Visual Studio Do-It-Yourself シリーズ

第 2 回 レイアウト

Microsoft®

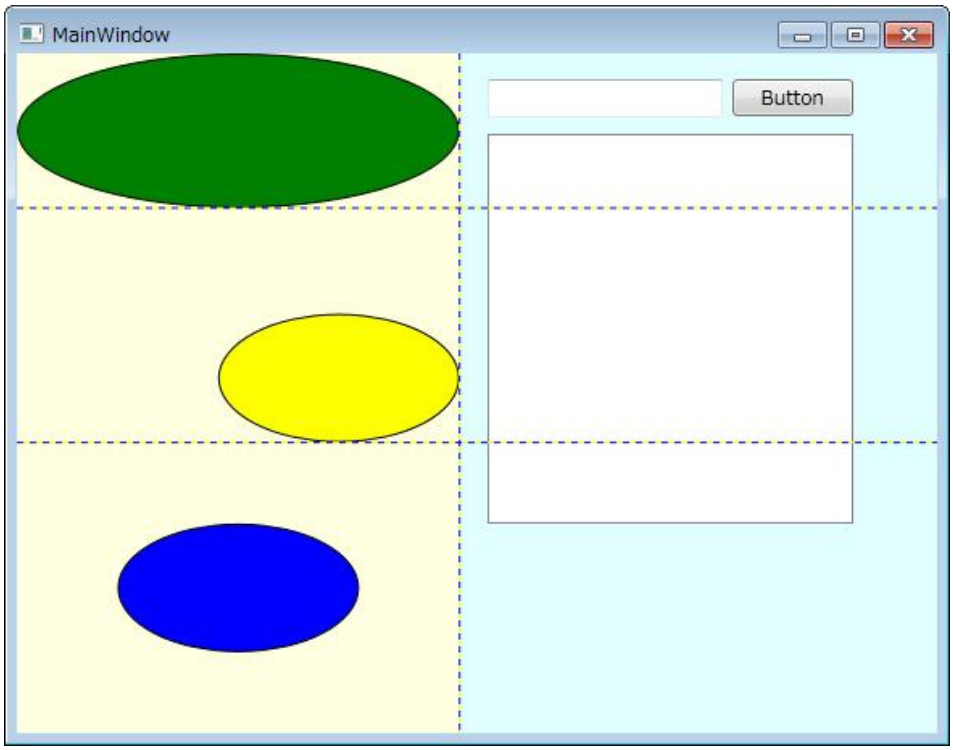
Visual Studio Do-It-Yourself
第 2 回 レイアウト

今回は、Windows アプリケーションを作成する際に基本となるレイアウトについて学びます。

ここでは、次のことを学習します。

- レイアウトの基本
- 各種のレイアウト
- レイアウトを組み合わせる

今回は、レイアウトの学習を通じて次のようなアプリケーションを作成します。

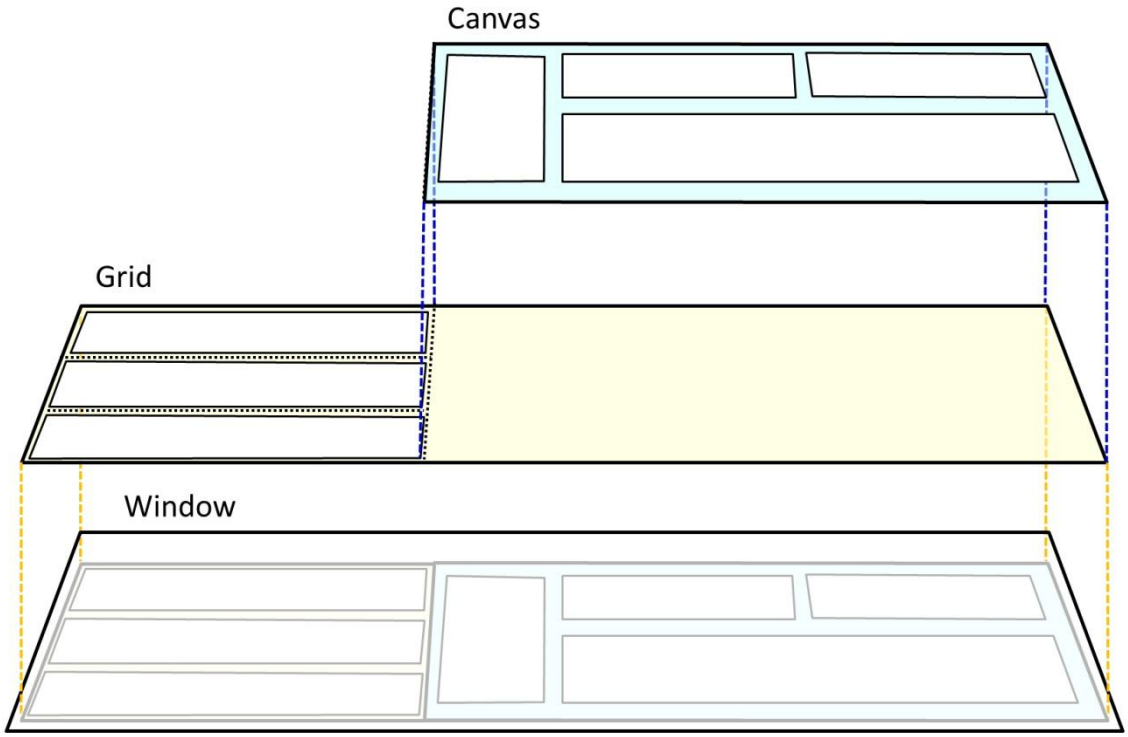


■レイアウトの基本

Visual Studio 2010 で Windows アプリケーション（WPF アプリケーション）を作成する場合、ウィンドウの上に、コントロールと呼ばれる部品を配置して、ユーザー インターフェイスを作り上げていきます。実はウィンドウ（Window）に直接配置できるコントロールは、ひとつだけです。しかし、第 1 回で Button と TextBox の 2 つのコントロールを配置したように、実際にはいくつものコントロールを配置しています。この複数のコントロールを置ける仕組みを提供しているものがレイアウト パネルと呼ばれるコントロールなのです。

レイアウトという言葉からは配置や間取りなどを想像できますが、「コントロールをどのように配置するか」を決めるのがレイアウト パネルの役割です。ただし、大きさが決まった部屋の間取りを考える場合と異なり、ウィンドウは大きさや条件が変わるため、その変化に合わせてどのようにコントロールの位置を調整するかを考えなければなりません。そのため、レイアウト パネルにはいくつかの種類が用意されています。

個々のレイアウトについては後述しますが、レイアウトは組み合わせて使うことができます。次の図は、ウィンドウ（Window）の上に、Grid レイアウトを配置し、その領域分割した部分にコントロールを載せたり、別のレイアウトである Canvas を載せたりしている様子です。



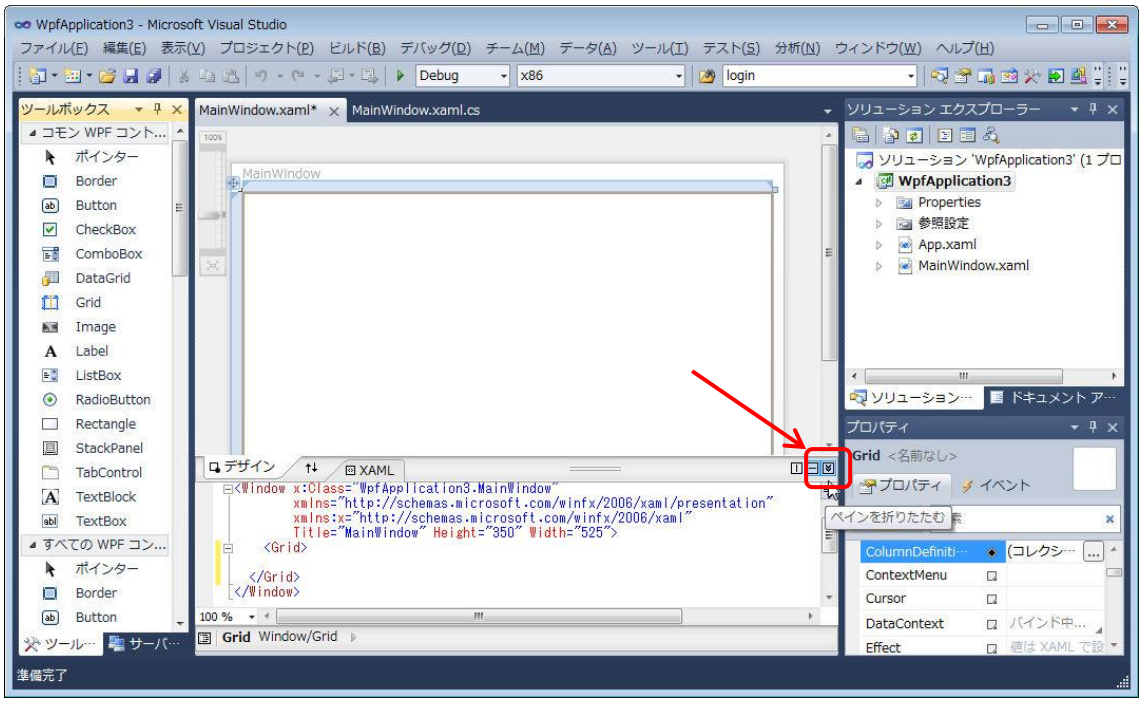
複数のコントロールを使いやすく配置するためには、レイアウト パネルをうまく活用することがポイントになります。

■各種のレイアウト

ここでは、以下の表に示す主要な 5 つのレイアウト パネルについて解説します。

レイアウト パネル	説明
Grid	格子状に領域を分割し、それぞれの領域にコントロールを配置する
Canvas	左上を基準とする座標値をもとにコントロールを配置する
StackPanel	コントロールを積み重ねる（縦または横に並べる）
WrapPanel	StackPanel と同様にコントロールを並べるが、収まりきらない場合に折り返す
DockPanel	上下左右の辺にコントロールをドッキングさせる

第 1 回の手順にしたがって、新しい「WPF アプリケーション」を作成した場合、ウィンドウには Grid レイアウトが配置されています。ユーザー インターフェイスは XAML という記述言語でもあらわされていますが、ここでは使わないので、下図の赤い矢印の先にある閉じるボタン（ペインを折りたたむ）をクリックしてください。

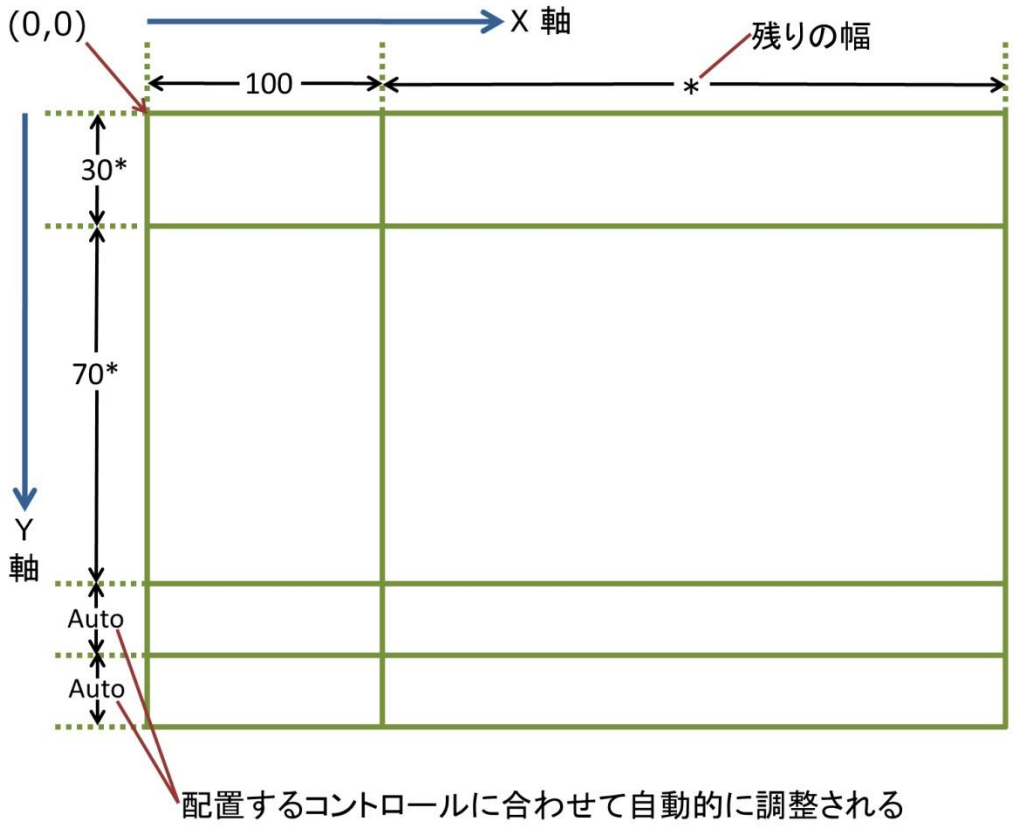


別のレイアウト パネルを使うときは、今、配置されているレイアウト パネルを削除して、新しいレイアウト パネルを配置します。レイアウト パネルを削除すると、そのパネル上に配置していたコントロールはすべて削除されるのでご注意ください。間違って削除した場合は、[編集]－[元に戻す]（または [Ctrl]+[Z]）で削除を取り消したり、[編集]－[やり直し]（または [Ctrl]+[Y]）で再実行したりできます。

配置したコントロールを残したい場合は、それらを選択してクリップボードにコピー（[編集]－[コピー] または [Ctrl]+[C]）して、パネルを入れ替えた後で貼り付け（[編集]－[貼り付け] または [Ctrl]+[V]）するか、XAML テキストを編集することができます。

●Grid

Grid レイアウトは、領域を縦横の格子状に分割して、それぞれの枠を基準にコントロールを配置するためのレイアウトパネルです。これはデフォルトのレイアウトであり、もっともよく使われるものです。次の図は、Grid を横に 2 分割、縦に 4 分割したイメージを示しています。

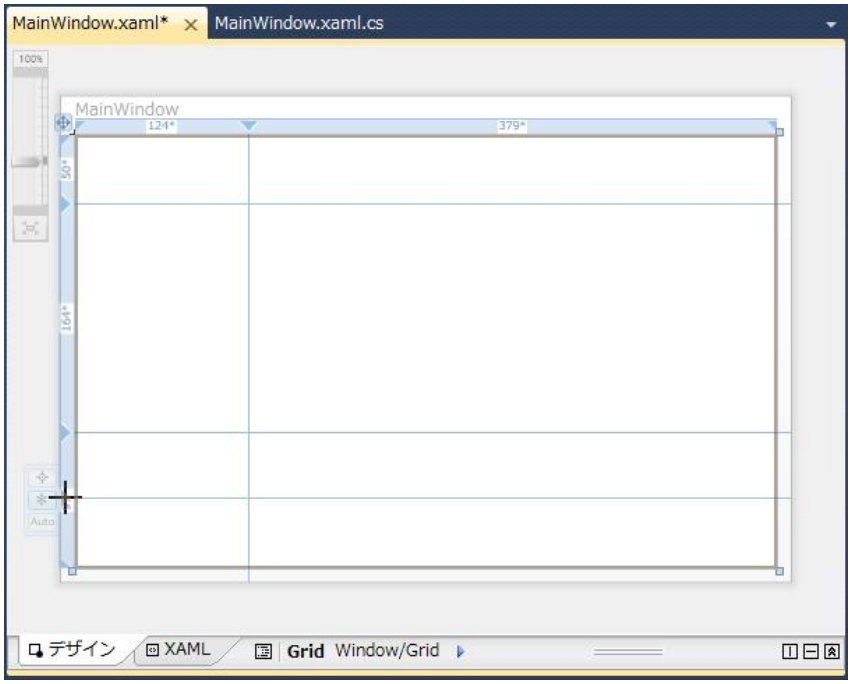


Grid レイアウトでは、分割された領域の長さ（幅と高さ）を決めるために、「固定」「スター」「自動」という設定があります。それぞれ次のような意味があります。

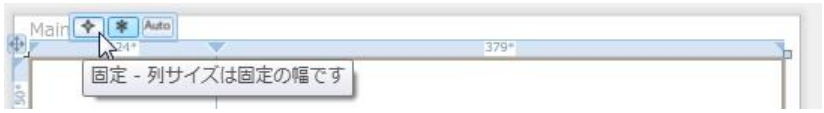
	表記	意味
固定	(数値)	数値で指定された長さ（幅または高さ）が確保される
スター	*、(数値)*	固定や自動で確保された残りを、スターの直前の数値で加重比率した長さ（数値が指定されていない場合は 1）
自動	Auto	配置されているコントロールが必要とするサイズに自動調整される

上図において、横方向は左から順に「100 という固定長」と「残りの長さ」の 2 つに分割されています。縦方向は上から順に「高さ全体から下 2 行を除いた長さのうちの 30%」と「同じく、その 70%」、「配置されているコントロールで自動調整される高さ」の行が 2 つの、計 4 つに分割されています。Grid レイアウトを配置しているウィンドウなどの大きさが変わると、この規則にしたがって分割場所が変わります。

実際に、Visual Studio で領域を分割してみましょう。ウィンドウ（MainWindow）の内側の細い線で囲まれた領域（クライアント領域）をクリックすると Grid を選択できます。このとき、上辺と左辺に表示されるバーをクリックすると、その位置で領域が分割されます。この様子を以下に示します。



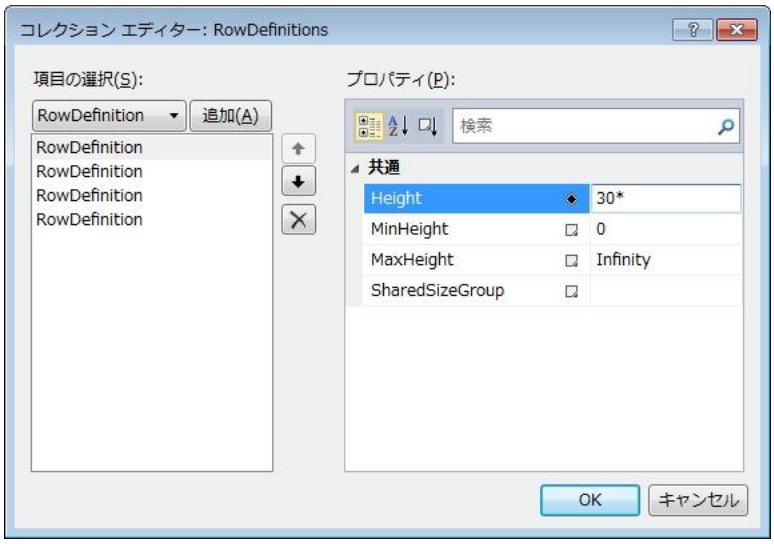
分割線はデザイン用のガイドであり、実行するときには表示されません。分割した領域のそれぞれは、最初は「スター」（加重比率）の設定になっています。上辺の場合は、それぞれの区切りの中央から上に、左辺の場合は中央から左にマウスカーソルをずらすことで、「固定」「スター」「自動」を切り替えるボタンが表示されます（下図）。ここでは、左側の「固定」を選びます。すると、小さな文字で「(数字)*」と表示されていたものが「(数字)」だけになることがわかります。



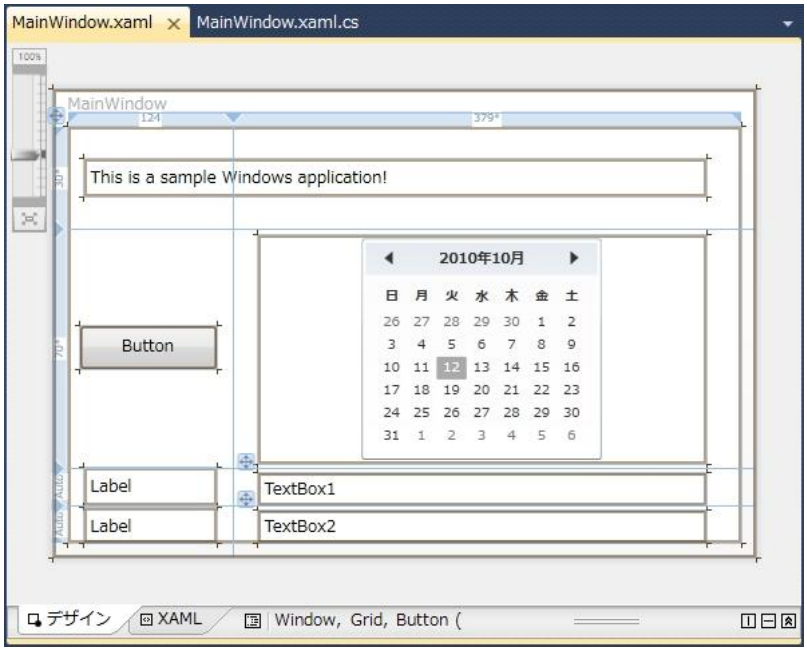
同じ手順で、左辺の区切りの下の 2 つを「自動」に変更できますが、まだ変更しないでください。コントロールを何も配置していない状態で「自動」を設定すると、サイズが 0 になってしまい、その領域にコントロールを配置できなくなるためです。

どのように分割するかは、プロパティ ウィンドウからも変更できます。列の定義は ColumnDefinitions、行の定義は RowDefinitions というプロパティであらわされます。いずれも「(コレクション)」と表示されている右側の […] ボタンをクリックすると、コレクション エディターというダイアログが表示されます。ここで、それぞれの分割領域の詳細を設定できます。

たとえば、RowDefinitions プロパティでは次のようなダイアログが表示されます。ここで、Height プロパティを設定すれば、各領域の高さをどのように調整するかを決められます。また、MinHeight（高さの最小値）や MaxHeight（高さの最大値）を設定しておけば、その領域の変動の範囲を制限できます。



実際に、いくつかのコントロールを配置してみましょう。ここでは Label や Button、Calendar、TextBox を配置しています。コントロールを配置するときは、どの領域に配置されるかがハイライトされて示されます。縦や横に領域をまたがって配置することもできます。コントロールを移動したり、サイズを変更したりするときには、スナップ線やガイド用の線が表示されるので、位置合わせも容易です。コントロールを配置したら、Grid の 2 つの行を自動（Auto）に変更しておきます。



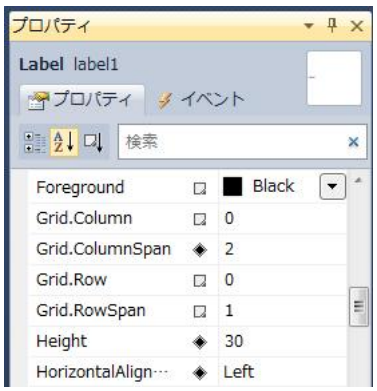
コントロールを選択しているときに [Esc] キーを押すと、そのコントロールが配置されているパネル、あるいはパネルを配置しているウィンドウを選択できます。ウィンドウの外側をクリックすると、メイン ウィンドウ（MainWindow）を選択できます。ウィンドウを選択した状態で、サイズを変更して、分割領域がどのように変わるかを確かめてみてください。

Grid に配置するコントロールのプロパティには、レイアウトに大きく影響するものがあります。これらのプロパティは、Grid 以外のレイアウトでも重要な意味を持つことがあります。

プロパティ	意味
Width	コントロールの幅。Auto を選択すると、自分が保持するコンテンツに応じて自動的に調整される。
Height	コントロールの高さ。Auto を選択すると、自分が保持するコンテンツに応じて自動的に調整される。
HorizontalAlignment	Left（領域の左側）、Center（領域の左右中央）、Right（領域の右側）、Stretch（領域の幅全体）のいずれか。
VerticalAlignment	Top（領域の上部）、Center（領域の上下中央）、Bottom（領域の下部）、Stretch（領域の高さ全体）
Margin	領域の各辺までの距離（すき間）。左、上、右、下の順で数値を指定。

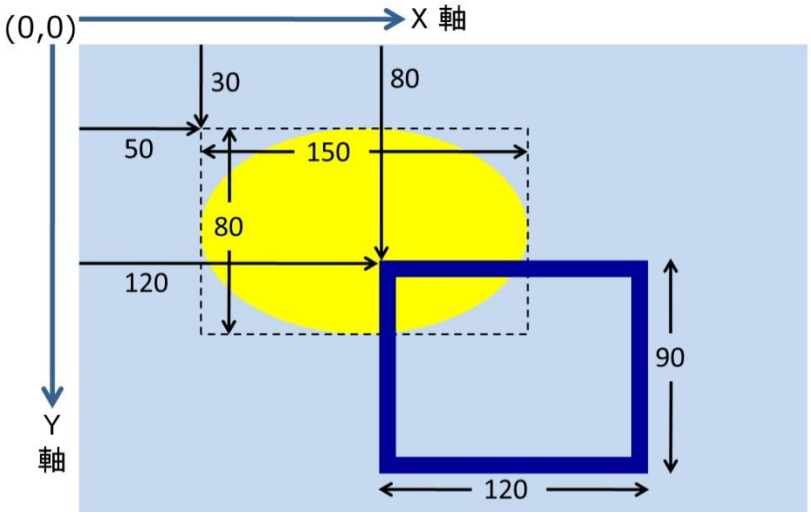
コントロールを配置すると、HorizontalAlignment は Left に、VerticalAlignment は Top になり、配置したコントロールの大きさに合わせて Margin プロパティが設定されます。つまり、枠の左上を基準にコントロールが移動し、右下を基準にサイズが変更されます。この設定で不都合があれば、それぞれのプロパティを修正してください。たとえば、コントロールを領域全体に広げるときは、Margin をすべて 0 にし、Width と Height を Auto に、HorizontalAlignment と VerticalAlignment を Stretch に設定します。

コントロールが複数の領域をまたいで配置されているときは、それらの領域を合わせた部分に対して作用します。特定の領域だけに限定するときは、いったんマウスでコントロールを領域に収めた後で、プロパティを変更してください。Grid に配置したコントロールは、プロパティ ウィンドウに Grid.Column（列の位置）、Grid.ColumnSpan（使用する列の数）、Grid.Row（行の位置）、Grid.RowSpan（使用する行の数）が表示されるので、これを使って設定することもできます。



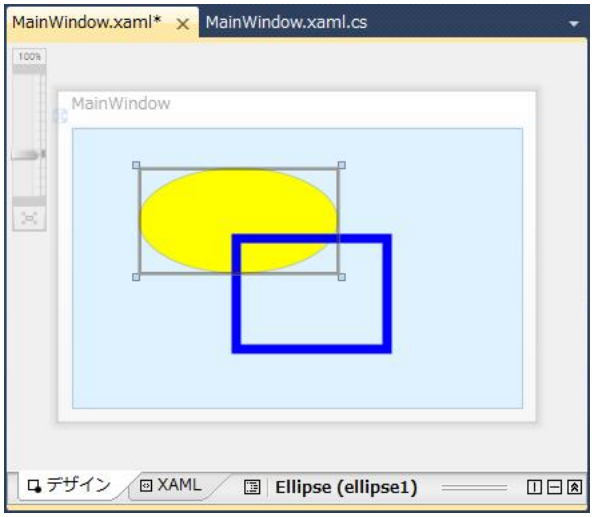
●**Canvas**

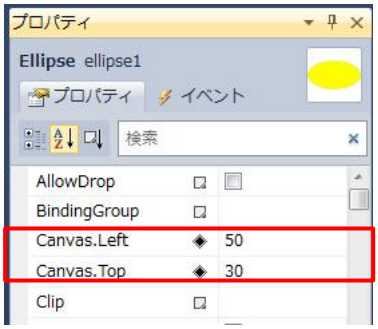
Canvas レイアウトは、単純なレイアウトです。どのコントロールも左上を基準にした位置と指定された大きさで配置されます。Canvas レイアウトのイメージを次の図に示します。



コントロールは左上を基準にしているので、ウィンドウの大きさを変更しても左上からの相対位置は変わりません。大きさが変更することもないのでわかりやすいのですが、ウィンドウを縮めた時にはコントロールが収まりきらないことが、ウィンドウを広げた時には余白が増えてしまうことが生じやすくなります。Canvas レイアウトは、配置する範囲が固定の場合に使います。

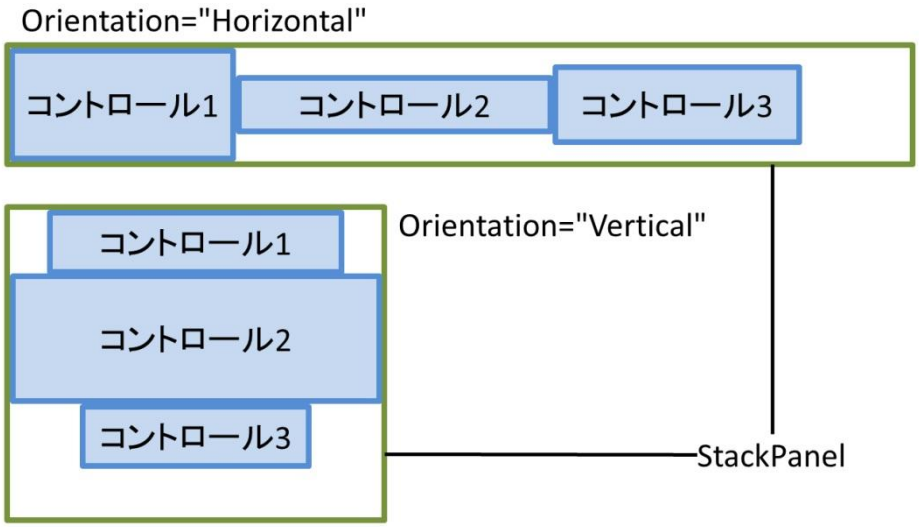
実際に Visual Studio でコントロールを配置してみましょう。Grid が配置されているときは、いったん削除して、Canvas コントロールを配置してください（Canvas の Width と Height プロパティは削除するか Auto にしておくとよいでしょう）。ここでは Ellipse と Rectangle を配置しています。Canvas にコントロールを配置すると、プロパティ ウィンドウに Canvas.Left と Canvas.Top というプロパティが表示され、ここでも座標値を設定できます。





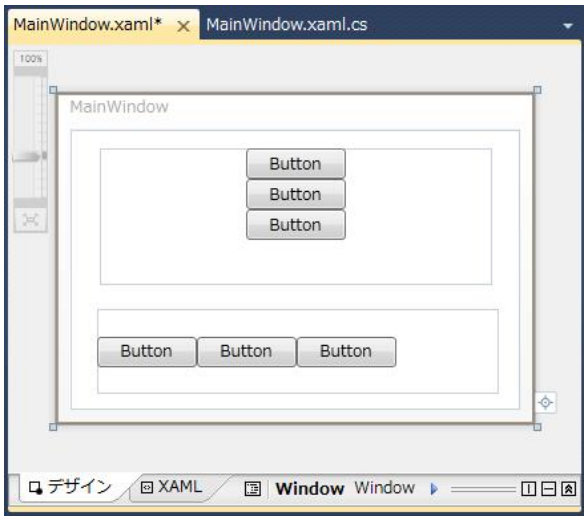
●StackPanel

スタック（Stack）とは「積み重ねる」という意味ですが、実際には縦または横にコントロールを並べて配置します。(通常、下から上に積み重ねることはできません)。スタックパネルのイメージを次に示します。



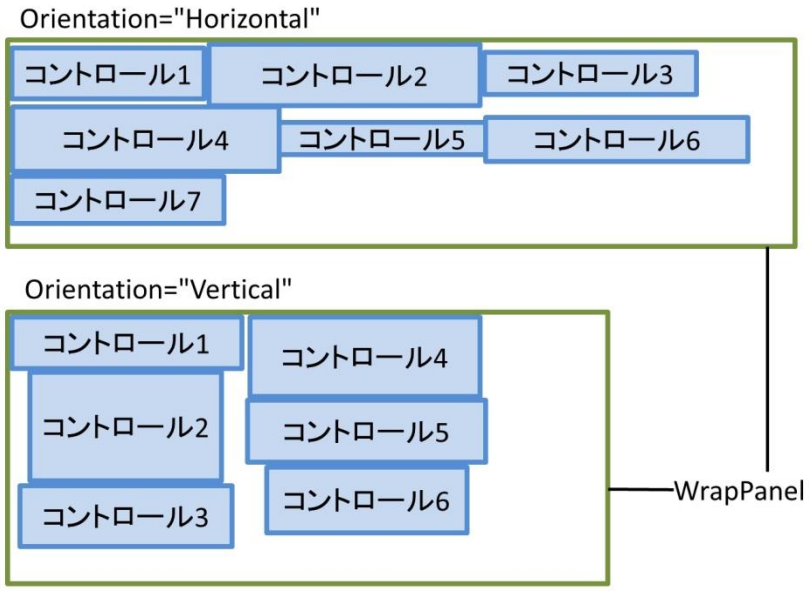
StackPanel の Orientation プロパティが、そこに配置するコントロールを縦に並べるか、横に並べるかを指定します（デフォルトでは縦）。コントロールを配置する際に大きさを指定することはできますが、位置は指定できません。すでに置かれているコントロールの隣（下または右）に配置されます。コントロールをドラッグして順序を変更することはできます。

実際に Visual Studio でコントロールを配置してみましょう。ここでは、Grid 上に 2 つの StackPanel を配置して、それぞれ Orientation を Vertical と Horizontal に設定します。それぞれの StackPanel に Button コントロールを 3 つずつ配置すると次の図のようになります。



●WrapPanel

WrapPanel は StackPanel に似たコントロールですが、左上隅から順に縦または横方向に並べていき、並べる方向でコントロールが表示しきれなくなったときに、配置する位置を折り返して、改めて上または左から並べます。WrapPanel のイメージを次に示します。

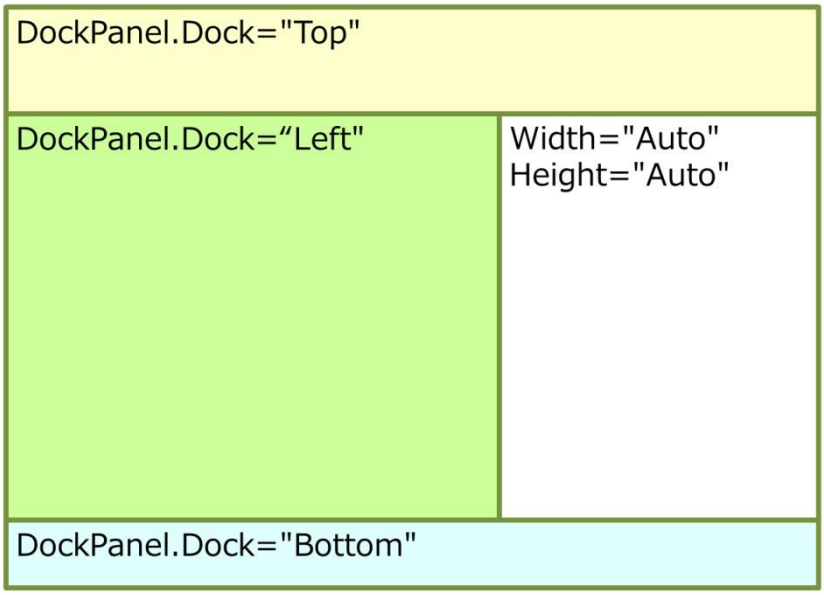


WrapPanel の Orientation は、Horizontal（水平方向）がデフォルトです（StackPanel とは異なります）。コントロールの大きさが揃っていない場合、それぞれのコントロールの VerticalAlignment または HorizontalAlignment の設定に従って、センタリングするか、上下または左右に寄せて表示されます。

実際に Visual Studio でコントロールを配置してみましょう。ここでは、Grid 上に 2 つの WrapPanel を配置して、それぞれ Orientation を Horizontal と Vertical に設定します。それぞれの WrapPanel に Button コントロールを 7 つずつ配置すると次の図のようになります。

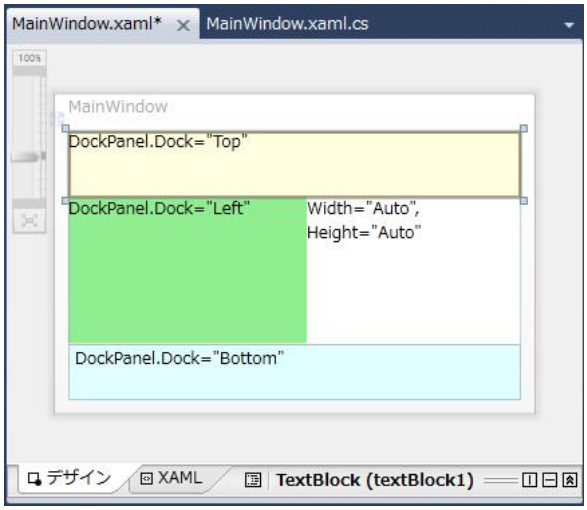
●DockPanel

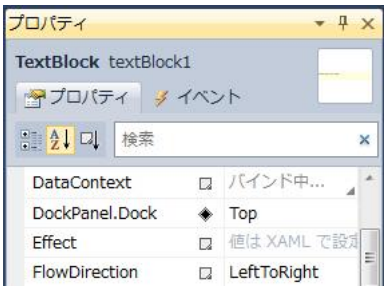
DockPanel は配置するコントロールを上下左右の辺にドッキングさせるパネルです。DockPanel のイメージを次に示します。



実際に Visual Studio でコントロールを配置してみましょう。Grid が配置されているときは、いったん削除して、DockPanel コントロールを配置してください（DockPanel の Width と Height プロパティは削除するか Auto にしておくとよいでしょう）。DockPanel にコントロールを配置すると、プロパティ ウィンドウに DockPanel.Dock プロパティが表示されます。このプロパティを使って、どの辺にドッキングするかを指定します。

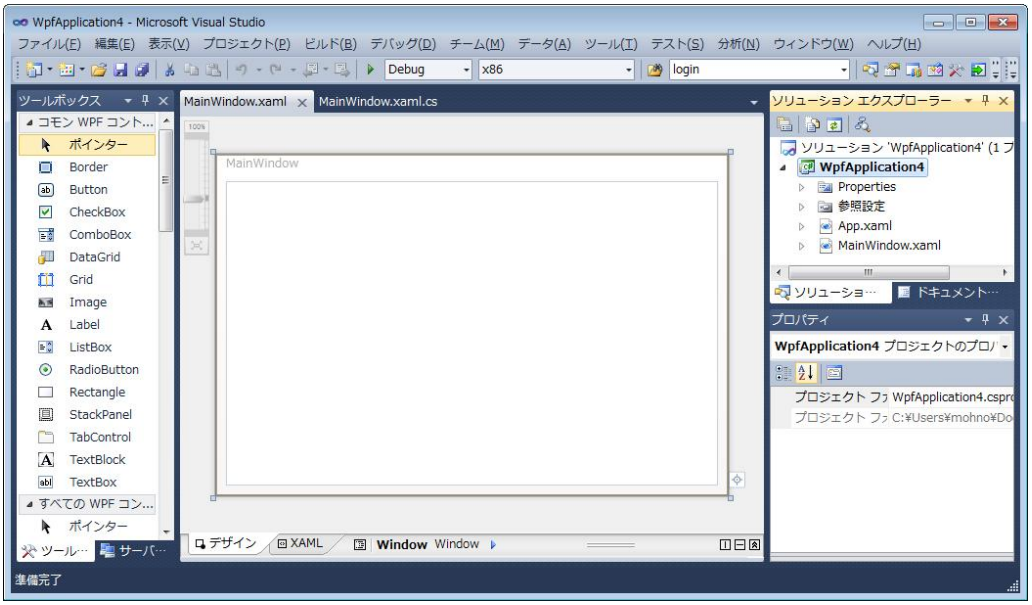
ここでは、4 つの TextBlock コントロールを配置し、それぞれ DockPanel.Dock プロパティを Top、Bottom、Left、Right に設定しています。また、最初の 2 つは Width を Auto に、次の 1 つは Height を Auto に、最後のひとつは Width/Height ともに Auto に設定しています。プロパティの設定を色々変えて試してみてください。



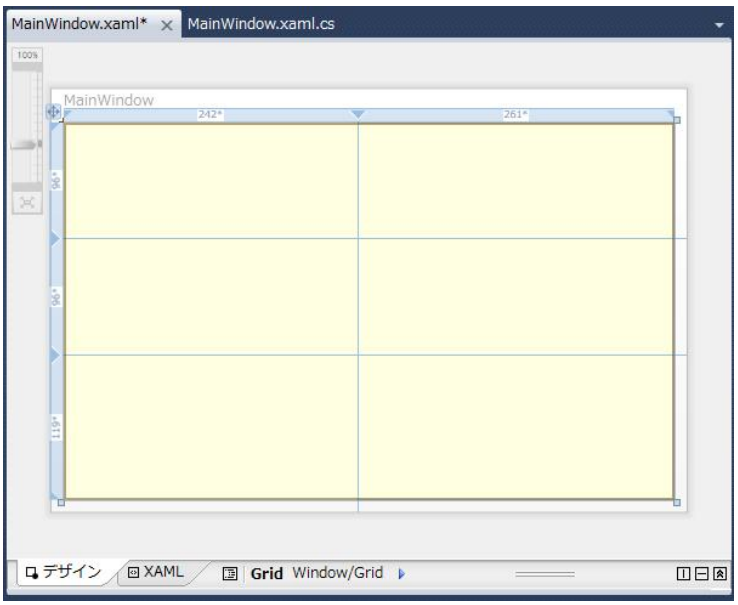


■レイアウトを組み合わせる

今回ご紹介したレイアウト パネルを組み合わせて使ってみます。まず、Visual Studio 2010 で、新しい WPF アプリケーションを作成します。（XAML ペインは閉じています）



中央部をクリックして、Grid を選び、Background プロパティを LightYellow にし、さらに ShowGridLines プロパティをチェックしてください。また、上辺の真ん中あたりをクリックして領域を左右に分割して、さらに左辺を 3 等分してください。WPF デザイナーは次のようになります。



また、左辺の 3 つの分割領域のうち、上のひとつについて小さく表示されている数字にマウスカーソルを移動させ、さらに左に表示される 3 つのボタンのうち上の「固定」ボタンを選んでください。これで、この領域の高さが固定されました。

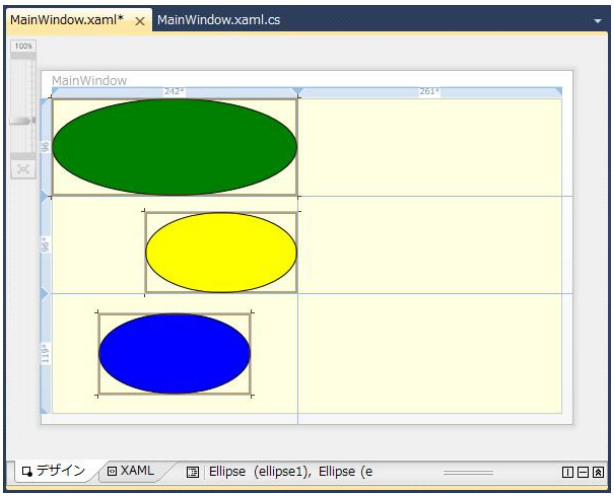
左側の 3 つの領域には、それぞれ Ellipse を配置してください。プロパティは次のように設定します。

ellipse1 のプロパティ	値	備考
Margin	0	0 だけを入力すると 4 つの値が 0 になります。
HorizontalAlignment	Stretch	
VerticalAlignment	Stretch	
Width	削除	Auto と同じです。
Height	削除	Auto と同じです。
Fill	Green	わかりやすくするために色を指定します

ellipse2 のプロパティ	値	備考
Margin	0	0 だけを入力すると 4 つの値が 0 になります。
HorizontalAlignment	Right	
VerticalAlignment	Bottom	
Width	150	領域に収まる範囲の幅
Height	80	領域に収まる範囲の高さ
Fill	Yellow	わかりやすくするために色を指定します

ellipse3 のプロパティ	値	備考
Margin	0	0 だけを入力すると 4 つの値が 0 になります。
HorizontalAlignment	Center	
VerticalAlignment	Center	
Width	150	領域に収まる範囲の幅
Height	80	領域に収まる範囲の高さ
Fill	Blue	わかりやすくするために色を指定します

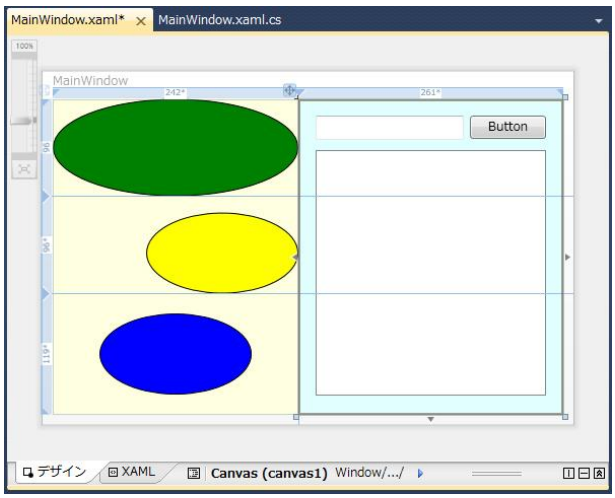
これで、WPF デザイナーは、次のようになります。



次に、ツールボックスから Canvas コントロールを選び、上図の右半分の 3 つの領域を覆うように配置します。このとき、3 つの領域にまたがっていれば、正確に配置する必要はありません。配置した Canvas のプロパティを次のように設定します。

プロパティ	値	備考
Margin	0	0 だけを入力すると 4 つの値が 0 になります。
HorizontalAlignment	Stretch	
VerticalAlignment	Stretch	
Width	削除	Auto と同じです。
Height	削除	Auto と同じです。
Background	LightCyan	わかりやすくするために色を指定します

これで、Canvas は正確に Grid の右半分をカバーするようになります。Canvas 上に、いくつかコントロールを配置します。次の図では、TextBox、Button、ListBox を配置しています。



できあがったら、[デバッグ開始（ ）] ボタンを押して、アプリケーションを実行させてください。最初の図に示した通り、Grid レイアウトのグリッド線が表示されているので、ウィンドウの大きさを変えると、それぞれの領域やコントロールの位置と大きさがどのように変化するかを確認できます。

■まとめ

今回は、ユーザー インターフェイスの基盤となるレイアウトについて学習しました。今回ご説明したレイアウト以外にも UniformGrid のように、あまり使われないレイアウト パネルもあります。また、実際にユーザー インターフェイスを作り上げていくためには、その他のコントロールの知識も必要です。

次回は、各種のコントロールについて学習します。