如何使用拆分

WinForms版版True DBGrid拥有类似于窗口性能的拆分，如Microsoft Excel和Word产品，在多个水平或垂直面板中使用拆分来显示数据，这些面板或拆分可以用不同颜色和字体来显示数据，拆分可以以一个单元或单独模块实现滚动，也可以显示不同的列集或相同的集，也可以禁止拆分对一个或多个列或行集合滚动，不像其他网格产品，WinForms版版True DBGrid
的固定（非滚动）列或行不必要在网格左侧开始，但它可以右侧或者中间，固定行和列的多个组可以在一个网格存在，拆分可以为您的应用程序提供数据呈现的不同可能。
无论您何时使用WinForms版版True DBGrid，您总会使用到拆分，网格包含至少一个水平拆分，拆分属性被设置为默认值，因而拆分的属性可以被忽略直到它被需要，因此如果您不需要在一个网格中创建和操纵多个拆分，你可以跳过此章。
通过使用Split 对象和SplitCollection 对象创建和操纵拆分，因为一个单独的列只能在一个拆分中可见，而在其他拆分中隐藏，每个Split 对象包含它自己的列集，即C1DisplayColumnCollection，该集合提供了每个拆分的外观及他所包含的列的外观。

引用拆分及其属性

C1TrueDBGrid 对象初始包含一个水平拆分，如果创建了一个额外的拆分，您可以使用网格的SplitIndex属性决定或设置当前的拆分(即聚焦的拆分):
To write code in Visual Basic

Visual Basic

```
’ 从0号索引读取当前的拆分。
Dim idx as Integer = Me.C1TrueDBGrid1.SplitIndex


’ 设置索引变量的聚焦拆分。
Me.C1TrueDBGrid1.SplitIndex = idx
```

To write code in C#

C#

```
// 从0号索引读取当前的拆分。
int idx = this.c1TrueDBGrid1.SplitIndex;


// 设置索引变量的聚焦拆分。
this.c1TrueDBGrid1.SplitIndex = idx;
```

同一数据源的每个拆分都有不同的视图，具有同独立网格相同的行为，如果不使用自定义拆分属性创建额外的拆分，所有的拆分将全部相同且每一个拆分的行为与原始网格将会非常类似。
注意一些属性，如RecordSelectors 和MarqueeStyle在C1TrueDBGrid和Split对象中均支持，三个主要规则应用于属性且对于网格和它的拆分是常规的：

1. 当您设置或获取Split 对象的属性时，您可以访问一个具体的拆分，而在同一个网格的其他拆分不会受到影响。
2. 当您获取C1TrueDBGrid 对象的属性时，您可以访问当前拆分中的同一属性。
3. 当您设置C1TrueDBGrid 对象的一个属性时，您可以对所有拆分中的同一属性进行设置。

为了理解如何在代码中使用这些规则，下述示例演示了一个带有两个水平拆分的网格，且当前拆分索引为1，为了确定当前使用的选取框样式，请参考以下声明:
To write code in Visual Basic

Visual Basic

```
marquee = Me.C1TrueDBGrid1.MarqueeStyle marquee = Me.C1TrueDBGrid1.Splits(1).MarqueeStyle marquee =
Me.C1TrueDBGrid1.Splits(Me.C1TrueDBGrid1.SplitIndex).MarqueeStyle
```

To write code in C#

C#

```
marquee = this.c1TrueDBGrid1.MarqueeStyle; marquee = this.c1TrueDBGrid1.Splits[1].MarqueeStyle; marquee =
this.c1TrueDBGrid1.Splits[this.csss1TrueDBGrid1.SplitIndex].MarqueeStyle;
```

为了对网格中所有拆分改变选取框的固定单元格边框，您可以使用：
To write code in Visual Basic

Visual Basic

```
Me.C1TrueDBGrid1.MarqueeStyle = MarqueeEnum.SolidCellBorder
```

To write code in C#

C#
this.c1TrueDBGrid1.MarqueeStyle = MarqueeEnum.SolidCellBorder;

注意该声明等价于：
To write code in Visual Basic

Visual Basic
Me.C1TrueDBGrid1.Splits(0).MarqueeStyle = MarqueeEnum.SolidCellBorder Me.C1TrueDBGrid1.Splits(1).MarqueeStyle = MarqueeEnum.SolidCellBorder

To write code in C#

C#
this.c1TrueDBGrid1.Splits(0).MarqueeStyle = MarqueeEnum.SolidCellBorder; this.c1TrueDBGrid1.Splits(1).MarqueeStyle = MarqueeEnum.SolidCellBorder;

同样的，可以对每个拆分设置选取框样式的不同值：
To write code in Visual Basic

Visual Basic
Me.C1TrueDBGrid1.Splits(0).MarqueeStyle = MarqueeEnum.NoMarquee Me.C1TrueDBGrid1.Splits(1).MarqueeStyle = MarqueeEnum.FloatingEditor

To write code in C#

C#
this.c1TrueDBGrid1.Splits(0).MarqueeStyle = MarqueeEnum.NoMarquee;

this.c1TrueDBGrid1.Splits(1).MarqueeStyle = MarqueeEnum.FloatingEditor;
这些规则仅应用于一个C1TrueDBGrid 对象并且它关联了Split 对象，没有其他的对象对拥有类似的关系。

C1TrueDBGrid常规拆分属性

以下属性在Split和C1TrueDBGrid 对象中均支持，这里附加以下规则对前一章中的内容进行描述：

Property	Description
AllowColMove	使交互列移动。
AllowColSelect	使交互列选择。
AllowRowSelect	使交互行选择。
AllowRowSizing	使交互行调整大小。
AlternatingRowStyle	控件是否奇/偶行样式应用于一个拆分。
CaptionStyle	控件拆分的标题样式。
CurrentCellVisible	设置/返回当前单元格的变更状态。
ExtendRightColumn	设置/返回拆分的扩展列。
FetchRowStyles	控件的FetchRowStyle 事件是否被触发。
FirstRow	书签行占据第一显示行。
LeftCol	返回最左侧可见列。
MarqueeStyle	设置/返回一个拆分的选取框样式。
RecordSelectors	显示/隐藏左边框的选择板。

注意注意: Caption 属性并不包含在该列表中，即使两个对象都支持，因为网格和拆分都包含分离的标题栏，网格的Caption 属性设置不能应

用于每个拆分标题的同一个字符串。以下属性在Split 对象中支持，但在C1TrueDBGrid中不支持，因此为了给整个网格应用一个值，可以单独的为每个拆分设置值。

Property	Description
AllowFocus	允许拆分中的单元格被聚焦。
HorizontalScrollGroup	控件水平滚动条。
Locked	如果禁止拆分的数据输入，可以设置为True 。
SplitSize	根据SizeMode设置/返回拆分的宽度。
SplitSizeMode	控件的拆分是否可以缩放或者是固定尺寸。
VerticalScrollGroup	控件的垂直滚动条。

拆分矩阵符号

当网格包含水平和垂直拆分，也就是说它可以被组织为一个二维拆分矩阵，在矩阵中拆分对象属性的引用和访问可以使用一个二维矩阵符号来完成，在拆分矩阵中的一个特定拆分索引是一个拆分行，此时列由逗号分隔，例如，在第三个水平拆分（行）中访问第二个垂直拆分（列）如下:；
To write code in Visual Basic

```
Visual Basic
Me.C1TrueDBGrid1.Splits.Item(2,1).Style.ForeColor = System.Drawing.Color.Blue
```

To write code in C#

```
C#
this.c1TrueDBGrid1.Splits[2,1].Style.ForeColor = System.Drawing.Color.Blue;
```

注意注意: Item 属性用于之前的示例，当使用拆分矩阵符号访问一个拆分，Item 属性必须明确指定，当在一个一维结构的网格中访问拆分时，Item 属性将被隐藏或被省略。例如，访问仅使用水平拆分网格的第二个拆分，其显示如下：
To write code in Visual Basic

```
Visual Basic
Me.C1TrueDBGrid1.Splits(1).Style.ForeColor = System.Drawing.Color.Blue
```

To write code in C#

```
C#
this.c1TrueDBGrid1.Splits(1).Style.ForeColor = System.Drawing.Color.Blue;
```

创建与移除拆分

在代码中，您必须使用RemoveHorizontalSplit, InsertHorizontalSplit, RemoveVerticalSplit,
及RemoveHorizontalSplit方法创建与移除拆分，每个方法都从0拆分开始：
To write code in Visual Basic

```
Visual Basic
Dim S As C1TrueDBGrid.Split

' 创建带有7个索引的拆分。
Me.C1TrueDBGrid1.InsertVerticalSplit(7)


' 移除带有5个索引的拆分。
Me.C1TrueDBGrid1.RemoveVerticalSplit(5)
```

To write code in C#

```
C#
C1TrueDBGrid.Split S;
// 创建带有7个索引的拆分。
this.c1TrueDBGrid1.InsertVerticalSplit(7);
// 移除带有5个索引的拆分。
this.c1TrueDBGrid1.RemoveVerticalSplit(5);
```

您使用SplitCollection的Count 属性决定网格中拆分的数量：
To write code in Visual Basic

```
Visual Basic
' 在C1TrueDBGrid1中设置变量等于拆分的数量。

variable = Me.C1TrueDBGrid1.Splits.Count
```

To write code in C#

```
C#
// 在C1TrueDBGrid1中设置变量等于拆分的数量。

variable = this.c1TrueDBGrid1.Splits.Count;
```

您可以使用Count 属性对所有的拆分进行迭代，例如：
To write code in Visual Basic

```
Visual Basic
For n = 0 To Me.C1TrueDBGrid1.Splits.Count - 1
Debug.WriteLine (Me.C1TrueDBGrid1.Splits.Caption)

Next n
```

To write code in C#

```
C#
for (n = 0 ; n < this.c1TrueDBGrid1.Splits.Count; n++) {

Console.WriteLine (this.c1TrueDBGrid1.Splits[n].Caption);


}
```

当让，更有效的方法可以使用For Each...Next循环:
To write code in Visual Basic

```
Visual Basic
Dim S As C1TrueDBGrid.Split
For Each S In Me.C1TrueDBGrid1.Splits

Debug.WriteLine (S.Caption)
Next
```

To write code in C#

```
C#
C1TrueDBGrid.Split S;
foreach (S In this.c1TrueDBGrid1.Splits)
{
Console.WriteLine (S);
}
```

创建的Split对象将继承集合中最后对象的所有属性。

在拆分中使用列

在WinForms版版True DBGrid中的每个拆分包含它自己的列集合，C1DisplayColumnCollection 对象提供了特定拆分中两个属性显示的属性访问，C1DisplayColumnCollection的特定属性为单个拆分提供巨大的灵活性，网格连接单个数据
源，因此拆分对于同一个数据呈现不同的视图，因此，在每个拆分中的C1DisplayColumnCollection 包含同样数量的列，列也绑定相同的数据字段。
然而，C1DisplayColumn对象属性的其他值，如Visible，会根据拆的不同而不同，这些属性被称为split-specific，例
如，在代码中创建的列默认是不可见的，因此在之前示例名为LastName的列是不可见的，以下代码在第二个拆分中设置可见：
To write code in Visual Basic

Visual Basic
Me.C1TrueDBGrid1.Splits(1).DisplayColumns("LastName").Visible = True

To write code in C#

C#
this.c1TrueDBGrid1.Splits(1).DisplayColumns("LastName").Visible = true;

Since Visible is a split-specific property, the LastName column remains invisible in other splits.

调整大小与缩放拆分

WinForms版版True DBGrid 提供对所有控件中的单个拆分的大小调整与缩放拆分，配置的拆分具有准确的宽度或高度，拥有固定的数目的行或列，或根据其他拆分调整大小比例，当开始使用WinForms版版True DBGrid控件时，拆分仍可以使用各种方式而不必掌握所有的细节。
在运行时，Split对象的实际尺寸根据它的SplitSize和SplitSizeMode 属性确定，SplitSizeMode属性指定了度量的单位，SplitSize属性指定了单位的数量，WinForms版版True DBGrid 支持三种不同的模式支持拆分，通过
SplitSizeMode属性确定：

Mode	Description
SizeModeEnum.Scalable	表示与其他拆分相对宽度。
SizeModeEnum.Exact	指定容器坐标中的固定宽度。
SizeModeEnum.NumberofColumns	指定固定数量的列。

一个可缩放拆分使用其SplitSize属性的值来确定拆分占据空间的百分比，对于任意的可缩放拆分，百分比是由所有其他可缩放的拆分的SplitSize值的总和除以其SplitSize值确定的。因此，考虑到可缩放拆分的SplitSize属性是分数的分子，分母是可缩放拆分尺寸的总和，可缩放拆分会在不可缩放拆分获取资源后分配剩余的空间，在默认情况下，所有的拆分都是可缩放的，所以他们会争夺整个网格的显示区域，当网格只包含一个拆分时SplitSizeMode始终是可缩放的。
一个精确的拆分使用其SplitSize属性的值作为容器坐标的固定宽度，如果他们不能适应网格的水平范围，则拆分将被截断，该模式不适用于当一个网格仅包含一个拆分的情况。
一个固定列的拆分使用SplitSize属性来指定拆分中显示的列数，如果所显示的列的大小被改变（无论是通过代码方式或用户交互方式），或者在拆分中的列被水平滚动，以致所显示列的宽度是不同的，这些拆分会自动重新配置整个网格，这种模式主要用于创建固定列而不是水平滚动，然而它也可以用于各种其他目的，当网格仅包含一个拆分时该模式并不适用。
注意，当仅存在一个拆分（网格的默认行为）且拆分跨越整个网格的宽度时，这个SplitSizeMode属性总是可缩放的，并且SplitSize属性总是1，当只有一个拆分时这些属性的设置不会有影响，如果有多个拆分，那么此时删除所有拆分只余一个，因此剩余拆分的SplitSizeMode和SplitSize属性将分别自动转化为0和1。
在默认情况下，新创建的拆分SplitSizeMode属性是SizeModeEnum.Scalable，并且SplitSize属性设置为1，例如，使用以下代码创建两个额外拆分：
To write code in Visual Basic

Visual Basic
’ 在左侧创建一个拆分。 Me.C1TrueDBGrid1.InsertHorizontalSplit(0) ’ 创建另一个。 Me.C1TrueDBGrid1.InsertHorizontalSplit(0)

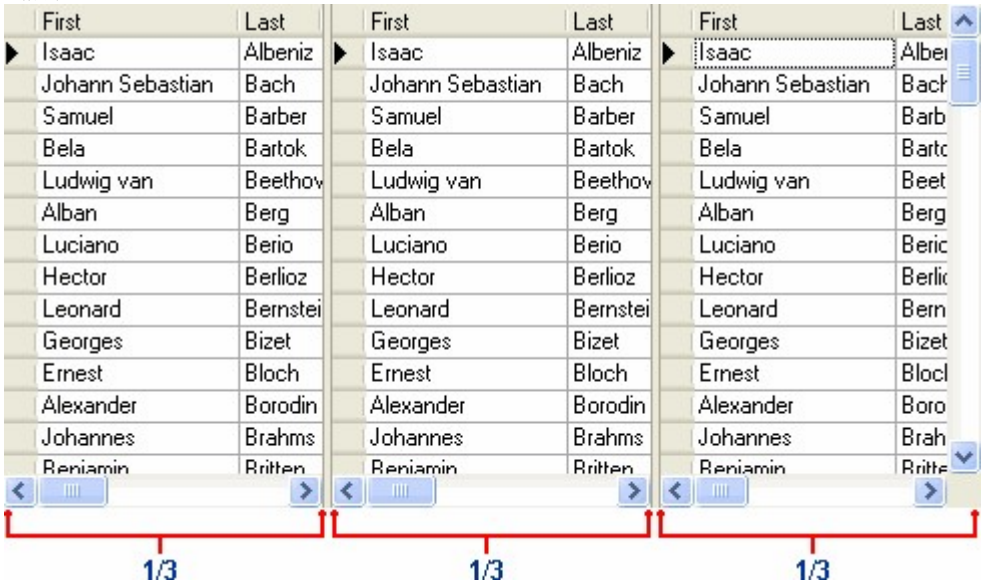
To write code in C#

C#

```
// 在左侧创建一个拆分。
this.c1TrueDBGrid1.InsertHorizontalSplit(0);


// 创建另一个。
this.c1TrueDBGrid1.InsertHorizontalSplit(0);
```

网格会显示如下：



注意到每个拆分占了整个网格控件的1/3，这是因为有三个可缩放的拆分，每个拆分有一个SplitSize为1的属性，如果拆分的尺寸分别变为1, 2, 和3：

To write code in Visual Basic

```
Visual Basic

' 更改相对尺寸为1。
Me.C1TrueDBGrid1.Splits(0).SplitSize = 1


' 更改相对尺寸为2。
Me.C1TrueDBGrid1.Splits(1).SplitSize = 2


' 更改相对尺寸为3。
Me.C1TrueDBGrid1.Splits(2).SplitSize = 3
```

To write code in C#

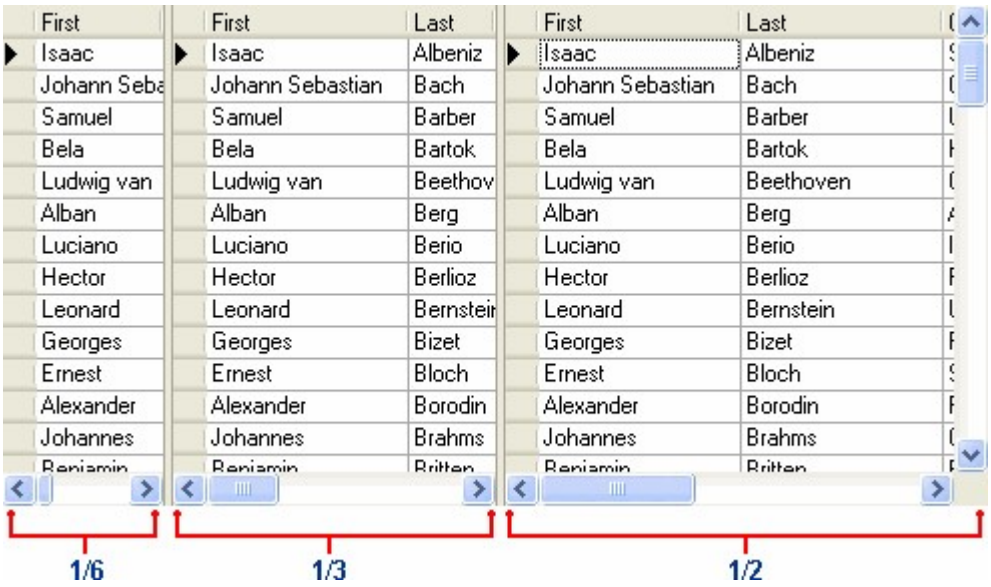
```
C#

// 更改相对尺寸为1。
this.c1TrueDBGrid1.Splits[0].SplitSize = 1;

// 更改相对尺寸为2。
this.c1TrueDBGrid1.Splits[1].SplitSize = 2;


// 更改相对尺寸为3。
this.c1TrueDBGrid1.Splits[2].SplitSize = 3;
```

网格显示将会如下:



注意到拆分尺寸的总数为(1+2+3)=6，因此每个拆分的尺寸是分子与其SplitSize属性值和为6的分数。
当拆分的SplitSizeMode 被设置为SizeModeEnum.Exact，则该拆分会接受其他拆分的空间，这个行为多少有些复杂，但理解如何缩放拆分是有帮助的，例如，拆分设置如下：
To write code in Visual Basic

Visual Basic

```
Me.C1TrueDBGrid1.Splits(0).SplitSizeMode = C1.Win.C1TrueDBGrid.SizeModeEnum.Scalable

Me.C1TrueDBGrid1.Splits(0).SplitSize = 1

Me.C1TrueDBGrid1.Splits(1).SplitSizeMode = C1.Win.C1TrueDBGrid.SizeModeEnum.Exact
Me.C1TrueDBGrid1.Splits(1).SplitSize = 250

Me.C1TrueDBGrid1.Splits(2).SplitSizeMode = C1.Win.C1TrueDBGrid.SizeModeEnum.Scalable

Me.C1TrueDBGrid1.Splits(2).SplitSize = 2
```

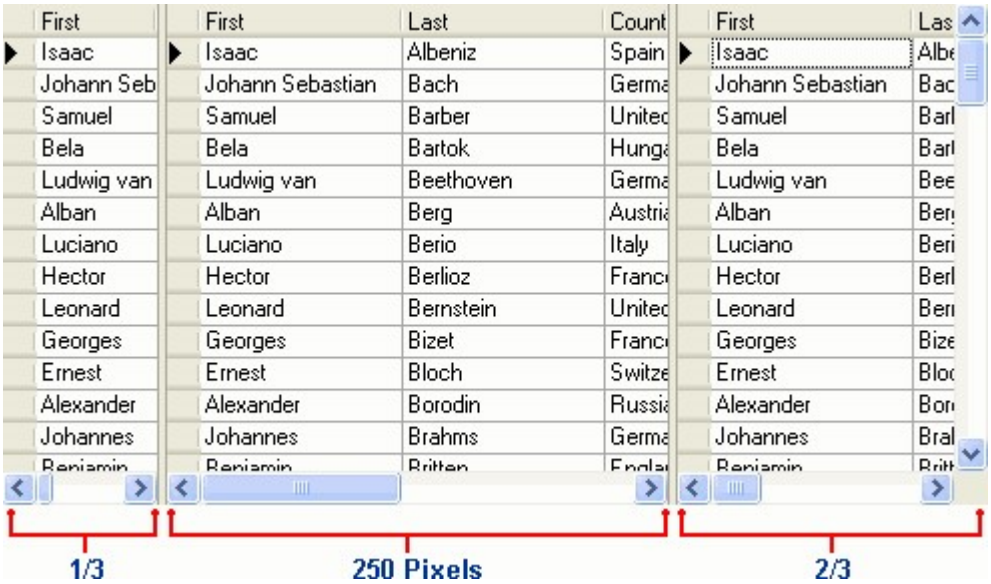
To write code in C#

C#

```
this.c1TrueDBGrid1.Splits[0].SplitSizeMode = C1.Win.C1TrueDBGrid.SizeModeEnum.Scalable;
this.c1TrueDBGrid1.Splits[0].SplitSize = 1; this.c1TrueDBGrid1.Splits[1].SplitSizeMode =
C1.Win.C1TrueDBGrid.SizeModeEnum.Exact; this.c1TrueDBGrid1.Splits[1].SplitSize = 250;

this.c1TrueDBGrid1.Splits[2].SplitSizeMode = C1.Win.C1TrueDBGrid.SizeModeEnum.Scalable;
this.c1TrueDBGrid1.Splits[2].SplitSize = 2;
```

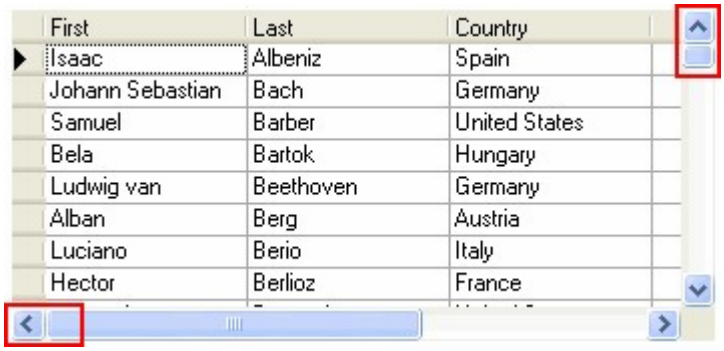
使用此方法配置拆分后，其结果显示如下：



在中间的固定大小的拆分（拆分1）被精确地配置为250个像素，并且其余的拆分争夺剩余的网格的空间，由于剩余的拆分都是可缩放的，它们根据使用SplitSize属性值计算出的百分比划分的剩余的空间，因而最左边的拆分占用的剩余三分之一的空间，最右边的分裂占据2/3。SplitSizeMode属性被设置为SizeModeEnum.NumberOfColumns的拆分行为等同于精确拆分，除了它们的尺寸是由列的整数倍的宽度决定，然而宽度是动态的，所以调整列的大小或滚动以致视图中不同的列可以让整个网格重新配置它本身。避免创建无缩放拆分网格，尽管WinForms版版TrueDBGrid能够处理这种情况，但这种方式很难配置网格，例如，如果
没有拆分是可缩放的，那么所有的拆分将具有确切的大小，它可能没有填充网格的水平宽度，如果拆分的总宽度过短，WinForms版版TrueDBGrid显示"空领域"将不会有任何拆分，如果拆分的总宽度比网格更宽，那么WinForms版版
TrueDBGrid将只显示无法显示拆分的分隔线。

通过用户交互创建并调整拆分的大小

拆分可以在代码中创建并调整大小，然而用户可以交互地创建并调整拆分，通过设置AllowHorizontalSplit 或AllowVerticalSplit 属性为True，默认情况下，两个属性均为False，以防止用户创建并调整拆分的大小。将这些属性设置为False的网格如下图显示，注意在水平滚动条的左边框或者垂直滚动条上方没有拆分框。



新的拆分将继承原始拆分的属性，这两个拆分的SplitSizeMode属性将自动设置为SizeModeEnum.Scalable，不管原始拆的分的SplitSizeMode，两个拆分的SplitSize属性将被设定为所述拆分的尺寸比例，SplitSize属性的值可能最终会被相当大，这是因为WinForms版版True DBGrid需要选择总拆分的尺寸的最小公分母，并且用户可以将指针拖动到任意的位置。

垂直拆分

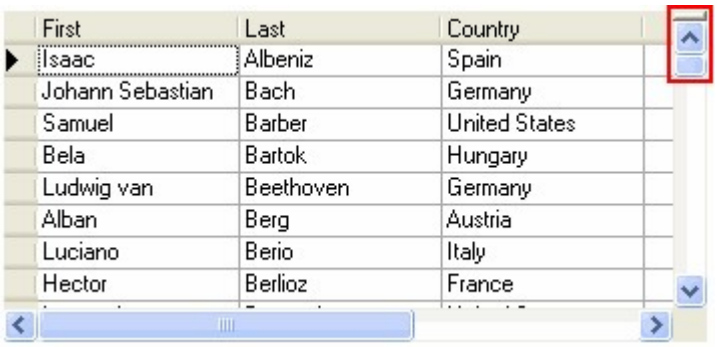
如果拆分的AllowVerticalSplit属性被设置为True:
To write code in Visual Basic

Visual Basic
Me.C1TrueDBGrid1.AllowVerticalSplit = True

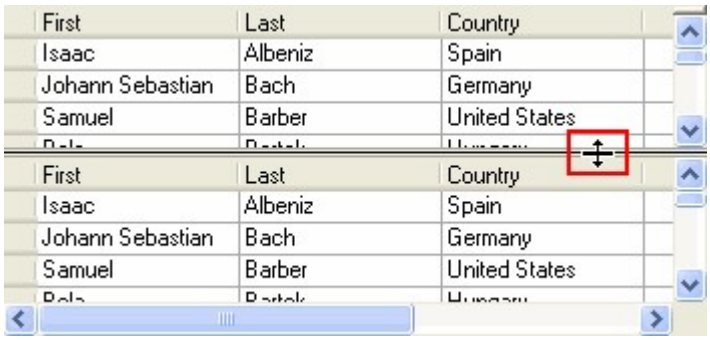
To write code in C#

C#
this.c1TrueDBGrid1.AllowVerticalSplit = true;

一个拆分框将会出现在垂直滚动条的顶端，并当光标在拆分框上方时，它将返回使用垂直箭头的两个水平栏，从拆分框拖动光标来创建一个新的拆分。



当一个拆分被创建，拖动光标上下调整拆分的相对尺寸。



水平拆分

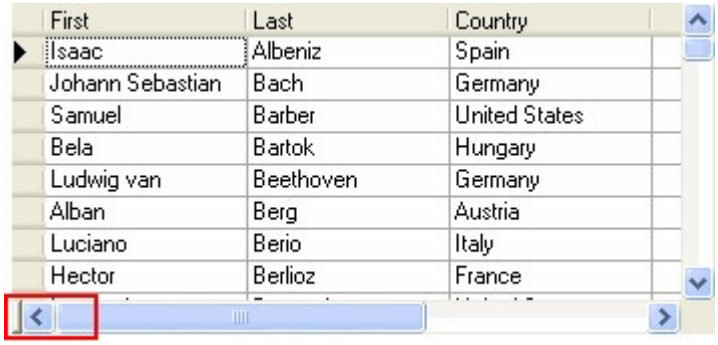
若拆分的AllowHorizontalSplit属性被设置为True:
To write code in Visual Basic

Visual Basic
Me.C1TrueDBGrid1.AllowHorizontalSplit = True

To write code in C#

C#
this.c1TrueDBGrid1.AllowHorizontalSplit = true;

一个拆分框将会出现在水平滚动条的左端，并当光标在拆分框上方时，它将返回使用水平箭头的两个垂直栏，从拆分框拖动光标来创建一个新的拆分。



当一个拆分被创建，拖动光标左右调整拆分的相对尺寸。

总结

拆分可以在代码中创建或调整尺寸，但AllowHorizontalSplit 和 AllowVerticalSplit
属性控制用户是否能在运行时交互的创建或调整拆分的大小。
用户可以重新调整两个拆分的相对尺寸，仅但两个拆分的AllowSizing属性为True时，当用户完成调整尺寸操作时，两
个拆分的总大小保持不变，但两个拆分的SplitSizeMode属性将被自动设置为SizeModeEnum.Scalable，无论其以前的设置为何值，两个拆分的SplitSize属性将被设置为反映其新尺寸的比例值。

垂直滚动与拆分分组

默认情况下，网格仅有从索引0开始的一个水平的拆分，并且它的HScrollBar 和 VScrollBar 样式属性被设置为ScrollBarStyleEnum.Automatic，即水平或垂直的滚动条会根据列的宽度和行内数据的数量出现，拆分的默认HorizontalScrollGroup 和 VerticalScrollGroup 属性被设置为1，Splits有相同的滚动组属性设置，可以同时设置垂直和水平滚动，当一个拆分被创建时，它将从两个滚动条中继承并且继承从父拆分中继承Scroll Group属性，若所有的拆分拥有相同的HorizontalScrollGroup或VerticalScrollGroup ，可以将它们的HScrollBar 和VScrollBar 样式属性设置为ScrollBarStyleEnum.Automatic，此时WinForms版版True DBGrid 在滚动组的最右或最低部显示垂直滚动条或水平条，更改单个滚动条将会同时引起同一个组里所有拆分的改变。
例如，两个额外的拆分通过以下代码创建：
To write code in Visual Basic

```
Visual Basic

' 在左边创建一个拆分。
Me.C1TrueDBGrid1.InsertHorizontalSplit(0)


' 创建另一个。
Me.C1TrueDBGrid1.InsertHorizontalSplit(0)
```

To write code in C#

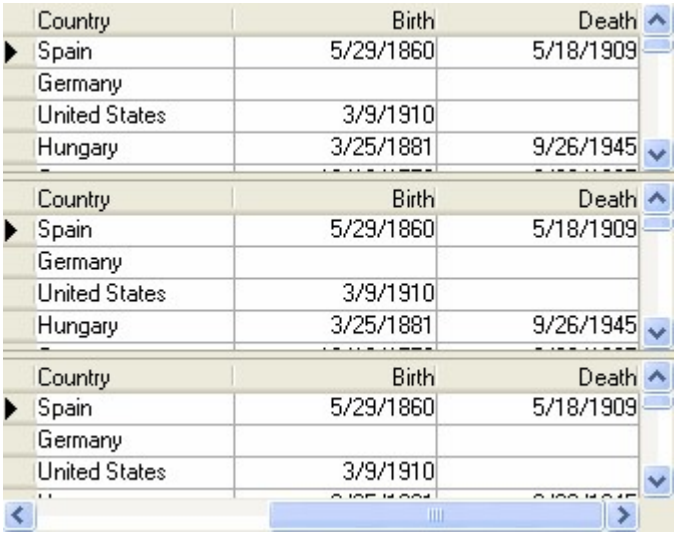
```
C#

// 在左边创建一个拆分。
this.c1TrueDBGrid1.InsertHorizontalSplit(0);


// 创建另一个。
this.c1TrueDBGrid1.InsertHorizontalSplit(0);
```

网格的显示如下：

三个拆分都有相同的HScrollBar 及 VScrollBar 设置，并且VerticalScrollGroup 被设置为1，此外，仅有一个垂直滚动条显示且在最右边的拆分中显示，当用户操作该滚动条时，所有三个拆分都会同时滚动。
垂直拆分会做出相同的反应，再添加两个垂直拆分至网格后，所有拆分都有相同的HorizontalScrollGroup 值且为1，因此网格的下方仅有一个水平滚动条，并且若滚动条被滚动，则三个拆分都会同时被滚动。



改变拆分的滚动组属性其中一个属性可以独立的创建拆分组，在之前的示例中，对中间两个拆分设置HorizontalScrollGroup 属性为2 创建一个新的滚动组：
To write code in Visual Basic

Visual Basic

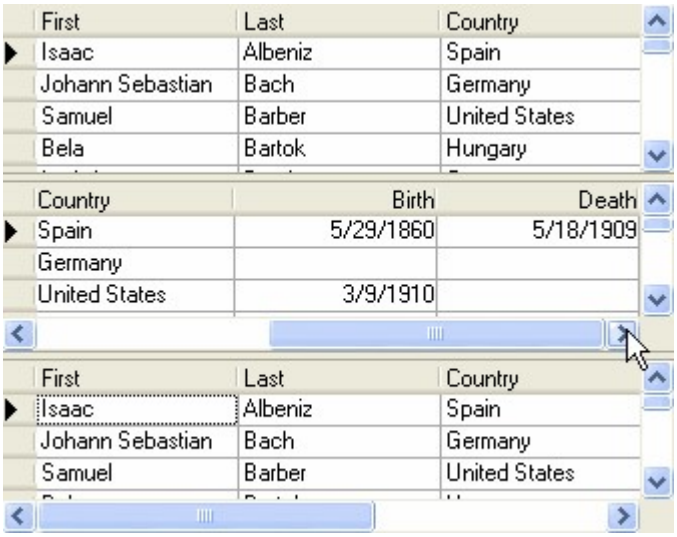
```
Me.C1TrueDBGrid1.Splits.Item(0,1).HorizontalScrollGroup = 2
```

To write code in C#

C#

```
this.c1TrueDBGrid1.Splits[0,1].HorizontalScrollGroup = 2;
```

在声明执行后，滚动中间拆分时不会打断其他的拆分，其显示如下图。



注意到中间的拆分现在包含一个水平滚动条，该滚动条仅能在中间的拆分中操作，因为它的HorizontalScrollGroup 属性为2，在最下方拆分的水平滚动条现在仅能控制底部和顶部的拆分，它不在影响中间的拆分。
该特性的一个常规应用是创建两个独立的拆分组，用户可以通过滚动每一个拆分对不同的记录比较字段值。

水平滚动与固定列

每个拆分的滚动是独立的，通常情况下，一个或多个列需要阻止水平或垂直滚动，使得列始终处于视图中，WinForms 版版True DBGrid 提供了简单的方法在网格中的任意位置保持滚动的列数（甚至在中间！），您仅需几个拆分属性。
实例中包含一个带有三个水平拆分，以下代码将在中间拆分中"固定"列0和列1：
To write code in Visual Basic

Visual Basic

```
' 隐藏Splits(1)中除了列0和1的所有列。
Dim Cols As C1TrueDBGrid.C1DisplayColumnCollection

Dim C As C1TrueDBGrid.C1DisplayColumn

Cols = Me.C1TrueDBGrid1.Splits(1).DisplayColumns
For Each C In Cols
C.Visible = False
Next C
Cols(0).Visible = True
Cols(1).Visible = True

' 配置Splits(1)准确显示两行并禁用调整大小。
With Me.C1TrueDBGrid1.Splits(1)
.SplitSizeMode = SizeModeEnum.NumberOfColumns
.SplitSize = 2
.AllowHorizontalSizing = False
End With
```

To write code in C#

C#

```
// 隐藏Splits(1)中除了列0和1的所有列。
```

```
C1TrueDBGrid.C1DisplayColumnCollection Cols;
C1TrueDBGrid.C1DisplayColumn C;
Cols = this.c1TrueDBGrid1.Splits[1].DisplayColumns foreach (C In Cols) {

C.Visible = false;
}
Cols(0).Visible = true;
Cols(1).Visible = true;

// 配置Splits(1)准确显示两行并禁用调整大小。
this.c1TrueDBGrid1.Splits[1].SplitSizeMode = SizeModeEnum.NumberOfColumns; this.c1TrueDBGrid1.Splits[1].SplitSize = 2;
this.c1TrueDBGrid1.Splits[1].AllowHorizontalSizing = false;
```

通常情况下，如果在一个拆分中列0和列1将不会滚动，在其他的拆分中它们将不可见：
To write code in Visual Basic

Visual Basic

```
’ 在拆分0和2中使列0和1不可见。
Dim Cols As C1TrueDBGrid.C1DisplayColumnCollection

Cols = Me.C1TrueDBGrid1.Splits(0).DisplayColumns
Cols(0).Visible = False
Cols(1).Visible = False
Cols = Me.C1TrueDBGrid1.Splits(2).DisplayColumns
Cols(0)Visible = False
Cols(1)Visible = False
```

To write code in C#

C#

```
// 在拆分0和2中使列0和1不可见。
C1TrueDBGrid.C1DisplayColumnCollection Cols;
Cols = this.c1TrueDBGrid1.Splits[0].DisplayColumns;

Cols[0].Visible = false;
Cols[1].Visible = false;
Cols = this.c1TrueDBGrid1.Splits[2].DisplayColumns;

Cols[0]Visible = false;
Cols[1]Visible = false;
```

跨拆分导航

跨拆分导航由网格的TabAcrossSplits 和每个拆分的AllowFocus 属性控制，跨导航拆分可以使用整个网格来讨论，获取更多信息，请参阅运行时交互。