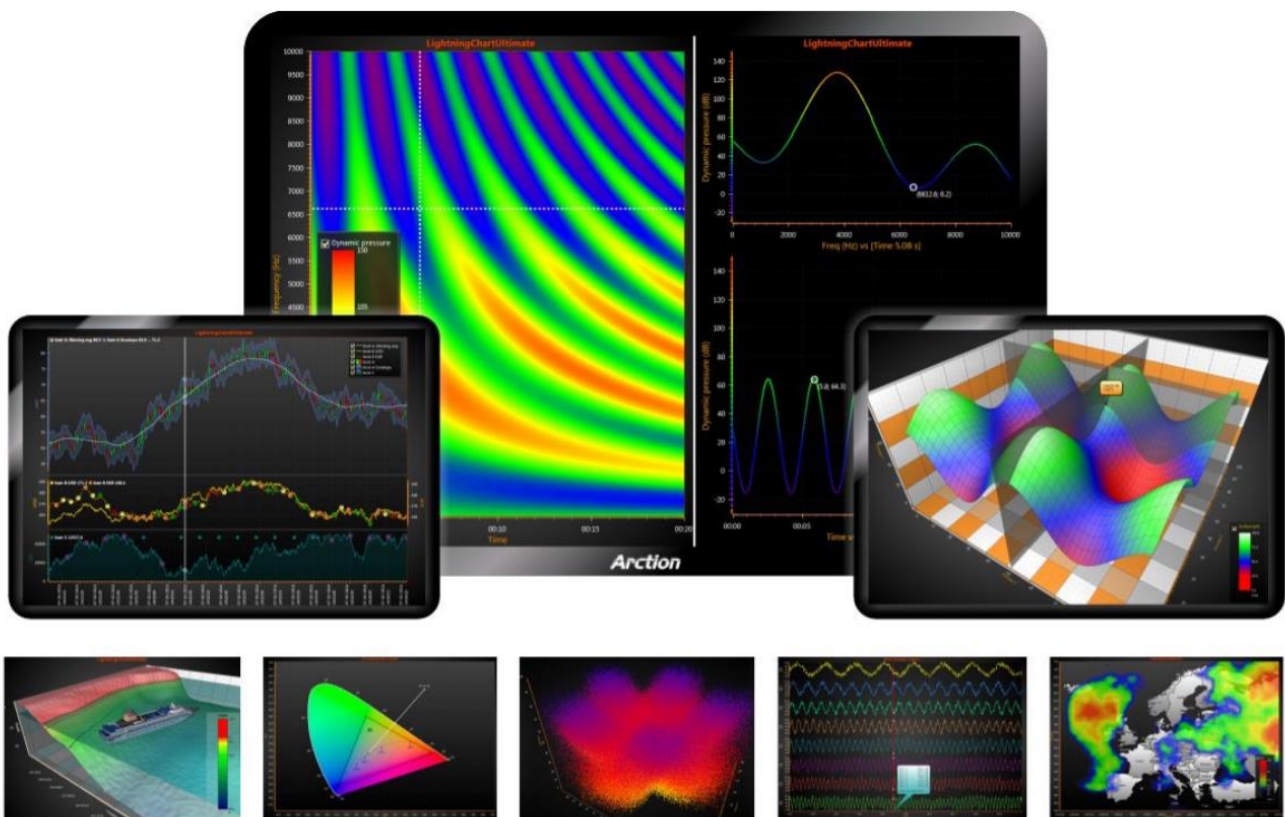




# LightningChart®.NET v.10

## 日本語スタートガイド



**XL**SOFT

エクセルソフト株式会社

# 目次

はじめに .....	3
<b>1. インストール方法</b> .....	4
1.1 インストール前に .....	4
1.2 setup wizard の実行 .....	4
1.3 Arction 製品のコンポーネントの手動インストール .....	4
<b>2. Dev Center</b> .....	5
2.1 Interactive Examples の実行 .....	6
2.2 シード プロジェクトの作成 .....	7
<b>3. ライセンスの管理</b> .....	8
3.1 ライセンスの追加 .....	8
3.2 ライセンスの削除 .....	10
3.3 ライセンスの更新 .....	10
3.4 デプロイメント キーの抽出 .....	10
3.5 アプリケーションにおけるデプロイメント キーの適用 .....	11
3.6 開発環境におけるデプロイメント キーの実行 .....	13
3.7 デバッガーの実行 .....	13
3.8 評価版の期間 .....	13
3.9 フローティング ライセンス .....	14
<b>4. ビューおよびチャート系列の種類</b> .....	14
4.1 XY チャート .....	14
4.2 3D チャート .....	16
4.3 ポーラー チャート .....	18
4.4 スミス チャート .....	18

# はじめに

この度は、Arction 社製品をご利用いただき誠にありがとうございます。

Arction 社は、フィンランドのクオピオに本拠を持ち、高解像度なグラフ描画によるデータ視覚化を実現する WPF および Windows Forms 向けの高速チャート コンポーネントである LightningChart® .NET を世界中の .NET 開発者へ提供しています。

本ドキュメントは、LightningChart®.NET をはじめて利用される方を対象に、製品のインストール方法、デモアプリケーションの実行方法、および購入ライセンスを使用した評価版から製品版への切り替え方法を説明しています。

すべての Arction 製品名は、Arction 社の商標または登録商標です。  
その他記載の商標、サービス マーク、商品名は各社の商標または登録商標です。

※このガイドは、Arction 社からの許可を得てエクセルソフト株式会社が LightningChart® .NET User's Manual の一部を日本語訳し、提供しています。エクセルソフト株式会社では、製品に関するご質問を日本語で承っております。

# 1. インストール方法

## 1.1 インストール前に

最初に、使用するマシンの構成とシステム要件が一致していることを確認します。

- DirectX 9.0c (shader model 3) レベル以降のグラフィック アダプターまたはグラフィック ハードウェアなしでレンダリング可能な DirectX11 と互換性のある OS  
※ DirectX11 と互換性のあるグラフィック ハードウェアが推奨されます。
- Windows Vista、7、8、10 (Windows 10 IoT Enterprise にも対応) および Windows Server 2008 R2、2012 R2、2016 またはそれ以降  
※ 上記の 32 ビット版および 64 ビット版がサポートされます。
- Visual Studio 2010、2012、2013、2015、2017、2019 (開発環境に限る)
- .NET Framework 4.0 またはそれ以降、.NET Core 3.0 (WPF、Windows Forms ライブラリ対応)  
※ .NET Core 3.0 ライブラリでのソース コードのコンパイルはサポートしていません。

## 1.2 setup wizard の実行

LightningChart .NET SDK v10.exe を右クリックすると、Visual Studio のツールボックス内にコンポーネントがインストールされます。また、ツールボックスの使用の際に役立つヘルプ ファイルも同時にインストールされます。コンポーネントやファイルが正常にインストールされない場合は、下記の項目に沿って、手動でインストールします。

LightningChart .NET の評価版を使用する場合、SetupDownloader.exe が使用されます。これにより、SDK がダウンロードおよびインストールされ、LightningChart.NET SDK v10.exe を明示的に実行する必要はありません。

## 1.3 Arction 製品のコンポーネントの手動インストール

### WinForms の場合：

1. Visual Studio を開き、新規の **WinForms** プロジェクトを作成します。ツールボックスを右クリックして [タブの追加] を選択し、[Arction] の名前を付けます。
2. [Arction] タブを右クリックして、[アイテムの選択...] を選択します。
3. [ツールボックス アイテムの選択] ウィンドウにて、**.NET Framework コンポーネント** ページを選択し、[参照] をクリックします。
4. コンポーネントをインストールしたフォルダーから、下記を開きます。  
**Arction.WinForms.Charting.LightningChartUltimate.dll**  
**Arction.WinForms.SignalProcessing.SignalTools.dll**  
 ※ 通常 **C:\Program Files (x86)\Arction\LightningChart .NET SDK v.10\LibNet4.**
5. ツールボックス内にコンポーネントが表示されます。

## WPF の場合：

1. Visual Studio を開き、新規の **WPF** プロジェクトを作成します。ツールボックスを右クリックして [タブの追加] を選択し、[Arction] の名前を付けます。
2. [Arction] タブを右クリックして、[アイテムの選択...] を選択します。
3. [ツールボックス アイテムの選択] ウィンドウにて、**.NET Framework コンポーネント** ページを選択し、[参照] をクリックします。
4. コンポーネントをインストールしたフォルダーから、下記を開きます。  
**Arction.Wpf.Charting.LightningChartUltimate.dl**  
**Arction.Wpf.SemibindableCharting.LightningChartUltimate.dl**  
**Arction.Wpf.SignalProcessing.SignalTools.dll**  
 ※ 通常 **C:\Program Files (x86)\Arction\LightningChart .NET SDK v.10\LibNet**。
5. ツールボックス内にコンポーネントが表示されます。

## 2. Dev Center

LightningChart®.NET v.8.5 以降、**LightningChart.NET SDK v10.exe** を実行する際、DevCenter が自動でインストールされるようになりました。Dev Center は、LightningChart®.NET の機能やリソースへの迅速なアクセスを実現する新しいアプリケーションです。たった数クリックで、以下のタスクを実行できます。

- **Interactive Examples** アプリケーションの起動
- シード プロジェクトの作成
- チュートリアルやユーザー マニュアル (英語) の参照
- Arction 社のサポート チームに対する英語での問い合わせ
- アプリケーション情報の自動収集によるテクニカル サポートの支援
- Arction 社に対するフィードバック提供のためのクイック リンク
- ライセンス状況の確認、ライセンス マネージャーの起動およびライセンスの更新
- Arction Portal を通した新しいライセンスの購入



図 2-1. さまざまなアクションに対するボタンを含む Dev Center のメイン ウィンドウ。

## 2.1 Interactive Examples の実行

Interactive Examples は、[スタート メニュー] > [Arction] 内のショートカットに表示される LightningChart®.NET のデモ アプリケーションです。また、Dev Center から実行可能です。

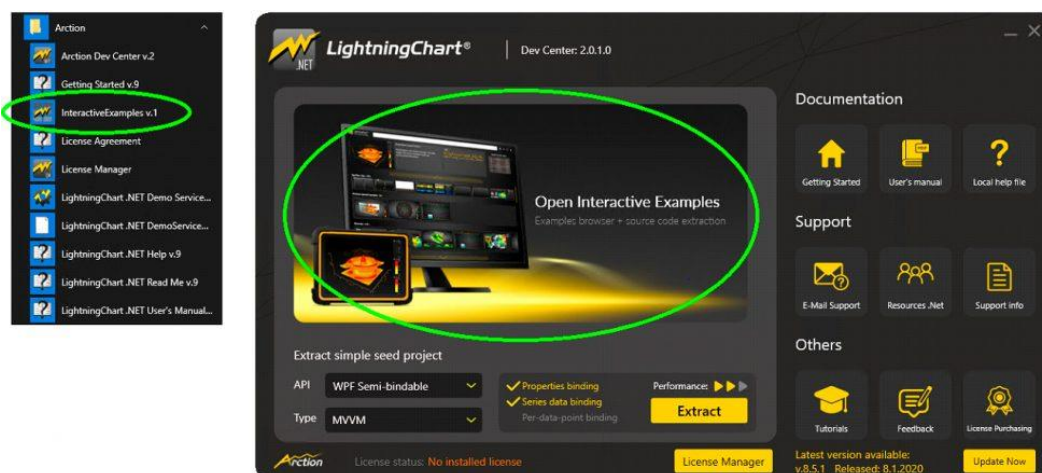


図 2-2. スタート メニューまたは Dev Center からアクセス可能な Interactive Examples の実行。

Interactive Examples には、さまざまなカテゴリ毎に多くのデモ サンプルが含まれています。カテゴリの閲覧またはアプリケーション上部の検索バーの使用により、サンプルを確認できます。

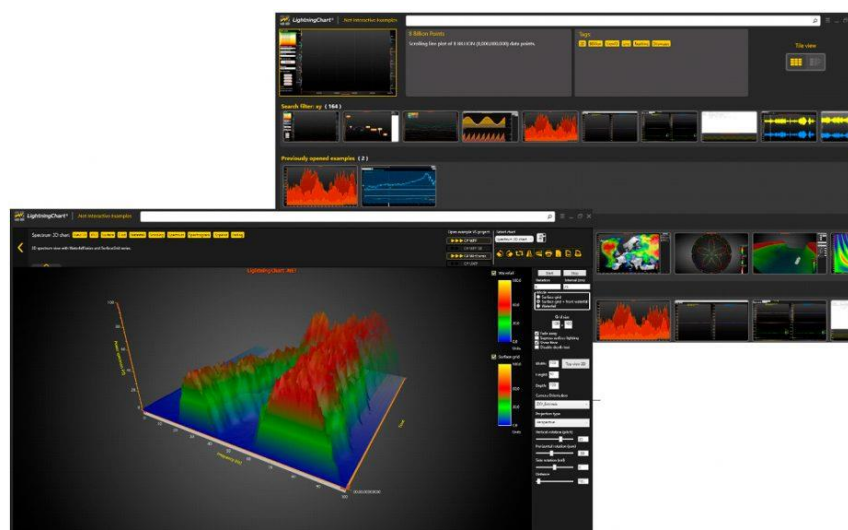


図 2-3. Interactive Examples のインターフェイス。

多くのサンプルでは、自由に変更可能ないくつかのチャート設定がユーザー インターフェイスに追加されています。さらに、[Properties] タブを使用して、ほとんどすべてのチャートのプロパティを個別に表示および編集できます。また、チャート画像ストリームを生成するサービスにクライアント アプリケーションを接続するための Service デモも提供されています。



すべてのデモ サンプルは、スタンドアロンの Visual Studio プロジェクトとして抽出でき、サンプル コードの変更や、ユーザー独自のアプリケーションの基盤としての使用が可能です。スタンドアロン プロジェクトは、Windows Forms、Non-Bindable WPF、MVVM モデルを使用する Semi-Bindable WPF や、UWP などの幅広いプラットフォームをターゲットできます。

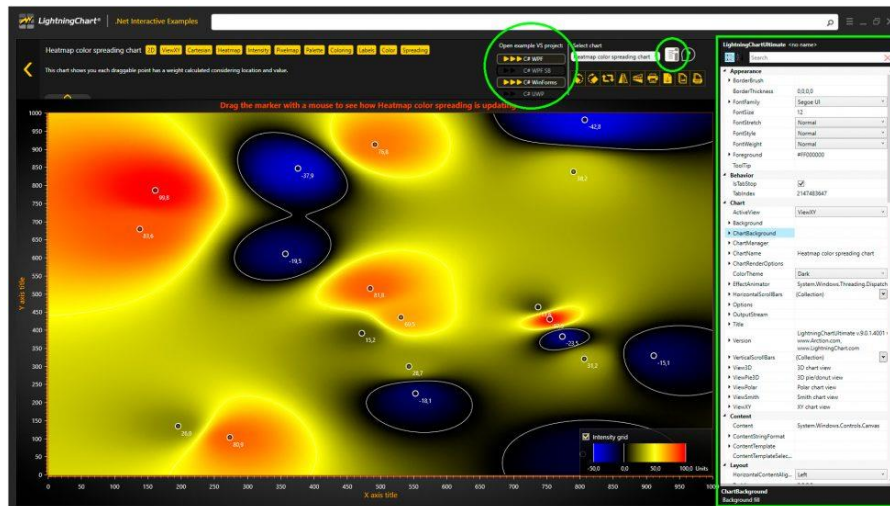


図 2-4. Interactive Examples における Properties タブ。

画面上部の [Open example VS project] からスタンドアロン プロジェクトとして抽出可能な個々のサンプル。

## 2.2 シード プロジェクトの作成

Dev Center を通して、いくつかのデータ ポイントとプロパティ設定とともに 1 つの系列が含まれる簡素的な .NET のシード プロジェクトを作成できます。これらのプロジェクトは、新しい LightningChart ベースのアプリケーションを作成する際の基盤として使用できます。

API バージョンおよびアプリケーションの種類を選択し、シード プロジェクトを作成します。種類の選択は、MVVM ベースのアプリケーションを作成する可能性がある Semi-Bindable のシード プロジェクトの作成時に主に関係します。[Extract] ボタンをクリックし、プロジェクトを作成するフォルダーを選択します。指定のフォルダーに既に同様のシード プロジェクトが保存されている場合、新しいプロジェクトは作成されず、既存のプロジェクトが上書きされます。

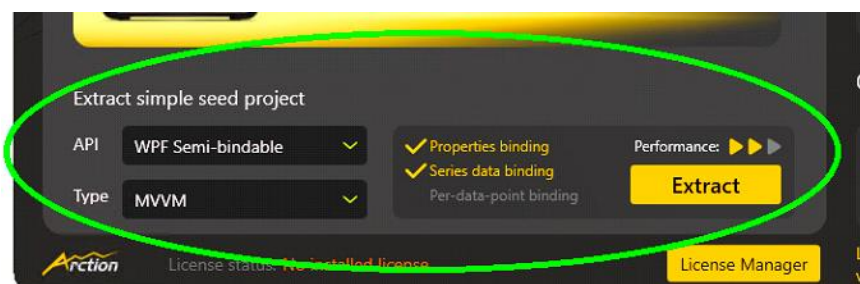


図 2-5. シード プロジェクトの作成画面。

## 3. ライセンスの管理

### 3.1 ライセンスの追加

ライセンスは、Dev Center からアクセス可能な **License Manager** アプリケーションで管理されます。または、Windows のスタート メニューから、[Programs] > [Arction] > [LightningChart.NET SDK] > [License Manager] を選択してアクセスすることもできます。

Arction 製品のコンポーネントは、ライセンス キーの保護システムを使用します。コンポーネントは、有効なライセンスのみにより使用できます。ライセンスは、以下の情報を保持します。

- 有効な機能：ViewXY(XY チャート)、View3D(3D チャート)、ViewPie3D(3D 円グラフ)、Maps(地図機能)、ViewPolar(ポーラー チャート)、ViewSmith(スミス チャート)、Volume Rendering(ボリューム レンダリング機能)、および Signal Tools(シグナル ツール機能) 等
- プラットフォーム：WPF、WinForms、UWP、またはバンドル
- ライセンスのアクティベーションが可能なマシンの台数 (標準台数は 1 台)
- サブスクリプションの有効期限 (アップデートおよびサポートの終了日)
- 技術サポートのレベル
- ライセンスの種類：開発者ライセンス、フローティング ライセンス、または学生ライセンス

Arction 製品のコンポーネントをツールボックスからアプリケーションに初めてドラッグする際、License Manager ウィンドウにてライセンス キーを要求されます。[Install license from file...] ボタンをクリックして、.alf ファイルを参照します。

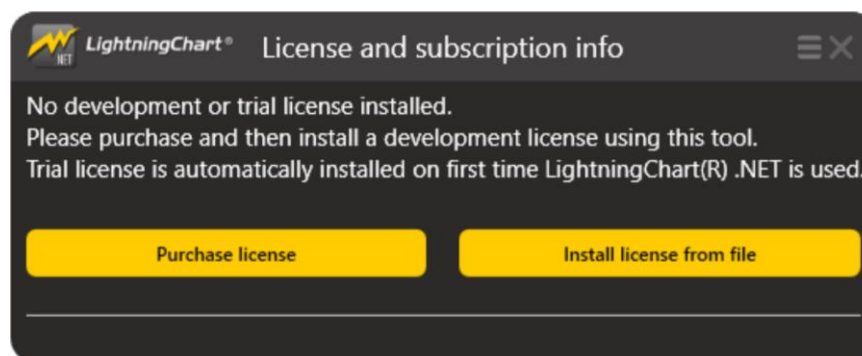


図 3-1. ライセンスがインストールされていない場合の License Manager。

[Install license from file] からライセンス ファイルを追加可能。

開発者ライセンスは、ライセンスを追加した後、インターネット接続を介して Arction 社のライセンス サーバーにて自動でアクティベーションが行われます。



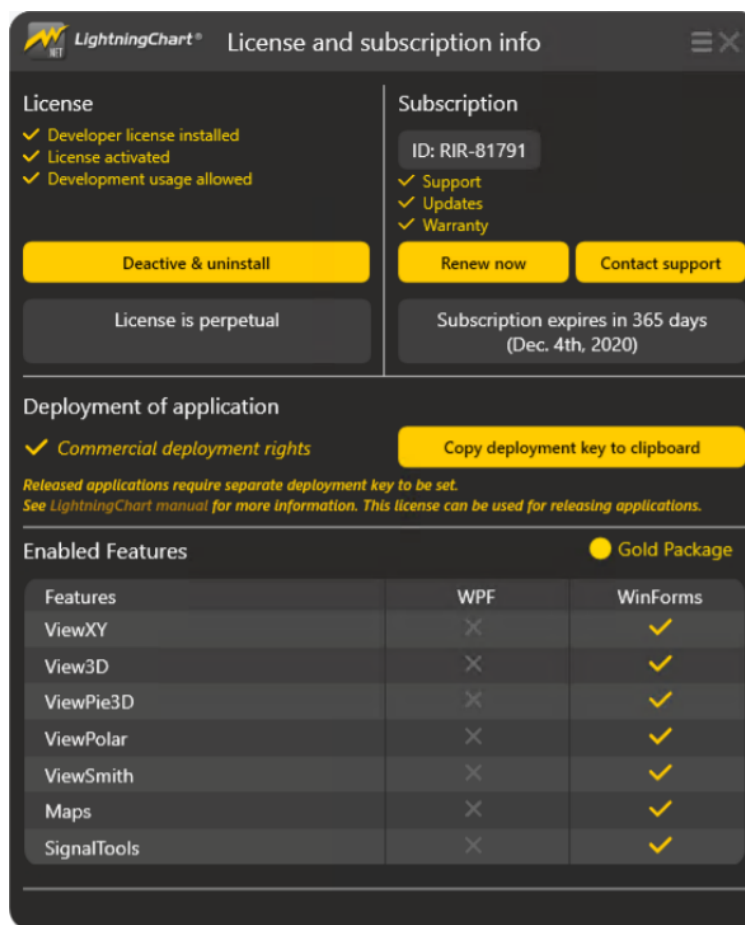


図 3-2. ライセンス ファイルが正常に追加された後の License Manager ウィンドウ。

インターネット接続が利用できない場合、Email を介してライセンスのアクティベーションを行うことができます。オンラインでのアクティベーションが数回失敗した場合、[Request offline activation] ボタンが利用可能になります。ライセンスのディアクティベーションも同様に対応可能です。

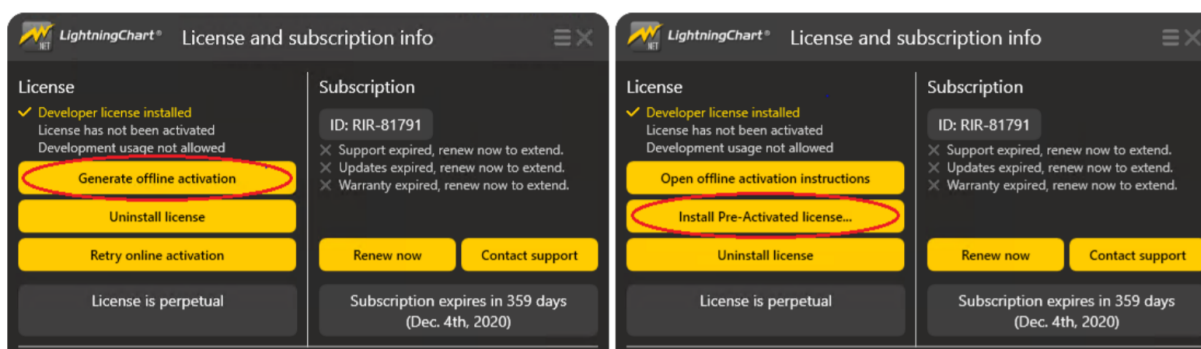


図 3-3. オンライン アクティベーション失敗後、オフライン アクティベーション オプションが利用可能。

画面の手順に沿って、Arction 社のライセンス チーム ([licensing@arction.com](mailto:licensing@arction.com)) へ Email を送信します。Email 送信後、2 営業日以内に Arction 社からライセンスをオフラインでインストールする方法が提供されます。

**注意：**ライセンス キーは文字数が多いため、電話からのアクティベーションおよびディアクティベーションは行っていませんのでご了承ください。

**注意：**LightningChart v.7.1 以降より、ChartManager コンポーネントの使用にライセンス キーは必要ありません。

**注意：**LightningChart v.8.0 以降では、LIC 形式のライセンス ファイルはサポートしていません。ALF 形式のライセンス ファイルが必要です。ALF 形式のライセンス ファイルをお持ちでない場合、弊社へお問い合わせください。

## 3.2 ライセンスの削除

ライセンスは、**[Deactivate & uninstall]** ボタンから削除できます。自動でディアクティベーションを行う場合、インターネット接続が必要です。インターネット接続が利用できない場合、「[3.1 ライセンスの追加](#)」の手順と同様に、Email を介してディアクティベーションを行うことができます。

ライセンスのディアクティベーション後、他のマシンでライセンスを使用することができます。

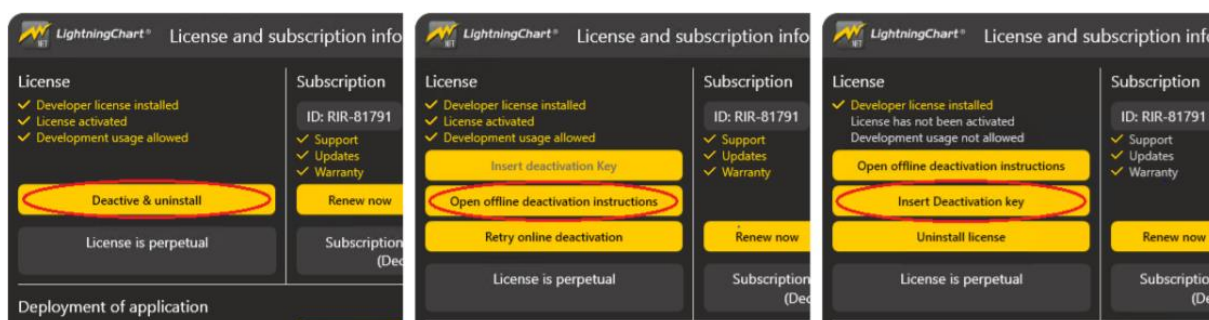


図 3-4. オンライン ディアクティベーション (左) 失敗後、オフライン オプション (右) が利用可能。

## 3.3 ライセンスの更新

サブスクリプションの有効期間を延長する場合や、より良いエディションにアップグレード、またはソース コードを購入した場合など、ライセンスのインストール後もライセンスを更新することができます。

**注意：**ライセンスは自動的に更新されません。開発を行うマシンのライセンスが更新されていることを確認してください。まず、古いライセンスのディアクティベーションを行い、削除してください。ライセンスの削除の手順につきましては、「[3.2 ライセンスの削除](#)」をご参照ください。

その後、ArctionPortal を通して .alf ファイルのライセンス キーを入手し、「[3.1 ライセンスの追加](#)」の手順に沿って、ライセンスを追加してください。

## 3.4 デプロイメント キーの抽出

ソフトウェアがデプロイされたマシンで LightningChart アプリケーションを実行する際、コード内にデプ

ロイメント キーを適用する必要があります。[Copy deployment key to Clipboard] ボタンを押下して、ライセンス キーからデプロイメント キーを抽出できます。

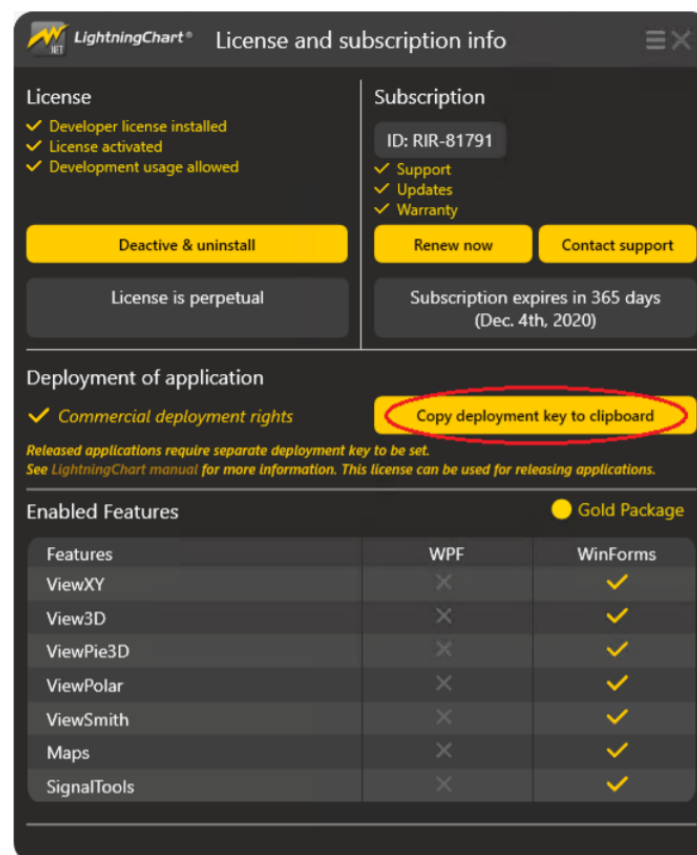


図 3-5. License Manager におけるクリップボードへのデプロイメント キーのコピー。

### 3.5 アプリケーションにおけるデプロイメント キーの適用

コンポーネントに対して静的な **SetDeploymentKey** メソッドをコード内で使用します。使用されていないコンポーネントに対してキーを設定する必要はありません。コンポーネントを使用する前に、**SetDeploymentKey** メソッドを呼び出します。チャートを使用するクラスの静的コンストラクター、またはアプリケーションのメイン クラス内で呼び出すことを推奨します。

#### Windows Forms の場合：

下記は、すべての WinForms アプリケーションで標準設定の一部として作成される Program クラスの静的コンストラクターに SetDeploymentKey を適用する例です。

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Windows.Forms;

namespace WindowsFormsApplication1
{
```

```

static class Program
{
    static Program()
    {
        // Arction 製品のコンポーネントにデプロイメント キーを設定します
        string deploymentKey = "VMalgCAA06kO1RgiNIBJABVcG.R..Kikfd...";

        Arction.WinForms.Charting.LightningChartUltimate.SetDeploymentKey(deploymentKey);
        Arction.WinForms.SignalProcessing.SignalGenerator.SetDeploymentKey(deploymentKey);
        Arction.WinForms.SignalProcessing.AudioInput.SetDeploymentKey(deploymentKey);
        Arction.WinForms.SignalProcessing.AudioOutput.SetDeploymentKey(deploymentKey);
        Arction.WinForms.SignalProcessing.SpectrumCalculator.SetDeploymentKey(deploymentKey);
        Arction.WinForms.SignalProcessing.SignalReader.SetDeploymentKey(deploymentKey);
    }

    // 残りのクラス...
}
}

```

### WPF の場合：

下記は、App クラスの静的コンストラクターにある App.xaml.cs の初めに SetDeploymentKey を適用する例です。

```

using System;
using System.Collections.Generic;
using System.Configuration;
using System.Data;
using System.Linq;
using System.Windows;
using Arction.Wpf.SignalProcessing;

namespace WpfApplication1
{
    /// <summary>
    /// App.xaml の相互作用ロジック
    /// </summary>
    public partial class App : Application
    {
        Static App()
        {
            // Arction 製品のコンポーネントにデプロイメント キーを設定します
            string deploymentKey = "- LICENSE MANAGER からデプロイメント キーを入力";

            // Bindable チャートを使用する場合、以下を設定します
            Arction.Wpf.BindableCharting.LightningChartUltimate
                .SetDeploymentKey(deploymentKey);

            // Semi-bindable チャートを使用する場合、以下を設定します
            Arction.Wpf.SemibindableCharting.LightningChartUltimate
                .SetDeploymentKey(deploymentKey);
        }
    }
}

```

```
// Non-bindable のチャートを使用する場合、以下を設定します
Arction.Wpf.Charting.LightningChartUltimate
.SetDeploymentKey(deploymentKey);

// Arction 製品のその他のコンポーネントにデプロイメント キーを設定します
SignalGenerator.SetDeploymentKey(deploymentKey);
AudioInput.SetDeploymentKey(deploymentKey);
AudioOutput.SetDeploymentKey(deploymentKey);
SpectrumCalculator.SetDeploymentKey(deploymentKey);
SignalReader.SetDeploymentKey(deploymentKey);
    }
}
}
```

**注意：**アプリケーション内でデプロイメント キーを設定しない場合、ターゲット マシンは 30 日間無償の評価モードに入ります。

## 3.6 開発環境におけるデプロイメント キーの実行

開発ライセンスがインストールされたマシン上で、**SetDeploymentKey** でデプロイメント キーが設定されたアプリケーションを実行する場合、**ライブラリは開発用のライセンス キーに優先順位を割り当てます**。ローカル環境でインストールしたライセンス (例: Silver パッケージ) より優先度の高い上位機能 (例: Gold パッケージ) がデプロイメント キーに含まれている場合、注意が必要です。

上記により、開発チーム内ですべて同じ種類のライセンスを使用することを推奨します。

## 3.7 デバッガーの実行

デプロイメント キーが正しく設定され、**Visual Studio** でデバッガーとともにプロジェクトを実行しているが、開発用のライセンス キーがシステム上に存在しない場合、チャートは最高 **~1 fps** のスロー レンダリング モードで実行され、チャート上にメッセージを表示します。

開発用のライセンス キーなしで **LightningChart** を使用した直接的な開発やデバッグを行うことは禁止されています。

## 3.8 評価版の期間

評価版は、30 日間使用可能です。期間終了後、製品の使用を延長する場合、ライセンスを購入する必要があります。評価版ライセンスでビルドしたすべてのプロジェクトは、製品版ライセンスへアップデートした後も動作します。評価版ライセンスでビルドしたチャート アプリケーションを実行する場合、評価版の **nag** メッセージが表示されます。

## 3.9 フローティング ライセンス

フローティング ライセンスは、マシンの台数に制限なくインストールできます。同時に使用可能な開発者数は、Arction 社によって構成されます。ライセンスを購入した同時ユーザー数に限り、一度に LightningChart.NET を使用できます。LightningChart.NET を終了後、次の開発者が使用開始できるまで 10 分から 15 分のタイムアウトがあります。

デプロイメント キーは、開発者ライセンスと同様の手順で設定する必要があります。

デフォルトでは、フローティング ライセンスは Arction 社のライセンス サーバーによって制御されます。開発の際には、継続的なインターネット接続が必要です。

カスタマー サイドのフローティング ライセンスもご利用可能いただけます。開発を行うマシンは、ローカル エリア ネットワークを介して、お客様の企業で用意したサーバーへ接続されます。Arction 社またはサード パーティとのオンライン通信は必要ありません。ライセンスとともに、コントローラー サービスおよびカスタマー サイドのフローティング ライセンスのインストール手順が別途提供されます。

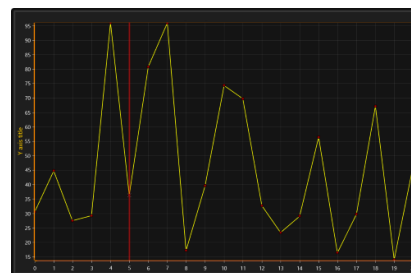
## 4. ビューおよびチャート系列の種類

### 4.1 XY チャート

XY チャート (ViewXY) は、あらゆるフォーマットによる 2 次元のデータ視覚化を実現します。すべての系列は、軸の値の範囲にバインドされます。

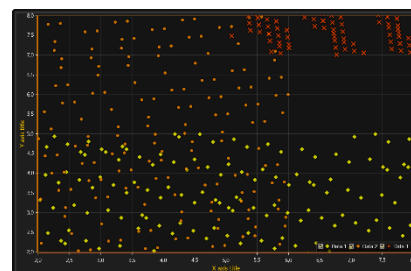
#### PointLineSeries:

PointLineSeries は、可変インターバル データを高速にレンダリングできる系列です。点および線、線のみ、または点のみ (散布図) を表示できます。また、X 値は累進的な順序で割り当てられる必要があります。



#### FreeformPointLineSeries:

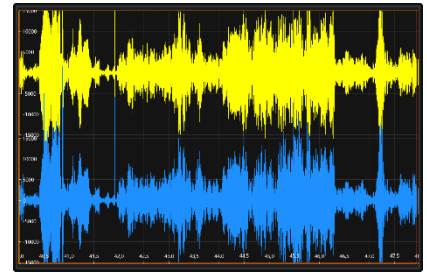
あらゆるデータの視覚化に役立つ FreeformPointLineSeries は、PointLineSeries に似ていますが、1 つ前の点から、任意の方向に線や点をレンダリングできます。ポイント数が非常に多い場合、レンダリングが重くなる可能性があります。



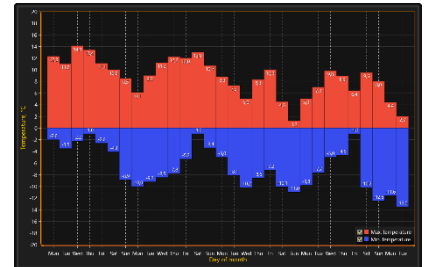


**SampleDataSeries:**

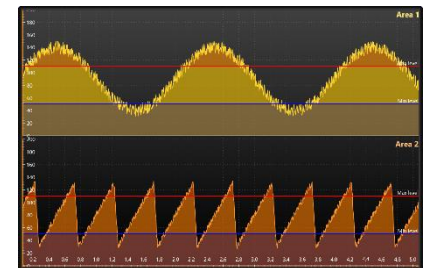
**SampleDataSeries** は、固定インターバルにおいて進行データを表示する系列であり、リアルタイムの監視アプリケーションに頻繁に使用されます。高速なレンダリングにより、点の値を格納するためのメモリを確保する必要はありません。

**BarSeries:**

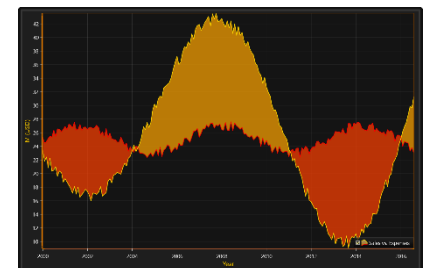
**BarSeries** は、水平方向または垂直方向での棒グラフによるデータの表示を実現します。

**AreaSeries:**

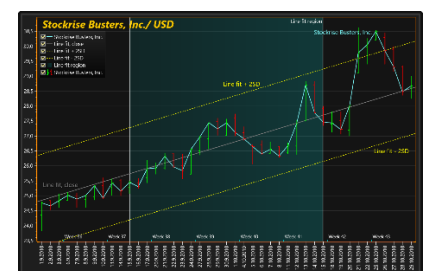
**AreaSeries** は、ベースとなるレベルと値の間を塗りつぶした面グラフによるデータの表示を実現します。超過の上限と下限を設定できます。この系列では、これらの上限と下限によって異なる色でデータを表示できます。

**High-Low Series:**

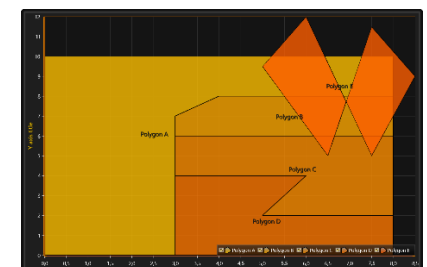
**High-Low Series** は、大きな値と小さな値の間を塗りつぶしてデータを表示します。**AreaSeries** と同様に、超過の上限と下限を設定できます。

**StockSeries:**

**StockSeries** は、ローソク足チャートおよび棒グラフによって証券取引所データを視覚化します。

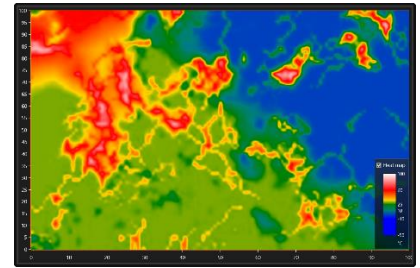
**PolygonSeries:**

**PolygonSeries** は、境界線のパスに沿って、領域の塗りつぶしや境界線をレンダリングします。

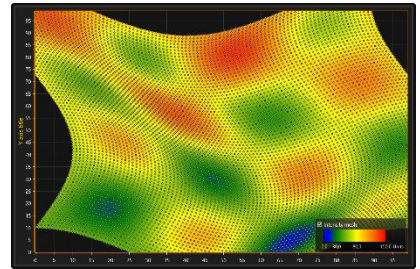


**IntensityGridSeries (ヒートマップ):**

**IntensityGridSeries** は、指定の値の範囲のパレットで色付けされた  $M \times N$  配列のノードの視覚化を実現します。これは、 $X$  ディメンションと  $Y$  ディメンションで等間隔に配置される長方形の系列です。等高線、ラベル、ワイヤーフレームをレンダリングできます。**PixelRendering** プロパティは、ピクセル単位または長方形とするノードのレンダリングを実現します。

**IntensityMeshSeries:**

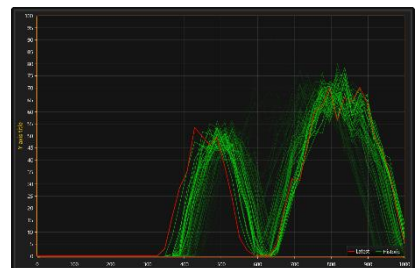
**IntensityMeshSeries** は、**IntensityGridSeries** と似ていますが、ノードが長方形である必要がない点が異なります。系列ノードは、 $X$  および  $Y$  空間の任意の位置に配置できます。

**Maps:**

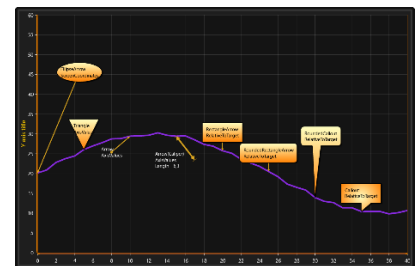
**Maps** プロパティおよびサブプロパティでは、地理地図を表示できます。ベクター形式の地図と、タイルマップが備えられています。これらの地図は正距円筒図法で表示され、**LightningChart.NET** の系列の種類と、実質的にすべてが  $X$  軸と  $Y$  軸にバインドされているその他のオブジェクトを地図と同時に使用できます。

**PersistentSeriesRenderingLayer:**

**PersistentSeriesRenderingLayer** は、幾度にわたり、同じ  $X$  と  $Y$  の範囲で繰り返しプロットされるデータの高速なレンダリングに使用されます。ビットマップのように、レンダリング データをインクリメンタルに追加できます。コマンドでクリアされる、または、データが古くなり表示がされなくなるまでグラフィックを保持します。

**Annotations:**

**Annotations** は、グラフ領域の任意の位置にて、マウス操作に対応するテキスト ラベルやグラフィックの表示を実現します。カスタマイズされたグラフィックを画面上でレンダリングする必要がある場合、**Annotations** を使用することで、さまざまなスタイルや形にレンダリングできます。



## 4.2 3D チャート

**3D チャート (View3D)** は、3D 空間におけるデータの視覚化を実現します。3D モデルは、さまざまな方法

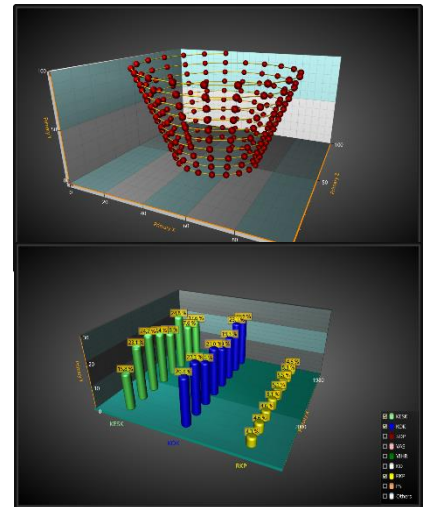
でズーム、回転、およびハイライトできます。1 つの 3D ビューに異なる系列の種類を配置することで、視覚化の組み合わせを実現することもできます。

### PointLineSeries3D:

PointLineSeries3D は、3D 空間において点や線の表示を実現します。XY チャートの PointLineSeries と同様に、点を線で繋げることができます。

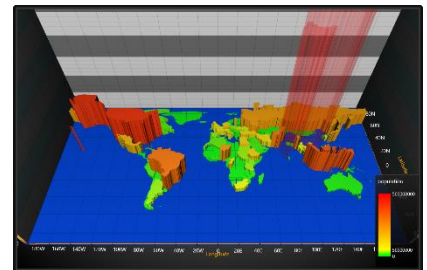
### BarSeries3D:

BarSeries3D は、3D 空間での棒グラフの表示を実現します。



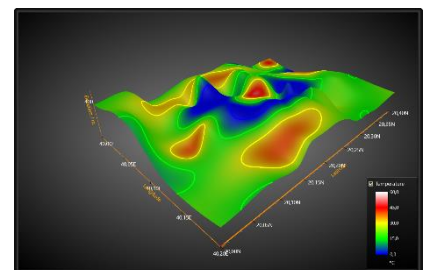
### Polygon3D:

Polygon3D オブジェクトは、2D ポリゴンを指定の Y 値の範囲にあわせてスケールします。



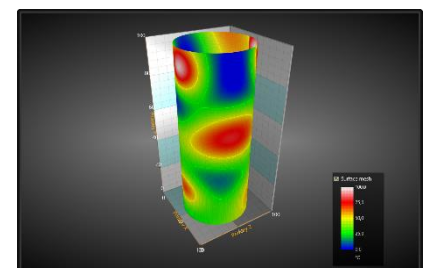
### SurfaceGridSeries3D:

SurfaceGridSeries3D は、3D サーフェスとしてデータを視覚化します。ノードは、X および Y ディメンション間で等間隔に配置されています。XY チャートの SurfaceGridSeries のように、等高線やワイヤーフレームをサポートします。サーフェスは、ビットマップ ファイルから作成できます。



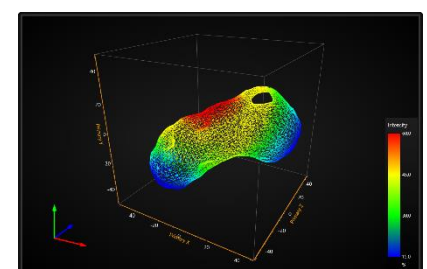
### SurfaceMeshSeries3D:

SurfaceMeshSeries3D は、SurfaceGridSeries3D と似ていますが、3D 空間でノードを自由に配置できる点が異なります。これにより、サーフェスをあらゆる形状に構築できます。



### MeshModels:

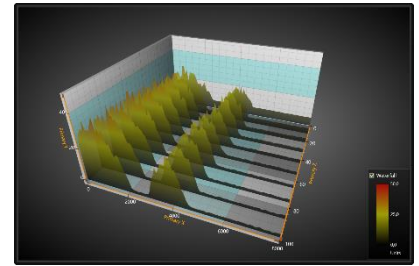
MeshModel プロパティは、外部の 3D モデル エディターから、LightningChart View3D への 3D モデルの挿入を実現します。また、プログラムによる MeshModel ジオメトリの構築もサポートしています。これにより、計算の結果として生成されたオブジェクトや



形状を視覚化できます。

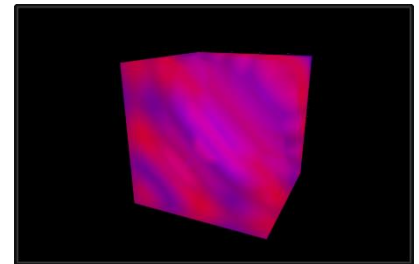
#### WaterfallSeries3D:

WaterfallSeries3D では、データがグラフ領域のストリップで視覚化されます。SurfaceMeshSeries3D のように、領域の塗りつぶし、ワイヤーフレーム、等高線を適用できます。



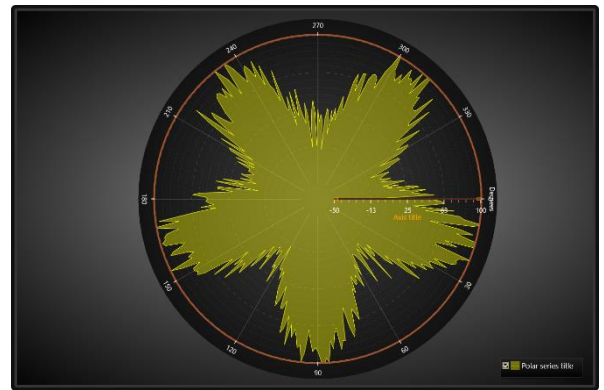
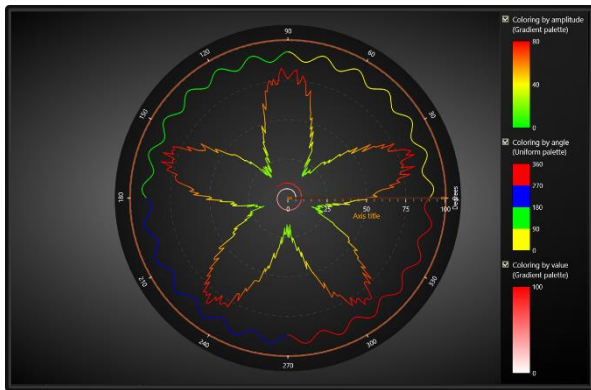
#### VolumeModels (ボリューム レンダリング):

VolumeModels は、Direct Volume Rendering (DVR) 手法を介してボリューム データを視覚化します。内部のボリューム データを取得し、視覚化します。データセット内部を通るレイの追跡とボリューム データのサンプリングを介して、アルゴリズムによる画像の生成を実現します。



## 4.3 ポーラー チャート

ポーラー チャート (ViewPolar) は、ポーラー形式によるデータの視覚化を実現します。データ ポイントの位置は、角度の値と振幅をもとに決定されます (ViewXY で角度となる X と振幅となる Y を比較します)。また、PointLineSeries、AreaSeries、パレットによる色付けなど、その他のチャートで実装できる機能の多くをサポートしています。



## 4.4 スミス チャート

スミス チャート (ViewSmith) は、インピーダンス測定や

インピーダンス マッチング アプリケーション等の電子機器で使用されます。スミス チャートの **PointLineSeries** は、その他のビューと同様に、線、点群、または線および点を描画するために使用できます。

