smaller then axis ticks available
Chart.Navigator.EnableCellVisualCues = true

- When the chart area is disabled, the chart area axis will also be collapsed.
- When the preview area supports cell selection, it will be available on the chart area axis.
- The chart area axis will inherit its background color from the chart area.

Preview Area

Specifying Preview Area Height
Chart.Navigator.PreviewAreaHeight = 35

Disabling/Hiding Preview Area
Chart.Navigator.PreviewAreaHeight = 0

Disable chart data drawing
Chart.Navigator.EnablePreviewAreaChart = false

Specifying UI Features
Chart.Navigator.PreviewAreaNavigationOptions =
    PreviewAreaNavigationOptions.Cells
    | PreviewAreaNavigationOptions.ArbitrarySelection
    | PreviewAreaNavigationOptions.ScrollWheel;

Omiting any options will disable them.
ScrollBar

Disabling the Scrollbar
Chart.Navigator.EnableXScrollbar = false

Disabling the scrollbar will remove the preview area as well.

Sample: **Feature Set > Navigator > NavigatorDetails (http://dotnetcharting.com/gallery/view.aspx? id=NavigatorDetails)**

General Options

Chart Sizing

A chart size can be specified in the traditional way. However, with the navigator, if the width or height is not specified, the chart will size to 100% of the available width or height.

Sample: **Feature Set > Navigator > NavigatorDynamicSizing (http://dotnetcharting.com/gallery/view.aspx? id=NavigatorDynamicSizing)**

Disable Browser Status Help Messages
Chart.Navigator.EnableBrowserStatusHelp = false

⚠ This may not work for some browsers by default. For example, to enable it in IE 8, choose Internet Options > Security tab > custom level > Under Scripting > Allow status bar updates via script, choose the Enable radio button. Using the SetHelpMessageHost javascript property of the client side chart, a text box or other element can be associated with this message.
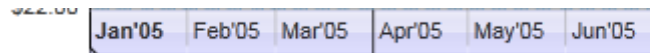
The preview area provides a visual representation of t and data the scrollbar represents. They work togethe scrollbar disables the preview as well. The preview ar and arbitrary selection, dragging handles *that mirror handles* and scrollwheel zooming..

Disabling the scrollbar will remove the preview area

Specify Animation Duration (in milliseconds)
Chart.Navigator.AnimationDuration = 200

Disable Mouse Tracking
Chart.Navigator.EnableMouseTracking = false

Limiting the zooming depth
Chart.Navigator.ZoomLimit = TimeInterval.Day
Chart.Navigator.ZoomLimit = TimeSpan.FromDays(5)

Set a starting zoom position
Chart.XAxis.Viewport.ScaleRange
    = new ScaleRange(new DateTime(2010,1,1),new D (2010,5,1))

Culcutre Settings
Server and client side charts may be running on mach different culture settings. By default the client side c the hosting machines culture. In some cases this may for example if using formatting such as currency for v clients default culture may incorrectly represent mon the other hand, using the clients culture for x axis tim useful. Culture settings may be specified for the entir specific axis:

Chart.DefaultCultureName = "en-US"

Silverlight plugin installation handling

The chart provides some options for handling users without silverlight installed. The options are defined by the **SilverlightInstall** enumeration.

Chart.Navigator.SilverlightInstall = SilverlightInstall.Automatic

Navigator Initialization

Loading Strings

When the navigator is initializing it displays strings such as "Please Wait...", "Downloading", "Loading". There is a property which allows overwriting these strings if they must be in a different language or just to change the wording.

Chart.Navigator.LoadingTextOverride = "Wait|Downloading(%Percentage)|Loading";

This property takes a single string which can have three different strings separated by a '|' character. The downloading string can contain the token "%Percentage" which will be replaced with the percent downloaded when the chart is loading.

⚠ WARNING: The %Percentage token is case sensitive.

LoadingAnimation

There is a number of loading animation visuals available for display while the Navigator is loading.

Chart.Navigator.LoadingAnimation = LoadingAnimation.Snake;

| Segmented Snake | Segmented Circle | Snake | Double Snake |

Chart.YAxis.CultureName = "en-US"

Windowless Mode

A property is available to allow the Silverlight Navigat operate in windowless mode. This feature is useful wh obscures other dynamic content on a page. For more i this feature, see **this page (http://msdn.microsoft.co us/library/cc838156(VS.95).aspx)**.

Chart.Navigator.EnableWindowlessMode = true;

Preview Image Genration

When the Navigator is loading, the chart is capable of preview of what the navigator will look like when it is provides a more seamless experience for end users.

Chart.Navigator.GenerateImagePreview = false;

Preview Image Performance

If the initial navigator view does not contain many ele image can be generated very quickly to show while al downloading. However, if the initial view contains a l elements (thousands for instance) the preview image moment to generate. In this scenario, it may be worth the preview image. This can improve the time necessa navigator to load, but the navigator must load fully be visible. For most cases, the preview image generates it's best to keep it enabled. When it becomes a proble affecting the website performance, disable it.

Client Side Programming Interface (Real-Time JavaScript API)

Several properties and methods allow the chart to be manipulated through JavaScript and allows the chart to call functions and update objects on a page. For example the standard navigation bar on top of Navigator charts can be created from HTML elements and communicate with the chart through Javascript directly.

Control The Chart

In order to use the JavaScript interface of the silverlight chart, a unique ID must be specified for the control like with the following asp.net code

Chart.Navigator.ControlID = "slPlugin"

Now the chart object can be accessed from JavaScript on the page. Consider the following code snippet:

```
<script type="text/javascript">
var chartID = "slPlugin";

function goback() {
    var control = document.getElementById(chartID);
    control.Content.Chart.PageLeft();
```

```
}
</script>
```

This code defines a function that accesses the chart object and calls the chart's PageLeft() method.

🧩 Sample: **Feature Set > Navigator > NavigatorPageInteraction
(http://dotnetcharting.com/gallery/view.aspx?id=NavigatorPageInteraction)**

Getting information from the Chart

The Javascript chart interface provides the ability to get information about the position status of the chart. The following JavaScript method gets the time range of the visible chart area and updates the visible range to show only the last 10 days from the current right position.

```
<script type="text/javascript">
var chartID = "slPlugin";

function leftten() {
    var control = document.getElementById(chartID);
    var range = control.Content.Chart.GetVisibleRange();
    var leftDate = range.ValueHigh;
    var tmpDate = range.ValueHigh;
    leftDate.setDate(leftDate.getDate() - 10);

    control.Content.Chart.NavigateToVisibleRange(leftDate, tmpDate);
}
</script>
```

🧩 Sample: **Feature Set > Navigator > NavigatorPageInteraction
(http://dotnetcharting.com/gallery/view.aspx?id=NavigatorPageInteraction)**

Chart Controlling the Page

The chart can call a Javascript functon when the viewport is updated by the user and it can update an HTML element with a help message text.

The following example will let the chart update the text of a div element with a help message while using the navigator. The first part of this task requires registering a method or property with the chart. Consider the following JavaScript code.

```
<script type="text/javascript">
var chartID = "slPlugin";

// This function registers the location update handler when the silverlight chart loads.
function initEvents() {
    var control = document.getElementById("slPlugin");
    if (control) {
        control.Content.Chart.SetHelpMessageHost("helpDiv.innerHTML");
        clearInterval(initID);
    }
}

// Try to register the location update handler with the chart until it is succeeds.
var initID = setInterval("initEvents()", 1000);
</script>
```

The initEvents() method tells the chart to update the innerHTML property of the div tag named helpDiv with a help message. The if, else statement ensures that the silverlight control is loaded before this property is assigned. If it is not loaded, it waits 1000 milliseconds before it tries again.

🧩 Samples:
   **Feature Set > Navigator > NavigatorPageInteraction (http://dotnetcharting.com/gallery/view.aspx?
   id=NavigatorPageInteraction)**
   **Feature Set > Navigator > NavigatorSelector (http://dotnetcharting.com/gallery/view.aspx?
   id=NavigatorSelector)**
   **Feature Set > Navigator > NavigatorSelectorControl (http://dotnetcharting.com/gallery/view.aspx?
   id=NavigatorSelectorControl)**
   Feature Set > Navigator > NavigatorSelectorSeparate (http://dotnetcharting.com/gallery/view.aspx?

id=NavigatorSelectorSeparate)

ChartArea Tokens

The navigator supports all the same label tokens as a static charts. However, it offers a couple additional tokens that are not supported by static charts. The reason this is not available with static charts is because these tokens are evaluated in real-time and based on the mouse position over the chart area. The tokens % XValue and %Value can be used in chart area hotspots. Consider this setting:

Chart.ChartArea.Hotspot.Tooltip = "%XValue, %YValue"

When the mouse is hovering the chart area, a tooltip will appear showing the axis values at the current mouse position. Then can be used in hotspot URL with enables passing the clicked axis values to javascript.

Sample: **Feature Set > Navigator > NavigatorGetSeriesValue**
(http://dotnetcharting.com/gallery/view.aspx?id=NavigatorGetSeriesValue)

Slideshow Presentation with Navigator

Including a Navigator chart in a slideshow presentation is easy to do. Refer to sample [Feature Set > Navigator > NavigatorSlideshow]

It contains information about the slideshow in javascript arrays including a text message, delay in milliseconds, and left and right positions to move the chart for each slide.

var myTextArr = [...];
var delayArr = [...];
var leftDTArr = [...];
var rightDTArr = [...];

A displaySlide function will update the page with the specified slide.

```
function displaySlide(frame) {
    var control = document.getElementById(chartID);
    control.Content.Chart.NavigateToVisibleRange(leftDTArr[frame], rightDTArr[frame], delayArr[frame])
    var tText = myTextArr[frame];
    setTimeout('displayText("' + tText + '");', delayArr[frame]);
}
```

The startSlideshow() function initiates the slide show and playSlide() function handles itarating the slides with a delay between them. playSlide() ensures only the available slides are shown and initiates the next slide by calling itself on a delay.

```
function startSlideshow() {
    playSlide(0)
}
function playSlide(frame) {
    displaySlide(frame);
    var nextFrame = frame + 1;
    if (nextFrame < myTextArr.length) {
        setTimeout('playSlide(' + nextFrame + ')', delayArr[frame] + slidesPause);
    }
}
```

Possible Uses

The interface is designed to allow users to use the chart to control content of the page or let the content of your page control the chart itself. The chart is able to animate between views of data such as zooming in or out which gives the user a better idea of how a subset of the the data relates to a bigger picture. A page that lets the user navigate articles related to specific dates for example can update the chart view to encompass data that is related to those articles.

A page that emulates a slideshow type movie can be connected to a chart so that when each slide is shown the chart animates to the appropriate data. Methods that control the navigation also provide the option to specify the duration any animation between the different ranges.

Sample: **FeatureSet > Navigator > NavigatorIncludeInAxisScale**

Silverlight Navigator

Javascript Interface Reference

Below is a list of available methods provided by this interface.

Method/Property	Description
Control The Chart	
PageLeft()	Pages the scrollbar left.
PageRight()	Pages the scrollbar right.
NavigatorForward()	Navigates the zoomer forward to the next location. Same as clicking the forward button on the navigation bar.
NavigateBack()	Navigates the zoomer back to the previous location. Same as clicking the back button on the navigation bar.
NavigateToCenter (object centerVal, int milliseconds)	Animates the zoomer to center at the specified x axis value maintaining the scrollbar width in the specified amount of time. • **centerVal** - X axis value to center on, this value can be numeric, DateTime or string. • **milliseconds** - Time of the animation in milliseconds
NavigateToCenter (object centerVal)	Animates the zoomer to center at the specified x axis value maintaining the scrollbar width. • **centerVal** - X axis value to center on, this value can be numeric, DateTime or string.
NavigateToVisibleRange (object leftVal, object rightVal, int milliseconds)	Animates the view to the specified x axis value range in the specified amount of time. • **leftVal** - Value of the new left position. • **rightVal** - Value of the new right position. • **milliseconds** - Time of the animation in milliseconds
NavigateToVisibleRange (object leftVal, object rightVal)	Animates the view to the specified x axis value range. • **leftVal** - Value of the new left position. • **rightVal** - Value of the new right position.
NavigateToVisibleRangePerc (double left, double right, int milliseconds)	Animates the view to the specified percentage range in the specified amount of time. The percentage range is a numeric value from 0 to 1 representing the maximum left and right positions. A range of 0 to 1 means the chart is fully zoomed out. • **left** - Left percent position from 0 to 1. Value must be less than the right position. • **right** - Right percent position from 0 to 1. Value must be more than the left position. • **milliseconds** - Time of the animation in milliseconds.
NavigateToVisibleRangePerc (double left, double right)	Navigates the view to the specified percentage range. The percentage range is a numeric value from 0 to 1 representing the maximum left and right positions. A range of 0 to 1 means the chart is fully zoomed out. • **left** - Left percent position from 0 to 1. Value must be less than the right position. • **right** - Right percent position from 0 to 1. Value must be more than the left position.
Get info from chart.	
ScaleRange GetVisibleRangePerc()	Gets the visible x axis percentage range. The percentage range is a numeric value from 0 to 1 representing the viewport left and right positions. A resulting range of 0 to 1 means the chart is fully zoomed out.
ScaleRange GetVisibleRange()	Gets the visible x axis range values.

double GetSeriesYValueAtX(string name,object xValue)	Gets the element y value of the series specified at a given x position. ● **name** - Name of the series to search for elements in. ● **xValue** - The x value to match with an element.
Have the chart control the page.	
SetLocationUpdateHandler(string)	Sets the name of a JavaScript method with two parameters (lowValue, highValue) on the current page that will be called when the chart viewport is updated and the new range will be passed to it.
SetHelpMessageHost(string)	Sets the property of an HTML or Javascript object on the page which will be updated by the chart with a help message when the user is interacting with the chart.

Notes:

JavaScript DateHandling

Unlike asp.net, when Javascript dates are created they assume the DateTime is given in universal GMT time. When passing dateTime values to the navigator, the navigator will adjust for the timezone difference so that dates passed from javascript to Navigator will be treated as the GMT time is actually the local time. This makes working with dates more convenient without having to adjust the time manually.

This conversion was introduced in version 6.1. If a chart passing dates in parameters to JS methods was developed with version 6.0, time zone conversions were necessary, but they can be removed if the same code will be running with version 6.1.

Range vs RangePerc

The Percentage range of the chart is a coordinate system where 0 = the maximum left and 1 = maximum right chart positions.

The range represents the actual values of the x axis and can be given in DateTime or numeric data types. Category scales also use a numeric scale range numbering each element name from 0-n.

Performance Considerations

This chart is designed to animate large sets of data smoothly utilizing silverlight which is a subset of WPF. While WPF has very powerful graphics capabilities, silverlight does not have all the same capabilities of WPF. There are a few things that can help improve the performance of your charts if they become to slow down.

Bar Charts

A bar series includes some extra graphical elements such as a borders and the bar face itself. A chart with thousands of such bars can be made much more efficient utilizing the finance bar series types. This will make the bars thin but when so many bars are available they will usually be very close together anyway. However, these bar widths can be controled with Series.Line.Width . Using the following code a series can be converted to utilize this performance enhancement.

C# Example

```
mySeries.Type = SeriesTypeFinancial.Bar;
// In order to improve the performance of this chart, the volume series will utilize the
finance bars instead of full columns.
foreach (Element el in mySeries.Elements)
{
    // This applies the y value to high and 0 to low so that the finance bars appear as
a column series would.
    el.High = el.YValue;
    el.Low = 0;
```

```
}
```

Visual Basic Example

```
mySeries.Type = SeriesTypeFinancial.Bar
' In order to improve the performance of this chart, the volume series will utilize the
finance bars instead of full columns.
For Each el As Element In mySeries.Elements
        ' This applies the y value to high and 0 to low so that the finance bars appear as
a column series would.
        el.High = el.YValue
        el.Low = 0
Next
```

Sample: **Feature Set > Navigator > NavigatorStockDB (http://dotnetcharting.com/gallery/view.aspx?
id=NavigatorStockDB)**

Line and AreaLine Marker Behavior

Lines with many points have element markers for each. When these elements exceed a count of 250 the
element markers will automatically be removed. However, in charts with fewer points, removing the
element markers can also improve performance.

Chart Size

The chart size has a great influence on animation performance. If the charts starts getting slow, making it
smaller is likely to speed it up a great deal.

Mouse Tracking

Disabling mouse tracking can improve performance of client side charts when the user hovers their mouse
over the chart areas.

Element Hotspots

Using element hotspots to make them clickable or with tooltips can add also add a performance penalty
when the user hovers their mouse over the chart areas.

Axis AlternateGridBackground & GridLines

Everything that needs to move on the chart area requires processing time. Disabling the
alternateGridBackground color can help as a start. In addition, disabling gridlines can help also.

Axis Scale

The chart is capable of scrolling easily, however, when the y axis range updates dynamically the chart area
must not only scroll but also scale everything on it. Setting a static axis range (YAxis.ScaleRange = ...) can
lock in the axis range making the chart very fast to scroll and zoom.

Caching

The server side processing can be cached so that each request of a page with charts on them doesnt require
generating them each time. To use this feature, you must set a static filename for the chart such as:
Chart.FileName = "ChartA1"; Then enable fragment caching for the hosting aspx page.

For more information see:

- **How To Perform Fragment Caching in ASP.NET by Using Visual C# .NET
(http://support.microsoft.com/kb/308378)**

- **Chart File Handling ('File Handling' in the on-line documentation)**