LC .NET version 12.0.1.1

Released June 10th, 2024.

In this release we added new features, improvements and fixed several bugs. We added ImageLayer in ViewXY, IntensityGridSeriesPolar or Heatmap in ViewPolar, added TransparencyRenderMode for all 3D series, improved DataCursor configuration options and made LightningChart control compatible with XAML serializer.

ImageLayer

ImageLayer is a very large image layer, which can be gradually filled with smaller images. The accumulation of smaller images is done in the background thread, thus keeping Chart UI thread responsive and fast (for example zooming could be done while images are loaded). While compounding **ImageLayer** from many smaller bitmaps requires a lot of memory, the chart itself remains very interactive & fast (during and after layer is loaded). On top of **ImageLayer** all ViewXY series can be rendered. It is possible to create multiple **ImageLayers** and each layer's visibility can be switched on/off independently. On top of main layers, several sublayers of images could be created, each of them reduces pixel count approximately by factor 4. This allows fast interaction in application with all zoom levels (works similarly as tiles of geographical maps).

This feature will be handy in Semiconductor industry, where optical measuring systems are used for surface analysis. The surface in question could be silicon wafers, micro-electromechanical systems (MEMS) or similar. However, any application which requires a display of a very large image and needs fast zooming, could use this feature. If there is information about each sub-image position and size (in XY axes units), users can create a collage of any size.

Polar Heatmap/ IntensityGrid

IntensityGridSeriesPolar is like intensity grid series in XY chart. IntensityGrid allows visualizing M x N array of nodes, colored by assigned value-range palette. The colors between the nodes are interpolated. **IntensityGridSeriesPolar** is evenly spaced in Polar space. The data is stored in Data property as two-dimensional array of doubles [Angle, Amplitude]. MinimumAmplitude and MaximumAmplitude properties define the amplitude range where the grid should be fitted, while BeginAngle and EndAngle properties set range for the angles. For streaming/real-time application users should use Series.UpdateData() method.

DataCursor improvements

Now **DataCursor** for all views (XY, 3D, Polar or Smith) has Configuration property tree. There user can find properties to control color, style and visibility of label for each axis independently. The default text color is *Series.Title.Color*, default label's fill color is tracked pixel color and default label's border color is white.

3D TransparencyRenderMode property/feature

In this release **TransparencyRenderMode** property was added to all remaining series (it was already added for 3D-surface-series and PointLineSeries3D in v10.5.1): BarSeries3D, MeshModels, Polygon3D, VolumeModel and Rectangle3D. When different types of translucent series/objects are rendered in 3D charts, *ShaderApproximation* option of **TransparencyRenderMode** should be used. The new feature helps to mitigate artifacts and improve translucent object handling in the chart. This new way to handle translucent surface/lines is available only when DirectX 11 renderer is used (i.e. RendererDeviceType is either HardwareOnlyD11 or SoftwareOnlyD11).

XAML serialization compatibility

LightningChart .NET now supports XAML serialization. This means that .NET class **XamlWriter** can be used to serialize LightningChart object into XAML markup, while .NET class **XamlReader** can be used to read XAML input and create a LightningChart object. By using *XamlWriter.Save()* and *XamlReader.Load()* methods user can save the state of the Chart (save value for the most of its properties) and import that state next time user opens the application.

Notes: 1) Series Data/Points properties are not serialized; 2) VolumeModel properties are not serialized; 3) starting from this version it is perfectly fine to create chart in *.xaml code with non-bindable edition of LightningChart, although if someone needs to bind chart properties, then MVVM edition of LightningChart should be used.

Interactive Examples App

New examples have been added to Interactives Examples App (Demo) to demonstrate the new features. In particular, *ExampleImageLayer*, *ExampleHeatMapPolar*, *ExampleXamlSerialization* and *ExampleTranslucentChart3D* have been added. The source code of the new examples can be extracted as non-bindable WPF or WinForms (for both .NET Framework 4.8 and .NET 6.0). While *ExampleXamlSerialization* shows how ‘state’ of a chart could be saved and restored, Demo allows to save XAML string of serialized Chart of any example (button ‘Extract Xml’).

Check out the rest of the changelog for detailed information.

This is a full listing of the changes in version 12.0.1 against 11.0.1

New features

• ImageLayer - a new chart layer is introduce for accumulating smaller bitmaps

ImageLayer is very large image layer, which can be gradually filled with smaller images. The accumulation of smaller images is done in the background thread, thus keeping Chart UI thread responsive and fast (for example zooming could be done while images are loaded). While compounding ImageLayer from many smaller bitmaps requires a lot of memory, the chart itself remains very interactive & fast (during and after layer is loaded). On top of ImageLayer all ViewXY series can be rendered. It is possible to create multiple ImageLayers and each layer's visibility can be switched on/off independently.

• TransparencyRenderMode property is added to all 3D series

Previously, in version 10.5.1, TransparencyRenderMode property was added 3D-surface-series and PointLineSeries3D. In the latest version all remaining series also got this feature: BarSeries3D, MeshModels, Polygon3D, VolumeModel and Rectangle3D. When different types of translucent series/objects are rendered in 3D charts, ShaderApproximation option of TransparencyRenderMode should be used. The new feature helps to mitigate artifacts and improve translucent object handling in the chart. This new way to handle translucent surface/lines is available only when DirectX 11 renderer is used (i.e. RendererDeviceType is either HardwareOnlyD11 or SoftwareOnlyD11).

• LightningChart .NET now supports XAML serialization - [BCB]

This means that .NET class XamlWriter can be used to serialize LightningChat object into XAML markup, while .NET class XamlReader can be used to read XAML input and create a LightningChat object. By using XamlWriter.Save() and XamlReader.Load() methods user can save the state of the Chart (save value for the most of properties) and import that state next time user opens the application.
As a result of these changes it is perfectly fine to create chart in *.xaml code with non-bindable edition of LightningChart, although if someone needs to bind chart properties, then MVVM edition of LightningChart should be used.
Notes: 1) Series Data/Points properties are not serialized; 2) VolumeModel properties are not serialized;

• Added IntensityGridSeriesPolar series type to ViewPolar - [BCB]

IntensityGridSeriesPolar is similar to intensity grid series in XY chart. IntensityGrid allows visualizing M x N array of nodes, colored by assigned value-range palette. The colors between the nodes are interpolated. IntensityGridSeriesPolar is an evenly spaced in Polar space. The data is stored in Data property as two-dimensional array of doubles [[Angle, Amplitude]]. MinimumAmplitude and MaximumAmplitude properties define amplitude range grid should be fitted, while BeginAngle and EndAngle set range for the angles.

Improvements

- **Made ClipAreas utilization more robust**

  Previously ClipAreas maybe rendered incorrectly or even throw out-of-memory exception, if End smaller than Begin value or ranges are not in order. Now it is fixed by making ClipAreas utilization more robust.

- **Improved look of ZoomBar control**

  In version 11.0.1 change of ColorTheme could make ZoomBar's Annotation Box opaque. Now it is fixed.

- **ViolinPlot custom-control added to WinForms edition of LightningChart**

  Build-in custom-controls ViolinPlot was added to LightningChart non-bindable WPF edition in version 10.4.1. In v11.0.1 ViolinPlot was partially added to WinForms edition of library, only .NET6 build. Now ViolinPlot is added to all WinForms edition (NET4 and NET6 build).

- **Now IntensityGridSeries and graph-border fitting YAxis height correctly**

  Previously, auto-fitted IntensityGridSeries was stretched 1 px too short (at the top and at the bottom). In addition, height of ViewXY Border was 1 px too long (at the top and at the bottom). Now IntensityGridSeries border and ViewXY border are the same length as YAxis. They are also aligned with corresponding YAxis in segmented/stacked YAxesLayout.

  IntensityGrid series should now fill whole graph area when rendering. Previously there were one pixel gaps on top and bottom near border.

- **LegendBox texts and CheckBoxes positioning and sizing improved.**

  Previously (in ViewXY), when SeriesTitleFont was much bigger than CheckBox, those two are not aligned by center. Now it is centered as with other size changes in Legendbox. In addition, gap between LegendBox titles in Polar and 3D views is reduced (to be similar to XY).

- **DataCursor got more properties to customize its colors - [BCB]**

  Now DataCursor for all views (XY, 3D, Polar or Smith) has Configuration property tree. There user can find properties to control color, style and visibility of label for each axis independently. The default text color is Series.Title.Color, default label's fill color is tracked pixel color and default label's border color is white.

- **Fixed possible errors when running several Chart export command in sequence**

  Previously running several Chart export commands in a sequence could cause errors (e.g. rendering error, undisposed objects or raised exception). Now it is fixed.

- **Added Indexed8 bitmap format support**

  Now LightningChart can handle images with Indexed8 format directly (without conversion). This will save memory and increase speed.

Error fixes

- **SeriesEventMarkers rendered with GDI are now obeying ClipInsideGraph.**

  Previously SeriesEventMarkers denoted as clipped were rendered outside graph area in vector format output (emf/svg/wmf). Now it is fixed.

- **Fixed DataCursor 3D serialization error**

  Previously, then WinForms control was dragged from Visual Studio Toolbox and View3D.DataCursor.Visible = True, the application would crash on start (Null Reference Exception). Now it is fixed.

- **DataCursor.SnapToNearestDataPoint now works with SampleDataSeries**

  Previously DataCursor would not track SampleDataSeries if SnapToNearestDataPoint is enabled. Now it is fixed.

- **IntensityGridSeries now exported to vector format without errors**

  File export to EMF/SVG format was missing IntensityGridSeries, if chart-image size was changed. SaveToFile() or SaveToStream() method gave the same (incorrect) result. This particular error appeared in LightningChart version 10.4.1 and 10.5.1. Now it is fixed.

- **Fixed PolygonSeries on logarithmic axis graphical glitch**

  Previously PolygonSeries looked strange when having logarithmic axis (in particular when Polygon's edges make sharp angles). For example, border and fill were a bit off. Now it is fixed.

- **Fixed occasional wrong symbol/shape rotation**

  Previously, some of rotation angles could be incorrectly estimated. For example, for symbol/shape Angle 180 deg, the result could be 0 deg instead (in vector format image output). Now it is fixed.

- **Bindable FirstSampleTimeStamp property properly update for SampleDataBlockSeries**

  Previously, once points deleted by viewXY.DropOldSeriesData or directly by DeleteSamplesBeforeX() method, the SampleDataBlockSeries FirstSampleTimeStamp property wass not properly updated in bindable edition of library. As a result reading FirstSampleTimeStamp value would give wrong result, although internally rendering goes fine. Now it is fixed.

- **Fixed possible crash with SampleDataBlockSeries**

  Previously, if SampleDataBlockSeries collection had a mix of visible and hidden series, then chart would crash when DeleteSamplesBeforeX() method was called (indirectly also when ViewXY.DropOldSeriesData properties is enabled). Now it is fixed.

- **Bindable marker.Label.Font properties now updated correctly**

  Previously, setting bindable (MVVM edition) marker.Label.Font properties (like Size, Family etc.) had no effect for rendering result. Now it is fixed.

- **Bindable BlockSeries' PointCount/BitCount property now updated correctly**

  Previously, calling Clear() method for bindable (MVVM edition) BlockSeries' (like SampleDataBlockSeries or DigitalLineSeries) failed to reset properties PointCount/BitCount. Therefore, reading this property in bindable edition of Chart gave incorrect result. Now it is fixed.

Backwards compatibility breaks

API changes

• DataCursor got more properties to customize its colors

Some of DataCursor properties at ViewPolar & ViewSmith has been renamed:
* renamed "Configure" -> "Configuration";
* word 'Dynamic' replaced with 'UseSeries'. E.g. AngleLabelDynamicColor -> AngleLabelUseSeriesColor;
* to clarify renamed '...BorderColor' properties to '...LabelBorderColor'. E.g. AngleBorderColor -> AngleLabelBorderColor;

From DataCursor base class (all Views) properties removed (replaced with more specific properties in DataCursor.Configuration):
* LabelUseSeriesColor and LabelFillColor.

• LightningChart .NET now supports XAML serialization

Changes may effect how user can interact with all List properties in WinForm or non-bindable WPF edition of LightningChart. There are around 60 such properties in library (e.g. XAxes, SampleDataBlockSeries, SeriesEventMarkers, CustomTicks etc.). In most cases (e.g. adding to or removing from the list) user don't need to change anything in the application's code. However, if user wants to address such property as (a whole) collection, then instead of creating type List<TypeName>, one should create TypeNameList class object.
For example,
instead of List<CustomAxisTick> ticks = new List<CustomAxisTick>();
should be CustomAxisTickList ticks = new CustomAxisTickList();

In addition, namespace change for ThemeBasics:
LightningChart.ThemeBasics -> ThemeBasics

• Added IntensityGridSeriesPolar series type to ViewPolar

Due to the changes in class inheritance in some Polar series, AreaSeriesPolar and PointLineSeries polar do not include AllowDragging property anymore. Also, it had no implementation before, so it had no use anyway.