

LineChart for ASP.NET Web Forms

2018.04.12 更新

グレースィティ株式会社

目次

製品の概要	3
ComponentOne for ASP.NET Web Forms のヘルプ	3
主な特長	4
機能一覧	5
折れ線グラフの機能	5
マーカーの機能	5
線の機能	5
チャート全般の機能	5-6
データ種別の機能	6
データ連結の機能	6
軸の機能	6-7
軸グリッド線の機能	7
軸目盛りの機能	7
軸ラベルの機能	7
値ラベルの機能	7
ヘッダ／フッタの機能	7-8
凡例の機能	8
ヒントの機能	8-9
アニメーションの機能	9
スタイルの機能	9-10
デザイン時の機能	10
クイックスタート	11
手順 1: アプリケーションの作成	11
手順 2: C1LineChart へのシリーズ一覧の追加	11
手順 3: コントロールのカスタマイズ	11
手順 4: アプリケーションの実行	11-12
デザイン時のサポート	13
C1LineChart スマートタグ	13
C1LineChart のコレクション エディター	13
LineChartSeries コレクション エディター	14
ChartStyle コレクション エディター	14-15

C1LineChart の基礎	16
折れ線グラフのタイプ	16
面グラフ	16
ベジェ曲線グラフ	16
スプライン面グラフ	16
Line のフィッティングタイプ	16-17
データ欠損	17
C1LineChart の要素	18
軸	18-19
軸の位置	19
軸の外観	19
軸のタイトルと回転	19
軸の目盛記号	20
軸のグリッド線	20
軸の範囲	20-21
軸の注釈	21
値の注釈	21
グラフのラベル	21
ヘッダーとフッター	21-22
凡例	22
系列	22-23
注釈	24
近似曲線	25-29
データ連結	30
エクスポートサービス	31-34
C1LineChart のアニメーション	35
遷移効果	35
アニメーション効果の持続時間	35

製品の概要

LineChart for ASP.NET Web Forms によって、カスタマイズした線形グラフを表示します。折れ線グラフは、相互につないだデータポイントとして各系列を描画します。

ComponentOne for ASP.NET Web Forms のヘルプ

ComponentOne for ASP.NET Web Forms の各コントロールで共通したトピック、アセンブリの追加、テーマの適用、クライアント側情報などについては「[ASP.NET Web Forms ユーザーガイド](#)」を参照してください。

主な特長

LineChart for ASP.NET Web Forms は、以下の独特な主要機能を備えています。

- **HTML5 グラフ**

HTML5 グラフが優れているのは、これらが Scalable Vector Graphics (SVG) を使用しているためです。SVG は HTML5 で充実した対話型の視覚化を作成するのに完璧です。SVG によるグラフ作成のメリットの中には、真のクライアント側グラフ作成、DOM フレンドリなグラフ、ハードウェアアクセラレーショングラフィックス、対話型のアニメーションとツールチップ、およびストリーミングの視覚化が含まれます。さらに、これらのグラフは追加のプラグインが不要です。

- **面グラフ**

面グラフを作成することによって、値の変化を視覚化します。面グラフはさまざまなデータポイントを結んでいる線の下でのグラフの部分に塗りつぶすだけで簡単にできます。

- **スプライングラフ**

スプライン面グラフを作成します。線は、ポイント間に直線を描画する代わりに、ポイント間に滑らかな曲線を描画します。

- **シンボル**

折れ線グラフのマーカーとして、画像などのシンボルを使用します。

- **複数の系列**

折れ線グラフに複数の系列を表示します。これは2つの関連するデータ系列を表示する場合に役立ちます。

- **データ連結**

C1LineChart は、サーバー上にある外部データソースからのデータのバインドをサポートしています。DataBinding を有効にするには、単に DataSourceID プロパティまたは DataSource プロパティと、DataBindings プロパティを設定するだけです。**C1LineChart** は、すべての標準 ASP.NET DataSource コントロール (SqlDataSource など) とすべての DataSource タイプ (DataTable など) をサポートしています。**C1LineChart** の DataBinding は非常に使い慣れた機能であり、デザイン時やコードに操作できます。

- **ライブデータ**

すべての ComponentOne グラフはライブデータのストリーミング用に最適化されています。これらのグラフは、データが変化するとアニメーション化し、遷移を表示してデータセット内のデータを視覚化します。

- **ツールチップ**

ツールチップやリンクをグラフ要素に関連付けて、グラフがマウスイベントに応答するようにします。これはクライアント側で処理される HTML 画像マップを使用して行われます。

- **外観**

C1LineChart の豊富なプロパティセットによって、グラフのルック&フィールを変更できます。系列や系列の項目を異なる色にし、グラフ要素に影に付けて、その外観を変更します。ユーザーの操作性をさらに高めるために、位置、サイズ、および色も調整可能です。

- **DateTime**

対応するデータを持つ日付を折れ線グラフに追加します。

- **自動軸**

C1LineChart はグラフの軸を自動的に作成します。

- **ラベル**

ラベルによって折れ線グラフの読みやすさを向上させます。重要なデータポイントを強調表示し、あるいはデータやグラフに関する情報を提供します。

機能一覧

このトピックでは、LineChart for ASP.NET Web Formsの主な機能の一覧と、機能を理解するのに役立つページへのリンクを記載しています。

折れ線グラフの機能

機能	サンプル	ヘルプ	リファレンス	KB
グラフ種別(折れ線、面)	○	○	○	
欠損値		○	○	
レイアウトの読み込み			○	
レイアウトの保存			○	
リロード時のアニメーション			○	
アニメーションの方向(水平、垂直)			○	

マーカーの機能

機能	サンプル	ヘルプ	リファレンス	KB
記号の種類(円、三角形、逆三角形、四角形、ひし形、十字)			○	
表示			○	
記号の高さ	○		○	
記号の幅	○		○	
記号を表示する目盛りのインデックス	○		○	
記号のURL	○		○	

線の機能

機能	サンプル	ヘルプ	リファレンス	KB
線の種類(直線、スプライン、ベジェ)	○	○	○	
目盛りのスタイル			○	

チャート全般の機能

機能	サンプル	ヘルプ	リファレンス	KB
チャート系列のスタイル			○	
マウスホバー時のスタイル			○	
チャートテキストのスタイル			○	
チャートラベルの表示		○	○	
チャートラベルの書式		○	○	

チャートラベルのスタイル		<input type="radio"/>	<input type="radio"/>	
テーマ			<input type="radio"/>	
影			<input type="radio"/>	
マージン			<input type="radio"/> 1、 <input type="radio"/> 2、 <input type="radio"/> 3、 <input type="radio"/> 4	
カルチャ情報			<input type="radio"/>	

データ種別の機能

機能	サンプル	ヘルプ	リファレンス	KB
数値(Double)			<input type="radio"/>	
日時	<input type="radio"/>		<input type="radio"/>	
文字列(X軸のみ)			<input type="radio"/>	

データ連結の機能

機能	サンプル	ヘルプ	リファレンス	KB
データソース(ASP.NETデータソースコントロール)	<input type="radio"/>	<input type="radio"/>		
連結するデータメンバ		<input type="radio"/>	<input type="radio"/>	
X値のフィールド名	<input type="radio"/>		<input type="radio"/>	
X値のデータ型(数値、日時、文字列)	<input type="radio"/>		<input type="radio"/>	
Y値のフィールド名	<input type="radio"/>		<input type="radio"/>	
Y値のデータ型(数値、日時)	<input type="radio"/>		<input type="radio"/>	

軸の機能

機能	サンプル	ヘルプ	リファレンス	KB
最大値		<input type="radio"/>	<input type="radio"/>	
最大値を自動的に計算		<input type="radio"/>	<input type="radio"/>	
最小値		<input type="radio"/>	<input type="radio"/>	
最小値を自動的に計算		<input type="radio"/>	<input type="radio"/>	
原点の値	<input type="radio"/>		<input type="radio"/>	
軸の表示			<input type="radio"/>	
軸の位置(上、下、左、右)		<input type="radio"/>	<input type="radio"/>	
軸のスタイル		<input type="radio"/>	<input type="radio"/>	
軸タイトルの表示			<input type="radio"/>	
軸タイトルの値		<input type="radio"/>	<input type="radio"/>	
軸タイトルのスタイル			<input type="radio"/>	

LineChart for ASP.NET Web Forms

軸タイトルの配置(左、中央、右)		<input type="radio"/>	<input type="radio"/>	
注釈の書式			<input type="radio"/>	
注釈の表示方法(値、ラベル)			<input type="radio"/>	
主目盛りの値を自動的に計算		<input type="radio"/>	<input type="radio"/>	
主目盛り記号の単位		<input type="radio"/>	<input type="radio"/>	
副目盛りの値を自動的に計算		<input type="radio"/>	<input type="radio"/>	
副目盛り記号の単位		<input type="radio"/>	<input type="radio"/>	

軸グリッド線の機能

機能	サンプル	ヘルプ	リファレンス	KB
スタイル			<input type="radio"/>	
表示			<input type="radio"/>	

軸目盛りの機能

機能	サンプル	ヘルプ	リファレンス	KB
長さ		<input type="radio"/>	<input type="radio"/>	
種類(内側、外側、両方、なし)		<input type="radio"/>	<input type="radio"/>	

軸ラベルの機能

機能	サンプル	ヘルプ	リファレンス	KB
スタイル			<input type="radio"/>	
テキストの配置(左、中央、右)			<input type="radio"/>	
幅			<input type="radio"/>	

値ラベルの機能

機能	サンプル	ヘルプ	リファレンス	KB
テキスト			<input type="radio"/>	
数値			<input type="radio"/>	
DateTime値			<input type="radio"/>	
グリッド線の表示			<input type="radio"/>	
グリッド線のスタイル			<input type="radio"/>	

ヘッダ／フッタの機能

機能	サンプル	ヘルプ	リファレンス	KB
----	------	-----	--------	----

表示			○	
位置(上、下、左、右)			○	
テキスト			○	
テキストのスタイル			○	
タイトルのスタイル			○	

凡例の機能

機能	サンプル	ヘルプ	リファレンス	KB
表示			○	
位置(上、下、左、右)			○	
方向(水平、垂直)			○	
アイコンのサイズ			○	
スタイル			○	
テキスト			○	
テキストの幅			○	
テキストの余白			○	
テキストのスタイル			○	
タイトルのスタイル			○	

ヒントの機能

機能	サンプル	ヘルプ	リファレンス	KB
表示			○	
位置(上、下、左、右)			○	
水平方向のオフセット			○	
垂直方向のオフセット			○	
スタイル			○	
コンテンツ			○	
コンテンツのスタイル			○	
タイトル			○	
タイトルのスタイル			○	
吹き出しの表示			○	
吹き出し			○	
吹き出しのスタイル			○	
アニメーション効果			○	

LineChart for ASP.NET Web Forms

表示／非表示にする時間			○	
イージング			○	
表示のアニメーション効果			○	
表示の遅延時間			○	
表示にする時間			○	
表示のイージング			○	
非表示のアニメーション効果			○	
非表示の遅延時間			○	
非表示にする時間			○	
非表示のイージング			○	

アニメーションの機能

機能	サンプル	ヘルプ	リファレンス	KB
有効			○	
イージング(7種類)		○	○	
持続時間		○	○	

スタイルの機能

機能	サンプル	ヘルプ	リファレンス	KB
Clip-Rect			○	
マウスカーソル			○	
CXポイント値			○	
CYポイント値			○	
塗りつぶし色(単色、線形グラデーション、放射状グラデーション)			○	
塗りつぶしの不透明度			○	
フォント			○	
フォントサイズ			○	
フォントの太さ			○	
高さ			○	
ハイパーリンク			○	
不透明度			○	
パス			○	
Rの長さ			○	
回転			○	
RXポイント値			○	

RYポイント値			○	
スケール			○	
Src			○	
ストロークの色(単色、線形グラデーション、放射状グラデーション)			○	
ストロークのダッシュ配列			○	
ストロークのラインキャップ			○	
ストロークのLineJoin			○	
ストロークのMiterLimit			○	
ストロークの不透明度			○	
ストロークの幅			○	
ターゲット			○	
テキストアンカー			○	
タイトル			○	
変換			○	
幅			○	
X座標			○	
Y座標			○	

デザイン時の機能

機能	サンプル	ヘルプ	リファレンス	KB
スマートタグ		○		
コレクションエディター		○		

クイックスタート

このクイックスタートは、ASP.NET コントロールである **C1LineChart** の基本的な使用方法を説明します。このクイックスタートでは、1つの **C1LineChart** コントロールを含む ASP.NET アプリケーションを作成し、データ系列を追加して **C1LineChart** をカスタマイズします。

手順 1: アプリケーションの作成

このトピックでは、**C1LineChart** コントロールを追加します。

1. まず ASP.NET Web アプリケーション, を作成します。
2. コントロールをツールボックスに追加します。
3. 「**デザイン**」タブを選択します。
4. デザインビューで、Visual Studio ツールボックスに移動し、**C1LineChart** コントロールをダブルクリックしてページのメインコンテンツに追加します。

手順 2: C1LineChart へのシリーズ一覧の追加

この手順では、DateTimeValueを X LineChartSeries に、DoubleValue を Y LineChartSeries に設定します。また系列の記号とラベルも変更します。

1. **LineChart** スマートタグをクリックし、[**C1LineChart タスク**]メニューから**シリーズ一覧**を選択します。
LineChartSeries コレクション エディターが表示されます。
2. **LineChartSeries コレクション エディター**で、<**追加**>ボタンをクリックして、新しい **LineChartSeries** を LineChart に追加します。
3. 右のプロパティペインで、**Data** ノードを拡張した後、X を拡張して **DateTimeValue** テキストボックスに「2005-07-01, 2005-08-01, 2005-10-01, 2005-12-01, 2006-03-01, 2006-12-01, 2008-01-01」を入力します。
4. **LineChartSeries コレクション エディター**の右プロパティペインで、Y ノードを拡張し、**DoubleValue** プロパティの横にデータ「1000000, 1000000, 1100000, 1400000, 1700000, 2000000, 2500000」を入力します。
5. **LineChartSeries** の **Label** プロパティを**北アメリカ**に設定し、グラフの凡例に表示されるようにします。
6. **Markers** ノードを拡張して、**Type** プロパティを **Box** に設定します。
7. **MarkerStyle** ノードを拡張した後、**Fill** ノードを拡張して、<...>ボタンをクリックし、色値 Hex={CC,33,00} を選択します。
<**OK**>をクリックして、[**その他の色**]ダイアログボックスを閉じます。
8. <**OK**>をクリックして、**LineChartSeries コレクション エディター**を閉じます。

手順 3: コントロールのカスタマイズ

このトピックでは、デザイン時に C1LineChart **プロパティ**ウィンドウを使用して、軸の注釈、グラフのヘッダー、およびグラフの凡例をカスタマイズします。

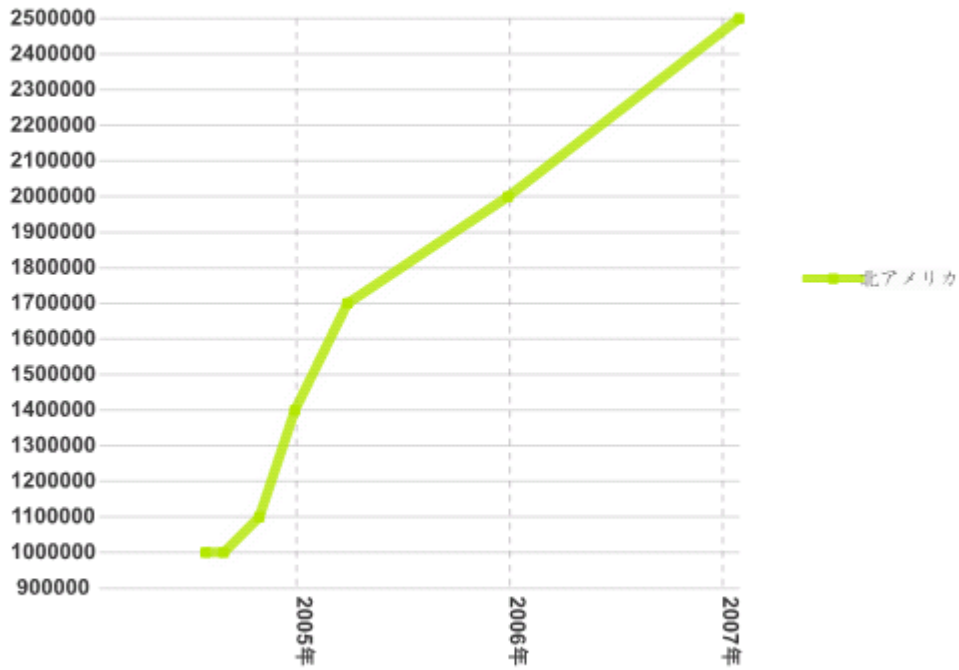
1. デザインビューで、**LineChart.Axis.X.Labels.Style.Rotation** プロパティを 90 に設定します。
2. **C1LineChart** の **Width** プロパティを 500 に設定します。
3. Header ノードを拡張し、**Header.Text** プロパティを「ゲームAのプレイヤー数」に設定します。

手順 4: アプリケーションの実行

[F5]を押して、プロジェクトを実行します。次のような表示になります。

- グラフヘッダーは、**Header.Text** プロパティを使用して追加されました。
- X 軸の注釈は、**LineChart.Axis.X.Labels.Style.Rotation** プロパティusing を使用して 90 度回転されています。
- 線系列のマーカは、**Type** プロパティが に設定されているため、四角形として表示されています。

ゲームAのプレーヤー数



おめでとうございます！ LineChart for ASP.NET Web Forms クイックスタート が正常に完了しました。

デザイン時のサポート

C1LineChart は、カスタマイズされたコンテキストメニュー、スマートタグ、および充実したデザイン時サポートを提供するデザイナを備えており、オブジェクトモデルの操作が簡素化されています。

以下のセクションでは、**C1LineChart** のデザイン時環境を使用して **C1LineChart** コントロールを設定する方法について説明します。

C1LineChart スマートタグ

Visual Studio では、**C1LineChart** コントロールはスマートタグを備えています。スマートタグは、**C1LineChart** で最もよく一般的に使用されるプロパティを提供するショートカットタスクメニューです。

C1LineChart コントロールでは、スマートタグによって、よく使用されるプロパティにすばやく簡単にアクセスできます。

[**C1LineChart** タスク]メニューにアクセスするには、**C1LineChart** コントロールの右上端にあるスマートタグ(🏷️)をクリックします。これによって、[**C1LineChart** タスク]メニューが開きます。



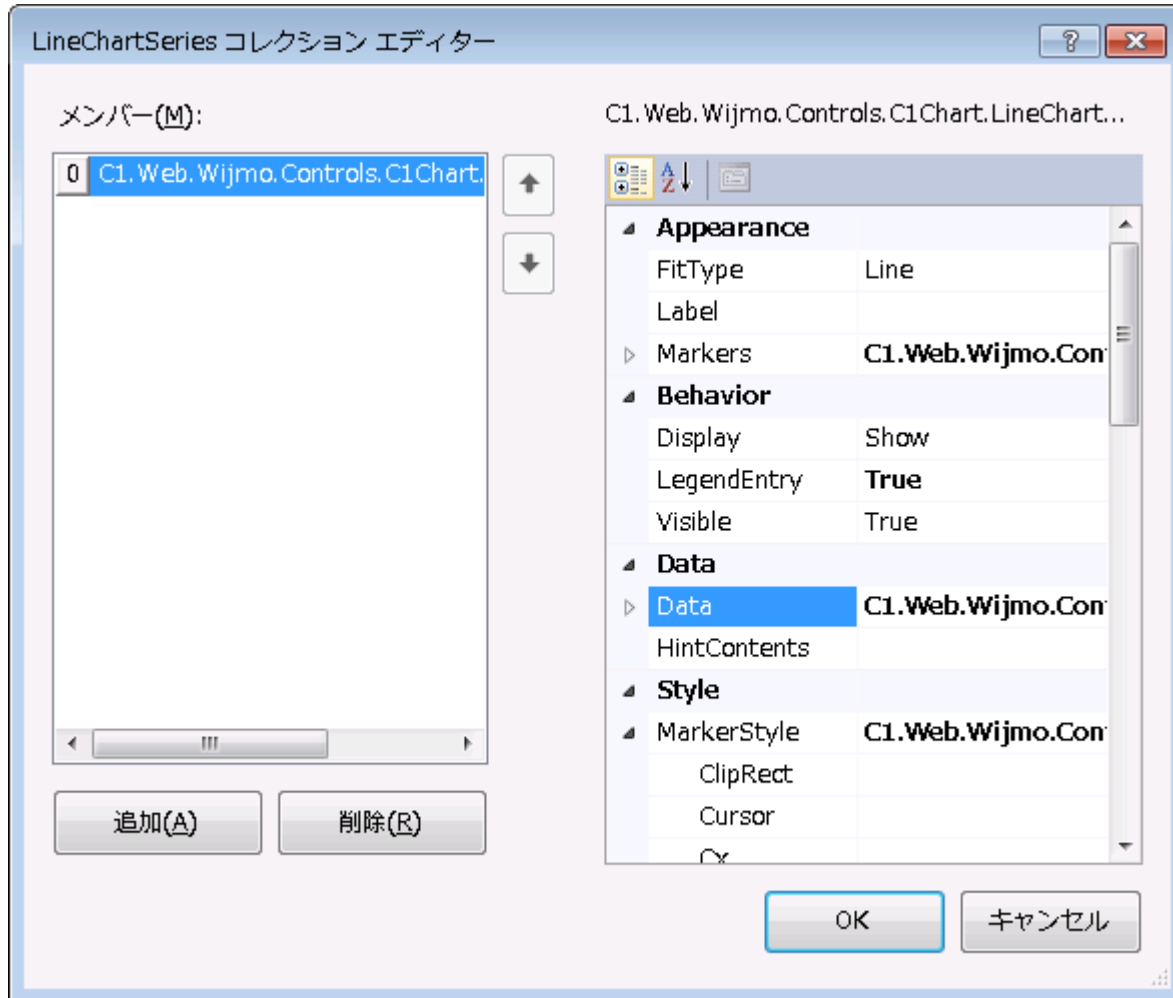
[**C1LineChart** タスク]メニューは、以下のように操作します。

- **データソースの選択**
項目[**データソースの選択**]をクリックすると、既存のデータソースやバインドする新しいデータソースを選択できるドロップダウンリストが開きます。
- **シリーズ一覧**
シリーズ一覧 項目を選択した場合、**C1LineChart** コントロールに**LineChartSeries** メンバを追加したり、削除したりできる[**LineChartSeries コレクション エディター**]ダイアログボックスが開きます。**LineChartSeries**メンバを追加したら、そのプロパティを変更できます。
- **シリーズスタイル**
シリーズスタイル 項目をクリックすると、**C1LineChart** コントロールに**ChartStyle** メンバを追加したり、削除したりできる[**ChartStyle コレクション エディター**]ダイアログボックスが開きます。**ChartStyle** メンバを追加したら、そのプロパティを変更します。
- **シリーズホバースタイル**
シリーズホバースタイル 項目をクリックすると、**C1LineChart** コントロールに**ChartStyle** メンバを追加したり、削除したりできる[**ChartStyle コレクション エディター**]ダイアログボックスが開きます。**ChartStyle** メンバを追加したら、そのプロパティを変更します。
- **CDN の使用**
[**CDN の使用**]チェックボックスを ON にすると、CDN からクライアントリソースがロードされます。これはデフォルトで OFF です。
- **CDN パス**
CDN の URL パスを表示します。
- **バージョン情報**
[**バージョン情報**]をクリックすると、製品のバージョン情報を確認できるダイアログボックスが表示されます。

C1LineChart のコレクション エディター

LineChartSeries コレクション エディター

LineChartSeries コレクション エディターによって、ユーザーは選択した **LineChartSeries** メンバを **C1LineChart** コントロールに対して追加または削除できます。**LineChartSeries** メンバが追加されたら、そのプロパティを変更できます。

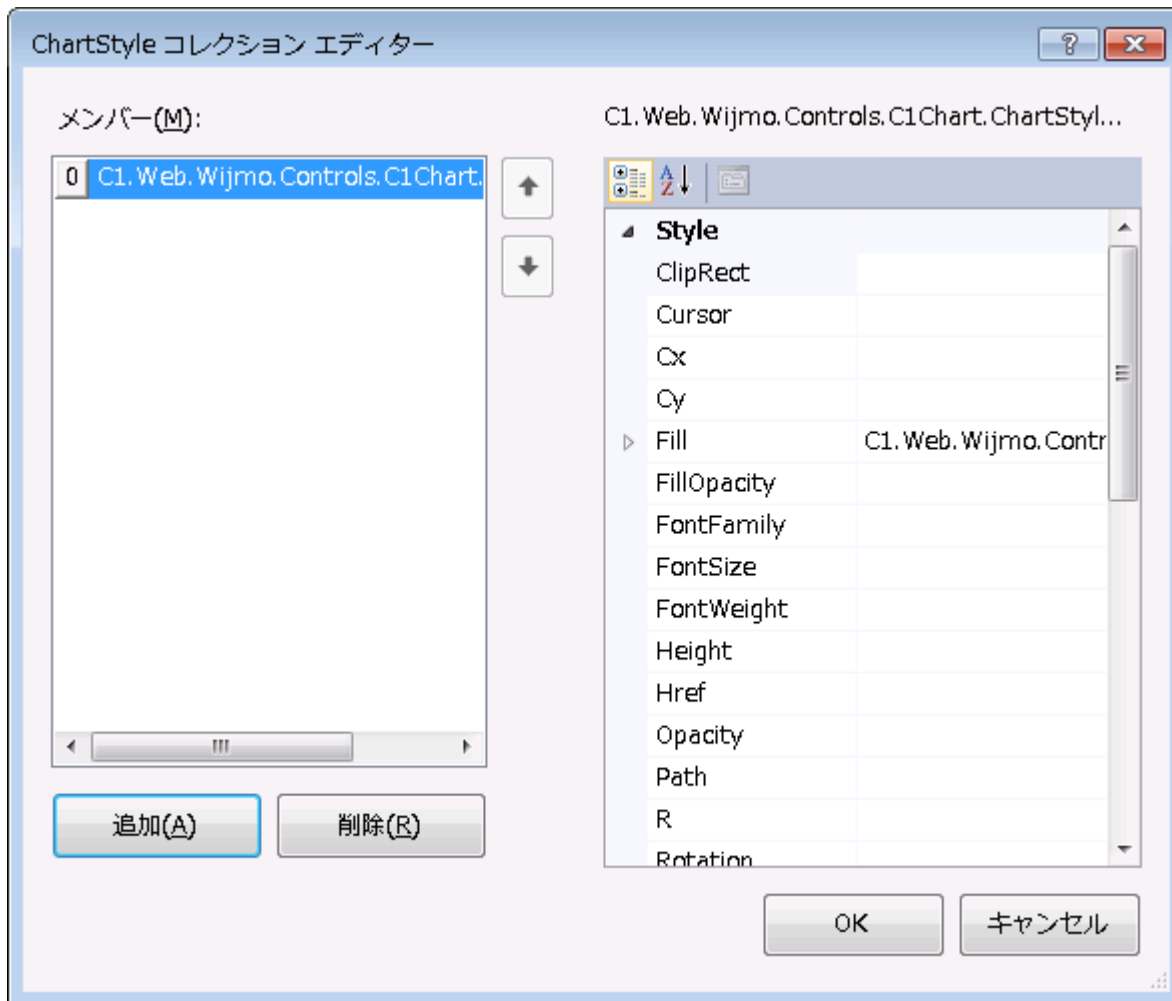


LineChartSeries コレクション エディターにアクセスするには、以下の手順を実行します。

1. **C1LineChart** コントロールを選択して、そのスマートタグをクリックします。
2. [**C1LineChart タスク**]メニューで、**シリーズ一覧**項目を選択します。これによって、[**LineChartSeries コレクション エディター**]ダイアログボックスが開きます。

ChartStyle コレクション エディター

ChartStyle コレクション エディターによって、ユーザーは **ChartStyle** メンバを**C1LineChart** コントロールに対して追加または削除できます。**ChartStyle** メンバが追加されたら、そのプロパティを変更できます。



ChartStyle コレクション エディターにアクセスするには、以下の手順を実行します。

1. **C1LineChart** コントロールを選択して、そのスマートタグをクリックします。
2. [**C1LineChart タスク**]メニューで、**シリーズスタイル**項目を選択します。これによって[**ChartStyle コレクション エディター**]ダイアログボックスが開きます。

C1LineChart の基礎

折れ線グラフでは、2つの数値型 XY 値を使用してデータポイントをプロットし、各データポイントは線で互いに結ばれています。これは時間上の傾向を示す際に役立ちます。

折れ線グラフのタイプ

C1LineChart コントロールは、以下のタイプの折れ線グラフを作成するために使用できます。

- 面グラフ
- ベジェ曲線グラフ
- スプライン面グラフ

面グラフ

面グラフでは、相互接続されたデータポイントとして各系列が表示され、それらのポイントの下の領域は塗りつぶされます。各系列は、前の系列の上に表示されます。ポイントの高さは Y 軸の値によって決定され、カテゴリのラベルは X 軸の値によって決定されます。

面グラフを作成するには、**Type** を Area に設定し、X 値と Y 値を持ついくつかの **LineChartSeries** を追加します。

ベジェ曲線グラフ

ベジェ曲線グラフは、折れ線グラフの一種で、**LineChartSeries** の X、Y データポイントによって角度の代わりにカーブをプロットします。単純な折れ線グラフと同様に、このグラフも一定期間の傾向を示すために使用されます。

ベジェ曲面グラフを作成するには、**Type** を Area、FitType を Spline に設定し、X 値と Y 値を持ついくつかの **LineChartSeries** を追加します。

スプライン面グラフ

スプライングラフは、線がプロットを通過してカーブしますが、滑らかにカーブしたプロット線を提供します。

スプライン面グラフを作成するには、**Type** を Area、FitType を Spline に設定し、X 値と Y 値を持ついくつかの **LineChartSeries** を追加します。

Line のフィッティングタイプ

FitType プロパティは、line がグラフのポイントにフィッティングする方法を設定します。このプロパティは **LineChartFitType** 列挙体を取ります。

LineChartFitType.Line

Line フィッティングは、プロット線のデフォルトのフィッティングで、線はポイント間をまっすぐ結びます。

LineChartFitType.Spline

Spline フィッティングでは、線はまだプロットを通りながらカーブしますが、滑らかにカーブしたプロット線を提供します。大部分の描画アプリケーションの曲線の作成方法に精通している場合は、Spline フィッティングタイプにも精通できます。

LineChartFitType.Bezier

Beziers フィッティングでは、線は必ずしもすべてのデータポイントを通る必要はありませんが、ポイントが磁石の役目を果たし、曲線を特定方向に引き付けて、曲線の曲がり方に影響します。

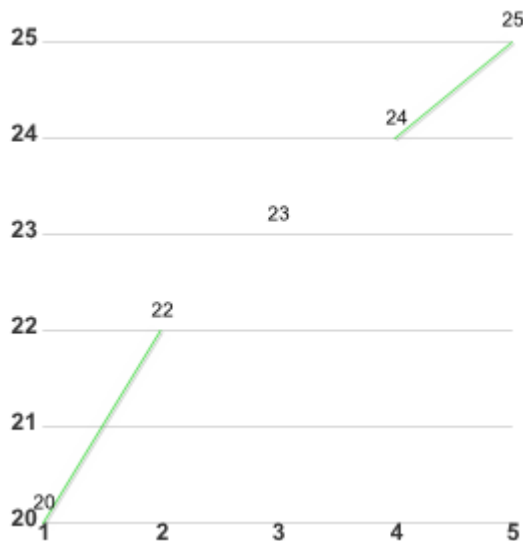
データ欠損

収集したデータをグラフ化するとき、多くの場合に、特定の値のデータが利用できないことをグラフ内で視覚的に表示する必要があります。たとえば、情報が欠けていることを示すことができます。データ欠損は、グラフ化するデータのブレイクで、データが欠けていることを視覚的に示すためにグラフ内で使用されます。

C1LineChart では、欠損値は、除外するデータポイントをデータ配列の特定の値かそうでなければ未使用の値に設定することによって指定されます。**ChartHole** オブジェクトの **NumericHole** プロパティは、使用される値を指定します。たとえば、欠損値を -1000 に設定して系列の3番目のポイントにデータが使用できないと仮定した場合、この系列ポイントの Y 値を -1000 に設定することによって、ブレイクまたは欠損値を系列プロットに導入できます。

C1LineChart Visual Studio のプロパティウィンドウで欠損値を指定するには

1. **Hole** ノードを拡張します。
2. **NumericHole** プロパティの横にある Y 値(23 など)を指定します。



C1LineChart の要素

このセクションでは、**C1LineChart** コントロールを構成する要素の概要について視覚的に説明します。各トピックは、**C1LineChart** コントロールのさまざまな特徴を表す各種の要素に分類されます。

軸

ChartAxes オブジェクトの X、Y の各プロパティは、グラフの軸の外観をカスタマイズできる **ChartAxis** オブジェクトを返します。各軸は、**ChartAxes** プロパティのサブプロパティである **X** と **Y** によって表されます。これらのプロパティは、それぞれ次のような主なプロパティを持つ **ChartAxis** オブジェクトを返します。:

Axis プロパティ	説明
Alignment	軸テキストの配置を示す値。このプロパティのデフォルト値は ChartAxisAlignment.Center です。
AnnoFormatString	注釈の書式文字列を示す値。
AnnoMethod	注釈のメソッドを示す値。このプロパティのデフォルト値は ChartAxisAnnoMethod.Values です。
AutoMajor	主目盛記号の値を自動的に計算するかどうかを示す値。このプロパティのデフォルト値は true です。
AutoMax	軸の最大値を自動的に計算するかどうかを示す値。
AutoMin	軸の最小値を自動的に計算するかどうかを示す値。このプロパティのデフォルト値は True です。
AutoMinor	補助目盛記号の値を自動的に計算するかどうかを示す値。このプロパティのデフォルト値は true です。
Compass	軸の方位を示す値。このプロパティのデフォルト値は ChartCompass.South です。
GridMajor	主グリッド線の情報を提供する値。
GridMinor	副グリッド線の情報を提供する値。
Labels	ラベルの情報を提供する値。
Max	軸の最大値を示す値。このプロパティのデフォルト値は 0 です。
Min	軸の最小値を示す値。このプロパティのデフォルト値は 0 です。
Origin	軸の起点の値を示す値。
AxisStyle	軸のスタイルを示す値。
Text	軸のテキストを示す値。
TextStyle	軸のテキストのスタイルを示す値。
TextVisible	軸テキストの表示／非表示を示す値。このプロパティのデフォルト値は True です。
TickMajor	主目盛の情報を提供する値。
TickMinor	補助目盛の情報を提供する値。
UnitMajor	主目盛記号の単位を示す値。このプロパティのデフォルト値は 0 です。
UnitMinor	補助目盛記号の単位を示す値。このプロパティのデフォルト値は 0 です。
ValueLabelList	軸の valueLabels のコレクションを表示する値。
Visible	軸の表示／非表示を示す値。このプロパティのデフォルト値は True です。

軸の位置

通常、軸の注釈は軸のそばに表示されます。これは、起点が軸の最小値や最大値でないグラフで問題となる場合があります。グラフはそのグラフタイプに応じて、さまざまな状況における注釈の配置場所を自動的に決定できます。**Compass** プロパティも注釈の位置関係を指定できます。X 軸の **Compass** 値は North または South、Y 軸の値は East または West に設定できます。デフォルトでは、X 軸は South に設定され、Y 軸は West に設定されます。

軸の外観

Alignment プロパティは、**Center**、**Near**、または **Far** の3つの異なる設定に設定できます。配置を Center に設定すると、軸タイトルはグラフ領域に対して中央に配置されます。配置を Near に設定すると、軸タイトルはグラフ領域の左側に配置されます。配置を Far に設定すると、軸タイトルはグラフ領域の右側に配置されます。

デザイン時に X 軸のラベルを変更するには、**Axis->X->Labels->AxisLabelStyle** を拡張し、**FontSize** プロパティを設定します。

Fill.Color プロパティは軸線、目盛記号、ラベル、およびタイトルの色を変更します。たとえば、デザイン時に X 軸ラベルのフォント色を変更するには、**Axis->X->Labels->AxisLabelStyle->Fill** を拡張した後、**Color** の横にある<...>ボタンをクリックして色を選択します。

以下の例は、設定された後の **Color** プロパティのソースビューを示しています。

ソースビュー

```
<Axis>
  <X Max="2010" AutoMin ="false" Min="2005" Text="年">
    <TextStyle FontSize="16">
      </TextStyle>
    <Labels>
      <Style FontSize="12">
        <Fill Color="#0033CC">
          </Fill>
        </Style>
      </Labels>
      <TickMajor Position="Outside">
        </TickMajor>
      </X>
```

軸のタイトルと回転

軸にタイトルを追加すると、その軸で何が表示されているかが明確になります。軸に沿ったタイトルまたは注釈も回転できます。

軸タイトルの追加

軸の **Text** プロパティを使用して、タイトルを軸に追加します。タイトルを削除するには、**Text** プロパティからテキストを削除します。

軸タイトルの回転

Rotation プロパティを使用して、軸タイトルを 90 度、180 度、または 270 度回転します。90 度と 270 度の回転は垂直軸に対して非常に効果的です。デザイン時に X 軸ラベルの回転を変更するには、**Axis->X->Labels->AxisLabelStyle** を拡張して、**Rotation** プロパティを設定します。

軸の目盛記号

目盛記号は、グラフの測定単位を表示するために軸に直交して表示される線です。**ChartAxisTick.Position** を **Cross** または **Outside** に設定した場合、折れ線グラフ上に主目盛記号と補助目盛記号を表示できます。目盛の間隔や属性のカスタマイズは、プロパティセットを操作するだけで容易です。

TickMajor、**TickMinor** の各プロパティは、軸の目盛記号の状態を設定します。このプロパティは、任意の **ChartAxisTickPosition** 値に設定できます。

目盛記号の位置

これらの値は目盛記号の表示位置と表示／非表示を設定します。

値	説明
ChartAxisTickPosition.None	軸の目盛記号なし。
ChartAxisTickPosition.Cross	目盛記号は軸上を横断。
ChartAxisTickPosition.Outside	目盛記号を軸上のグラフ領域の外側に配置。
ChartAxisTickPosition.Inside	目盛記号を軸上のグラフ領域の内側に配置。


目盛記号の間隔

AutoMajor プロパティと **AutoMinor** プロパティは、目盛記号がグラフによって自動的に設定されるかどうかを設定します。これらのプロパティの両方を **True** に設定した場合、グラフは現在のデータを使用して主目盛記号と補助目盛記号を論理的に配置します。**AutoMajor** プロパティが **true** の場合、軸の注釈の重なりを有効にする必要はありません。

UnitMajor プロパティと **UnitMinor** プロパティは、目盛を配置する際の単位を設定します。**UnitMajor** プロパティを設定すると、**UnitMinor** プロパティはグラフによって **UnitMajor** 値の半分の値に自動的に設定されます。グラフが **UnitMinor** プロパティを自動的に設定しますが、手動で異なる値に変更することもできます。

目盛記号の長さ

Factor プロパティを使用し、主目盛記号と補助目盛記号の長さを伸ばすことができます。**Factor** プロパティを使用する前に、**Position** を **Outside** または **Cross** に設定します。目盛記号は、軸線の太さと目盛係数に基づいてサイズ調整されます。目盛係数を2倍にすると、軸の目盛記号の長さは2倍になります。X 軸の目盛記号に負の値を使用する場合、目盛記号は X 軸ラベルの上側に表示されます。Y 軸の目盛記号に負の値を使用する場合、目盛記号は Y 軸ラベルの左側に表示されます。

 **注意:** **ChartAxisTick.Factor** プロパティの値を増加した場合、適切な **MarginBottom**、**MarginLeft**、**MarginRight**、または **MarginTop** の各プロパティも目盛記号の長さの伸長に合わせて適切なスペースを増加させる必要があります。

軸のグリッド線

グリッド線は、主／副単位間隔ごとに主目盛記号および補助目盛記号と直交して表示されます。主間隔の軸に直交して表示される線は **GridMajor** プロパティによって制御され、副間隔の軸に直交して表示される線は **GridMinor** プロパティによって制御されます。グリッド線を設定すれば、正確な値を確認する際のグラフの読みやすさが向上します。

軸の範囲

通常、グラフでは、含まれているデータがすべて表示されます。しかし、軸の範囲を調整することによって、グラフの特定の部分を表示することもできます。

LineChart for ASP.NET Web Forms

グラフでは、最低値と最高値、および数値の増分を考慮することによって、各軸の範囲が決まります。**Min**、**Max**、**AutoMin**、および **AutoMax** の各プロパティを設定すれば、このプロセスをカスタマイズできます。

軸の最小値と最大値

特定の軸の値でグラフの枠を設定するには、**Min** プロパティと **Max** プロパティを使用します。グラフの X 軸の値が 0 ~ 100 の場合、**Min** を 0、**Max** を 10 に設定すると、値は 10 までしか表示されません。

グラフでは、**Min** と **Max** の値を自動的に計算することもできます。**AutoMax** プロパティと **AutoMin** プロパティを **True** に設定した場合、グラフでは、現在のデータセットに合わせて軸の数値が自動設定されます。

軸の注釈

各軸の注釈は、どのようなグラフでも重要な部分です。グラフは、データが変化しても可能なかぎり自然な注釈を自動生成します。

以下のプロパティは、**C1LineChart** の軸の注釈の書式とレイアウトを表します。

Axis プロパティ	説明
AnnoFormatString	注釈の書式文字列を示す値。
AnnoMethod	注釈のメソッドを示す値。
ValueLabelsList	軸の valueLabels のコレクションを表示する値。

値の注釈

値の注釈は、グラフがデータ自体に基づいて数値型の注釈を自動的に生成する実装です。値の注釈は任意の軸に対し、任意のグラフタイプと任意のデータレイアウトで使用できます。値の注釈は軸の以下のプロパティによって制御されます。

プロパティ	説明
AnnoFormatString	注釈の書式文字列を示す値。

グラフのラベル

折れ線グラフのグラフラベルは、データポイントの各セットの内側またはすぐ外側に **C1LineChart** の x 値と y 値を表示するラベルを表します。

グラフラベルは、重要データポイントを強調表示する際に役立ちますが、一般に情報をデータ上やグラフ上に提供する際にも使用できます。

x、y 値を指定すると、グラフラベルはデータポイントの内側に自動表示されます。グラフを移動させるには、**ChartLabels** は、**ShowChartLabels** プロパティを **False** に設定して非表示にできます。

グラフラベルの書式設定

グラフラベルは、**ChartLabelFormatString** プロパティを使用して書式設定できます。

グラフラベルの外観

ChartLabelStyle プロパティを使用し、グラフラベルの外観をカスタマイズできます。

ヘッダーとフッター

ヘッダー要素とフッター要素は、グラフに関する説明情報を表示するために使用されます。これらは、**Header** プロパティと **Footer** プロパティによって制御されます。

グラフのヘッダーとフッターの各プロパティは、以下の主なプロパティを含む **ChartTitle** オブジェクトを返します。

プロパティ	説明
Compass	タイトルの方位を示す値。これは、タイトルの位置関係を North(グラフの上側)、South(グラフの下側)、East(グラフの右側)、および West(グラフの左側)から決定します。
LegendStyle	凡例のスタイルを示す値。
Text	タイトルの位置を決定します。
TextStyle	タイトルテキストのスタイルを示す値。
Visible	タイトルを表示するかどうかを決定します。

C1Chart は、タイトルのサイズと位置をそのコンテンツと**Compass** の設定値に基づいて自動的に設定します。

ヘッダー要素とフッター要素のカスタマイズ

ヘッダー要素とフッター要素のテキストと配置、位置、枠、色、フォントは、**ChartTitle** のプロパティを使用してカスタマイズできます。

ヘッダー要素は、**X** プロパティを使用して左側(負の値)または右側(正の値)に移動でき、**Y** プロパティを使用して上側(正の値)または下側(負の値)に移動できます。

凡例

凡例要素は、グラフの各データ系列に関する情報を表示します。グラフの凡例は、物理的な色とデータ系列の間のマッピングを表示します。

C1LineChart は、グラフにデータが存在して、**LegendEntry** プロパティが有効の場合は常に **Legend** を自動的に生成します。**Label** プロパティを指定すると、各系列の名前が凡例に示されます。**Label** プロパティに指定された値が存在しない場合は、系列名は未定義として凡例に表示されます。

凡例は、**Legend** プロパティによって制御され、以下の主なプロパティを持つ **ChartLegend** オブジェクトを返します。

プロパティ	説明
Text	凡例タイトルに表示するテキストを含みます。
LegendStyle	凡例のスタイルを示す値。
Compass	凡例の位置を決定します。
Visible	凡例を表示するかどうかを決定します。
Orientation	凡例の項目を水平または垂直方向に表示するかどうかを決定します。

C1Chart は、凡例のサイズと位置をそのコンテンツと **Compass** プロパティおよび **Orientation** プロパティに基づいて自動生成します。

凡例の向きは、**Orientation** プロパティから水平または垂直に設定でき、その位置は**Compass** プロパティから north、south、east、または west に設定できます。

系列

LineChart for ASP.NET Web Forms

LineChartSeries オブジェクトは、折れ線グラフにプロットされているデータを表します。1つ以上の **LineChartSeries** を **C1LineChart** 上に持つことができます。各 **LineChartSeries** は異なる色で表されます。**LineChartSeries** は、デザインビューで **LineChartSeries コレクション エディター** から追加でき、ソースビューでは **LineChartSeries** 要素から追加できます。また、プログラムによって **LineChartSeries** オブジェクトから追加できます。

デザインビューでの LineChartSeries の追加

1. **C1LineChart** コントロールを選択して、そのスマートタグをクリックします。
2. [**C1LineChart タスク**]メニューで、シリーズ一覧項目を選択します。これによって、[**LineChartSeries コレクション エディター**]ダイアログボックスが開きます。
3. <追加>をクリックし、**LineChartSeries** メンバをシリーズ一覧コレクションに追加します。

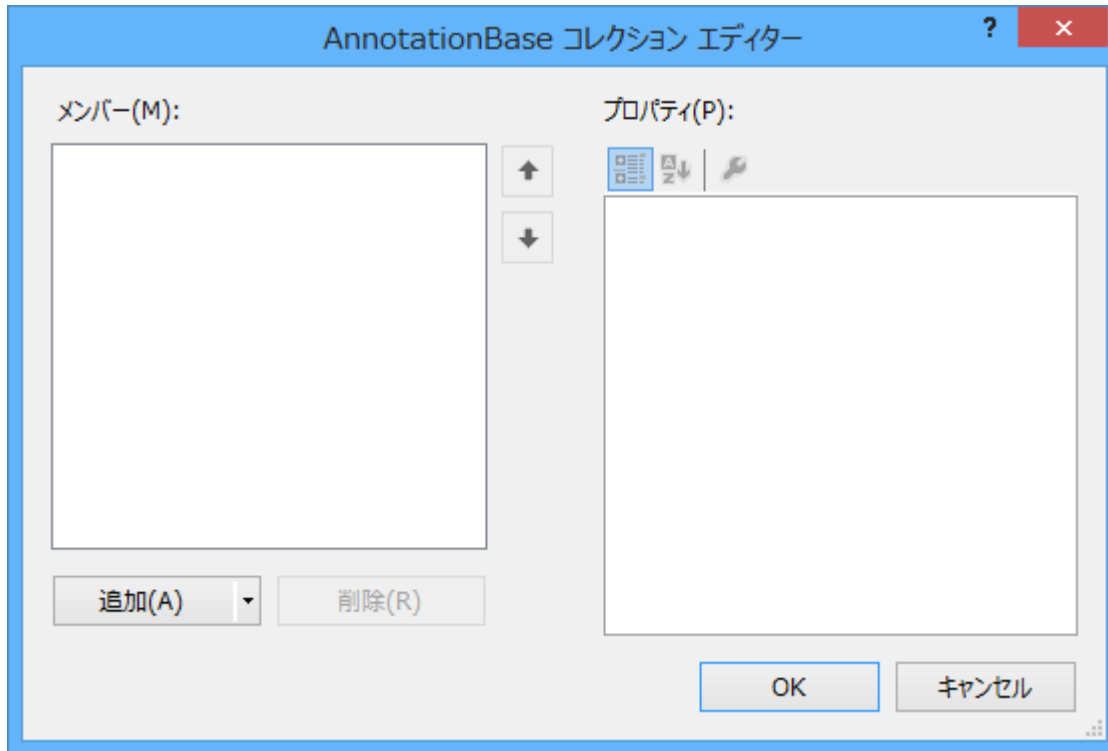
ソースビューでの LineChartSeries の追加

ソースビュー

```
<SeriesList>
  <cc1:LineChartSeries Label="#Wijmo" LegendEntry="true">
    <Markers Visible="true" Type="Circle">
      <Symbol>
        <cc1:LineChartMarkerSymbol Index="6" Url="Images/wijmo.png" Width="30" Height="30" />
        <cc1:LineChartMarkerSymbol Index="2" Url="Images/hatemo.png" Width="30" Height="30" />
      </Symbol>
    </Markers>
    <Data>
      <X>
        <Values>
          <cc1:ChartXData DateTimeValue="2010-11-21" />
          <cc1:ChartXData DateTimeValue="2010-11-22" />
          <cc1:ChartXData DateTimeValue="2010-11-23" />
          <cc1:ChartXData DateTimeValue="2010-11-24" />
          <cc1:ChartXData DateTimeValue="2010-11-25" />
          <cc1:ChartXData DateTimeValue="2010-11-26" />
          <cc1:ChartXData DateTimeValue="2010-11-27" />
          <cc1:ChartXData DateTimeValue="2010-11-28" />
          <cc1:ChartXData DateTimeValue="2010-11-29" />
        </Values>
      </X>
      <Y>
        <Values>
          <cc1:ChartYData DoubleValue="12" />
          <cc1:ChartYData DoubleValue="30" />
          <cc1:ChartYData DoubleValue="6" />
          <cc1:ChartYData DoubleValue="22" />
          <cc1:ChartYData DoubleValue="14" />
          <cc1:ChartYData DoubleValue="25" />
          <cc1:ChartYData DoubleValue="41" />
          <cc1:ChartYData DoubleValue="14" />
          <cc1:ChartYData DoubleValue="3" />
        </Values>
      </Y>
    </Data>
  </cc1:LineChartSeries>
</SeriesList>
```


注釈

C1BarChartでは、**Annotations** プロパティを設定して注釈を追加できます。注釈を追加またはカスタマイズするために AnnotationBaseコレクションエディタを使用することができます。

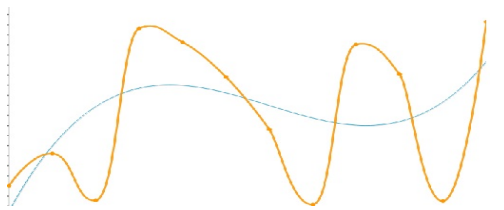
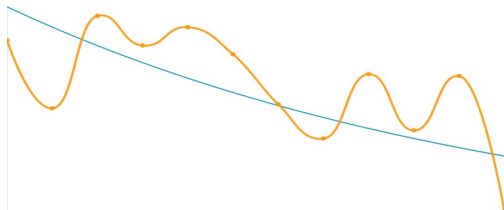
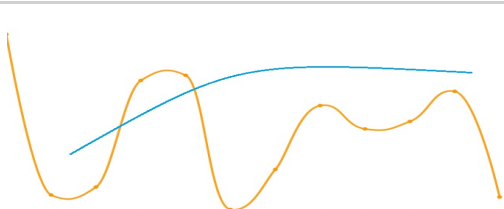
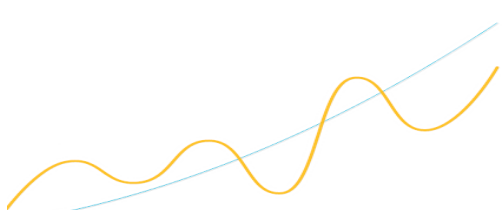
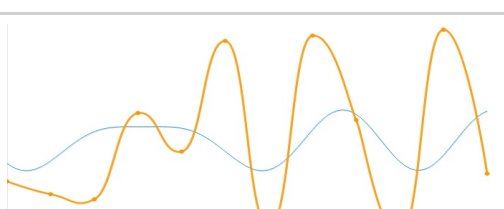
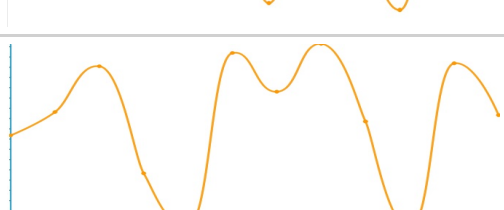


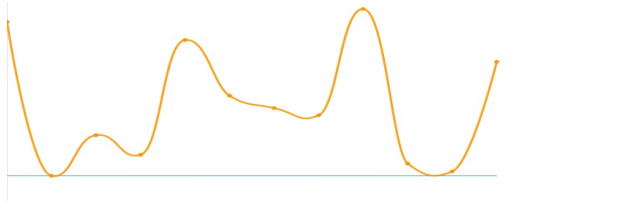
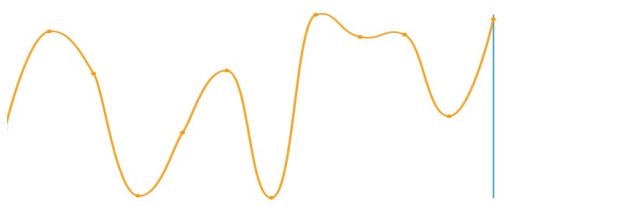
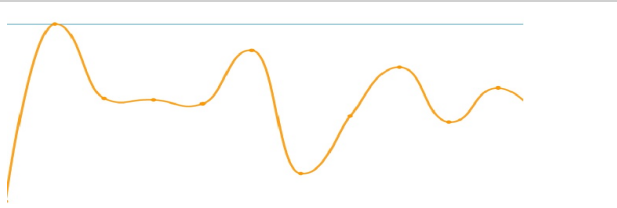
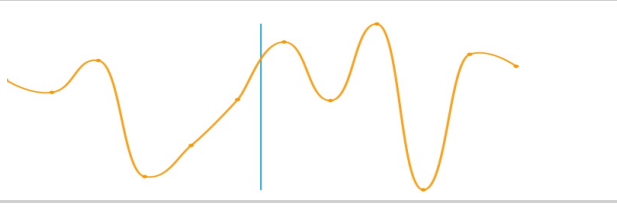
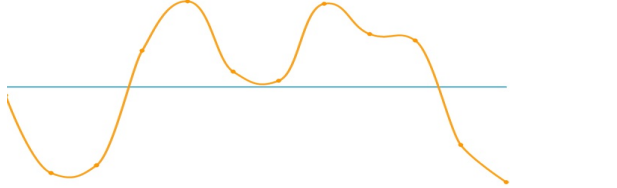
近似曲線

近似曲線は、データの傾向を示し、予測問題の検討に利用されるツールです。価格チャートや金融チャートで使用されるのが多いですが、株価のテクニカル分析で使用される取引指標、MACD(Moving Average Convergence Divergence、移動平均収束拡散法)や金融市場の解析に使用されるテクニカル指標、RSI(Relative Strength Index、相対力指数)のようなテクニカル分析グラフでも使用されます。

近似曲線の種類

以下の一覧で対応するすべてのFitTypesが説明されています。近似曲線の各種類がそれぞれの計算式に基づいて作成されます。

FitType	説明	例画像
Polynom	データが変動する場合に使用される曲線です。大量のデータで利益や損失を分析するのに採用されます。	
Exponent	データ値の増減率が高くなっていく場合に適します。0 または負の値が含まれているデータの場合、指数近似曲線を作成できません。	
Logarithmic	データの可視化に最適する曲線です。データの変化率が急速に増加または減少してから平になる場合に適します。この曲線では負の値と正の値を使用できます。	
Power	特定の比率で増加する測定を比較するデータセットの場合に適します。たとえば、超特急列車の加速を1秒間隔で計測した値など。	
Fourier	フーリエ級数式で作成され、波動関数を簡単な正弦波の組み合わせとして表示する方法です。	
Min X	グラフからXの最小値を取得して近似曲線を作成します。	

Min Y	>グラフからYの最小値を取得して近似曲線を作成します。	
Max X	グラフからXの最大値を取得して近似曲線を作成します	
Max Y	グラフからYの最大値を取得して近似曲線を作成します。	
Average X	グラフからXの平均値を取得して近似曲線を作成します。	
Average Y	グラフからYの平均値を取得して近似曲線を作成します。	

近似曲線をデザイナーまたは、ソースビューやコードでも追加できます。C1BarChartに近似曲線を追加するには、以下の手順に従います。

デザイナーの場合

以下の手順では、チャートにデータが追加されていることを前提します。詳細については、C1LineChartの[クイックスタート](#)を参照してください。

1. **LineChart** スマートタグをクリックし、[**C1LineChart タスク**]メニューから **シリーズ一覧** を選択します。
LineChartSeries コレクションエディターが表示されます。
2. **LineChartSeriesコレクションエディター**で、**<追加>**ボタンをクリックして、新しい **LineChartSeries** をLineChartに追加します。
3. 新規に追加された系列の**isTrendline**プロパティをTrueに設定します。
4. **Trendline Series**のプロパティグループを展開して以下のプロパティを設定できます。

▲ TrendlineSeries	C1.Web.Wijmo.Control
▷ Data	C1.Web.Wijmo.Contro
FitType	MaxY
Order	4
SampleCount	150

- **FitType** -近似曲線の種類を指定します。
- **SampleCount**-近似曲線の関数計算用にサンプルカウントを指定します。FitTypeがpolynom、power、exponent、logarithmicやfourierの場合のみ機能します。

LineChart for ASP.NET Web Forms

- **Order** - 多項式の次数を指定します。FitTypeがpolynom、power、exponent、logarithmicやfourierの場合のみ機能します。
5. TrendlineSeries.DataのプロパティグループにあるTrendlineSeries.Data.X.Valuesプロパティの横にある<...>ボタンをクリックして**ChartXData コレクションエディター**を開きます。
 6. X軸に表示される値を入力して、**OK**をクリックします。
 7. TrendlineSeries.DataのプロパティグループにあるTrendlineSeries.Data.Y.Valuesプロパティの横にある<...>ボタンをクリックして**ChartYData コレクションエディター**を開きます。
 8. Y軸に表示される値を入力して、**OK**をクリックします。
 9. **FitType**プロパティ、**Order**プロパティと**SampleCount**プロパティを設定します。
 10. **<OK>**をクリックして保存し、**BarChartSeriesコレクションエディター**を閉じます。

ソースビューの場合

<SeriesList></SeriesList>タグ内に以下のマークアップを追加してグラフに近似曲線を追加します。

SourceView

```
<ccl:LineChartSeries LegendEntry="True" IsTrendline="true">

  <TrendlineSeries FitType="Polynom">
    <Data>
      <X>
        <Values>
          <ccl:ChartXData StringValue="QTR1" />
          <ccl:ChartXData StringValue="QTR2" />
          <ccl:ChartXData StringValue="QTR3" />
          <ccl:ChartXData StringValue="QTR4" />
        </Values>
      </X>
      <Y>
        <Values>
          <ccl:ChartYData DoubleValue="13" />
          <ccl:ChartYData DoubleValue="4" />
          <ccl:ChartYData DoubleValue="18" />
          <ccl:ChartYData DoubleValue="8" />
        </Values>
      </Y>
    </Data>
  </TrendlineSeries>

</ccl:LineChartSeries>
```

コードの場合

Page_Loadイベントに以下のコードを追加することでチャートに近似曲線を追加できます。

C#コードの書き方

C#

```
// 新規系列を作成します。
var seriesTrendline = new LineChartSeries();
```

```

seriesTrendline.IsTrendline = true;
seriesTrendline.Label = "Trendline";
seriesTrendline.TrendlineSeries.FitType = TrendlineFitType.Polynom;
seriesTrendline.TrendlineSeries.Order = 4;
seriesTrendline.TrendlineSeries.SampleCount = 100;

// チャートに系列を追加します。
this.C1BarChart1.SeriesList.Add(seriesTrendline);

// X軸のデータを追加します。
seriesTrendline.TrendlineSeries.Data.X.Add("QTR1");
seriesTrendline.TrendlineSeries.Data.X.Add("QTR2");
seriesTrendline.TrendlineSeries.Data.X.Add("QTR3");
seriesTrendline.TrendlineSeries.Data.X.Add("QTR4");

// Y軸のデータを追加します。
seriesTrendline.TrendlineSeries.Data.Y.Add(13);
seriesTrendline.TrendlineSeries.Data.Y.Add(4);
seriesTrendline.TrendlineSeries.Data.Y.Add(18);
seriesTrendline.TrendlineSeries.Data.Y.Add(8);

```

Visual Basicコードの書き方

VB

```

' 新規系列を作成します。
Dim seriesTrendline = New LineChartSeries()
seriesTrendline.IsTrendline = True
seriesTrendline.Label = "Trendline"
seriesTrendline.TrendlineSeries.FitType = TrendlineFitType.Polynom
seriesTrendline.TrendlineSeries.Order = 4
seriesTrendline.TrendlineSeries.SampleCount = 100

' チャートに系列を追加します。
Me.C1BarChart1.SeriesList.Add(seriesTrendline)

' X軸のデータを追加します。
seriesTrendline.TrendlineSeries.Data.X.Add("QTR1")
seriesTrendline.TrendlineSeries.Data.X.Add("QTR2")
seriesTrendline.TrendlineSeries.Data.X.Add("QTR3")
seriesTrendline.TrendlineSeries.Data.X.Add("QTR4")

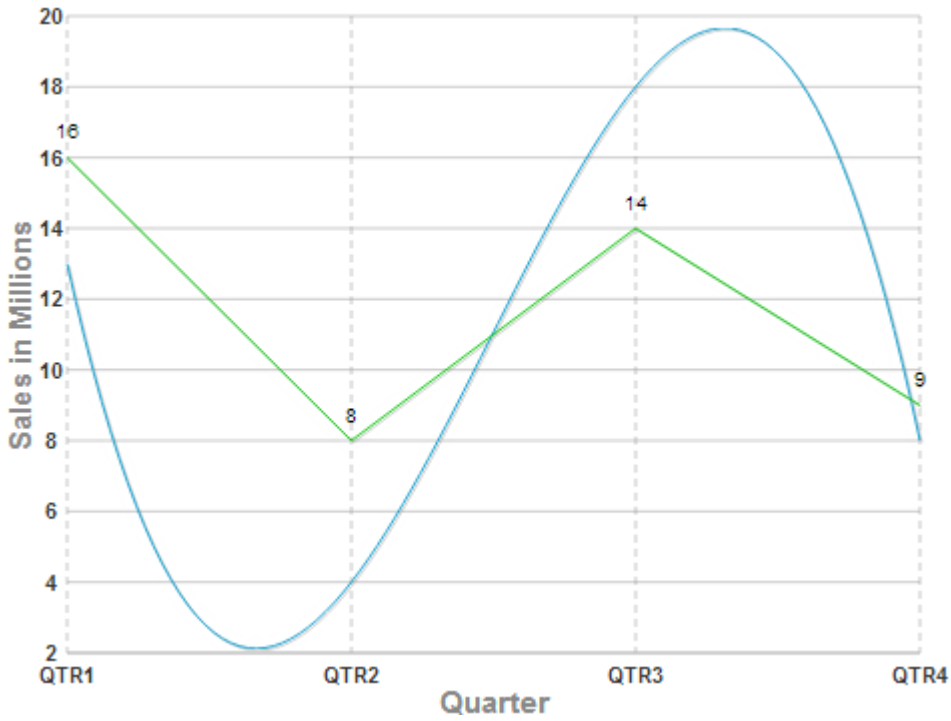
' Y軸のデータを追加します。
seriesTrendline.TrendlineSeries.Data.Y.Add(13)
seriesTrendline.TrendlineSeries.Data.Y.Add(4)
seriesTrendline.TrendlineSeries.Data.Y.Add(18)
seriesTrendline.TrendlineSeries.Data.Y.Add(8)

```

LineChart for ASP.NET Web Forms

このトピックの作業結果

プロジェクトを実行すると、チャートに緑色の近似曲線が表示されます。



データ連結

C1LineChart は、サーバー上の外部データソースからのデータの連結をサポートしています。DataBinding は、DataSourceID または DataSource と、DataBindings を設定すると可能になります。以下のプロパティは、X 値と Y 値を指定されたデータフィールドに連結するために使用されます。

- DataSourceID
- DataBindings
- C1BubbleChartBinding.XField
- C1BubbleChartBinding.XFieldType
- C1BubbleChartBinding.YField
- C1BubbleChartBinding.YFieldType
- C1BubbleChartBinding.Y1Field

DataBindings は、C1LineChartBindings のインスタンスを含むコレクションです。C1LineChartBinding は以下のプロパティを含んでいます。

- **DataMember** - このプロパティは、データソースが複数のリストを含む場合にデータのリストの名前を指定するために使用されます。
- **HintField** - このプロパティは Hint コンテンツを指定されたフィールド名にバインドするために使用されます。HintField を設定した場合、マウスが系列に移動すると、その系列と同じインデックスを持つヒント値を表示します。

エクスポートサービス

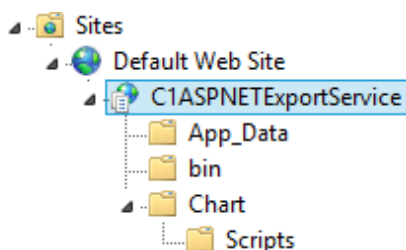
C1 ASP.NET Export Service を使用すると、複雑なエクスポートアプリケーションを作成しなくても、チャートを画像または PDF としてエクスポートできます。このサービスはアプリケーションサーバー上に置かれます。

C1 ASP.NET Export Service を使用する利点は次のとおりです。

- 書式設定を維持したままチャートをエクスポートできます。
- エクスポートされたファイルの設定を必要に応じて変更できます。

インストール

Export Service は、インターネットインフォメーションサービス(IIS)上に配布される Web アプリケーションです。C:\Program Files\ComponentOne\ASP.NET Web Forms フォルダにある **C1ASPNETExportService** インストーラを実行してください。次のファイルが IIS にインストールされます。



これらのファイルは、次の場所にも格納されています。

C:\ProgramData\ComponentOne\C1ASPNET\C1APNETExportService

システム要件

サービスホストのシステム要件は次のとおりです。

- Microsoft Windows 7 以上。
- ASP.NET 4.0 以上 (.NET Framework 4.0) を含む IIS 7.0 以上。
- サービスホスト上に IE9 以上。

 **Microsoft Windows 7** または **Microsoft Windows Server 2008 R2** では .Net Framework 4.0 をアップデートできません。詳細については、「<http://support.microsoft.com/kb/2468871>」を参照してください。

チャートのエクスポート設定

チャートをエクスポートするための設定は次のとおりです。

画像としてエクスポート

- **ファイル形式**: .jpg、.bmp、.gif、.png、または .tiff 画像としてエクスポートします。
- **作成者**: データの作成に対して責任を負う人または組織の名前を指定します。
- **サーバー URL**: サーバーの URL を設定します。"<サーバーの URL>/exportapi/chart" を入力します。
- **ファイル名**: エクスポートされる画像に使用するファイル名を設定します。

PDF にエクスポート

- **幅の自動調整**: 自動調整を有効にします。
- **横**: 横長モードを有効にします。
- **ファイルの内容**:
 - 画質: 画質を Low(低)、Medium(中)、または High(高)に設定します。
 - 圧縮: 圧縮レベルを Default(デフォルト)、None(なし)、Best Speed(速度優先)、または Best Compression(圧縮率優先)に設定します。

トピックの内容

インストール
システム要件
チャートのエクスポート設定
用途

- フォントタイプ: フォントタイプを True Type または Embedded (埋め込み) に設定します。
- **ドキュメント情報:**
 - 作成者: ドキュメントを作成した人または組織の名前を設定します。
 - 作成元アプリケーション: 元のドキュメントを作成したアプリケーションの名前を設定します。
 - サブタイトル: ドキュメントのサブタイトルを設定します。
 - タイトル: タイトルバーに表示されるドキュメントのタイトルを設定します。
 - PDF 作成アプリケーション: PDF ドキュメントを作成したアプリケーションの名前を設定します。
 - キーワード: PDF ドキュメントに関連付けられるキーワードを設定します。これらのキーワードは、ドキュメントの検索に使用できます。
- **ドキュメントのセキュリティ:**
 - 暗号化の種類: 暗号化の種類を NotPermit (許可しない)、Standard40、Standard128、または Aes128 に設定します。
 - 所有者パスワード: ドキュメントの権限を編集するために必要なパスワードを設定します。
 - ユーザーパスワード: ドキュメントを開くために必要なパスワードを設定します。
 - コンテンツのコピーを許可: コンテンツのコピーを許可または禁止します。
 - 注釈の編集を許可: ユーザーが注釈を編集することを許可または禁止します。
 - コンテンツの編集を許可: ユーザーがドキュメントの内容を編集することを許可または禁止します。
 - 印刷を許可: ドキュメントの印刷を許可または禁止します。
- **構成設定:**
 - サーバー URL: サーバーの URL を設定します。"<サーバーの URL>/exportapi/chart" を入力します。
 - ファイル名: エクスポートされる PDF に使用するファイル名を設定します。

用途

チャートを画像または PDF としてエクスポートするには、`exportChart` メソッドを呼び出します。チャートを画像にエクスポートするには、`<head>` タグと `</head>` タグの間に次のコードを追加します。

```
<script src="http://code.jquery.com/jquery-1.9.1.min.js" type="text/javascript">
</script>
<asp:PlaceHolder runat="server">
  <!--Export メソッド-->
  <script type="text/javascript">
    $(function () {
      $("#Button1").click(exportImage);
    });
    function getChart() { return $("#<%=C1LineChart1.ClientID%>"); }

    // Export 関数
    function exportImage() {
      var fileName = "ExportImage";
      var type = "Png";
      var url = "http://demos.componentone.com/ASPNET/ExportService" + "/exportapi/chart";
      var chart = getChart();
      $("#<%=C1LineChart1.ClientID%>").c1linechart("exportChart", fileName, type, url);
    }
  </script>
</asp:PlaceHolder>
```

チャートを PDF にエクスポートするには、`<head>` タグと `</head>` タグの間に次のコードを追加します。

```
<script src="http://code.jquery.com/jquery-1.9.1.min.js" type="text/javascript">
</script>
<asp:PlaceHolder runat="server">
<script type="text/javascript">
  $(function () {
    $("#Button1").click(exportPdf);
  });
```

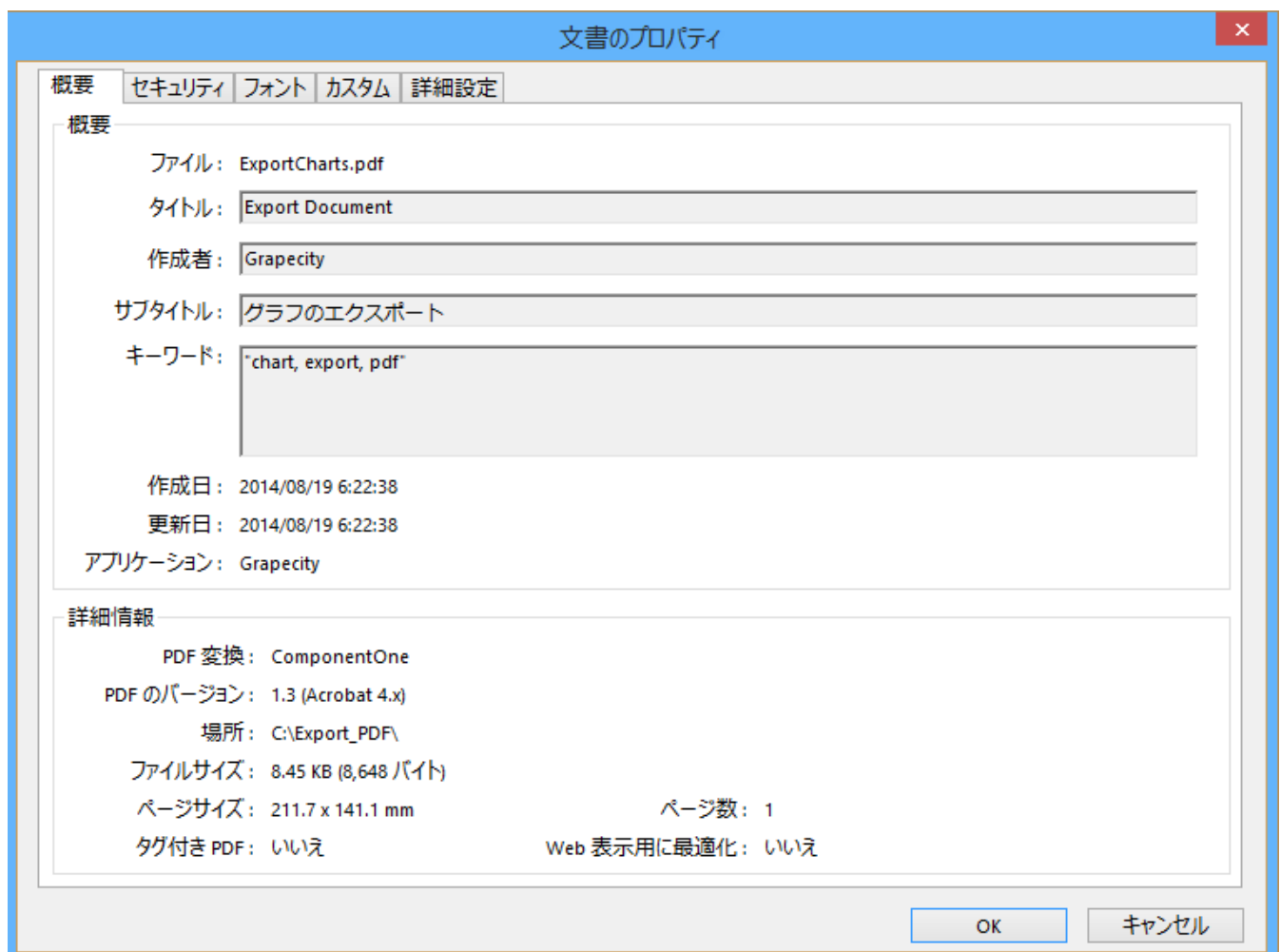
LineChart for ASP.NET Web Forms

```
// Export function
function exportPdf() {
var fileName = "ExportCharts";
var url = "http://demos.componentone.com/ASPNET/ExportService" + "/exportapi/chart";
var pdfSetting = {
    imageQuality: 'Low',
    compression: 'BestCompression',
    fontType: 'TrueType',
    author: 'Grapecity',
    creator: 'Grapecity',
    subject: 'グラフのエクスポート',
    keywords: 'chart, export, pdf',
    allowCopyContent: true,
    allowEditAnnotations: true,
    allowEditContent: true,
    allowPrint: true
}

$("#<%=C1LineChart1.ClientID%>").c1linechart("exportChart", fileName, "pdf",

    pdfSetting, url);
}
</script>
</asp:Placeholder>
```

生成される PDF のプロパティを次の図に示します。



ファイルが Internet Explorer でダウンロードされない場合は、Internet Explorer の保護モードをオフにしてファイルをエクスポートするか、Internet Explorer を管理者として実行します。保護モードをオフにするには、次のようになります。

- Internet Explorer の[設定]を開き、[インターネットオプション]を選択します。
- [セキュリティ]タブで、[インターネット]を選択し、[保護モードを有効にする]のチェックを外します。

C1LineChart のアニメーション

C1LineChartの系列は、**Duration** プロパティと **Easing** プロパティを使用してアニメーション化できます。

遷移効果

Enabled プロパティが true のとき、アニメーション効果を折れ線グラフの系列に適用できます。アニメーション化されたスライド状態／フェード状態の間に遷移効果を追加すれば、それらの状態間にシームレスな流れが生まれ、折れ線グラフの魅力を高めることができます。ロード時に左から右へスムーズに移動する折れ線グラフ系列の代わりに、折れ線グラフを系列のスライドイン時にバウンドインさせ、系列のスライドアウト時にバウンドアウトさせることができます。デフォルトでは、**Easing** プロパティは **EaseLinear** に設定され、棒グラフをリロードすると、各系列はスムーズで直線的な遷移効果でリロードされます。

以下の遷移効果は、状態間の遷移をアニメーション化するために使用できます。これにより、折れ線グラフ系列をロードする際にユーザーにとって動きがスムーズに見えます。

遷移の名前	遷移の説明
EaseInBack	バックのイーasingイン。開始は遅く、それから加速します。
EaseInCubic	3次型のイーasingイン。開始は速度ゼロで、それから加速します。
EaseInOutCubic	3次型のイーasingインとイーasingアウト。開始は速度ゼロで、途中まで加速し、それから再び速度ゼロまで減速します。
EaseOutBack	バックのイーasingアウト。開始は速く、それから減速します。
EaseOutBounce	バウンドしながらのイーasingアウト。開始は速く、それから減速します。バウンドの回数は持続時間に関係します。持続時間が延びれば、バウンドの回数は多くなります。
EaseOutCubic	3次型のイーasingインとイーasingアウト。開始は全速で、それからゼロまで減速します。
EaseOutElastic	5次型のイーasingアウト。開始は全速で、それからゼロまで減速します。

アニメーション効果の持続時間

CLineChart のアニメーション効果の長さは、**ChartAnimation.Duration** プロパティを使用して設定できます。アニメーション効果の持続時間の指定に使用される時間の単位はミリ秒であり、**Duration** プロパティのデフォルト設定値は **500** ミリ秒(0.5秒)です。アニメーション効果を長くするにはこの値を増加させ、短くするにはこの値を減少させます。