



SPREAD

For Windows Forms 5.0

使用指南

使用指南简介

Spread for Windows Forms 是一个综合性的、用于微软.NET 平台的 Windows Forms 应用程序开发的表格控件。它集成了丰富的表格功能、工作表功能，并可以与多种数据源进行绑定。一个 Spread 控件可以处理多达 20 亿个工作表，每个工作表可以有 20 亿行和 20 亿列，并且支持跨工作表的数据引用和跨工作表的公式引用。Spread 控件已经被业界公认为功能最为强大的表格控件。Spread 在中国也有了超过 10 年的使用历史，得到了许多软件开发人员的认可。

随着 5.0 版本的推出，Spread 增加了对于图表功能的支持。通过超过 85 种不同类型的图表和内置的用户界面，为创建和定制图表提供了全面支持，丰富了数据的展示方式。同时，Spread 与 Excel 文件的兼容性得到了进一步的提升。

虽然 Spread 产品附带了大量的文档和示例代码，但不可否认的是，对于初次接触 Spread 的中国软件开发人员来讲，有时难免会觉得 Spread 入门不太容易。本使用指南旨在帮助软件开发人员较快地了解和运用 Spread 的常用功能。开发人员可以在了解 Spread 整体功能的基础上，循序渐进地学习单元格操作、行列操作和工作表操作。每个操作都附带了具体的 C#和 VB 代码，读者可以直接运行例子代码进行联系并查看效果。在表格数据操作的基础上，本指南也针对高级数据操作、图形图表、以及其他数据格式的交换方面做了进一步的介绍，并且在最后提供了常用词汇的中英文对照表，方便读者查看。

对于项目管理人员和技术主管来讲，通过对本指南中 Spread 主要功能的概览，可以帮助他们决策 Spread 是否适合于其项目的需要。

有关 Spread 产品的更多信息，请访问：

Spread 产品网站 <http://www.grapecity.cn/powertools>

Spread 技术支持论坛 <http://gcdn.grapecity.com>

目 录

使用指南简介	1
1. 从例子入门：创建 CHECKBOOK REGISTER	4
1.1 添加 SPREAD 控件到 CHECKBOOK 工程	4
1.2 设置行和列	5
1.3 设置单元格类型	7
1.4 添加公式	10
2. SPREAD 产品介绍	12
2.1 产品概述	12
2.2 基本功能介绍	13
2.3 SPREAD 5.0 FOR WINDOWS FORMS 的主要新增功能.....	17
3. 单元格操作	19
3.1 设置单元格类型	19
3.2 设置单元格的顏色	21
3.3 合并单元格	23
3.4 锁定、解锁单元格	24
3.5 给单元格添加批注	26
3.6 设置公式	28
4. 行列操作	31
4.1 设置行列数	31
4.2 移动行列	31
4.3 调整行高、列宽	33
4.4 冻结行列	34
4.5 使用分组	36
4.6 定制行头、列头的文字	37
4.7 设置多行行头、多列列头	39
5. 工作表操作	42
5.1 使用当前工作表	42
5.2 设置背景色或背景图	42
5.3 增加工作表	44
5.4 删除工作表	45
5.5 移动工作表	46
5.6 显示或隐藏工作表	46
5.7 添加标题和子标题	47
6. 高级数据操作	50

6.1	数据绑定示例	50
6.1.1	把 Spread 添加到一个数据绑定项目.....	50
6.1.2	设置数据库链接	50
6.1.3	指定需要使用的数据	50
6.1.4	创建数据集	52
6.1.5	把 Spread 绑定到数据库.....	52
6.1.6	设置单元格类型, 改善显示效果.....	53
6.2	数据排序	54
6.3	数据过滤	55
6.3.1	允许数据过滤	55
6.3.2	使用数据过滤	56
7.	使用图形	58
7.1	创建图形对象	58
7.2	设置图形属性	58
7.3	图形旋转	59
7.4	图形缩放	59
7.5	图形移动	60
7.6	图形锁定	60
8.	使用图表	61
8.1	创建图表对象	62
8.2	使用图表设计器	68
8.3	绑定图表	69
8.4	允许用户改变图表	70
9.	与其他数据格式交互	72
9.1	打开 EXCEL 文件.....	72
9.2	保存为 EXCEL 文件.....	74
9.3	导出 PDF	75
10.	SPREAD WIN 5 中英文术语对照.....	77

1. 从例子入门：创建 Checkbook Register

1.1 添加 Spread 控件到 Checkbook 工程

创建一个新的 Visual Studio .NET 工程并命名为 Checkbook。将工程中的窗体(Form) 重命名为 Register。将 FpSpread 控件添加到当前工程，然后将该控件添加到窗体中。

如果您对 .net 平台不是很了解的话，您可能不熟悉如何启动一个新项目来使用新的控件。要使用这个产品，您需要将这个控件添加到 Visual Studio .NET 环境中。

第一步就是在 Visual Studio .NET 创建一个新的工程，然后添加该控件到该工程中。

1. 启动 Visual Studio .NET。
2. 在 File 菜单中，选择 New->Project。
3. 在新工程对话框的工程类型区域，选择您需要开发使用的语言，例如在这个工程类型中选择 Visual C#工程。
4. 在新工程对话框中
 - a. 在工程类型列表中选择 C#工程或者 Visual Basic 工程。
 - b. 在模板列表中选择 windows Application。
 - c. 在名称输入框中输入新工程的名称 Checkbook，默认的名称 WindowsApplication1。
 - d. 在地址输入框中有一个默认的工程路径，用户也可以点击浏览选择一个新的路径。
 - e. 点击 OK。
 - f. 如果您的工程看不到解决方案浏览器，您可以在“视图”菜单中找到。
5. 在解决方案浏览器中，右键点击 form 的名称 Form1, 在弹出菜单选择重命名，然后输入 register.

下一步就是添加该控件到 Visual Studio .NET 的工具箱中，这一步只需要做一次就可以了。

1. 如果工具箱没有显示，用户可以在 view 菜单的找到并显示它。
2. 如果工具箱已经显示，查看 windows forms 目录（或 spread 的安装目录）。
3. 如果 Spread 控件没有在工具箱中，右键点击工具箱并在弹出菜单中选择 Customize Toolbox, Add/Remove Items 或者 Choose Items.（这个需要根据 Visual Studio 的版本决定）。
4. 在 Customize Toolbox 对话框中，点击 .NET Framework Components 标签。
5. 在 .NET Framework Components 标签中，Spread 控件应该显示在控件列表中，选中 Spread 控件的复选框后点击 ok 按钮。如果 Spread 控件没有显示在控件列表中，点击 Browse 查看控件的安装目录，在安装目录选择 FarPoint.Win.Spread.dll 然后点击 Open。Spread 控件这时应该显示在控件列表中了，选择它然后点击 ok 按钮。
6. 您可以把这个控件添加到一个工程中来测试它。

下一步是把这个控件添加到一个工程中。

1. 在一个打开的工程中，在工具箱的 Windows Forms 目录（或其他 spread 控件添加的目录），选择 Spread 控件。
2. 在您的 windows Forms 中，您可以通过拖动鼠标画一个矩形来添加一个 spread 控件并初始化该控件的大小。

3. 您已经添加了一个 Spread 控件到您的工程中了。

1.2 设置行和列

Spread 控件在 form 中已经有了一个工作表，您可以定制这个工作表。在这一步您将会设置这个工作表的行、列和单元格。

示例

第 1 步：设置电子表单的宽，高以及行数和列数

```
[C#]
// 设置电子表单的宽、高以及行数和列数.
fpSpread1.Height = 330;
fpSpread1.Width = 765;
fpSpread1.Sheets[0].ColumnCount = 8;
fpSpread1.Sheets[0].RowCount = 100;

[Visual Basic]
'设置电子表单的宽，高以及行数和列数.
FpSpread1.Height = 330
FpSpread1.Width = 765
FpSpread1.Sheets(0).ColumnCount = 8
FpSpread1.Sheets(0).RowCount = 100
```

第 2 步：设置列头文本

```
[C#]
// 设置列头文本。
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 0].Text = "Check #";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 1].Text = "Date";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 2].Text = "Description";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 3].Text = "Tax?";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 4].Text = "Cleared?";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 5].Text = "Debit";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 6].Text = "Credit";
fpSpread1.Sheets[0].ColumnHeader.Cells[0, 7].Text = "Balance";

[Visual Basic]
'设置列头文本。
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 0).Text = "Check #"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 1).Text = "Date"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 2).Text = "Description"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 3).Text = "Tax?"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 4).Text = "Cleared?"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 5).Text = "Debit"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 6).Text = "Credit"
FpSpread1.Sheets(0).ColumnHeader.Cells(0, 7).Text = "Balance"
```

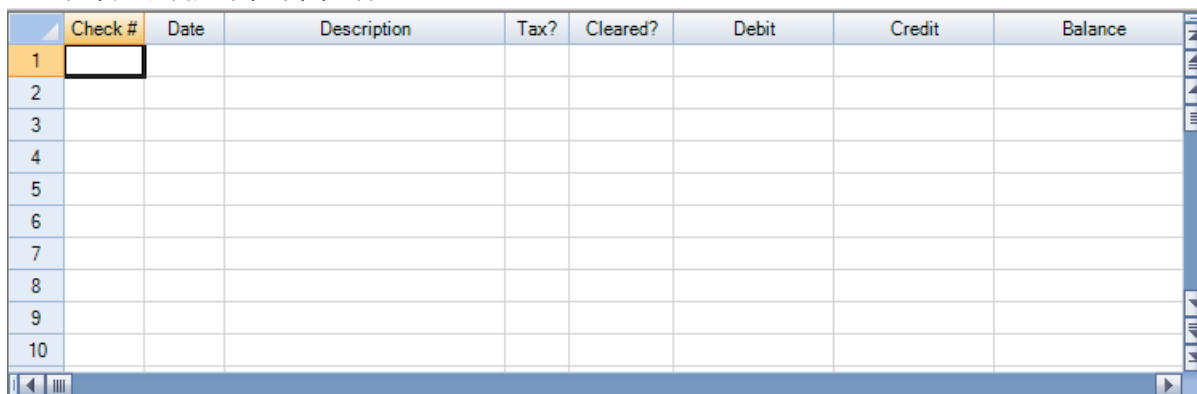
第 3 步：调整列宽显示标题和数据

```
[C#]
//设置列宽.
fpSpread1.Sheets[0].Columns[0].Width = 50;
fpSpread1.Sheets[0].Columns[1].Width = 50;
fpSpread1.Sheets[0].Columns[2].Width = 175;
fpSpread1.Sheets[0].Columns[3].Width = 40;
fpSpread1.Sheets[0].Columns[4].Width = 65;
fpSpread1.Sheets[0].Columns[5].Width = 100;
fpSpread1.Sheets[0].Columns[6].Width = 100;
fpSpread1.Sheets[0].Columns[7].Width = 125;

[Visual Basic]
' 设置列宽.
FpSpread1.Sheets(0).Columns(0).Width = 50
FpSpread1.Sheets(0).Columns(1).Width = 50
FpSpread1.Sheets(0).Columns(2).Width = 175
FpSpread1.Sheets(0).Columns(3).Width = 40
FpSpread1.Sheets(0).Columns(4).Width = 65
FpSpread1.Sheets(0).Columns(5).Width = 100
FpSpread1.Sheets(0).Columns(6).Width = 100
FpSpread1.Sheets(0).Columns(7).Width = 125
```

第 4 步：保存您的工程然后从“编译”菜单中选择“开始”来运行您的工程

您的表单应该看起来和下图一样：



Check #	Date	Description	Tax?	Cleared?	Debit	Credit	Balance
1							
2							
3							
4							
5							
6							
7							
8							
9							
10							

1.3 设置单元格类型

Spread 支持多达 22 种单单元格类型。设置单元格类型时，对每一种单元格类型，您需要创建一个单元格类型对象，设置其属性，然后将其设为一个或多个单元格的 CellType 属性。

通过以下代码设置一些单元格类型到当前数据表的列上。

示例：设置数字单元格类型到 Check #列

```
[C#]
// 把 Check #列的单元格类型设置成数字
FarPoint.Win.Spread.CellType.NumberCellType objNumCell = new
FarPoint.Win.Spread.CellType.NumberCellType();
objNumCell.DecimalPlaces = 0;
objNumCell.MinimumValue = 1;
objNumCell.MaximumValue = 9999;
objNumCell.ShowSeparator = false;
fpSpread1.Sheets[0].Columns[0].CellType = objNumCell;
```

[Visual Basic]

```
'把 Check #列的单元格类型设置成数字
Dim objNumCell As New FarPoint.Win.Spread.CellType.NumberCellType()
objNumCell.DecimalPlaces = 0
objNumCell.MinimumValue = 1
objNumCell.MaximumValue = 9999
objNumCell.ShowSeparator = False
FpSpread1.Sheets(0).Columns(0).CellType = objNumCell
```

示例：设置日期单元格类型到 Date 列

```
[C#]
// 把 Date 列的单元格类型设置成日期
FarPoint.Win.Spread.CellType.DateTimeCellType objDateCell = new
FarPoint.Win.Spread.CellType.DateTimeCellType();
objDateCell.DateTimeFormat = FarPoint.Win.Spread.CellType.DateTimeFormat.ShortDate;
fpSpread1.Sheets[0].Columns[1].CellType = objDateCell;
```

[Visual Basic]

```
'把 Date 列的单元格类型设置成日期
Dim objDateCell As New FarPoint.Win.Spread.CellType.DateTimeCellType()
objDateCell.DateTimeFormat = FarPoint.Win.Spread.CellType.DateTimeFormat.ShortDate
FpSpread1.Sheets(0).Columns(1).CellType = objDateCell
```

示例：设置文本单元格类型到 Description 列

```
[C#]
// 把 Description 列单元格类型设置成文本类型
FarPoint.Win.Spread.CellType.TextCellType objTextCell = new FarPoint.Win.Spread.CellType.TextCellType();
objTextCell.MaxLength = 100;
fpSpread1.Sheets[0].Columns[2].CellType = objTextCell;

[Visual Basic]
'把 Description 列单元格类型设置成文本类型
Dim objTextCell As New FarPoint.Win.Spread.CellType.TextCellType()
objTextCell.MaxLength = 100
FpSpread1.Sheets(0).Columns(2).CellType = objTextCell
```

示例：设置单选框单元格类型到 “Tax?” 和 “Cleared?” 列

```
[C#]
//把 “Tax?” 和 “Cleared?” 列的单元格类型设置成单选框类型
FarPoint.Win.Spread.CellType.CheckBoxCellType objCheckCell = new
FarPoint.Win.Spread.CellType.CheckBoxCellType();
objCheckCell.ThreeState = false;
fpSpread1.Sheets[0].Columns[3].CellType = objCheckCell;
fpSpread1.Sheets[0].Columns[4].CellType = objCheckCell;

[Visual Basic]
'把 “Tax?” 和 “Cleared?” 列的单元格类型设置成单选框类型
Dim objCheckCell As New FarPoint.Win.Spread.CellType.CheckBoxCellType()
objCheckCell.ThreeState = False
FpSpread1.Sheets(0).Columns(3).CellType = objCheckCell
FpSpread1.Sheets(0).Columns(4).CellType = objCheckCell
```

示例：设置货币框单元格类型到 Debit , Credit 和 Balance 列

```
[C#]
// 把 Debit, Credit 和 Balance 列的单元格类型设置成货币类型
FarPoint.Win.Spread.CellType.CurrencyCellType objCurrCell = new
FarPoint.Win.Spread.CellType.CurrencyCellType();
objCurrCell.LeadingZero = FarPoint.Win.Spread.CellType.LeadingZero.Yes;
objCurrCell.NegativeRed = true;
objCurrCell.FixedPoint = true;
fpSpread1.Sheets[0].Columns[5].CellType = objCurrCell;
fpSpread1.Sheets[0].Columns[6].CellType = objCurrCell;
fpSpread1.Sheets[0].Columns[7].CellType = objCurrCell;

[Visual Basic]
'把 Debit, Credit 和 Balance 列的单元格类型设置成货币类型
Dim objCurrCell As New FarPoint.Win.Spread.CellType.CurrencyCellType()
objCurrCell.LeadingZero = FarPoint.Win.Spread.CellType.LeadingZero.Yes
objCurrCell.NegativeRed = True
objCurrCell.FixedPoint = True
FpSpread1.Sheets(0).Columns(5).CellType = objCurrCell
FpSpread1.Sheets(0).Columns(6).CellType = objCurrCell
FpSpread1.Sheets(0).Columns(7).CellType = objCurrCell
```

保存您的工程然后从“编译”菜单中选择“开始”来运行您的工程。存储工程，选择“编译”菜单“开始”运行程序。

您的窗体将会如下图：

	Check #	Date	Description	Tax?	Cleared?	Debit	Credit	Balance
1				<input type="checkbox"/>	<input type="checkbox"/>			
2				<input type="checkbox"/>	<input type="checkbox"/>			
3				<input type="checkbox"/>	<input type="checkbox"/>			
4				<input type="checkbox"/>	<input type="checkbox"/>			
5				<input type="checkbox"/>	<input type="checkbox"/>			
6				<input type="checkbox"/>	<input type="checkbox"/>			
7				<input type="checkbox"/>	<input type="checkbox"/>			
8				<input type="checkbox"/>	<input type="checkbox"/>			
9				<input type="checkbox"/>	<input type="checkbox"/>			
10				<input type="checkbox"/>	<input type="checkbox"/>			

1.4 添加公式

您的工程现在看起来像一个支票登记簿。但是，在您登记的时候它不会自动计算收支平衡。

下列步骤通过 Spread 提供的公式功能来自动计算收支平衡。

示例

```
[C#]
// 设置公式计算收支平衡
fpSpread1.Sheets[0].ReferenceStyle = FarPoint.Win.Spread.Model.ReferenceStyle.R1C1;
int i;
for (i = 0; i <= fpSpread1.ActiveSheet.RowCount - 1; i++)
{
if (i == 0)
fpSpread1.Sheets[0].Cells[i, 7].Formula = "RC[-1] - RC[-2]";
else
fpSpread1.Sheets[0].Cells[i, 7].Formula = "RC[-1] - RC[-2] + R[-1]C";
}

[Visual Basic]
'设置公式计算收支平衡
FpSpread1.Sheets(0).ReferenceStyle = FarPoint.Win.Spread.Model.ReferenceStyle.R1C1
Dim i As Integer
For i = 0 To FpSpread1.ActiveSheet.RowCount - 1
If i = 0 Then
FpSpread1.Sheets(0).Cells(i, 7).Formula = "RC[-1] - RC[-2]"
Else
FpSpread1.Sheets(0).Cells(i, 7).Formula = "RC[-1]-RC[-2]+R[-1]C"
End If
Next
```

保存您的工程，从“编译”菜单中选择“开始”来运行您的工程。

此时您的窗体应该看起来如下图所示。输入数据来测试一下它是否正常工作吧！

Check #	Date	Description	Tax?	Cleared?	Debit	Credit	Balance
1			<input type="checkbox"/>	<input type="checkbox"/>			\$0.00
2	10/30/2007		<input type="checkbox"/>	<input type="checkbox"/>	\$55.00	\$100.00	\$45.00
3			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
4			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
5			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
6			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
7			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
8			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
9			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00
10			<input type="checkbox"/>	<input type="checkbox"/>			\$45.00

现在，您已经用 Spread 控件成功地创建了一个 Checkbook Register。您已经完成了这个教程。

2. Spread 产品介绍

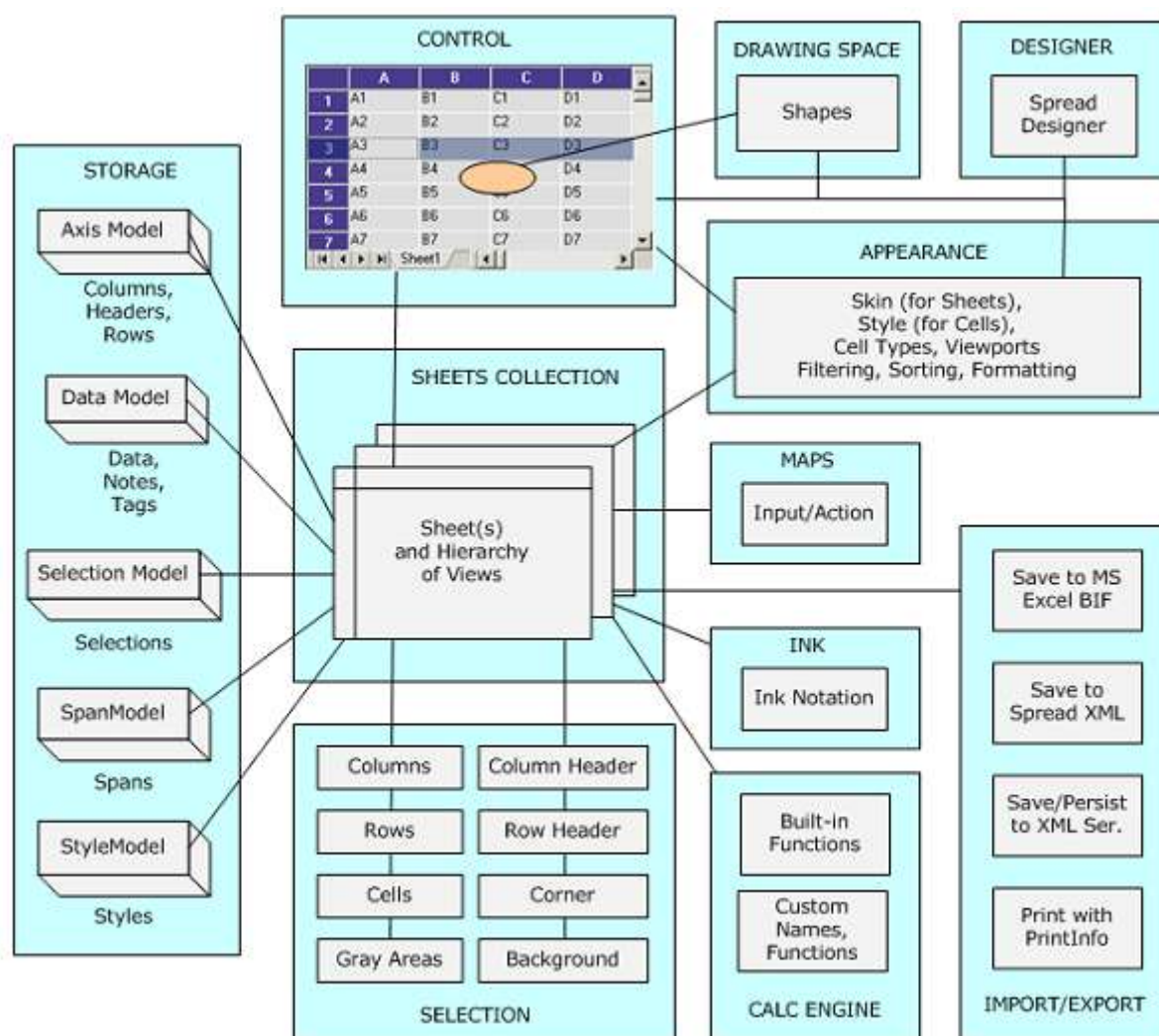
2.1 产品概述

Spread for Windows Forms 是一个综合性的、用于 Windows Forms 应用程序开发的表格控件。它集成了表格功能、工作表功能，并提供了与多种数据源进行绑定的功能。一个 Spread 控件可以处理多达 20 亿个工作表，每个工作表可以有 20 亿行和 20 亿列。Spread 支持跨工作表的数据引用和跨工作表的公式引用。

Spread 控件能被简单的拖放到一个 Windows Forms 上然后根据不同应用程序的需要来定制。Spread 为用户提供了许多方法来定制它的外观及和用户之间的交互方式。使用 Spread 设计器，您能够快速创建您的应用程序原形，完成您的设计。高级的开发者甚至能够通过编码实现对 Spread 控件的完全控制。

Spread 支持和 Microsoft Excel 数据间的导入/导出。Spread 的工作表数据能够被保存为或导入从以下格式：1. 逗号分隔的文本文件；2. BIFF8 格式的 Excel 文件 (Excel97-2003 格式)；3. XML 格式的 Excel 文件 (Excel2007 格式)。

下图是 Spread Win 控件的一个概念、功能模型图：



2.2 基本功能介绍

Spread 支持导入和导出 Microsoft Excel 格式的文件，支持 PDF 导出，支持多工作表、跨工作表公式索引、分层显示、分组、有条件的格式、排序、行过滤、搜索、缩放、撤销/重做、数据绑定或非绑定模式、拆分条等功能使您可以为任何应用程序创建解决方案。Spread 支持多达 22 种单元格类型，甚至支持创建自定义单元格类型，在单元格级别上支持全面的客户定制，提供单元格合并、多表头、320 种内建的计算函数、单元格提示和注释、浮动的公式条、动态公式范围选择、输入智能提示等。

多工作表

在一个工作簿中支持多个工作表，用多个工作表来对信息进行分类。这与 Excel 的工作表 (worksheet) 类似。

定制的外观 (皮肤)

通过预先定制的皮肤来轻松、快速地配置工作表的外观。定制的皮肤可在开发团队中共享，以实现控件外观的跨应用程序的一致性。

单元格合并

支持多个单元格合并。通过单元格合并将多个单元格合并到一起。支持对数据单元格、表头的合并，还支持将数值相同的单元格进行自动合并。

表头中包含多列和多行

您还可以合并表头单元格，用多列或多行表头对您的数据进行组织。

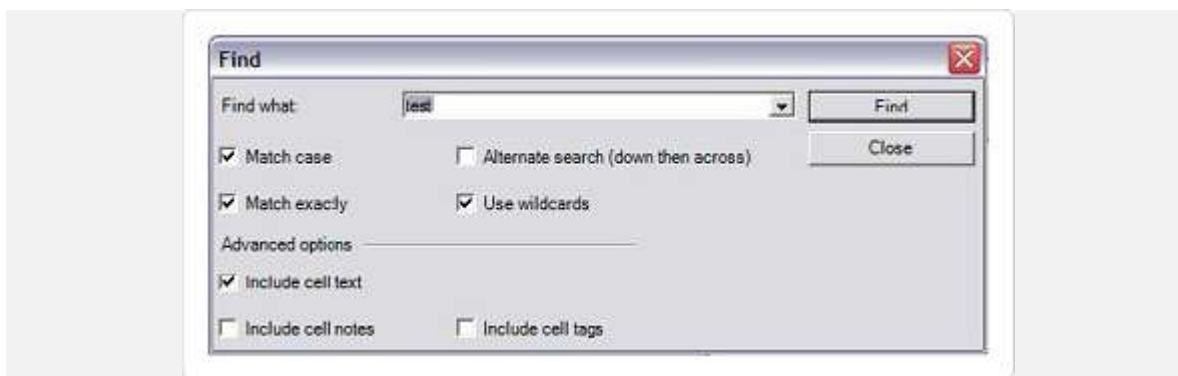
排序功能

支持对多行或多列数据进行排序，或对某个范围内的数据进行排序。可以通过代码实现，也可以通过用户点击列表头进行自动排序。

对行和列的交互进行定制

- 多行或多列的移动：允许用户拖动多行或多列。
- 多行或多列的冻结：支持冻结一个工作表中的任意数量的行或列。被冻结的行会一直置顶，冻结的列会始终位于最左侧。
- 行或列的缩放：支持对一个控件内的行或列的尺寸进行调整。

搜索数据



支持通过指定工作表和待搜索数据串，在工作簿中的任何单元格中搜索数据。

数据的过滤

支持对过滤条件的定制，只显示满足条件的数据。可在过滤的基础上，改变数据的显示。

用户与单元格之间的交互

支持对用户与单元格（或某个范围的单元格）之间的操作进行控制：

- 允许用户把数据从一个范围的单元格拖动到另一个范围。您可以规定用户是否可以选择一个单元格或某个范围内的单元格并拖动到同一个工作表或另一个工作表内的新位置。
- 允许用户将数据从一个单元格或某个范围内的单元格拖动并填充到另一个单元格或单元格范围。在选择了一个单元格或某个范围的单元格后，您可以对位于一行（或多行）或一列（或多列）内的单元格进行填充。
- 支持为单元格或一个范围内的单元格添加注释。
- 支持为单元格或一个范围内的单元格添加标签。
- 支持将单元格或一个范围内的单元格锁定。支持为锁定的单元格设置不同的外观（字体、颜色），以区别于其它单元格。
- 在单元格内设置条件格式，根据所设定的条件来确定单元格的格式化方式。例如可设定当数值小于 0 时，让其字体颜色变为红色等等。
- 使用 ButtonDrawMode 属性来定制单元格是否可以显示按钮。

数据选择

可以控制用户可以选择什么，以及外观是什么样的。还可以控制用户是否只能选择：

- 单元格
- 行
- 列
- 工作表
- 单元格，单元格范围，多个单元格范围
- 行，不能编辑
- 行，可以编辑
- 多个连续行，不能编辑

- 多个非连续行，不能编辑

丰富的单元格类型

Spread 支持丰富的单元格类型。通过使用 Spread 提供的内置单元格类型或自己定制的单元格类型，确定在一个单元格中可以输入什么样的数据，避免程序员不必要的检查和验证，并为用户提供一种自然的输入数据的方式。

分层显示

支持在一行内创建一个子工作表，以分层显示关系型数据，用父行和子行显示相关的数据。

内置的函数

Spread 的内置函数超过 200 多种。Spread 还支持通过内置函数和运算符来编制公式。支持的函数包括日期、时间函数、工程计算函数、财务计算函数、逻辑函数、数学和三角函数、统计函数、文本函数等。Spread 支持：

- 在单元格中放置公式
- 在公式中指定单元格索引
- 在公式中使用循环引用
- 在公式中嵌入函数
- 公式的自动重新计算和更新
- 允许用户输入公式
- 自定义函数
- 为自定义函数创建名称

打印

支持对表单任意一部分的打印。支持可缩放的预览功能。

- 通过调用 `FpSpread.PrintSheet` 方法，打印一个工作表或指定工作表的指定区域数据，打印所有页或指定页，打印分层工作表中的子表，并对其打印方式进行控制。
- 通过设置 `PrintInfo` 对象的属性，对打印进行定制。可以提供表头和表尾文字，使之出现在打印的每一页面上。还可以打印背景图形或水印。同时，Spread 还可自动确定最适合的打印方式。
- 支持最终用户对打印进行预览功能。

数据绑定

支持与数据集的绑定。数据集可以是一个数据库中的数据集或任何 .NET 框架允许的数据集如 `IList` 对象。Spread 支持：

- 绑定到一个数据集
- 在工作表中增加一个非绑定行
- 为绑定的工作表对列表头进行定制
- 为绑定的工作表对单元格类型进行定制
- 对列和字段的绑定进行定制
- 支持分层数据显示

多观察窗口

工作表支持一个以上的观察窗口，以便在不同的观察窗口中显示来自工作表的不同部分的数据。支持最终客户对各观察窗口的显示进行定制。

Spread 设计器

Spread 设计器用来设计并快速创建一个工作表原型。Spread 设计器通过其直观、易用的界面，使您在设计阶段能对表单进行各方面的定制，从而缩短开发时间。

Spread 设计器可为表单创建一个快照。当所有的更改完成后，这些更改可直接应用于工作表。Spread 设计器支持打开已有的设计文件并将您的设计更改保存进入该设计文件或保存为新的设计文件。

导入和导出的功能

Spread 支持多种数据格式的导入、导出。不仅在设计状态下可通过 Spread 设计器来导入、导出数据，在运行状态下，您也可以通过代码，把所有、指定表单或特定单元格范围内的数据导入、导出为不同的文件类型或流文件。

导出文件类型：

- Spread 的 XML 文件
- Excel 文件
- 文本文件

导入文件类型：

- Spread XML 文件
- Excel 文件
- Spread 文件
- 文本文件

定制光标显示类型

支持在工作表的不同区域显示不同的光标类型。如您可为锁定的单元格区域指定一种特殊的光标类型，为非锁定单元格区域指定另一种光标类型。

允许用户进行缩放操作

允许用户对表单的显示比例进行更改，即放大和缩小。可通过设置 AllowUserZoom 属性为 true 来实现之。此时用户可通过按下 Ctrl 键和转动鼠标滚轮来放大和缩小表单。

为表单增加一个上下文菜单

您可通过设置 ContextMenu 属性，创建一个自定义的右键菜单。

滚动条

支持滚动条的定制：

- 自定义是否在行和列的两端都显示滚动条
- 自定义滚动条的维数

- 自定义在用户拖动滚动框时，表单中的数据是否联动
- 自定义可滚动的行数、列数
- 自定义滚动时的对齐方式

2.3 Spread 5.0 for Windows Forms 的主要新增功能

使用 Spread Chart 图形化显示工作表数据

- 完全集成的图表支持：通过超过 85 种不同类型的 Chart 和内置的用户界面，为创建和定制 Chart 提供了全面支持。
- 使用与 Spread 集成的 Chart 图形化显示您的数据。
- 使用独立于 Spread 的 FpChart 图形化显示您的数据。

使用 Chart 设计器和预置的菜单、对话框, 不写代码，创建、操作 Spread Chart

- 使用“Change Chart Type and Select Data”对话框，在设计时创建 Chart。最终用户也能够使用之在运行时修改 Chart。
- 在设计时或运行时使用 Chart Designer 定制 Chart。

增强与 Microsoft Excel 兼容的表单功能

- 数据缓存提供了对包含宏、脚本代码、以及以前不支持内容的 Microsoft Excel 文件的无损编辑。
- 为 Microsoft Excel 支持了导出图形和图表。
- 支持导入和导出用 ISO-OOXML 兼容性密码(Excel2007 新功能)的 Excel2007 表单。



使用新的 Spread 设计器可以简化您的开发同时您也可以设计出强大的用户界面。

- 类似于 Microsoft Excel 2007 的新彩带栏界面。
- 在头单元格上可以设置多行文本。
- 在设计时可以设置 Microsoft Outlook 风格的分组。
- 设置条件格式。
- 支持对整个工作表的剪切复制。



利用新的快速启动向导 (Quick Start Wizard) ，帮您简单、快速地设计出强大的用户界面。您可以用它：

- 设置数据绑定
- 设置行和列的绑定和非绑定属性
- 其它更多的属性



使用以下新函数来进行高级运算

- Averagelf
- Averagelfs

- Countifs
- IfError
- Match
- Search
- Sumifs
- Text

利用如下新改进的易用性功能增强用户体验

- 排序数字列过滤列表
- 显示当前行图标
- 移动多行或多列
- 增强的鼠标与单元格内的按钮交互功能
- 自定义滚动增量
- 使用新的显示选项来显示表单标题、字幕、行预览和滚动
- 增强的剪贴板
- 表单缩放时的选择、移动、设置图形、图表的支持
- 在一个超链接单元格内设置多个超链接同时也可以修改显示字符串
- 在表单的灰色区域（最后行和最后列以外）自定义光标
- 新的列脚和分组列脚的计算和数据格式化

对现有功能的增强：

- 编辑模式下增加了更多控制
- 改进了表单导出文本的功能
- 增强了日期和时间类型的数据输入
- 在列表框单元格中可以选择多个列表项
- 在单元格和注释中可以自定义显示文本
- 在单元格中可以显示按钮
- 在按钮类型的单元格中可以显示提示
- 改进了空值单元格的显示
- 在单元格的任意角位置都可以显示单元格批注
- 在任何行头可以显示每列的列排序和列过滤图标
- 自定义显示的图像为展开/折叠、行选择器、列过滤、列排序、和新的工作表选项卡等功能的显示图标
- 增强了 Outlook 风格的数据分组

3. 单元格操作

3.1 设置单元格类型

Spread Win 5 支持 22 种单元格类型：

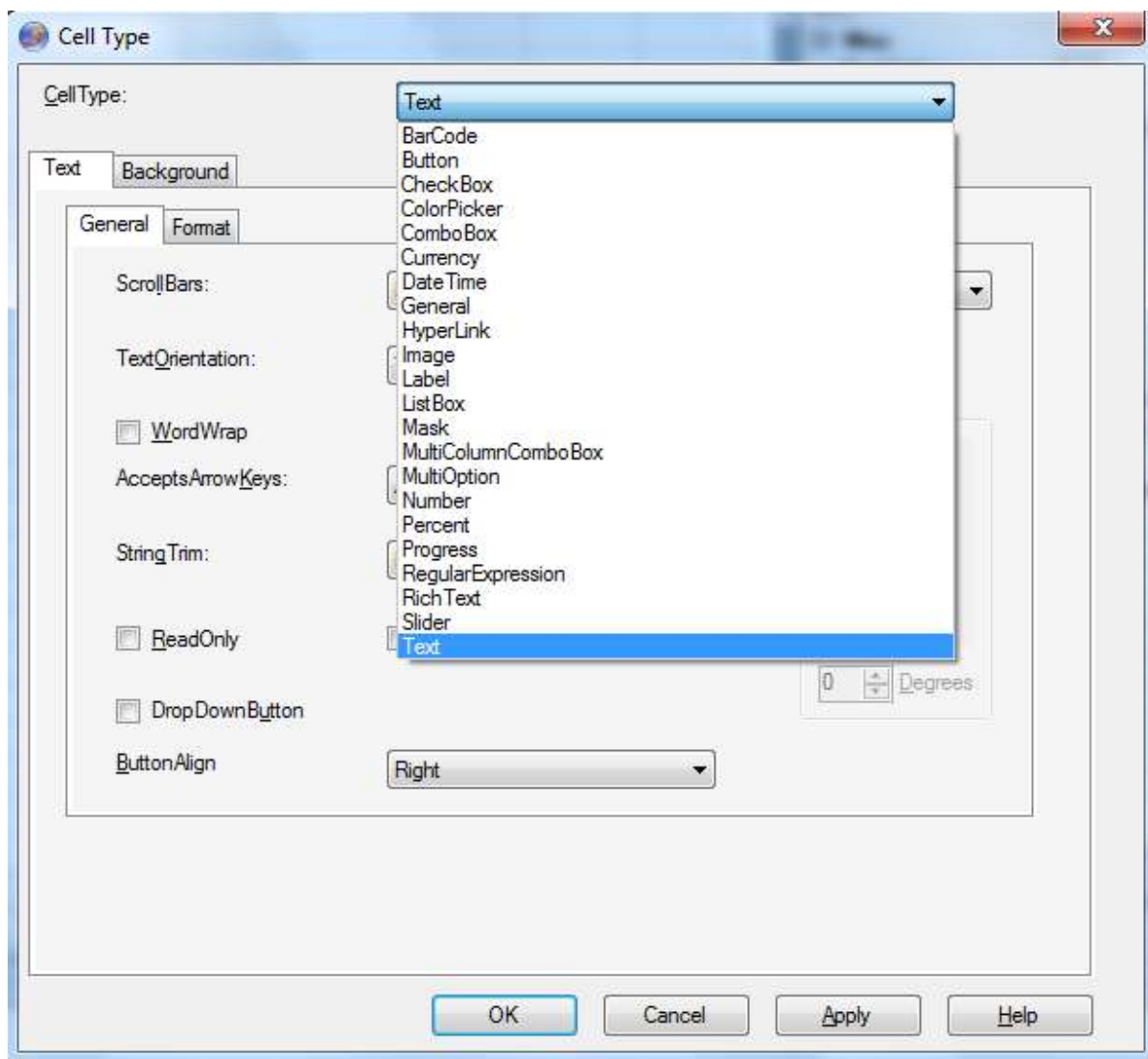
- 1) 普通型
- 2) 货币型
- 3) 日期-时间型
- 4) 数值型
- 5) 百分比型
- 6) 标签型
- 7) 文本型
- 8) 格式文本型
- 9) 单选型
- 10) 按钮型
- 11) 开关型
- 12) 列表框型
- 13) 组合框型
- 14) 多列组合框型
- 15) 超链接型
- 16) 图像型
- 17) 掩码型
- 18) 进度条型
- 19) 颜色选择型
- 20) 滑块型
- 21) 正则表达式型
- 22) 条形码型

您能够使用 Spread 设计器或者通过编码设置它们。

使用 Spread 设计器

使用 Spread 设计器的“Cell Type”对话框设置单元格类型：

您能够通过 Spread 设计器的右键菜单“Cell Types”的子菜单，或者通过 Ribbon Bar 的 Home 标签页上的 CellType 按钮，打开如下“Cell Type”对话框，进行各种单元格类型的设置：



使用编码

您能够使用编码设置各种单元格类型。下面是设置货币型单元格的一个例子：

1. 使用 CurrencyCellType 类，定义一个货币类型单元格。
2. 通过设置 CurrencyCellType 对象的 CurrencySymbol 和其他属性来指定货币单元格格式。
3. 通过设置单元格、单元格范围、列、或行的 CellType 属性把一个单元格或单元格区域的单元格类型设置成货币类型。

示例代码：

[C#]

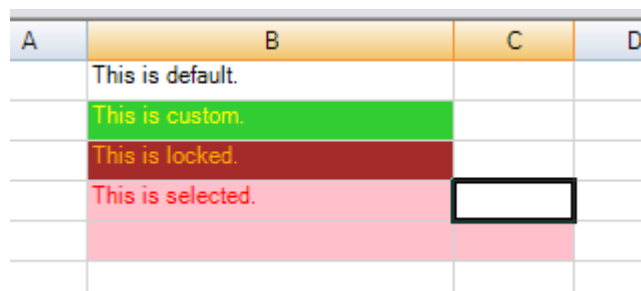
```
FarPoint.Win.Spread.CellType.CurrencyCellType currcell =  
new FarPoint.Win.Spread.CellType.CurrencyCellType();  
currcell.CurrencySymbol = "US$";  
currcell.DecimalSeparator = ".";  
currcell.DecimalPlaces = 8;  
fpSpread1.ActiveSheet.Cells[1,1].CellType = currcell;
```

[Visual Basic]

```
Dim currcell As New FarPoint.Win.Spread.CellType.CurrencyCellType()  
currcell.CurrencySymbol = "US$"  
currcell.DecimalSeparator = "."  
currcell.DecimalPlaces = 8  
FpSpread1.ActiveSheet.Cells(1,1).CellType = currcell
```

3.2 设置单元格的顏色

您可以设置一个或一组单元格的背景和前景（文字）的颜色。下图的例子说明了使用不同方式来设置单元格颜色。



A	B	C	D
	This is default.		
	This is custom.		
	This is locked.		
	This is selected.		

您可以使用当前单元格的 `BackColor` 属性用代码设置一个单元格的背景色，也可以使用 `ForeColor` 属性用代码来设置文本颜色。

同样地您也可以使用表单的 `SelectionBackColor` 和 `SelectionForeColor` 来设置选择单元格的这些颜色。同时您也可以使用 表单或者 Appearance 对象的 `LockBackColor` 和 `LockForeColor` 属性为锁定的单元格设置一个不同的颜色(背景色或者是文本颜色)。

使用代码

这段代码为第二个单元格设置背景色和前景色，为锁定的和选择的单元格设置颜色。

[C#]

```
fpSpread1.ActiveSheet.Cells[0,1].Value = "This is default.";
fpSpread1.ActiveSheet.Cells[1,1].Value = "This is custom.";
fpSpread1.ActiveSheet.Cells[2,1].Value = "This is locked.";
fpSpread1.ActiveSheet.Cells[3,1].Value = "This is selected.";
```

```
fpSpread1.ActiveSheet.Cells[1,1].BackColor = Color.LimeGreen;
fpSpread1.ActiveSheet.Cells[1,1].ForeColor = Color.Yellow;
```

```
fpSpread1.ActiveSheet.Cells[2,1].Locked = true;
fpSpread1.ActiveSheet.Protect = true;
fpSpread1.ActiveSheet.LockBackColor = Color.Brown;
fpSpread1.ActiveSheet.LockForeColor = Color.Orange;
```

```
fpSpread1.ActiveSheet.SelectionStyle = FarPoint.Win.Spread.SelectionStyles.SelectionColors;
fpSpread1.ActiveSheet.SelectionPolicy = FarPoint.Win.Spread.Model.SelectionPolicy.Range;
fpSpread1.ActiveSheet.SelectionUnit = FarPoint.Win.Spread.Model.SelectionUnit.Cell;
fpSpread1.ActiveSheet.SelectionBackColor = Color.Pink;
fpSpread1.ActiveSheet.SelectionForeColor = Color.Red;
```

[Visual Basic]

```
FpSpread1.ActiveSheet.Cells(0,1).Value = "This is default."
FpSpread1.ActiveSheet.Cells(1,1).Value = "This is custom."
FpSpread1.ActiveSheet.Cells(2,1).Value = "This is locked."
FpSpread1.ActiveSheet.Cells(3,1).Value = "This is selected."
```

```
FpSpread1.ActiveSheet.Cells(1,1).BackColor = Color.LimeGreen
FpSpread1.ActiveSheet.Cells(1,1).ForeColor = Color.Yellow
```

```
FpSpread1.ActiveSheet.Cells(2,1).Locked = True
FpSpread1.ActiveSheet.Protect = True
FpSpread1.ActiveSheet.LockBackColor = Color.Brown
FpSpread1.ActiveSheet.LockForeColor = Color.Orange
```

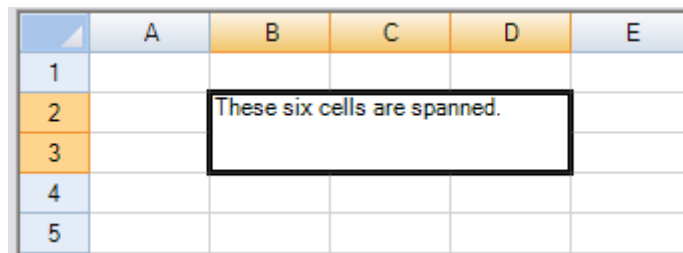
```
FpSpread1.ActiveSheet.SelectionStyle = FarPoint.Win.Spread.SelectionStyles.SelectionColors
FpSpread1.ActiveSheet.SelectionPolicy = FarPoint.Win.Spread.Model.SelectionPolicy.Range
FpSpread1.ActiveSheet.SelectionUnit = FarPoint.Win.Spread.Model.SelectionUnit.Cell
FpSpread1.ActiveSheet.SelectionBackColor = Color.Pink
FpSpread1.ActiveSheet.SelectionForeColor = Color.Red
```

使用 Spread 设计器

1. 在工作区域，选择一个或一组您想设置颜色的单元格。
2. 在属性列表中(在 Misc 分类中)选择 BackColor 属性来设置背景色。
3. 点击下拉按钮来显示颜色选择器，选择一个可用的颜色。
4. 设置前景色的话请重复以上步骤，选择属性列表的 ForeColor 属性。
5. 在“文件”菜单中先择“应用并退出”来应用组件的改变并退出 Spread 设计器。

3.3 合并单元格

您可以通过组合一些单元格来创建一个更大的合并单元格。比如，假如您想创建一个从 B2 到 D3 的合并单元格，单元格 B2 将占用从 B2 到 D3 的空间来显示这个合并的单元格。



	A	B	C	D	E
1					
2		These six cells are spanned.			
3					
4					
5					

Spread 工作表分为四个区域：表单交叉区，列标题区，行标题区和数据区。您可以在一个区域内创建合并的单元格，但您不能跨区域来创建合并的单元格，您不能在数据区域和行标题区域之间创建一个合并的单元格，也不可以在表单交叉区域和列标题之间创建一个合并的单元格。下面主要讨论在数据区合并单元格。

当您创建一个合并单元格，在合并单元格中的第一个单元格（称为锚定格）中的数据占用所有的合并空间。当您创建一个合并单元格，其中每一个单元格的数据仍然在每个单元格中，但不显示。这些数据只是隐藏在合并范围内。如果您删除了一组单元格的合并，其中以前是隐藏的单元格的内容将显示。通过调用 `AddSpanCell` 方法来创建一个合并单元格。在合并范围内的单元格类型没有改变。合并的单元格类型将是左上方单元格类型。

您可以用 `GetSpanCell` 方法来获得指定的单元格是否在一个合并单元格内。

您可以通过调用 `RemoveSpanCell` 方法来删除一个合并单元格，您也可以调用 `RemoveSpanCell` 方法同时指定合并单元格内的锚定格来删除一个合并单元格。当您删除一个合并单元格，先前合并单元格内的数据将重新显示在单元格中，数据并没有被删除，只是被简单地隐藏在了合并单元格。

注意：在一个排序表单内，合并单元格是不会显示的。当表单的任意部分使用 `SortRange` 方式排序以后，合并区域将会隐藏。如果单元格区间内包含合并单元格，就不能使用 `SortRange` 来排序。

在锚定单元格上设置的属性（包括单元格批注）应用到合并单元格。如果您在合并单元格内非锚定单元格中设置单元格批注，单元格批注将不显示。当您创建合并单元格，在合并中的第一个单元格中的数据（指定的 `Col` 和 `Row` 参数）将占有所有合并空间。当您创建一个合并单元格时，原始的每一个单元格仍然存在，但不显示，其数据也只是隐藏。另外，合并单元格中原始的每个单元格类型也没有改变。`GetCellSpan` 方法只能在当前被选择的表单中被调用。当您设置了当前表单后，这个方法也起作用。

调用 `GetCellSpan` 方法会返回指定的单元格是否在一个合并的单元格内。如果它在一个合并的单元格内，该方法将返回一个 `CellRange` 对象，其中包含锚定单元格的行和列以及合并区域内的行数和列数。这个方法只能在当前选择的表单中被调用。当您设置了当前表单后，这个方法也起作用。

如果一个合并的列与一个合并单元格交叉，合并的列将会替换合并的单元格。所以在合并的时候不

要合并那些已经是合并单元格的一部分的单元格。

使用代码

您能够使用如下方法来合并单元格或者删除合并的单元格:

- AddSpanCell, GetSpanCell, 和 RemoveSpanCell
- AddColumnHeaderSpan 和 AddRowHeaderSpan

示例: 调用 Sheets 对象的 AddSpanCell 方法来合并单元格

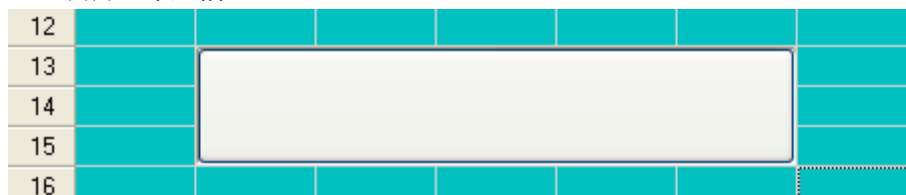
下面的代码定义了一些单元格的内容然后合并 6 个相邻的单元格。

```
[C#]
//设置 2 个单元格的文本。
fpSpread1.ActiveSheet.Cells[1,1].Text = "These six cells are spanned.";
fpSpread1.ActiveSheet.Cells[2,2].Text = "This is text in 2,2.";
// 合并 6 个包含不同内容的单元格
fpSpread1.ActiveSheet.AddSpanCell(1, 1, 2, 3);
[Visual Basic]
' 设置 2 个单元格的文本。
fpSpread1.ActiveSheet.Cells(1,1).Text = "These six cells are spanned."
fpSpread1.ActiveSheet.Cells(2,2).Text = "This is text in 2,2."
' 合并 6 个包含不同内容的单元格
fpSpread1.ActiveSheet.AddSpanCell(1, 1, 2, 3)
```

使用 Spread 设计器

1. 在电子表单上选择一些单元格准备合并。
2. 点击鼠标右键然后选择合并。
3. 另一种方式是,在属性列表(在 Misc 分类中), 选择 RowSpan 或者 ColumnSpan 属性,设置一个大于 1 的数字去合并单元格。 如果想要删除合并的单元格,则把其重新设为 1。

下图显示了合并的单元格。



12							
13							
14							
15							
16							

4. 从“文件”菜单选择“应用并退出”来应用变更到 Spread 然后退出 Spread 设计器。

3.4 锁定、解锁单元格

您可以锁定一个单元格或者一个范围内的单元格,使得它们不能再被最终用户编辑。您可以给这些

锁定的单元格设置不同的外观以使用户注意到它们。

您可以使用 Cell, Column, Row 或者 AlternationRow 这些对象的 `Locked` 属性来锁定单元格。您也可以使用 `StyleInfo` 对象的 `Locked` 属性，然后应用格式到您想要锁定的单元格。为使锁定得到效果，同时您必须设置 `SheetView` 对象的 `Protect` 属性为 `True`。

`Locked` 属性标记了这个单元格是否被锁定，而 `Protect` 属性设置了是否这个单元格需要被锁定。当单元格被锁定以防止用户输入时，`Protected` 属性必须设置成 `True`。`Protected` 的默认值也是 `True`。如果设置 `Protected` 为 `False`，用户依旧可以和单元格交互。

另一种锁定单元格的方式是设置文本框单元格(使用 `TextCellType`)的 `ReadOnly` 属性。这样也可以让单元格不能被编辑。

您可以使用 `SheetView` 或者是 `Appearance` 对象的 `LockBackColor`, `LockForeColor`, 和 `LockFont` 属性在锁定的单元格内指定一种不同的颜色(设置背景或者是文本)或者字体。

锁定单元格不会锁定任何单元格之上的图形(浮动对象)。一个受保护的表单意味着表单中所有被标记成被锁定的单元格都会被锁定，但不会应用到表单的图形上。

使用代码

使用单元格、行或列的 `Locked` 属性 您可以把单元格标记成被锁定的。如果您想要单元格被锁定来阻止用户输入，`SheetView` 的 `Protect` 属性必须设成 `True`。

示例

```
[C#]
fpSpread1.ActiveSheet.Protect = true;

fpSpread1.ActiveSheet.LockBackColor = Color.LightCyan;
fpSpread1.ActiveSheet.LockForeColor = Color.Green;

fpSpread1.ActiveSheet.Columns[0, 3]. Locked = true;

fpSpread1.ActiveSheet.Cells[1,1,1,2]. Locked = false;
[Visual Basic]
FpSpread1.ActiveSheet.Protect = True

FpSpread1.ActiveSheet.LockBackColor = Color.LightCyan
FpSpread1.ActiveSheet.LockForeColor = Color.Green

FpSpread1.ActiveSheet.Columns[0, 3].Locked = True

FpSpread1.ActiveSheet.Cells[1,1,1,2]. Locked = False
```

使用 Spread 设计器

1. 在工作区，通过拖拽一片区域的单元格或者选择一行或者一列的头来选择一些您想要锁定的单

元格。

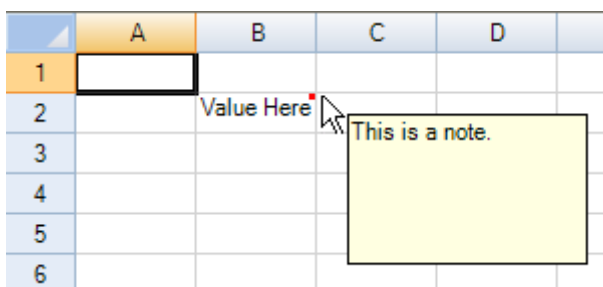
2. 在属性列表(在 Misc 分类中)选择 Locked 属性，然后选择 True (另一种方式是选择 Cells 属性，点击按钮以弹出 “Cells,Columns, and RowsEditor”来选择编辑器中的单元格)。
3. 点击包含单元格的电子表单的名字标签，从属性列表(在行为分组中)选择 Protect 属性设置为 True。

从 “文件” 菜单选择 “应用并退出” 来应用变更到组件然后退出设计器。

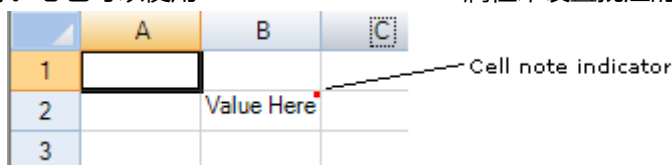
3.5 给单元格添加批注

您也可以给一个单元格或者一个范围内的单元格添加批注。这个批注包含一些诸如注释，问题或者文档性的描述来说明单元格的原始数据。在单元格的右上角有一个批注图标来说明这个单元格包含批注(默认是一个小的红色方块)。当鼠标划过批注图标时，批注将会以一个方框的形式显示在单元格的旁边以显示批注内容。另外您也可以让单元格批注始终显示而不仅仅在鼠标划过批注图标时显示。单元格批注是一种弹出式的批注，它们的显示方式类似于文本提示。

如下图所示：

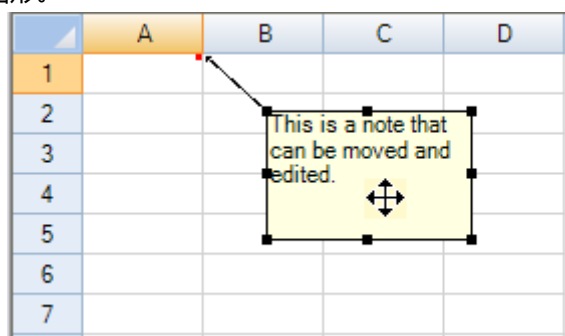


单元格右上角的红色方块说明那里有一个批注。您可以用 Spread 的 `CellNoteIndicatorVisible` 属性来隐藏这个单元格批注图标。您也可以使用 `NoteIndicatorPosition` 属性来设置批注的位置。



定制批注的行为

您可以让批注就像即时贴一样一直显示。这时会有一个矩形框将会出现在单元格的旁边。同时还有一根延长线连接批注和单元格。这根延长线也可以让用户移动批注。下图显示了一个选中的像即时贴一样的批注。为了得到此种效果，单元格的 `NoteStyle` 属性必须设置为 `StickyNote`。在这个例子中即时贴批注是一个可以被移动的图形。



为了移动批注，当鼠标停留在批注上时，鼠标左键按下，拖拽到目标位置即可。从批注图标出发的那根线会自适应延伸到批注的位置。

如果批注始终显示，用户就可以编辑批注。为了让用户编辑批注，您必须设置表单的 `AllowNoteEdit` 属性为 `True`，来允许用户编辑一直显示的单元格批注。

单元格批注包含更多的可读信息给最终用户，也可以允许用户在单元格批注中附加说明自己的信息。这些资料对最终用户来说可能是十分有意义的。比如，最终用户可能会利用单元格的批注，以显示单元格的原始值的来源（比如单元格批注=“值是从一个消费者报告杂志 7 月号上的文章获得”）。

单元格批注还包含如下功能：

- 根据内容批注大小自动调整
- 定制单元格批注的位置
- 单元格的批注可以被放置在任何位置
- 定制单元格批注图标 (见下节)
- 打印单元格批注

您也可以使用标准批注的 `NoteStyleInfo` 类还有固定批注的 `StickyNoteStyleInfo` 类来定制单元格批注的外观。

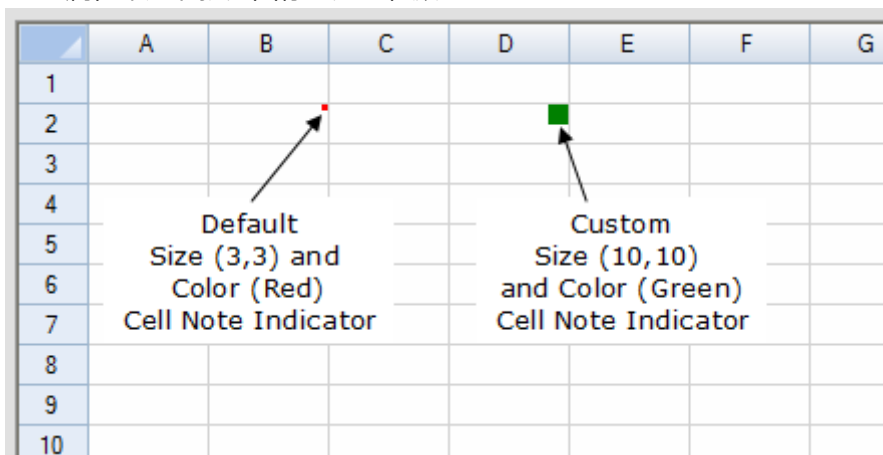
限制

当使用和显示单元格批注时有一些限制：

- 当单元格对象的 `NoteStyle` 属性被设置成隐藏时，批注将不会显示。
- 当单元格类型的 `IsReservedLocation` 被设置成 `true` 以后，批注不能显示在一些单元格类型上。这可能发生在 `Checkbox` 单元格，或者是不能编辑的 `ComboBox` 单元格或者是当鼠标滑过 `hyperlink` 单元格的超链接时。
- 单元格批注图标在单元格编辑时不会出现。
- 在锚定单元格上的批注会显示在一个合并单元格内。但是合并单元格内其他单元格的批注将不会显示。
- 当单元格包含一个红色批注时请谨慎选择红色作为单元格的背景。红色的批注图标在红色背景下是显示不出来的。

定制批注图标

您可以改变批注图标的尺寸和颜色。批注图标的默认大小是 3x3 的方块。您可以设置 NoteIndicator 的宽和高为任意的正整数值。批注图标的默认颜色是红色。不过您可以设置任意颜色给它。下图显示了一个使用默认值和一个使用自定义尺寸和颜色的批注图标。使用 NoteIndicatorColor 和 NoteIndicatorSize 属性来定制批注图标的大小和颜色。



使用代码示例

如下示例代码设置了一个可编辑的批注给一片区域的单元格然后把批注图标的颜色设置成了绿色(从红色)。

```
[C#]
fpSpread1.Sheets[0].AllowNoteEdit = true;
fpSpread1.Sheets[0].Cells[1, 1, 3, 3].Note = "test";
fpSpread1.Sheets[0].Cells[1, 1, 3, 3].NoteIndicatorColor = Color.Green;
fpSpread1.Sheets[0].Cells[1, 1, 3, 3].NoteStyle = FarPoint.Win.Spread.NoteStyle.StickyNote;

[Visual Basic]
FpSpread1.Sheets(0).AllowNoteEdit = True
FpSpread1.Sheets(0).Cells(1, 1, 3, 3).Note = "test"
FpSpread1.Sheets(0).Cells(1, 1, 3, 3).NoteIndicatorColor = Color.Green
FpSpread1.Sheets(0).Cells(1, 1, 3, 3).NoteStyle = FarPoint.Win.Spread.NoteStyle.StickyNote
```

使用 Spread 设计器

1. 在工作区域内，选择一个或者是几个您想设置批注的单元格。
2. 在属性列表(在 Misc 分组中)中选择 Note 属性然后输入一些文本。
3. 另外一种方式是选择 Cells 属性然后点击按钮弹出 “Cells, Columns, and Rows Editor”来选择编辑器中的单元格。
4. 从 “文件” 菜单选择 “应用并退出” 来应用变更到组件然后退出 Spread 设计器。

3.6 设置公式

您可以给单元格或一片区域的单元格添加公式。您也可以给所有行或列的单元格添加公式。该公式

是一个用字符串来说明的公式表达式，通常包含一些功能、操作和常量的组合。

当设置行或列的公式时，您给该行或列指定了一个默认的公式。换句话说，该公式可以被用到行或列上任何的单元格（假设该公式不是在单元格级别上被覆盖）。对于行或列的公式，Spread 使用行或列中的第一个单元格为基础位置。如果使用相对寻址，该公式为列 A 的每个单元格计算出不同结果。如果您想要列 A 的每个单元格得到 C2 和 D2（不是列 C 和列 D 上所有行的数据）的总和，您可能需要用到表达式 $\$C\$2+\$D\2 ，这个表达式用了绝对路径。

您可以通过指定对象的 Formula 属性或在设计器上输入公式来为一个对象设置公式。如何使用代码来完成这个操作，请参考下面的内容。有关使用设计器来输入一个公式的说明，请参考在设计器中输入公式中的指南。您可以让最终用户键入等号，然后输入表达式来完成输入公式的操作。

在查找数据时，要小心单元格的类型，以及您是否使用 Text 或 Value 属性作为公式中的数据计算。当您使用 Text 属性计算单元格数据，电子表单使用单元格类型把指定的字符串解析成所需的数据类型。当您把单元格数据保存在 Value 属性上时，工作表接受指定的数据，也没有解析发生。所以如果您设定了一个字符串，它仍然是一个字符串。有些和数字相关的功能（如 SUM）会忽略在单元格区域中的非数字数值。例如，如果单元格区域 A1:A3 包含值（1，“2”，3），则公式 SUM（A1:A3）的值为 4，因为 SUM 函数忽略字符串“2”。要确保您设置的公式计算正确请使用正确的数据类型。

一个公式中包含的字符串常量可以包含诸如新行字符的特殊字符（即'\n'）。请确保您的公式中用正确的方式描述了引用的字符串常量。下面的 C# 代码创建了一个多行文本单元格，其设置的公式包含了一个字符串常量，一个换行字符。

```
TextCellType ct = new TextCellType();
ct.Multiline = true;
fpspread1.Sheets[0].Cells[0,0].Formula = "\"line1\nline2\"";
```

使用代码

这种使用代码添加公式的方式只有在运行时才能看到效果。您可以通过设置单元格、行或者列的 Formula 属性来为单元格、行或者列添加一个公式。

示例

下面这个 示例演示了从第一个单元格中查找第五次的 Product 的值，把结果放置到另外一个单元格中。然后计算出单元格区域(A1 到 A4)的总计把结果放到第三列中。

```
[C#]
FpSpread1.ActiveSheet.Cells[2, 0].Formula = "PRODUCT(A1,5)";
FpSpread1.ActiveSheet.Columns[3].Formula = "SUM(A1:A4)";

[Visual Basic]
FpSpread1.ActiveSheet.Cells(2, 0).Formula = "PRODUCT(A1,5)"
FpSpread1.ActiveSheet.Columns(3).Formula = "SUM(A1:A4)"
```

使用 Formula 编辑器

在运行时，您可以在 Spread 设计器中使用 Formula 条或者 Formula Editor 在单元格中输入一个公式。Formula Editor 也可以从 Properties Window 打开使用。关于创建公式的详细功能和操作，请参考

Spread 的 .NET Formula Reference。

1. 选择一个电子表单。
2. 选择电子表单的一个或多个单元格。
3. 在 Formula 属性上，点击箭头按钮可以打开 Formula Editor。
4. 在 Formula Editor 上，您可以在编辑框内键入公式。您可以在编辑框的某个特定功能名称上双击以帮助输入公式，您可以键入操作符和常量来完善您的公式。
5. 完成之后，点击“应用”，点击“OK”关闭编辑器。
6. 如果您在 Spread 设计器上工作，从“文件”菜单选择“应用并退出”来应用变更然后退出 Spread 设计器。

4. 行列操作

4.1 设置行列数

当您创建一个电子表单的时候，它会自动创建 500 行和 500 列。您可以把行数或者列数设置为 0 到 2000000 之间的任意数字。

更多信息，请参考 [SheetView.RowCount](#) 属性和 [SheetView.ColumnCount](#) 属性。

使用代码

为 Sheets 对象设置 ColumnCount 或者 RowCount 属性。

示例

这个例子演示了设置第一个电子表单的行数为 100,列数为 10。

```
[C#]
fpSpread1.Sheets[0].ColumnCount = 10;
fpSpread1.Sheets[0].RowCount = 100;
[Visual Basic]
FpSpread1.Sheets(0).ColumnCount = 10
FpSpread1.Sheets(0).RowCount = 100
```

使用 Spread 设计器

1. 选择您想设置行数或者列数的电子表单。
2. 在电子表单的属性列表里设置 ColumnCount 和 RowCount 属性。
3. 从“文件”菜单选择“应用并退出”来应用变更到组件然后退出 Spread 设计器。

4.2 移动行列

用户可以拖拽并移动行或列。设置 [AllowRowMove](#) 属性可以让用户移动行，设置 [AllowColumnMove](#) 属性可以让用户移动列。如果您想要用户移动多行或多列，请设置 [AllowRowMoveMultiple](#) 或者 [AllowColumnMoveMultiple](#) 属性。

用户如果想要移动行或列，他们只要简单地在头区域单击行头或列头来移动和拖动头向前或者向后移动，然后在目标头上释放鼠标(多行的话,首先要选择需要移动的行和列)。移动中的行或者列在鼠标下是半透明的。如下图所示，第四列准备移向左边。

	ID	CompanyName	ContactTitle	ContactTitle
1	1	Alfreds Futterkiste	Maria Anders	Sales Representative
2	2	Ana Trujillo Emparedados	Ana Trujillo	Owner
3	3	Antonio Moreno Taquería	Antonio Moreno	Owner
4	4	Around the Horn	Thomas Ashworth	Sales Representative
5	5	Berglunds snabbköp	Christina Berglund	Order Administrator
6	6	Blauer See Delikatessen	Hanna Christman	Sales Representative
7	7	Blondesddsl père et fils	Frédérique Citeaux	Marketing Manager
8	8	Bólido Comidas preparadas	Martín Chocón	Owner
9	9	Bon app'	Laure Coates	Owner
10	10	Bottom-Dollar Markets	Elizabeth Dolan	Accounting Manager
11	11	B's Beverages	Victor Dolan	Sales Representative
12	12	Cactus Comidas para llevar	Patricio Dominguez	Sales Agent
13	13	Centro comercial Moctez	Franco Escobedo	Marketing Manager
14	14	Chop-suey Chinese	Yang Feng	Owner

用代码移动行和列

用编程方式实现移动一或多行，可以使用 `SheetView.MoveRow` 方法。用编程方式实现移动一列或多列，可以使用 `SheetView.MoveColumn` 方法。如果您对模块比较熟悉，您可以使用 `DefaultSheetAxisModel.Move` 方法。

使用代码

设置 `FpSpread` 组件的 `AllowRowMove` 或 `AllowColumnMove` 属性。

示例

```
[C#]
fpSpread1.AllowRowMove = true;
fpSpread1.AllowColumnMove = true;

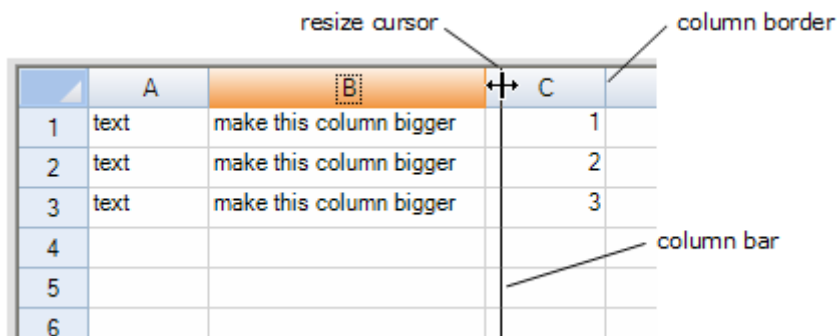
[Visual Basic]
FpSpread1.AllowRowMove = True
FpSpread1.AllowColumnMove = True
```

使用 Spread 设计器

1. 在设计器的右上角的下拉框里选择 Spread。
2. 在 Misc 分组里设置 `AllowRowMove` 或 `AllowColumnMove` 属性为 True 或 False。
3. 在“文件”菜单中，选择“保存并退出”保存变更。

4.3 调整行高、列宽

用户也可以调整表单中的行高和列宽。为调整行高、列宽，首先必须设置行或者列的 `Resizable` 属性为 `True`。如果用户想要调整行高或列宽，只需要点击行头或列头的边缘位置来调整和拖动头的边缘然后在合适的尺寸处释放鼠标。当鼠标左键按下，一条线就会随着鼠标移动。如下图所示。请注意该线是在行的右边和列的下边。双击行的边缘可以让行调整高度以显示行中最高文本。双击列的边缘可以让列调整宽度以显示列中最宽文本。



在默认情况下，用户可以调整行和列的数据区的大小，头区是不允许的。在代码中，可以调整行头和列头，而不仅仅是行和列的数据区。您可以使用 `Resizable` 的属性覆盖默认行为，防止用户调整大小。

下面的代码可以把行头上的第一列设置成可调整大小的：

```
spread.Sheets[0].RowHeader.Columns[0].Resizable = true;
```

下面的代码可以把行头上的所有列都设置成可调整大小的：

```
spread.Sheets[0].RowHeader.Columns.Default.Resizable = true;
```

您可以使用 `SheetView` 这个类上的如下方法来得到或者设置特定的行、列能否调整大小。

- `GetColumnSizeable`
- `SetColumnSizeable`
- `GetRowSizeable`
- `SetRowSizeable`

使用代码

设置行或者列的 `Resizable` 属性把第一行和第一列设置成可以调整大小的。

示例

```
[C#]
fpSpread1.Sheets[0].Columns[0].Resizable = true;
fpSpread1.Sheets[0].Rows[0].Resizable = true;

[Visual Basic]
FpSpread1.Sheets(0).Columns(0).Resizable = True
FpSpread1.Sheets(0).Rows(0).Resizable = True
```

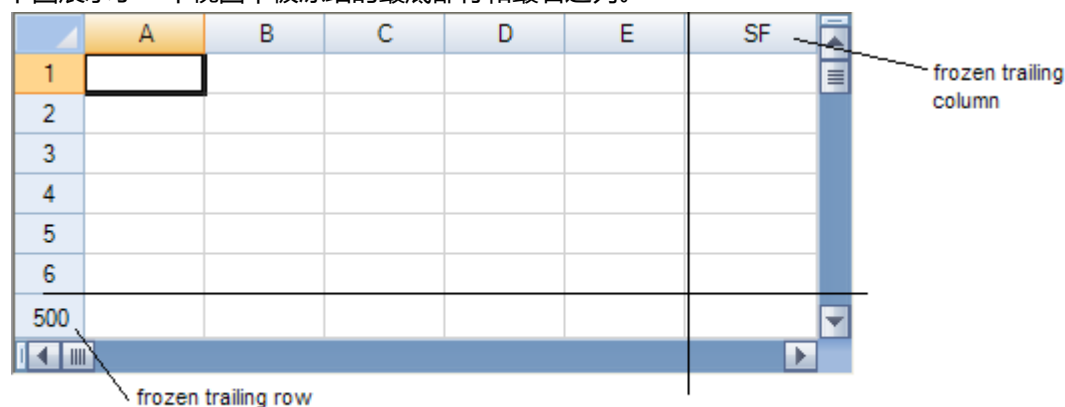
使用 Spread 设计器

1. 在设计器的右上角从下拉框中选择 Sheet。
2. 在外观分类中选择行或者选择列,然后单击 “Cells,Columns, and Rows Editor”按钮。
3. 在编辑器中选择一行或者一列。
4. 选择 Resizable 属性。
5. 从下拉框中选择一个值 (True, False)。
6. 在 “文件” 菜单中, 选择 “保存并退出” 来保存修改。

4.4 冻结行列

您可以冻结(使其不可滚动)一定的行数,列数。您可以冻结上面的任意行(被称为领先行),您也可以冻结最左边的任意列(被称为领先列)。您也可以冻结最底部的任意行数或者最右边的任意列数。冻结的领先行和领先列永远留在最顶部以及最左边,冻结的最底部的行(末尾行)和最右边的列(末尾列)也永远停留在那里,无论鼠标滚轮如何滚动。

下图展示了一个视图中被冻结的最底部行和最右边列。



与冻结行和冻结列相关的属性有：

- FrozenRowCount
- FrozenColumnCount
- FrozenTrailingColumnCount
- FrozenTrailingRowCount

在运行时,冻结行和冻结列是不能滚动的。但是在设计时,他们依然可以滚动。

底部的冻结行和右边的冻结列不能重复打印在每页的最底部和最右边,只能作为最后一行和最后一列打印。领先行和领先列可以重复打印。

领先行和领先列是一个独立的视图，末尾行和末尾列也是一个独立的视图。冻结视图的索引是：

```
-1 = leading frozen  
0 = first scrollable  
1 = second scrollable  
...  
n-1 = last scrollable  
n = trailing frozen
```

在级联显示中，冻结领先行和冻结领先列可以在子表单中使用。

使用代码

设置 Sheet 的 FrozenRowCount、FrozenColumnCount、FrozenTrailingColumnCount、或 FrozenTrailingRowCount 属性。

示例

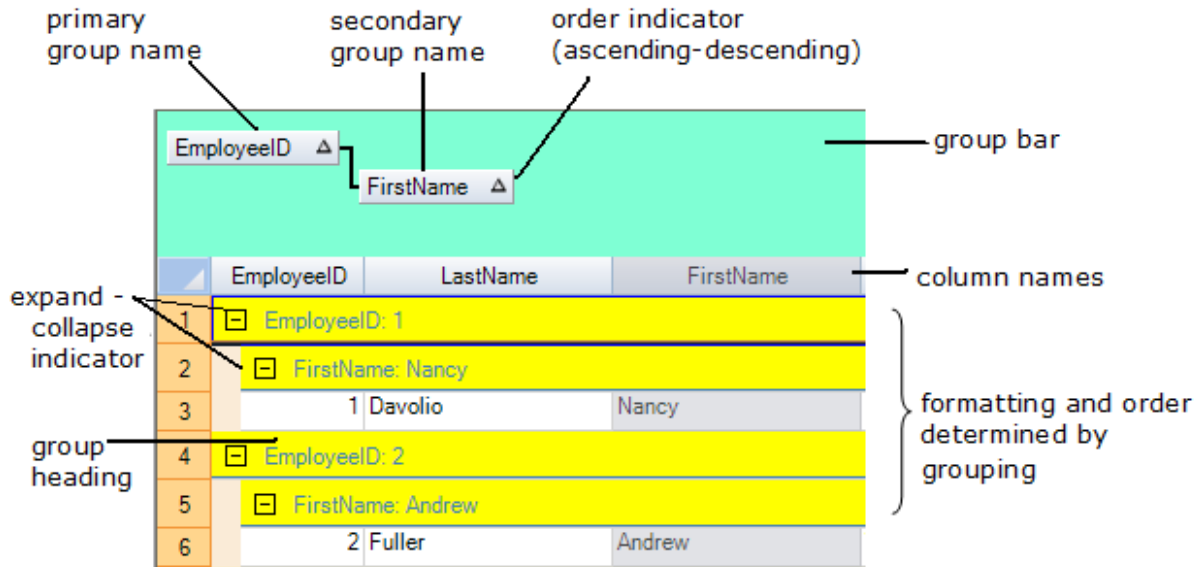
```
[C#]  
fpSpread1.Sheets[0].FrozenColumnCount = 2;  
fpSpread1.Sheets[0].FrozenRowCount = 2;  
fpSpread1.Sheets[0].FrozenTrailingColumnCount = 2;  
fpSpread1.Sheets[0].FrozenTrailingRowCount = 2;  
[Visual Basic]  
FpSpread1.Sheets(0).FrozenColumnCount = 2  
FpSpread1.Sheets(0).FrozenRowCount = 2  
FpSpread1.Sheets(0).FrozenTrailingColumnCount = 2  
FpSpread1.Sheets(0).FrozenTrailingRowCount = 2
```

使用 Spread 设计器

1. 在设计器的右上角的下拉框中选择 Sheet。
2. 在 SheetView 的 property 中，在 FrozenColumnCount、FrozenRowCount、FrozenTrailingColumnCount、或 FrozenTrailingRowCount 属性上键入数字。
3. 在“文件”菜单中，选择“保存和退出”来保存变更。

4.5 使用分组

您可以设置行像 Outlook 那样进行分组。在大数据量的时候，以用户需要的方式来显示数据对用户是非常有帮助的。用户选择想要排序的列，组件将以级联方式组织行和显示数据。如果要分组显示数据，请选择一列，双击列标题或者单击并拖拽列到页面顶端的分组栏中。下图展示了一个分组的例子。



您可以拖拽更多的列标题到分组区来生成多级分组以便排序数据。下图显示了两级的分组：您可以通过点击加号(+)或者减号(-)来展开和收起分组。

EmployeeID ▲

LastName

EmployeeID	LastName	First
1	EmployeeID: 1	
2	1 Davolio	Nancy
3	EmployeeID: 2	
4	2 Fuller	Andrew
5	EmployeeID: 3	
6	3 Leverling	Janet

在次要分组之前：拖拽列头到分组栏中。

EmployeeID ▲

FirstName ▲

EmployeeID	LastName	First
1	EmployeeID: 1	
2	FirstName: Nancy	
3	1 Davolio	Nancy
4	EmployeeID: 2	
5	FirstName: Andrew	
6	2 Fuller	Andrew
7	EmployeeID: 3	
8	FirstName: Janet	
9	3 Leverling	Janet

次要分组之后：现在第二层的级联显示。

当多个级别被选择，上级称为父组,下级称为子组。在上面的第二张图片，雇员 ID 是父组，名字是子组。

默认情况下，电子表格不允许出现行的分组。您可以打开此功能，对整个表的行分组。除了允许分组，您还需要允许列移动，因为用户通过单击并拖动到头分组栏来分组，这个行为类似移动列。此外，该分组栏必须是可见的，同时列头（至少 1 行）也应可见。

使用电子表单上的 `AllowGroup` 属性打开分组功能。使用电子表单的 `GroupBarVisible` 属性来显示分组栏（在电子表单的上方用户可以把列拖拽到这个区域）。别忘记把 `AllowColumnMove` 属性设成 `true` 来允许用户单击拖拽列标题。如果未默认设置，请设置 `ColumnHeaderVisible` 属性为 `true` 来保证列标题区域是可见的。

您可以打开或者关闭行头，在分组显示时，行头没有用。

您可以设置分组的最大级别。这个受最大可以拖拽到分组栏的列数限制。

想要了解更多关于分组，请看 [Using Grouping](#)。

使用代码

```
[C#]
FpSpread1.AllowColumnMove = true;
FpSpread1.ActiveSheet.GroupBarVisible = true;
FpSpread1.ActiveSheet.AllowGroup = true;

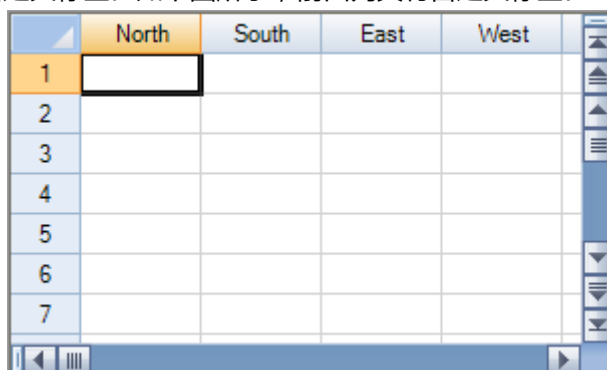
[Visual Basic]
FpSpread1.AllowColumnMove = True
FpSpread1.ActiveSheet.GroupBarVisible = True
FpSpread1.ActiveSheet.AllowGroup = True
```

示例

此例显示了如何允许分组功能：

4.6 定制行头、列头的文字

默认情况下 `Spread` 组件在列头显示字母、在行头显示数字。除了这种自动文本，您可以在任何或所有的头单元格上添加自定义标签。如下图所示，前四列具有自定义标签。



	North	South	East	West
1				
2				
3				
4				
5				
6				
7				

要给一个头单元格指定自定义文本，您可以使用该 `ColumnHeader.Column` 或 `RowHeader.Row` 的 `Label` 属性。也可以使用单元格的 `Text` 属性。对于多列头和多行头，您使用 `Cells` 的 `Text` 属性。更多信息，请参阅列标签属性或行标签属性。

在头上的单元格和数据区的单元格是分开的，因此在头单元格的坐标从 0,0 坐标开始。工作表交叉区域的单元格也是独立的，不计入头单元格的坐标。

使用代码

- 如果您想要改变头单元格的文本或者在头单元格中显示文本，请设置 ColumnHeader 的 Cell 的 Text 来定制您想要显示的文本。如果您想要设置多个头单元格的文本，请调用 ColumnHeader 的 SetClip 或 SetClipValue 方法。
- 您也可以调用 ColumnHeader 的 Column 的 Label 属性来定制您想要显示的文本。

示例

下例显示了定制前四列的标签。

```
[C#]
// 设置列 A 到列 D 的自定义文本.
fpSpread1.Sheets[0].ColumnHeader.Columns[0].Label = "North";
fpSpread1.Sheets[0].ColumnHeader.Columns[1].Label = "South";
fpSpread1.Sheets[0].ColumnHeader.Columns[2].Label = "East";
fpSpread1.Sheets[0].ColumnHeader.Columns[3].Label = "West";

[Visual Basic]
'设置列 A 到列 D 的自定义文本.
FpSpread1.Sheets(0).ColumnHeader.Columns(0).Label = "North"
FpSpread1.Sheets(0).ColumnHeader.Columns(1).Label = "South"
FpSpread1.Sheets(0).ColumnHeader.Columns(2).Label = "East"
FpSpread1.Sheets(0).ColumnHeader.Columns(3).Label = "West"
```

使用 Spread 设计器

1. 选择您想要定制头文本的表单。
2. 选择您想要定制的行或者列然后右键点击选择“Headers...”。
3. 在“Header Editor”中，双击您想要定制文本的头,如果只有一个头，点击一个有“<Default>”文本的单元格。
4. 编辑您想要显示的文本然后按 Enter 键终止编辑行为。
5. 当您编辑完所有您想编辑的头文本，点击 OK，关闭“HeaderEditor”然后单击 Yes 应用您的设置。
6. 在“文件”菜单中选择“保存和退出”来应用您的变更并退出 Spread 设计器。

4.7 设置多行行头、多列列头

您可以提供多行多列的列头和行头。如下面图所示，头可能有不同的列和行数。

The screenshot shows a spreadsheet with a multi-level header. The top row is a yellow header cell containing 'Fiscal Year 2004'. Below it, the first row of the data area has four blue header cells: '1st Quarter', '2nd Quarter', '3rd Quarter', and '4th Quarter'. The second row of the data area has eight blue header cells: 'East', 'West', 'East', 'West', 'East', 'West', 'East', and 'West'. The first column of the data area has a yellow header cell containing 'Branch #'. The data area consists of 9 rows and 8 columns. The cell at row 1, column 1 (under 'Branch #', '1st Quarter', 'East') is highlighted with a black border.

		Fiscal Year 2004							
		1st Quarter		2nd Quarter		3rd Quarter		4th Quarter	
		East	West	East	West	East	West	East	West
Branch #	1								
	2								
	3								
	4								
	5								
	6								
	7								
	8								
	9								

头区域的行和列同样能够包含合并单元格。

使用代码

设置 Sheets 对象的 ColumnHeaderRowCount 属性和 RowHeaderColumnCount 属性。使用 AddColumnHeaderSpanCell 方法来合并头单元格。使用 Label 和 Text 属性来添加头单元格的标签。

示例

下例包含一个电子表单，在行头上有 2 个列，在列头上有三行。

```
[C#]
// 设置行头列数和列头行数
fpSpread1.Sheets[0].ColumnHeaderRowCount = 3;
fpSpread1.Sheets[0].RowHeaderColumnCount = 2;
// 合并头上的单元格.
fpSpread1.Sheets[0].AddColumnHeaderSpanCell(1, 0, 1, 2);
fpSpread1.Sheets[0].AddColumnHeaderSpanCell(1, 2, 1, 2);
fpSpread1.Sheets[0].AddColumnHeaderSpanCell(1, 4, 1, 2);
fpSpread1.Sheets[0].AddColumnHeaderSpanCell(1, 6, 1, 2);
fpSpread1.Sheets[0].AddColumnHeaderSpanCell(0, 0, 1, 8);
fpSpread1.Sheets[0].AddRowHeaderSpanCell(0, 0, 12, 1);
// 使用 Label 或者单元格的 Text 属性设置需要的标签
fpSpread1.Sheets[0].ColumnHeader.Columns[0].Label = "East";
fpSpread1.Sheets[0].ColumnHeader.Columns[1].Label = "West";
fpSpread1.Sheets[0].ColumnHeader.Columns[2].Label = "East";
fpSpread1.Sheets[0].ColumnHeader.Columns[3].Label = "West";
fpSpread1.Sheets[0].ColumnHeader.Columns[4].Label = "East";
fpSpread1.Sheets[0].ColumnHeader.Columns[5].Label = "West";
fpSpread1.Sheets[0].ColumnHeader.Columns[6].Label = "East";
fpSpread1.Sheets[0].ColumnHeader.Columns[7].Label = "West";
fpSpread1.Sheets[0].ColumnHeader.Cells[0,0].Text = "Fiscal Year 2004";
fpSpread1.Sheets[0].ColumnHeader.Cells[1,0].Text = "1st Quarter";
fpSpread1.Sheets[0].ColumnHeader.Cells[1,2].Text = "2nd Quarter";
fpSpread1.Sheets[0].ColumnHeader.Cells[1,4].Text = "3rd Quarter";
fpSpread1.Sheets[0].ColumnHeader.Cells[1,6].Text = "4th Quarter";
// 设置行头以便显示
fpSpread1.Sheets[0].RowHeader.Columns[0].Width = 45;
fpSpread1.Sheets[0].RowHeader.Cells[0,0].Text = "Branch #";
```

```
[Visual Basic]
' 设置行头列数和列头行数
FpSpread1.Sheets(0).ColumnHeaderRowCount = 3
FpSpread1.Sheets(0).RowHeaderColumnCount = 2
' 合并头单元格
FpSpread1.Sheets(0).AddColumnHeaderSpanCell(1, 0, 1, 2)
FpSpread1.Sheets(0).AddColumnHeaderSpanCell(1, 2, 1, 2)
FpSpread1.Sheets(0).AddColumnHeaderSpanCell(1, 4, 1, 2)
FpSpread1.Sheets(0).AddColumnHeaderSpanCell(1, 6, 1, 2)
FpSpread1.Sheets(0).AddColumnHeaderSpanCell(0, 0, 1, 8)
FpSpread1.Sheets(0).AddRowHeaderSpanCell(0, 0, 12, 1)
' 使用 Label 或者单元格的 Text 属性设置需要的标签
FpSpread1.Sheets(0).ColumnHeader.Columns(0).Label = "East"
FpSpread1.Sheets(0).ColumnHeader.Columns(1).Label = "West"
FpSpread1.Sheets(0).ColumnHeader.Columns(2).Label = "East"
FpSpread1.Sheets(0).ColumnHeader.Columns(3).Label = "West"
FpSpread1.Sheets(0).ColumnHeader.Columns(4).Label = "East"
FpSpread1.Sheets(0).ColumnHeader.Columns(5).Label = "West"
FpSpread1.Sheets(0).ColumnHeader.Columns(6).Label = "East"
FpSpread1.Sheets(0).ColumnHeader.Columns(7).Label = "West"
FpSpread1.Sheets(0).ColumnHeader.Cells(0,0).Text = "Fiscal Year 2004"
FpSpread1.Sheets(0).ColumnHeader.Cells(1,0).Text = "1st Quarter"
FpSpread1.Sheets(0).ColumnHeader.Cells(1,2).Text = "2nd Quarter"
FpSpread1.Sheets(0).ColumnHeader.Cells(1,4).Text = "3rd Quarter"
FpSpread1.Sheets(0).ColumnHeader.Cells(1,6).Text = "4th Quarter"
' 设置行头以便显示
FpSpread1.Sheets(0).RowHeader.Columns(0).Width = 45
FpSpread1.Sheets(0).RowHeader.Cells(0,0).Text = "Branch #"
```

使用 Spread 设计器

1. 选择您想要设置多行头和多列头的电子表单。
2. 在属性列表中设置 ColumnHeaderRowCount 属性来指定您想要在列头中需要显示的行数。或者是 RowHeaderColumnCount 来指定您想要在行头中需要显示的列数。
3. 在“文件”菜单中选择“应用并退出”以应用变更并退出设计器。

5. 工作表操作

5.1 使用当前工作表

当前工作表是指目前可以和用户交互的表。您可以使用 FpSpread 对象的 ActiveSheet 属性用编程方式来指定当前工作表。您还可以使用 ActiveSheetIndex 属性指定当前工作表的索引。

通常当前表在其他表之上。

您可以使用 ActiveSheet 来设置当前工作表的其他属性。

示例

设置当前工作表的属性并且设置当前工作表的索引为 2。

```
[C#]
// 添加三个表单
fpSpread1.Sheets.Count = 3;
// 设置第三个表单(以 0 开始的顺序)为当前表单.
fpSpread1.ActiveSheetIndex = 2;
// 设置当前表单的一些属性。
fpSpread1.ActiveSheet.ColumnCount = 8;
fpSpread1.InterfaceRenderer = NULL;
fpSpread1.ActiveSheet.GrayAreaBackColor = Color.Purple;

[Visual Basic]
'添加三个表单
FpSpread1.Sheets.Count = 3
'设置第三个表单(以 0 开始的顺序)为当前表单.
FpSpread1.ActiveSheetIndex = 2
'设置当前表单的一些属性。..
FpSpread1.ActiveSheet.ColumnCount = 8
FpSpread1.InterfaceRenderer = Nothing
FpSpread1.ActiveSheet.GrayAreaBackColor = Color.Purple
```

5.2 设置背景色或背景图

表单中的所有单元格的背景色如同其他属性一样，可以使用默认格式设置。

使用代码

设置 Sheets 对象的 `GrayAreaBackColor` 属性。

示例

此示例把第一个表单的背景色设成了淡黄色。

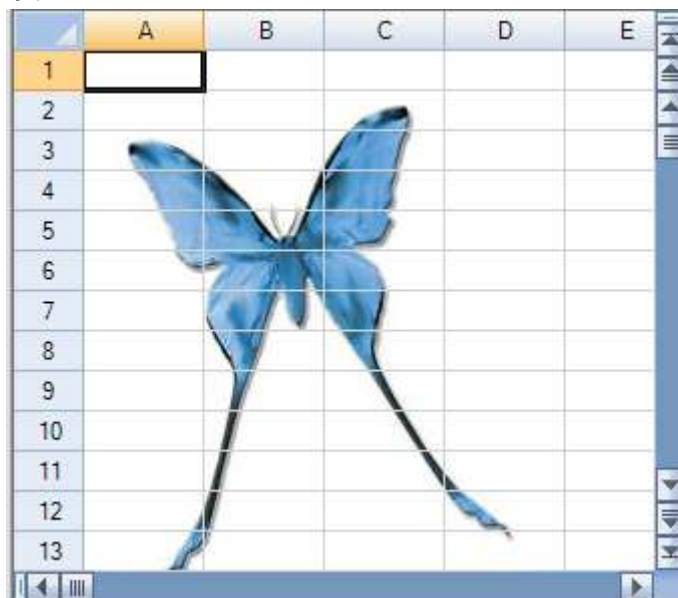
```
[C#]
// 设置第一个表单的背景色为淡黄色
fpSpread1.InterfaceRenderer = NULL;
fpSpread1.Sheets[0].GrayAreaBackColor = Color.LightYellow;

[Visual Basic]
'设置第一个表单的背景色为淡黄色.
FpSpread1.InterfaceRenderer = Nothing
FpSpread1.Sheets(0).GrayAreaBackColor = Color.LightYellow
```

使用 Spread 设计器

1. 选择您想要设置背景色的电子表单。
2. 在属性列表中选择 GrayAreaBackColor 属性。
3. 点击下拉箭头来显示颜色选择器。
4. 从颜色选择器中选择一个颜色。
5. 从“文件”菜单中选择“应用并退出”来应用您的变更并退出设计器。

您可以设置一张图片作为数据区域单元格的背景。依赖于图片和电子表单的大小，图片有可能在整个表单的单元格上重复显示。



示例

```
[C#]
private void Form1_Load(object sender, System.EventArgs e)
{
    //设置背景图片。
    fpSpread1.BackgroundImage = Image.FromFile("D:\\images\\butterfly.gif");

    //把表单的默认背景设成透明色。
    fpSpread1.ActiveSheet.DefaultStyle.BackColor = Color.Transparent;
}

[Visual Basic]
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load

    '设置背景图片。
    FpSpread1.BackgroundImage = Image.FromFile("D:\\images\\butterfly.gif")

    '把表单的默认背景设成透明色。
    FpSpread1.ActiveSheet.DefaultStyle.BackColor = Color.Transparent
End Sub
```

5.3 增加工作表

您可以增加一个或多个表单到 FarPoint Spread 组件中。默认情况下，这个组件包含一个命名为 Sheet1 的表单，表单索引为 0。表单的索引是从 0 开始的。您可以改变表单的数目或者添加电子表单。如果您想要定制表单的名字，直接修改表单名字就可以了。

默认情况下 Spread 允许用户添加新电子表单。您可以通过点击标签条旁边的新工作表图标（确保标签条首先是可见的）来添加。如果工作表计数大于 1，标签条默认是显示的。您可以通过设置 Spreadde 的 TabStripPolicy 属性改变标签条的显示。您可以设置 TabStripInsertTab 属性为 false（FpSpread1.TabStripInsertTab =False）阻止用户添加表单。

使用代码

1. 创建一个新的 SheetView 对象。
2. 如果有必要，请修改表单的属性，如名字。
3. 调用 Sheets 对象的 Add 方法来添加一个新的电子表单到 Spread 的 SheetViewCollection 中。

示例

下面的代码可以添加一个新的电子表单到组件中，表单的名字为“North”，有 10 列，100 行。

```
[C#]
// 创建一个新的表单。
FarPoint.Win.Spread.SheetView newsheet = new FarPoint.Win.Spread.SheetView();
newsheet.SheetName = "North";
newsheet.ColumnCount = 10;
newsheet.RowCount = 100;
//添加一个新的表单到组件里。
fpSpread1.Sheets.Add(newsheet);

[Visual Basic]
'创建一个新的表单。
Dim newsheet As New FarPoint.Win.Spread.SheetView()
newsheet.SheetName = "North"
newsheet.ColumnCount = 10
newsheet.RowCount = 100
'添加一个新的表单到组件里。
FpSpread1.Sheets.Add(newsheet)
```

5.4 删除工作表

如果您有多个工作表，您可以删除一个或多个工作表。但是要确保必须有至少一个工作表。工作表索引是从 0 开始的。在代码中，您可以简单地改变表的数目，也可以显式删除表。

删除现有的表不改变默认提供给其他工作表的名称。比如，一个 FarPoint Spread 组件有 3 个默认名字是 Sheet1，Sheet2 和 Sheet3 的表。如果您删除第二个表，其余的工作表的名称仍然是 Sheet1 的和 Sheet3，且这两张表的索引变成了 0 和 1。您还可以隐藏工作表。欲了解更多信息，请参阅显示或隐藏工作表部分。

使用代码

调用 Sheets 对象的 [Remove](#) 方法 (从组件的 [SheetViewCollection](#) 中删除表单)来删除指定的表单。

示例

下例从 FarPoint Spread 组件（包含 2 个表）中删除了第二张表。

```
[C#]
// 删除第二个表单。
fpSpread1.Sheets.Remove(fpSpread1.Sheets[1]);

[Visual Basic]
'删除第二个表单。
FpSpread1.Sheets.Remove(FpSpread1.Sheets(1))
```

5.5 移动工作表

如果您有多张工作表，您可以移动工作表。如果您将第一张表位置移动到最后一张工作表的位置，其他工作表将向左边移动。如果您把一个工作表移动到这张表的旁边，那么这个行为方式类似于交换。工作表索引是从零开始的。移动的开始位置和最终放置位置都是使用表单索引。移动工作表不会改变表的名称。

FpSpread 类的 AllowSheetMove 属性可以被设置为 true，以允许用户使用工作表标签移动表。您也可以隐藏工作表。欲了解更多信息，请参阅显示或隐藏工作表。

使用代码

调用 Sheets Move 方法，把电子表单从一个位置移动到另外一个位置。

示例

这个 示例代码把第二张表单移动到第三张表单的位置。

```
[C#]
// 把第二个表单移动到第三个表单的位置。
FpSpread1.Sheets.Count = 5;
FpSpread1.Sheets.Move(2, 3);

[Visual Basic]
'把第二个表单移动到第三个表单的位置。
FpSpread1.Sheets.Count = 5
FpSpread1.Sheets.Move(2, 3)
```

5.6 显示或隐藏工作表

如果在组件中您有一个以上的工作表，您可以隐藏工作表，以便它不会显示给用户。虽然它没有显示，但是并没有从组件中删除。这些操作都必须确保组件中至少有一张表。

隐藏工作表不改变默认表名字。比如，一个 FarPoint Spread 组件中有 3 张默认名字是 Sheet1，Sheet2 和 Sheet3 的表，如果您隐藏第二个表，其余工作表的名称依然是 Sheet1 的和 Sheet3。

隐藏工作表不会删除表也不会影响该工作表或表引用的公式。欲了解更多有关删除工作表的信息，

请参考删除一个表。

使用代码

设置 Sheets 对象的 `Visible` 属性。

示例

下例从一个包含了 8 张表的组件中隐藏了第二和第四张表。

```
[C#]
private void Form1_Load(object sender, System.EventArgs e)
{
    // 设置 8 个表单到 FarPointSpread 里。
    fpSpread1.Sheets.Count = 8;

    //隐藏第二和第四个表单。
    fpSpread1.Sheets[1].Visible = false;
    fpSpread1.Sheets[3].Visible = false;
}

[Visual Basic]
Private Sub Form1_Load(ByVal sender As Object, ByVal e As System.EventArgs) Handles MyBase.Load

    '设置 8 个表单到 FarPointSpread 里。
    FpSpread1.Sheets.Count = 8

    '隐藏第二和第四个表单。
    FpSpread1.Sheets(1).Visible = False
    FpSpread1.Sheets(3).Visible = False

End Sub
```

使用 Spread 设计器

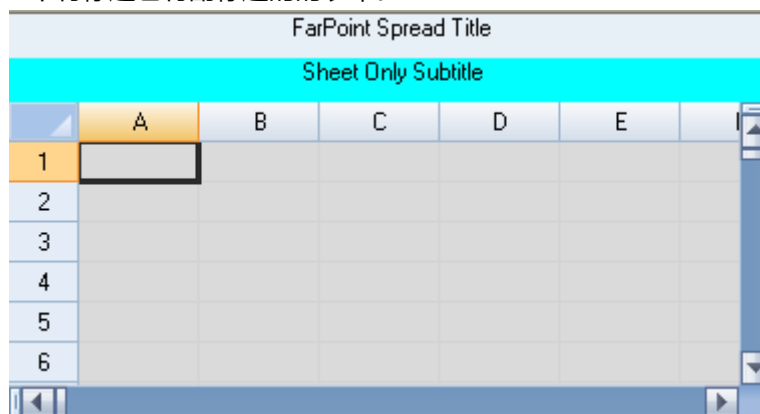
1. 选择您想要设置属性的表单。
2. 在属性列表中，选择 `Visible` 属性。
3. 设置为 `False`。
4. 从“文件”菜单中选择“应用并退出”来应用您的变更并退出设计器。

5.7 添加标题和子标题

您可以在电子表单的顶部添加一个特殊的格式化区域用来显示整个电子表单的标题或者一个电子表

单的副标题。

下例就说明了一个有标题也有副标题的的表单。



FarPoint Spread Title						
Sheet Only Subtitle						
	A	B	C	D	E	
1						
2						
3						
4						
5						
6						

标题是在 FpSpread 级别上使用 TitleInfo 属性来设置的。副标题是使用 Sheet 级别上的 TitleInfo 来设置的。也可以认为标题是应用到 Spread 组件上的，子标题则用来区分不同的电子表单。

与标题关联的 API 有：

- TitleInfo 类及所有成员
- SheetView TitleInfo 属性
- FpSpread TitleInfo 属性

使用代码

设置 TitleInfo 类的属性

示例

下面的示例演示了组件的标题和表单的副标题。

```
[C#]
// 设置整个 FarPoint 电子表单的标题。
FpSpread1.TitleInfo.Visible = true;
FpSpread1.TitleInfo.Text = "FarPoint Spread Title";
FpSpread1.TitleInfo.HorizontalAlign = FarPoint.Win.Spread.CellHorizontalAlignment.Center;
// 设置单个表单的副标题。
FpSpread1.Sheets[0].TitleInfo.Visible = true;
FpSpread1.Sheets[0].TitleInfo.Text = "Sheet Only Subtitle";
FpSpread1.Sheets[0].TitleInfo.HorizontalAlign = FarPoint.Win.Spread.CellHorizontalAlignment.Center;
FpSpread1.Sheets[0].TitleInfo.BackColor = System.Drawing.Color.Aqua;

[Visual Basic]
'设置整个 FarPoint 表单的标题
FpSpread1.TitleInfo.Visible = True
FpSpread1.TitleInfo.Text = "FarPoint Spread Title"
FpSpread1.TitleInfo.HorizontalAlign = FarPoint.Win.Spread.CellHorizontalAlignment.Center
'设置单个表单的副标题。
FpSpread1.Sheets(0).TitleInfo.Visible = True
FpSpread1.Sheets(0).TitleInfo.Text = "Sheet Only Subtitle"
FpSpread1.Sheets(0).TitleInfo.HorizontalAlign = FarPoint.Win.Spread.CellHorizontalAlignment.Center
FpSpread1.Sheets(0).TitleInfo.BackColor = System.Drawing.Color.Aqua
```

6. 高级数据操作

6.1 数据绑定示例

下面的教程将引导您创建项目和绑定数据库到 Spread 控件上。

1. 把 Spread 添加到一个数据绑定项目。
2. 设置数据库链接。
3. 指定需要使用的数据。
4. 创建数据集。
5. 把 Spread 绑定到数据库。
6. 设置单元格类型，改善显示效果。

6.1.1 把 Spread 添加到一个数据绑定项目

创建一个新的 Visual Studio.NET 项目，将项目命名为 databind。命名窗体文件的名字为 binding.cs (或.vb)。添加 FpSpread 组件到您的项目，然后在窗体上放置组件。

6.1.2 设置数据库链接

您必须告诉项目您要使用的数据库。在这一步，您将添加一个 OleDbConnection 的控件到您的窗体上，设置需要绑定的数据库的名称。

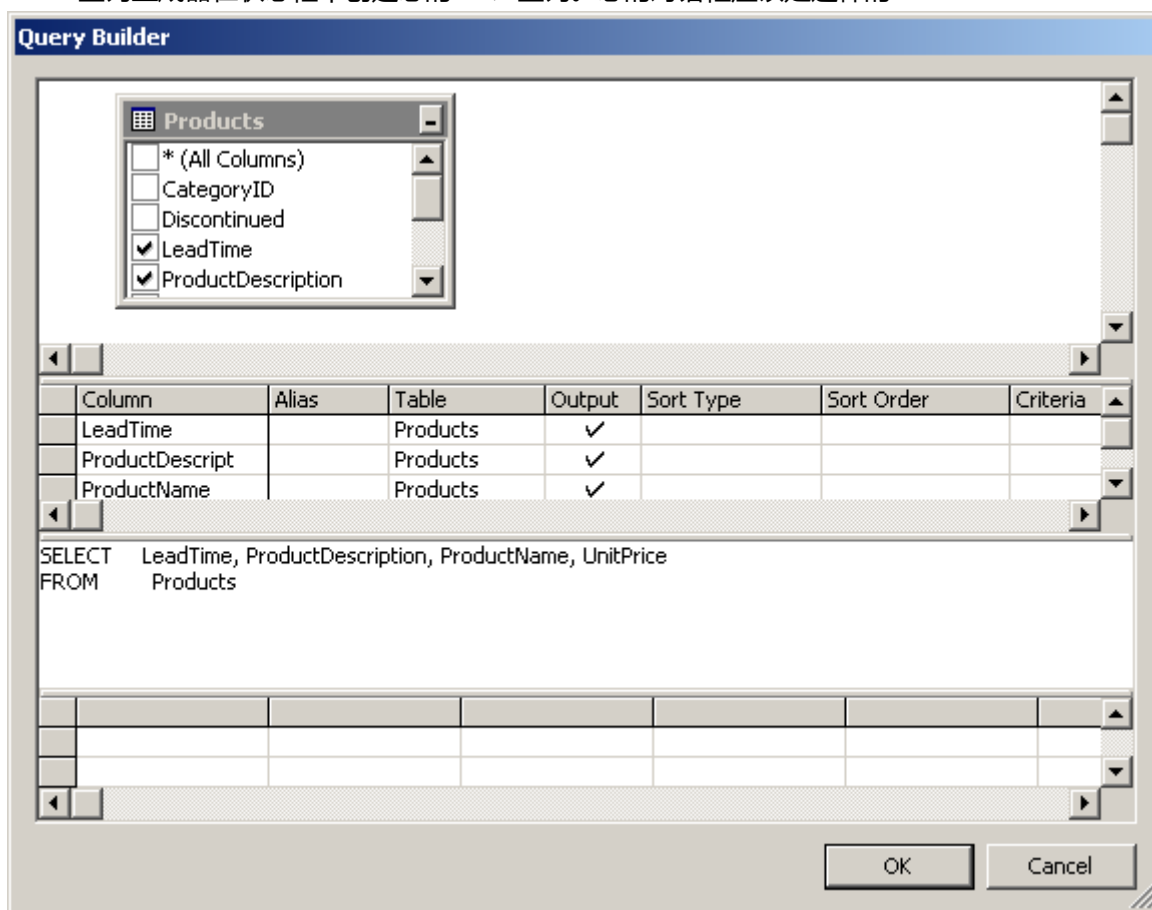
1. 如果“工具箱”没有显示，在“视图”菜单上选择使之显示。
2. 单击数据选项卡来显示可用的数据控件。
3. 双击 OleDbConnection 控件,将其添加到您的窗体上。OleDbConnection 控件添加到您的窗体上以后，一个新的可视区域将会显示在窗体的下面。在这个演练中您创建的数据控件将被放置到这个区域，取而代之窗体的可视区域。
4. 按 F4 来显示 OleDbConnection 控件的 Properties window 属性。
5. 在属性窗口上，设置控件的名字为 dbConnect。
6. 在属性窗口上，点击 ConnectionString 属性。
7. 单击设置区域的向下箭头然后在下拉列表中选择创建一个新的连接。Data Link Properties 对话框将会显示。
8. 单击 Provider 选项卡，然后在列表中选择 Microsoft Jet 4.0 OLE DB Provider。
9. 单击 Next。
10. 在 Select 或者输入数据库名字框的旁边，点击一个浏览按钮。
11. 浏览 \Spread.WinForm\Docs\TutorialFiles\databind.mdb 然后选择打开。
12. 单击 测试连接按钮。
13. 如果您没有收到“Test connection succeeded”的消息框,请重做步骤 6 到 12。
14. 如果您收到了“Test connection succeeded,”消息。您的连接完成了。单击 OK 然后选择 Data Link Properties 对话框。

6.1.3 指定需要使用的数据

现在，您已经指定要使用的数据库了。您还需要从数据库中检索出要显示在您的 FarPoint Spread

组件中的数据。要做到这一点，您将会用到 OleDbDataAdapter 控件。

1. 如果“工具箱”未显示，从“视图”菜单中选择工具箱。
2. 单击数据选项卡以显示可用的数据控件。
3. 双击 OleDbDataAdapter 控件，将其添加到您的窗体上。OleDbDataAdapter 控件将被添加到 Form 的一块可视区域内，数据适配器配置向导出现。
4. 开始选择“下一步”完成向导。
5. 在选择您的数据连接对话框，数据连接应使用哪种数据适配器？从下拉列表中选择您在步骤 2 中创建的连接。然后选择“下一步”。
6. 在选择查询类型对话框中，选择使用 SQL 语句，然后选择“下一步”。
7. 在生成的 SQL 语句对话框中，选择查询生成器。添加表对话框让您指定在数据库中使用的表。
8. 从列表中选择产品表并选择添加，然后选择关闭。
9. 在查询生成器对话框中，Product 表将出现，其中包含表中的可用字段的列表。选择以下字段：
 - LeadTime
 - ProductDescription
 - ProductName
 - ProductName
10. 查询生成器在状态框中创建您的 SQL 查询。您的对话框应该是这样的：



11. 选择 OK 以关闭查询生成器对话框，然后在向导中选择“下一步”。
12. 该向导总结了您的选择。选择“完成”以完成向导。
13. 按“F4”键显示该 OleDbDataAdapter 控件属性窗口。

14.在属性窗口中，更改控件的名称为 dbAdapt。

6.1.4 创建数据集

现在您已经从数据库中指定了数据库和数据，接下来您将在 FarPoint Spread 组件中创建一个包含您的数据的数据集。

使用代码

1. 选择窗体上的 dbAdapt OleDbDataAdapter 控件。
2. 如果属性窗口没有显示，按“F4”来显示控件的属性窗口。
3. 点击属性窗口下的“生成数据集”。
4. 生成数据集对话框将出现。
5. 点击 OK 关闭生成数据集对话框。新的数据集控件将添加到窗体上。
6. 按“F4”打开新的数据集控件的属性窗口。
7. 在属性窗口中,把控件名字修改为 dbDataSet。
8. 双击您工程中的窗体,打开代码窗口。
9. 在 Form_Load 事件中键入如下代码：

示例

```
[C#]
DataSet ds;
ds = dbDataSet;
dbAdapt.Fill(ds);

[Visual Basic]
Dim ds As DataSet
ds = dbDataSet
dbAdapt.Fill(ds)
```

这些代码使用您指定的数据库的数据来填充数据集，填充字段为 OleDbDataAdapter 控件中您指定的字段。

6.1.5 把 Spread 绑定到数据库

您的数据集已经准备好，现在您需要提供代码来绑定 FarPoint Spread 组件到数据集中。

1. 按 F4 显示 FarPoint Spread 组件的属性窗口。
2. 在属性窗口中，设置 DataSource 属性为数据集的名字“dbDataSet”。
3. 注意在 FarPoint Spread 组件中列头变成了 Products 表中字段的名字。
4. 保存您的工程。

5. 运行您的工程，您就会看到类似下面的图片：

	LeadTime	ProductDescription	ProdcutName	UnitPrice	ProductID
1	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 40 lbs.	30.00	1
2	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 20 lbs.	15.00	2
3	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 6 lbs.	5.00	3
4	1 week	Lamb & Rice food for puppies	Oregon Natural Lamb & Rice Puppy 6 lbs.	5.00	4
5	1 week	Lamb & Rice food for puppies	Oregon Natural Lamb & Rice Puppy 20 lbs.	15.00	5
6	2 weeks	Organic dog food for adult dogs	Doggy Delight Organic 20 lb.	15.00	6
7	2 weeks	Organic dog food for adult dogs	Doggy Delight Organic 40 lb.	30.00	7
8	2 weeks	Organic food for adult cats	Kitty Delight Organic 20 lb.	15.00	8
9	2 weeks	Organic food for adult cats	Kitty Delight Organic 8 lb.	5.00	9

6. 如果您的窗体和上面的不同,请调整您的 FarPoint Spread 组件的大小然后重新检查您的步骤。

7. 结束工程。

6.1.6 设置单元格类型，改善显示效果

使用代码

在这一步，您可以改变一系列的单元格类型以方便显示数据库的数据。

运行您的工程，会发现一张类似下面的图：

1. 双击窗体以打开代码窗口。
2. 通过下面的代码设置 UnitPrice 列的单元格类型。
3. 保存您的工程。

示例

[C#]

```
FarPoint.Win.Spread.CellType.CurrencyCellType CurrCell = new
FarPoint.Win.Spread.CellType.CurrencyCellType();
CurrCell.DecimalPlaces = 2;
fpSpread1.Sheets[0].Columns[3].CellType = CurrCell;
```

[Visual Basic]

```
Dim CurrCell As New FarPoint.Win.Spread.CellType.CurrencyCellType()
CurrCell.DecimalPlaces = 2
FpSpread1.Sheets(0).Columns(3).CellType = CurrCell
```

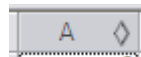
	LeadTime	ProductDescription	ProdcutName	UnitPrice	ProductID
1	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 40 lbs.	30.00	1
2	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 20 lbs.	15.00	2
3	1 week	Lamb & Rice food for adult dogs	Oregon Natural Lamb & Rice 6 lbs.	5.00	3
4	1 week	Lamb & Rice food for puppies	Oregon Natural Lamb & Rice Puppy 6 lbs.	5.00	4
5	1 week	Lamb & Rice food for puppies	Oregon Natural Lamb & Rice Puppy 20 lbs.	15.00	5
6	2 weeks	Organic dog food for adult dogs	Doggy Delight Organic 20 lb.	15.00	6
7	2 weeks	Organic dog food for adult dogs	Doggy Delight Organic 40 lb.	30.00	7
8	2 weeks	Organic food for adult cats	Kitty Delight Organic 20 lb.	15.00	8
9	2 weeks	Organic food for adult cats	Kitty Delight Organic 8 lb.	5.00	9

现在，您已经实现了绑定企业数据库到 Spread 上。恭喜您已经完成了此教程。

6.2 数据排序

当用户点击列头时就可以自动排序数据列。首次列标题被点击（选中），未排序图标将显示。第二次单击，排序图标将显示，列数据将进行排序。如果用户在同一列单击 2 次，那么排序的方向正好相反。此操作不会影响数据类型，只会影响数据如何显示。

下图显示了未排序图标：



使用 `Column` 对象的 `AllowAutoSort` 属性或者 `SheetView` 对象的 `SetColumnAllowAutoSort` 方法来实现当用户单击时的自动排序功能。设置列上的 `SortIndicator` 属性来显示排序图标。

`SetColumnShowSortIndicator` 方法或者 `ShowSortIndicator` 属性可以显示或者隐藏排序图标。排序图标在列标题上的显示如下图所示（包含升序和降序的图标）：

Ascending Sort Indicator			Descending Sort Indicator		
	A ▲	B		A ▼	B
1	Alignment		1	CarbAdjust	
2	Brakes		2	Brakes	
3	CarbAdjust		3	Alignment	
4			4		

当用户排序数据时，`AutoSortingColumn` 事件将在排序之前触发，`AutoSortedColumn` 事件将在排序之后触发。

使用代码

使用列的 `AllowAutoSort` 属性或者 `Sheets` 的 `SetColumnAllowAutoSort` 方法可以自动排序指定列。

示例

下例演示了如何给表单的前 30 列排序。

```
[C#]
fpSpread1.Sheets[0].Columns[0,29].AllowAutoSort = true;
或
fpSpread1.Sheets[0].SetColumnAllowAutoSort(0,30,true);
[Visual Basic]
FpSpread1.Sheets(0).Columns(0,29).AllowAutoSort = True
或
FpSpread1.Sheets(0).SetColumnAllowAutoSort(0,30,True)
```

使用 Spread 设计器

1. 选择一个您想排序的表单选项卡。
2. 在电子表单的属性列表中，选择列，单击打开“Cells, Columns, and Rows Editor”编辑器。
3. 选择您需要自动排序的列。
4. 在属性列表中,选择 AllowAutoSort 属性，将其设置成 True。
5. 在“文件”菜单中选择“应用并退出”来应用您的变更到组件中并退出设计器。

6.3 数据过滤

6.3.1 允许数据过滤

在默认情况下，电子表单不允许用户过滤行上的数据。您可以打开这个功能然后允许过滤所有或指定列。

使用 `HideRowFilter` 类来设置是否要隐藏过滤功能或 `StyleRowFilter` 类来设置过滤行的格式。使用 `Column.AllowAutoFilter` 属性打开一个给定列的过滤功能。

然后用户可以点击下拉按钮来过滤行。要了解最终用户的过滤功能，请参考行过滤。

使用代码

允许行过滤（隐藏过滤掉的行）是基于一系列上的值，所以请确保列标题是可见的。

使用 `HideRowFilter` 类来选择行过滤的类型。使用 `AllowAutoFilter` 属性，允许过滤指定列。此示例假设单元格上有一些数据，无论是指定值或是绑定到数据源。

示例

```
[C#]
fpSpread1.ActiveSheet.ColumnHeaderVisible = true;
FarPoint.Win.Spread.HideRowFilter hideRowFilter = new
FarPoint.Win.Spread.HideRowFilter(fpSpread1.ActiveSheet);

fpSpread1.ActiveSheet.Columns[1,3].AllowAutoFilter = true;

[Visual Basic]
FpSpread1.ActiveSheet.ColumnHeaderVisible = True
Dim hideRowFilter As New FarPoint.Win.Spread.HideRowFilter(FpSpread1.ActiveSheet)

FpSpread1.ActiveSheet.Columns(1, 3).AllowAutoFilter = True
```

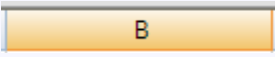
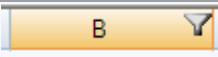
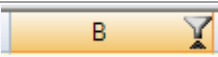
使用 Spread 设计器

1. 点击一个列标题去选择一列。
2. 设置属性窗口里（Misc 分类下）的 AllowAutoFilter 属性为 True。
3. 从“文件”菜单选择“应用并退出”以保存修改。

6.3.2 使用数据过滤

本主题概述了最终用户如何和过滤功能进行交互。

一旦您把行过滤应用于列，过滤图标将出现在列头上。此表总结了该过滤图标的不同外观：

过滤图标	描述
	头单元格没有过滤的外观；这个会在没有过滤或者过滤功能被关掉的时候发生。
	这个是头单元格有过滤但是没有被过滤的外观；这个产生在过滤被设置成全部而此列没有行被过滤的情况。
	这个是头单元格有过滤并且一些行被过滤的外观；这个产生在过滤掉此列中一些行的情况下。

当过滤时，列标题将显示行过滤图标及一个下拉箭头符号。点击过滤图标将出现一个下拉式过滤的选项列表。您可以从这个列表中选择需要的选项，Spread 将会过滤出所有此列满足该选项条件的行显示出来。默认的下拉列表包含在此列中的所有单元格文本值。下图显示了一个下拉式的过滤列表。

A	
Fender	AST-100 DM
Gibson	Les Paul St
Ibanez	ST58-70TX
Yamaha	AGS83B
(Blanks)	
(NonBlanks)	
5 Gibson	Les Paul Su
6 Yamaha	

这张表总结了下拉表中的条目：

过滤列表选项	描述
(All)	包含此列中所有行。
[内容]	包含此列中和“内容”相同的行。
(Blanks)	包含此列中所有空白单元格行。
(NonBlanks)	包含此列中所有非空白单元格行,换言之,就是所有包含内容的单元格行。

您可以自定义此列表显示方式，如同自定义过滤列表中所描述的那样。您甚至可以创建自定义过滤列表添加到下拉列表中，如同创建一个完全自定义过滤中描述的那样。

对于一个给定表单，可能有多个列包含过滤设置。不同的列可能有不同的过滤，这取决于该列单元格的内容。过滤的结果将类似于主要键和次要键排序时的数据。最初的列过滤将过滤某些行，后续的过滤将在留下的行中进行过滤。通过多个过滤，最终结果只包括那些满足所有过滤条件的行。

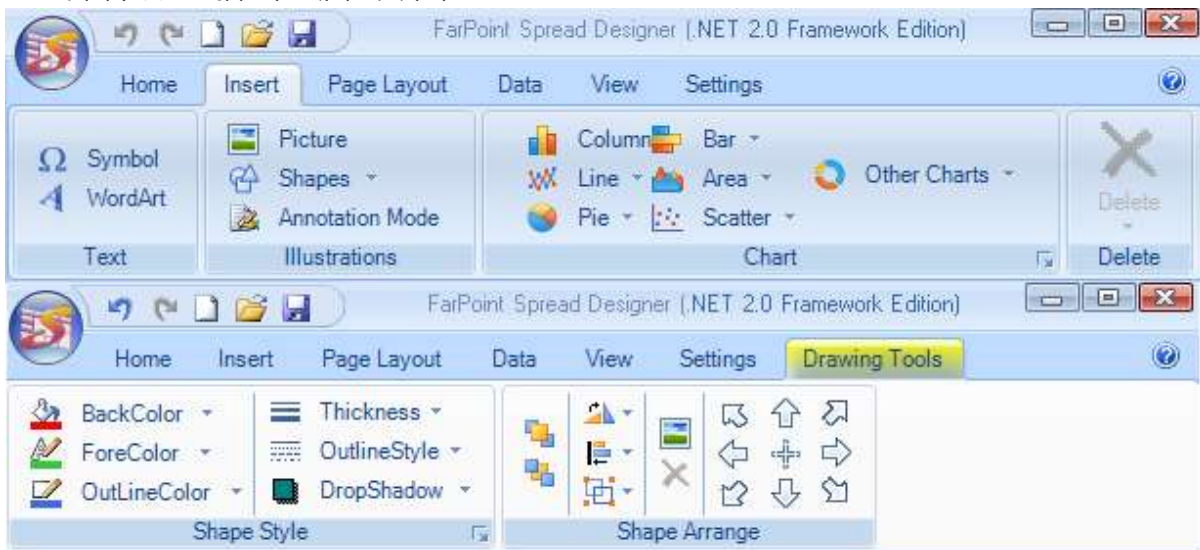
7. 使用图形

7.1 创建图形对象

创建图形的最简单的方法是利用 Spread 设计器中插入菜单来设计图形。您创建一个图形之后，绘图工具菜单会显示。插入菜单上的选项用来创建不同类型的图形对象，也可用于保存、加载、或删除图形对象。在插入菜单上提供了一种快速的方式方便使用电子表格的绘图空间创建图形。绘图工具菜单包含用于设置颜色、线条粗细、位置、比例和方向等几个绘图相关的图标。

关于图形的更多信息，请参考图形的相关章节。

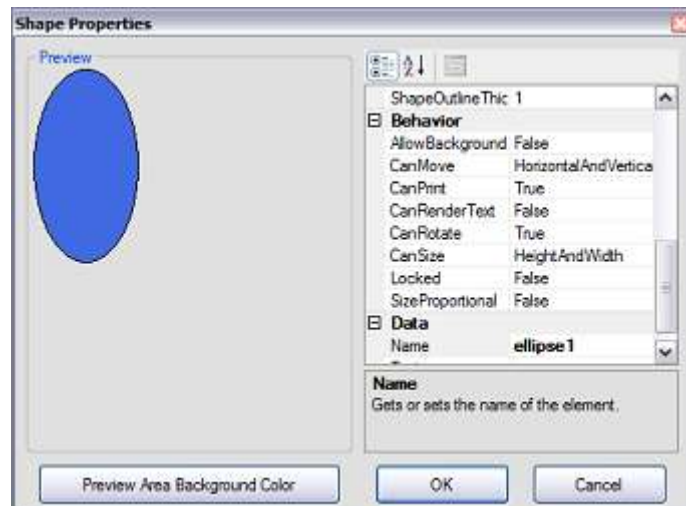
下图中演示了插入和绘图工具菜单：



“插入”菜单中的图形选项允许您选择一个内置的图形。“绘图工具”菜单的“图形风格”部分允许您设置颜色、轮廓以及阴影。

7.2 设置图形属性

使用 Spread 设计器可以设置图形的属性。您可以使用“插入”和绘图工具菜单或者使用图形属性对话框。为了打开这个对话框，选择一个图形然后点击右键，从右键菜单中选择属性，图形属性对话框将会出现如下图所示。在下图中，有一个椭圆形在左边，它的背景颜色被改成了蓝色。

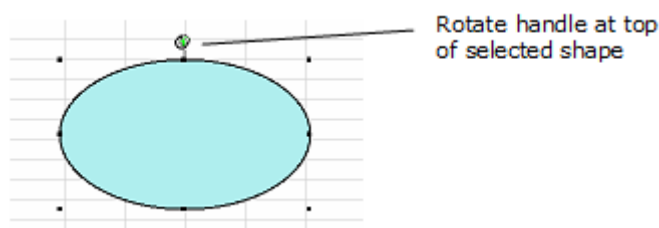


上图的这个属性列表不同于 Spread 设计器中的单元格的属性。当您选择一个图形时，请不要使用 Spread 设计器中的属性列表，而要使用此属性对话框。因此您应确保您选择的属性对话框是在某个图形被选择后并且是从其右键菜单中打开的。

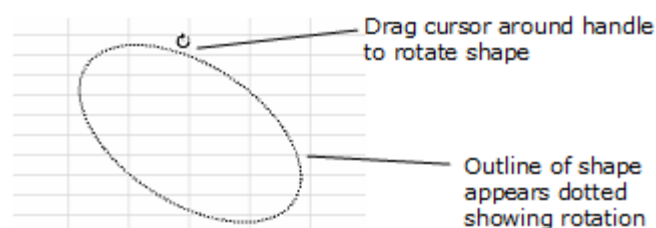
一旦图形被插入可视区域，图形属性对话框就可以用来定制图形了。通过显示或预览图形以及属性列表，图形属性对话框会在改变属性时立即显示预览效果。

7.3 图形旋转

在 Spread 设计器中为了旋转图形，请选择图形然后移动旋转柄(小的绿色的点)来调整图形，让其旋转到您期望的位置。单击在图形上面的旋转柄然后在旋转柄周围拖拽鼠标，图形将会临时消失只留下虚线轮廓，直到释放鼠标。



当您单击在旋转柄，一个小绿色圆圈就会出现在选择对象的上面，拖拽光标，显示就会改变。虚线轮廓可以让您了解旋转图形的效果。请看下图示例。



光标的变化表明您正在旋转图形。当您旋转时，释放鼠标就可以把图形显示在合适的位置上。

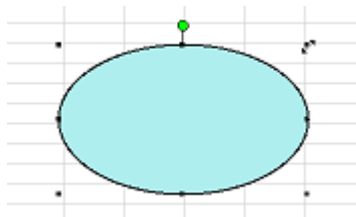
如果您通过旋转柄来旋转图形，每次鼠标移动可以让图形旋转 5 度。如果在旋转时按住 Ctrl 键就可以每次只旋转一度。

您可以允许最终用户来旋转图形或者是通过 CanRotate 属性来阻止他这么做。

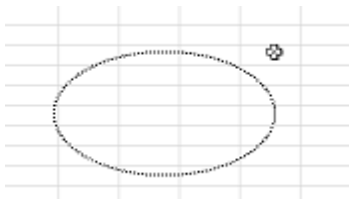
7.4 图形缩放

您可以调整一个图形的大小。

在 Spread 设计器中，选择一个图形然后移动尺寸调整柄就可以调整图形的大小了。这些尺寸调整柄都出现在图形的角上或者是边缘。为了成比例地调整图形大小，您可以选择角上的尺寸调整柄。单击选择对象，然后单击调整柄，拖拽它。



在您拖拽尺寸调整柄的时候，这些图形将会临时消失，但是会用虚线临时勾勒一个改变后的图形轮廓。光标如果改变就意味着您在调整图形的大小。



当调整大小时，鼠标释放在合适位置，图形将会用新的尺寸和面积来显示。

对于圆角的矩形框，您可以点击和拖拽弧形调整柄(黄色方框)来调整圆角的弧度。

您可以允许最终用户调整图形的大小也可以用 `CanSize` 属性来阻止用户这么做。

7.5 图形移动

您可以移动一个图形。

为了移动图形，在 Spread 设计器中，选择图形然后拖拽图形到新的位置。

您可以允许最终用户移动图形或者使用 `CanMove` 属性来阻止用户这么做。

7.6 图形锁定

您可以锁定一个图形防止用户和它交互。图形仍然是可见的，但是不能移动、旋转或者修改。

锁定一个单元格不能锁定任何在单元格之上的图形(浮在上面的对象)。一个受保护的表单只意味着所有表单中的单元格都被标记成锁定或非锁定的，这个不能应用到图形上。为了锁定一个图形，需要改变图形的 `Locked` 属性为 `true`。在 Spread 设计器上，这个可以通过右键点击一个图形然后选择属性。这样可以打开图形属性对话框，在这个对话框中，设置 `Locked` 属性为 `true`。使用代码也可以设置图形的 `Locked` 属性为 `true`。

每个图形都可以被单独锁定。

锁定意味着图形不能被选择以及和用户进行交互。

8. 使用图表

图表控件有多种图表类型，每个类型都有不同的视图。

以下为不同的图表类型列表：

- 柱形图
- 折线图
- 饼图
- 条形图
- 面积图
- XY 散点图
- 气泡图
- 股价图
- 曲面图
- 圆环图
- 雷达图
- 极地图

柱形图有以下类型 – 簇状柱形图，堆积柱形图，百分比堆积柱形图，高低柱形图，三维簇状柱形图，三维堆积柱形图，百分比三维堆积柱形图，三维柱形图，三维高低柱形图，簇状圆柱图，堆积圆柱图，百分比堆积圆柱图，三维圆柱图，高低列圆柱图，簇状圆锥图，堆积圆锥图，百分比堆积圆锥图，三维圆锥图，高低列圆锥图，簇状棱锥图，堆积棱锥图，百分比堆积棱锥图，三维棱锥图，高低列棱锥图。

折线图有以下类型 – 折线图，堆积折线图，百分比堆积折线图，带数据标记的折线图，带数据标记的堆积折线图，百分比带数据标记的堆积折线图，三维折线图。

饼图有以下类型 – 二维饼图，三维饼图，二维分离型饼图和三维分离型饼图。

条形图有以下类型 – 簇状条形图，堆积条形图，百分比堆积条形图，高低条形图，三维簇状条形图，三维堆积条形图，百分比三维堆积条形图，三维高低条形图，簇状水平圆柱图，堆积水平圆柱图，百分比堆积水平圆柱图，高低水平圆柱图，簇状水平圆锥图，堆积水平圆锥图，百分比堆积水平圆锥图，高低圆锥图，簇状水平棱锥图，堆积棱锥图，百分比堆积棱锥图和高低列棱锥图。

面积图的类型有以下 – 面积图，堆积面积图，百分比堆积面积图，高低面积图，三维面积图，三维堆积面积图，百分比三维堆积面积图，三维高低面积图。

XY 散点图有以下类型 – XY 散点图，带直线的散点图和数据标记的散点图。

气泡图有以下类型 – 二维气泡图和三维气泡图。

股价图有以下类型 – 盘高-盘低-收盘图，开盘-盘高-盘低-收盘图，和 **K 线图**。

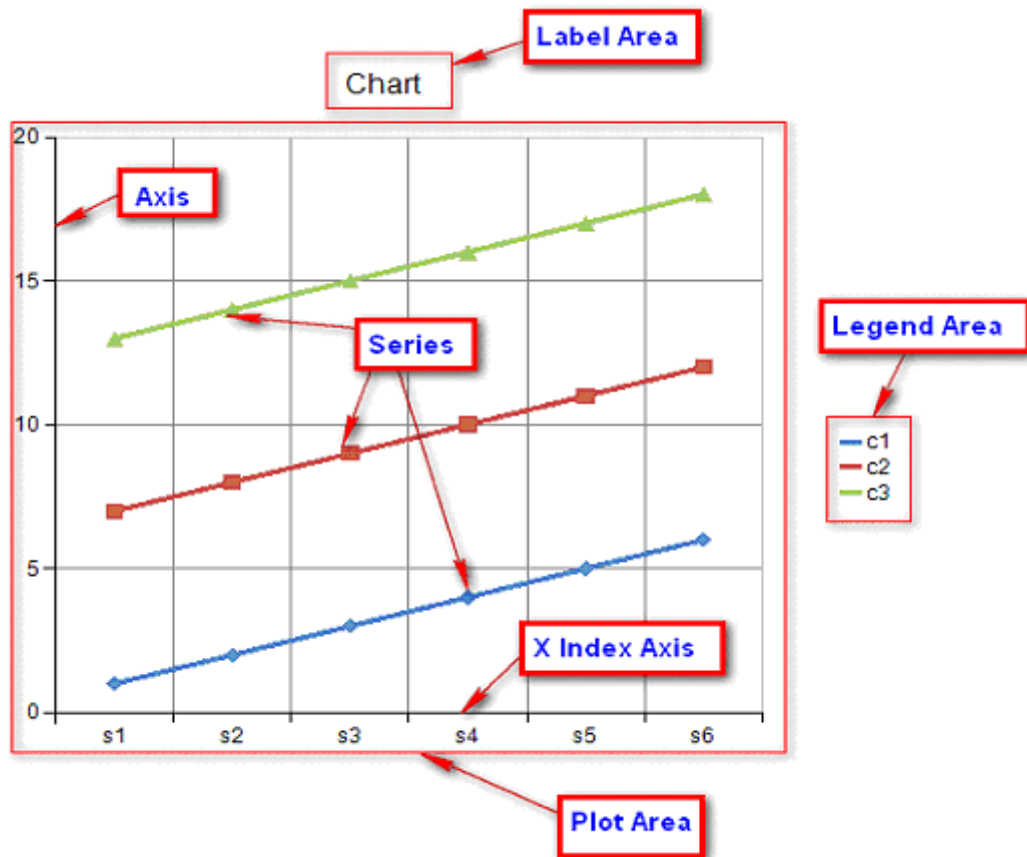
曲面图有以下类型 – 曲面线图，带数据标记的曲面线图，曲面点图，曲面图。

圆环图有以下类型 – 圆环图，分离圆环图。

雷达图有以下类型 – 雷达图，带数据标记的雷达图，雷达点图，填充雷达图。

极地图有以下类型 – 极地图，带数据标记的极地图，极地点图，极地区域图。

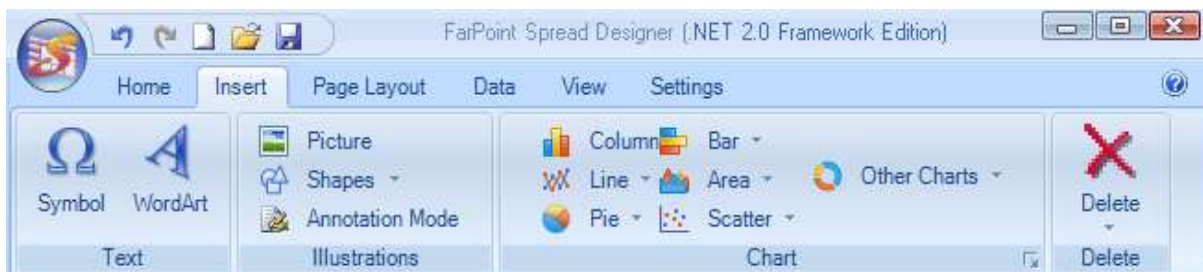
图表有几个视觉元素：图例、图表标题、坐标轴和数据系列。图表标题包含该图表的相关信息。图例可以帮助最终用户识别不同的图表元素以及数据系列。坐标轴是用来显示单一维度绘图区的大小。每个数据系列是一组数据点的组合。绘图区是数据点绘制区域。

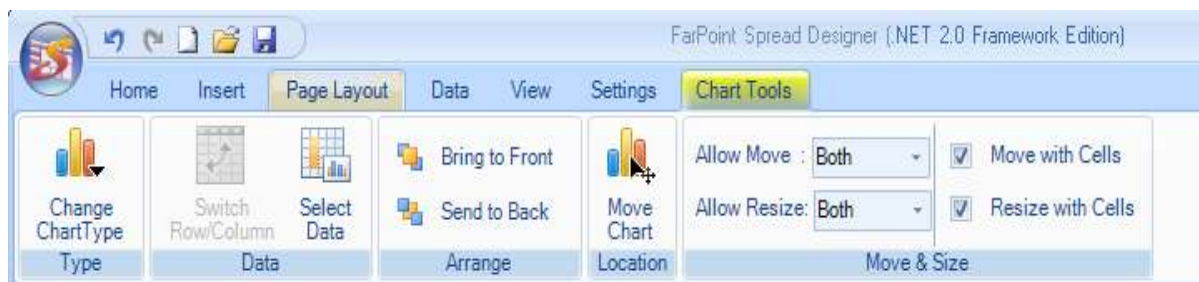


您可以使用代码、Spread 设计器或者是 Chart 设计器来添加一个图表。您可以绑定图表，让最终用户可以在运行时修改图表。

8.1 创建图表对象

您可以使用代码或 Spread 设计器给表单添加图表控件。您也可以让用户调整图表控件中使用的数据范围并且允许用户调整图表的大小。下图显示 Spread 设计器中插入菜单中的图表一节。第二个图显示图表工具菜单选项（该显示在图表插入以后出现）。





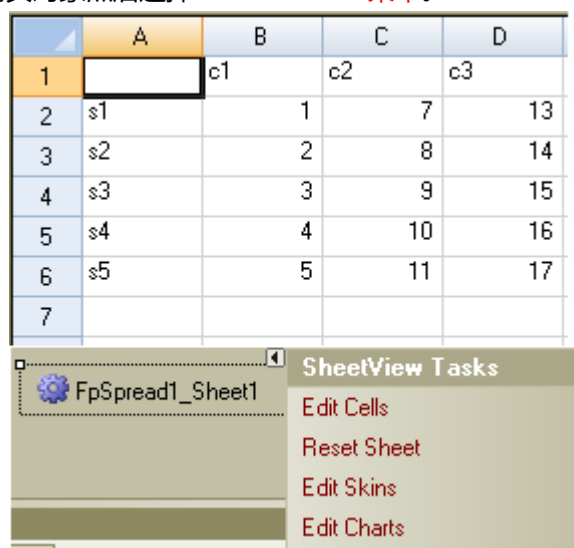
以下示例工程文件可以在产品安装目录下找到。

使用 Spread 设计器 或 “Edit Charts” 菜单

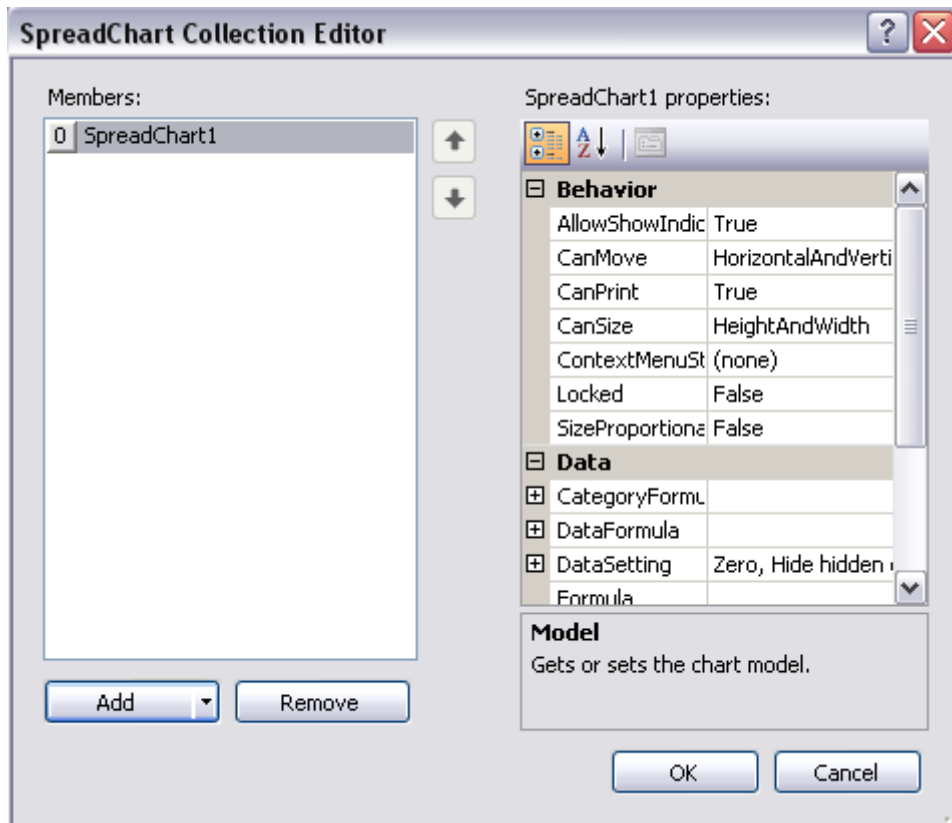
1. 打开 Spread 设计器然后键入单元格中的图表数据(后面章节有这个章节的代码演示)
2. 选择一个有数据的单元格范围。
3. 点击“插入”菜单然后选择图标类型(如上图)。
4. 图表工具菜单将出现。
5. 关闭 Spread 设计器然后保存变更。

或者

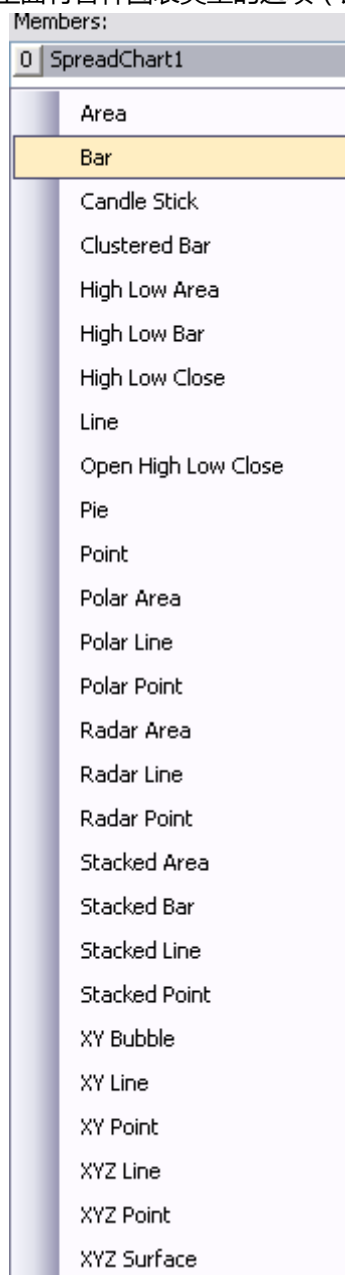
1. 点击页面底部的 FpSpread1_Sheet1 对象。
2. 点击对象右上角的箭头对象然后选择 “Edit Charts” 菜单。



3. Spread Chart Collection 编辑器将会打开。点击“添加”来添加一个图表然后设置图表属性。

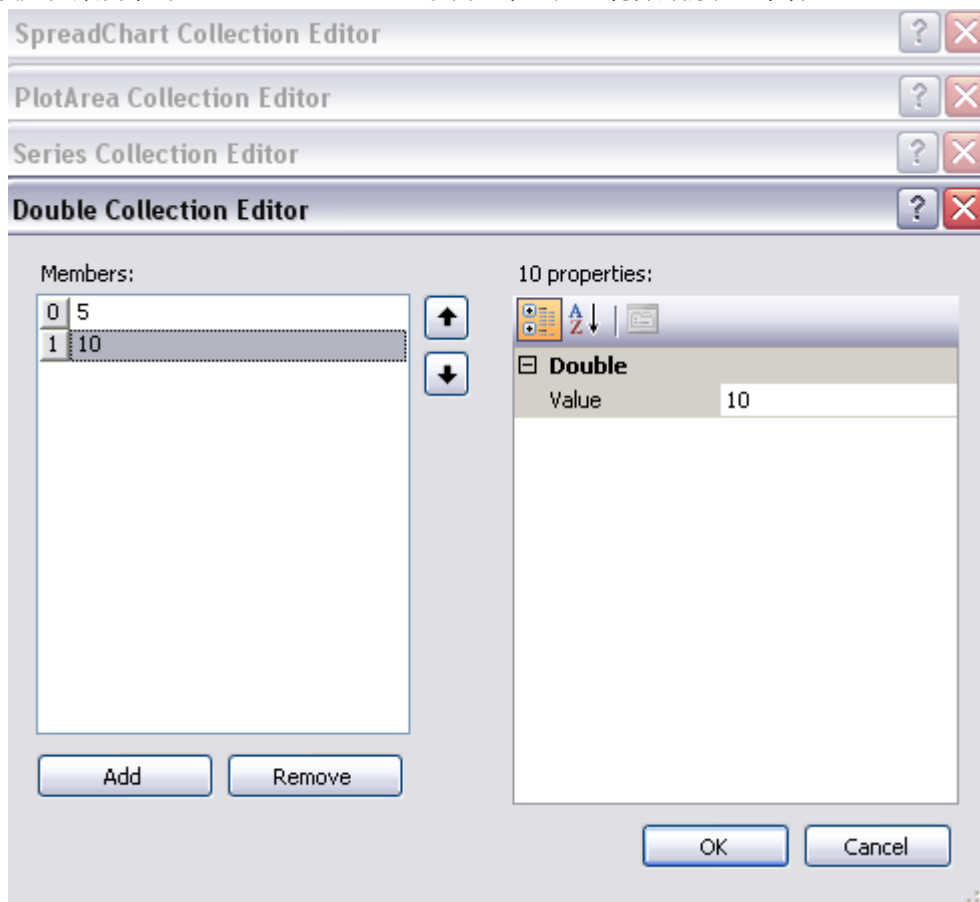


4. 添加按钮有一个下拉菜单，里面有各种图表类型的选项（比如柱形图）。



5. 在 SpreadChart 集合编辑器中选择“模型”，弹出图表设计器或在“模型”下选择 PlotAreas 集合。使用添加按钮下拉菜单选择绘图区类型（比如 YPlotArea）。
6. 使用该数据系列集合添加一种数据系列类型（比如 BarSeries）。添加按钮下拉菜单有您可以添加的各种数据系列类型。

7. 使用值集合，弹出 DoubleCollection 编辑器，可用于将数据添加到图表。



8. 点击每个对话框的 OK 按钮。

使用代码

您可以使用代码给 Spread 控件中添加一个 Chart 控件。这个例子设置了一些单元格的值然后添加图表控件。

示例

这个例子演示了如何创建一个图表控件。

```
[C#]
fpSpread1.Sheets[0].Cells[0, 1].Value = "c1";
fpSpread1.Sheets[0].Cells[0, 2].Value = "c2";
fpSpread1.Sheets[0].Cells[0, 3].Value = "c3";
fpSpread1.Sheets[0].Cells[1, 0].Value = "s1";
fpSpread1.Sheets[0].Cells[2, 0].Value = "s2";
fpSpread1.Sheets[0].Cells[3, 0].Value = "s3";
fpSpread1.Sheets[0].Cells[4, 0].Value = "s4";
fpSpread1.Sheets[0].Cells[5, 0].Value = "s5";
fpSpread1.Sheets[0].Cells[6, 0].Value = "s6";
fpSpread1.Sheets[0].Cells[1, 1].Value = 1;
```

```
fpSpread1.Sheets[0].Cells[2, 1].Value = 2;
fpSpread1.Sheets[0].Cells[3, 1].Value = 3;
fpSpread1.Sheets[0].Cells[4, 1].Value = 4;
fpSpread1.Sheets[0].Cells[5, 1].Value = 5;
fpSpread1.Sheets[0].Cells[6, 1].Value = 6;
fpSpread1.Sheets[0].Cells[1, 2].Value = 7;
fpSpread1.Sheets[0].Cells[2, 2].Value = 8;
fpSpread1.Sheets[0].Cells[3, 2].Value = 9;
fpSpread1.Sheets[0].Cells[4, 2].Value = 10;
fpSpread1.Sheets[0].Cells[5, 2].Value = 11;
fpSpread1.Sheets[0].Cells[6, 2].Value = 12;
fpSpread1.Sheets[0].Cells[1, 3].Value = 13;
fpSpread1.Sheets[0].Cells[2, 3].Value = 14;
fpSpread1.Sheets[0].Cells[3, 3].Value = 15;
fpSpread1.Sheets[0].Cells[4, 3].Value = 16;
fpSpread1.Sheets[0].Cells[5, 3].Value = 17;
fpSpread1.Sheets[0].Cells[6, 3].Value = 18;
FarPoint.Win.Spread.Model.CellRange range = new FarPoint.Win.Spread.Model.CellRange(0, 0, 7, 4);
fpSpread1.Sheets[0].AddChart(range, typeof(FarPoint.Win.Chart.BarSeries), 400, 300, 0, 0,
FarPoint.Win.Chart.ChartViewType.View3D, false);
```

```
[Visual Basic]
FpSpread1.Sheets(0).Cells(0, 1).Value = "c1"
FpSpread1.Sheets(0).Cells(0, 2).Value = "c2"
FpSpread1.Sheets(0).Cells(0, 3).Value = "c3"
FpSpread1.Sheets(0).Cells(1, 0).Value = "s1"
FpSpread1.Sheets(0).Cells(2, 0).Value = "s2"
FpSpread1.Sheets(0).Cells(3, 0).Value = "s3"
FpSpread1.Sheets(0).Cells(4, 0).Value = "s4"
FpSpread1.Sheets(0).Cells(5, 0).Value = "s5"
FpSpread1.Sheets(0).Cells(6, 0).Value = "s6"
FpSpread1.Sheets(0).Cells(1, 1).Value = 1
FpSpread1.Sheets(0).Cells(2, 1).Value = 2
FpSpread1.Sheets(0).Cells(3, 1).Value = 3
FpSpread1.Sheets(0).Cells(4, 1).Value = 4
FpSpread1.Sheets(0).Cells(5, 1).Value = 5
FpSpread1.Sheets(0).Cells(6, 1).Value = 6
FpSpread1.Sheets(0).Cells(1, 2).Value = 7
FpSpread1.Sheets(0).Cells(2, 2).Value = 8
FpSpread1.Sheets(0).Cells(3, 2).Value = 9
FpSpread1.Sheets(0).Cells(4, 2).Value = 10
FpSpread1.Sheets(0).Cells(5, 2).Value = 11
FpSpread1.Sheets(0).Cells(6, 2).Value = 12
FpSpread1.Sheets(0).Cells(1, 3).Value = 13
FpSpread1.Sheets(0).Cells(2, 3).Value = 14
FpSpread1.Sheets(0).Cells(3, 3).Value = 15
FpSpread1.Sheets(0).Cells(4, 3).Value = 16
FpSpread1.Sheets(0).Cells(5, 3).Value = 17
FpSpread1.Sheets(0).Cells(6, 3).Value = 18
Dim range As New FarPoint.Win.Spread.Model.CellRange(0, 0, 7, 4)
FpSpread1.Sheets(0).AddChart(range, GetType(FarPoint.Win.Chart.BarSeries), 400, 300, 0, 0,
FarPoint.Win.Chart.ChartViewType.View3D, False)
```

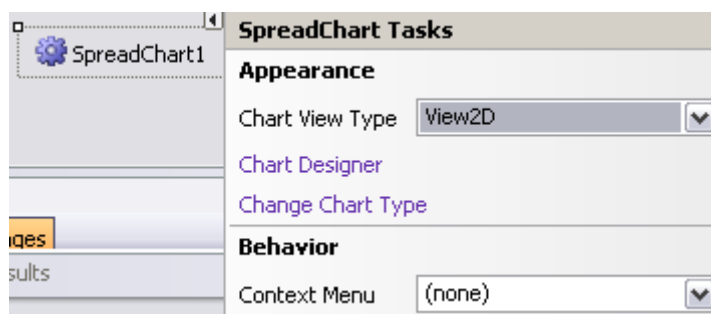
8.2 使用图表设计器

您可以使用图表设计器来添加一个图表。

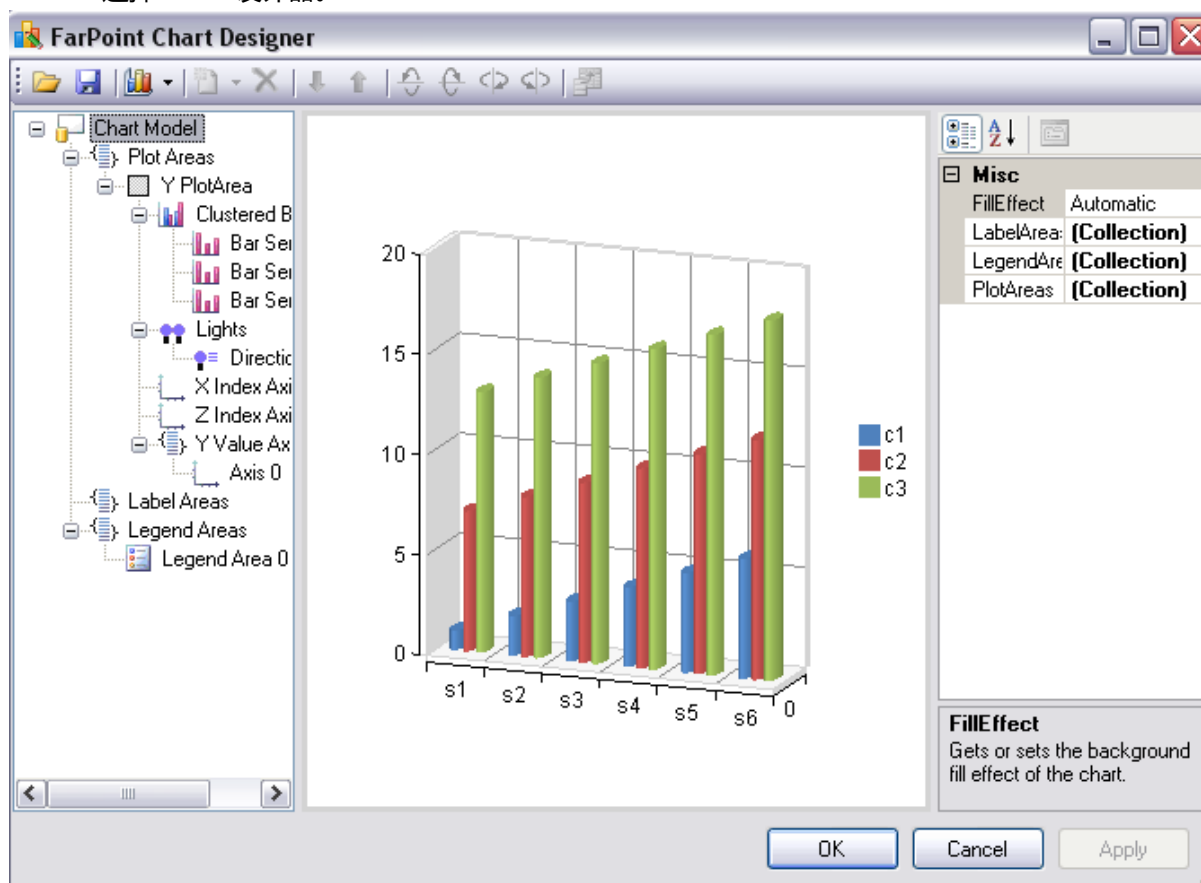
下面的示例演示了这个功能，该示例您可以在产品的安装目录下找到。

使用 Chart 设计器

1. 使用 **“Edit Charts” 菜单** 或者使用 Spread 设计器的插入菜单在窗体上创建一个 Spread Chart 对象。



2. 点击 Spread Chart 对象的 smart tag 或者 verb。
3. 选择 Chart 设计器。



4. Chart 设计器允许您创建一个图表并且设置额外的选项。使用 PlotAreas Collection 来创建绘图数据系列并添加数据。使用 LegendAreas Collection 来创建一个图表的图例。使用 LabelAreas Collection 集合创造图表标签。
5. 点击“应用”来保存变更。

8.3 绑定图表

一个系列包含三个部分（分类、系列名称和数据）。您可以绑定每个部分到系列数据字段的实例。

整个图表控件不能用数据绑定，但是您可以使用单元格区域或公式，让数据显示在图表中。

使用代码

您可以用一个数据或者数据集给 Spread 添加数据然后使用一个单元格区间让数据显示在图表控件中。

示例

下例演示了如何使用数组在控件中放置数据。

```
[C#]
object[,] values = { { "lg1", "lg2", "lg3" }, { "tt1", 2.0, 5.0 }, { "tt2", 4.0, 5.0 } };
fpSpread1.Sheets[0].SetArray(0, 0, values);
FarPoint.Win.Spread.Model.CellRange cellRange = new
FarPoint.Win.Spread.Model.CellRange(0,0,values.GetLength(0),values.GetLength(1));
fpSpread1.Sheets[0].AddChart(cellRange, typeof(FarPoint.Win.Chart.BarSeries), 400, 400, 0, 0);
private void button1_Click(object sender, EventArgs e)
{
FarPoint.Win.Chart.BarSeries series =
(FarPoint.Win.Chart.BarSeries)fpSpread1.Sheets[0].Charts[0].Model.PlotAreas[0].Series[0];
FarPoint.Win.Spread.Chart.SeriesDataField data =
(FarPoint.Win.Spread.Chart.SeriesDataField)series.Values.DataSource;
data.Formula = "Sheet1!$B$2:$E$1";
}
[Visual Basic]
Dim values As Object(,) = {{"lg1", "lg2", "lg3"}, {"tt1", 2.0R, 5.0R}, {"tt2", 4.0R, 5.0R}}
FpSpread1.Sheets(0).SetArray(0, 0, values)
Dim cellRange As New FarPoint.Win.Spread.Model.CellRange(0, 0, values.GetLength(0), values.GetLength(1))
FpSpread1.Sheets(0).AddChart(cellRange, GetType(FarPoint.Win.Chart.BarSeries), 400, 400, 0, 0)
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click
Dim series As FarPoint.Win.Chart.BarSeries =
DirectCast(FpSpread1.Sheets(0).Charts(0).Model.PlotAreas(0).Series(0), FarPoint.Win.Chart.BarSeries)
Dim data As FarPoint.Win.Spread.Chart.SeriesDataField = DirectCast(series.Values.DataSource,
FarPoint.Win.Spread.Chart.SeriesDataField)
data.Formula = "Sheet1!$B$2:$E$1"
End Sub
```

8.4 允许用户改变图表

您可以允许用户调整大小、移动、或者改变显示在图表中的范围元素。

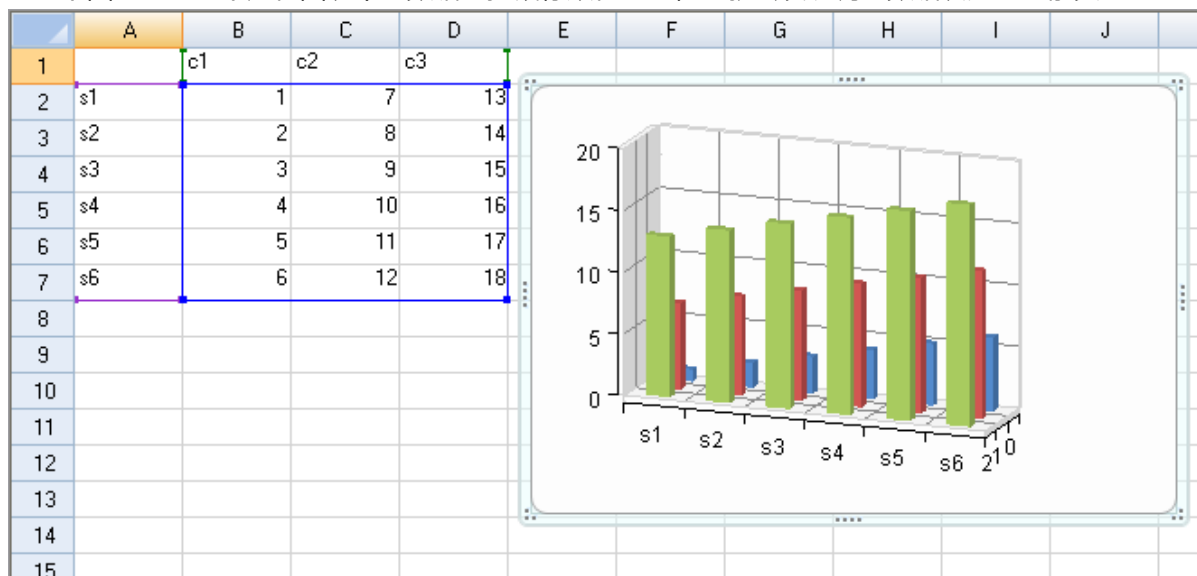
在运行时,用户可以做如下修改：

- 用户可以选择图表然后移动或调整图表大小。
- 用户可以选择图表和图表中使用的数据范围，用户也可以调整选择的数据块大小来改变图表中

数据范围。

- 用户可以编辑单元格的值来修改图标中的数据。

下图显示了一个用在图表中的数据。把鼠标放在蓝色框边就会发现调整数据块大小的箭头。



您可以使用 Locked 属性来禁止用户移动或者调整图表的大小。

使用代码

示例

此示例设置了 Locked, CanMove, 和 CanResize 属性。

```
[C#]
FarPoint.Win.Spread.Chart.SpreadChart chart;
chart = fpSpread1.Sheets[0].AddChart(0, 0, typeof(FarPoint.Win.Chart.BarSeries), 400, 400, 200, 80,
FarPoint.Win.Chart.ChartViewType.View2D, true);
chart.Locked = true;
//chart.CanSize = FarPoint.Win.Spread.DrawingSpace.Sizing.None;
//chart.CanMove = FarPoint.Win.Spread.DrawingSpace.Moving.Horizontal;

[Visual Basic]
Dim chart As FarPoint.Win.Spread.Chart.SpreadChart
Dim range As New FarPoint.Win.Spread.Model.CellRange(0, 0, 7, 4)
chart = FpSpread1.Sheets(0).AddChart(range, GetType(FarPoint.Win.Chart.BarSeries), 400, 300, 300, 80,
FarPoint.Win.Chart.ChartViewType.View3D, False)
chart.Locked = True
'chart.CanSize = FarPoint.Win.Spread.DrawingSpace.Sizing.None
'chart.CanMove = FarPoint.Win.Spread.DrawingSpace.Moving.Horizontal
```

9. 与其他数据格式交互

9.1 打开 Excel 文件

在 FarPoint Spread 中，您可以打开一个已经存在的 Excel 格式文件(Biff8 格式或者是 xlsx)或者是一个用逗号隔开的文本文件。

在 FarPoint Spread 组件中，您可以打开全部多表单文件或者指定一个单独的表单(通过名字或者索引)然后在指定的电子表单中打开。

FarPoint Spread 可以在绑定和非绑定两种模式下使用。打开一个 Excel 文件, FarPoint Spread 可以在非绑定模式下使用。在非绑定模式下 DataSource 属性返回 null (在 Visual Basic 下 Nothing)。

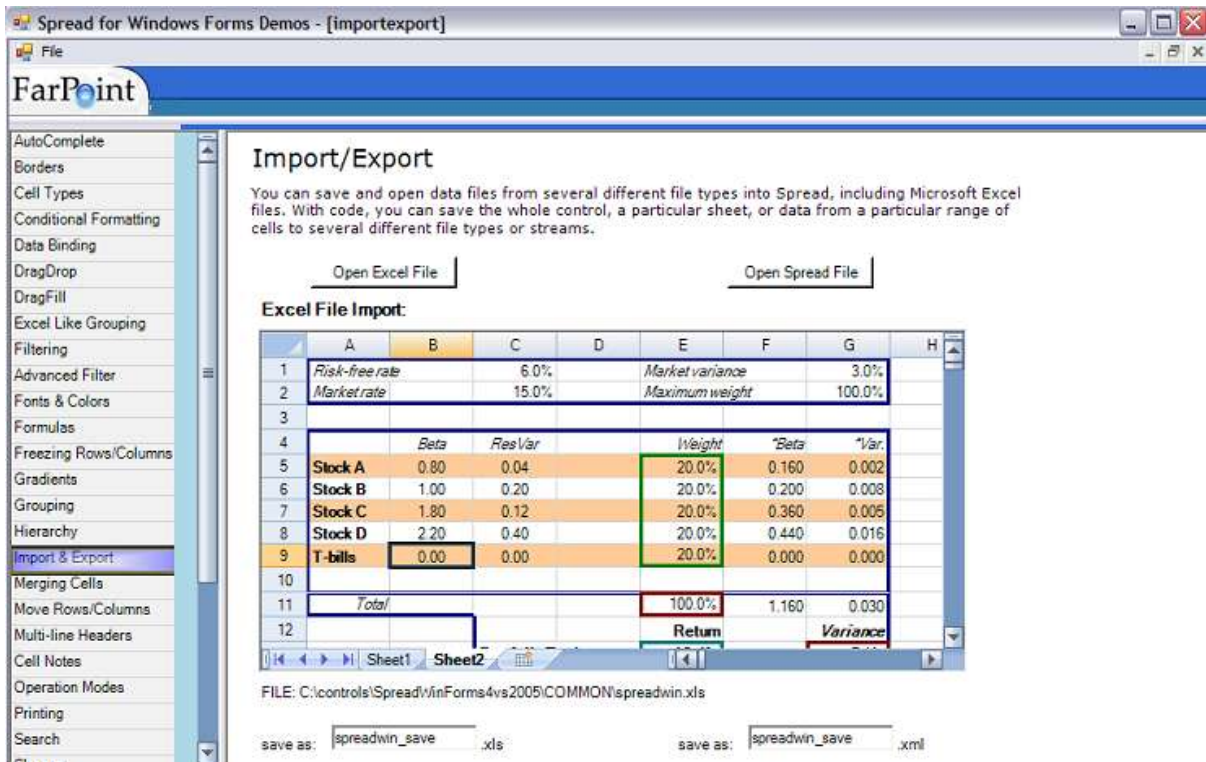
使用 FpSpread 类的一个 OpenExcel 方法来打开 Excel 文件中的所有电子表单。在打开 Excel 文件的时候需要提供文件的路径和名字和其他额外的信息来打开 Excel 文件。您可以使用 ExcelOpenFlags 枚举设置额外的打开选项。这个枚举允许您决定冻结行和冻结列如何导出，是否只有数据导出，等等其他选项。使用 SheetView 类中的 OpenExcel 方法中的一种方法，通过指定电子表单的名字和数字打开一个指定的 Excel 文件的电子表单。

在 ExcelOpenFlags 或 ExcelSaveFlags 枚举，文件缓存选项允许用户打开、编辑和保存无损的高级文件内容和格式。如果打开文件格式类似保存的文件格式就可以确保内容无损。如果文件内容除了 xls 文件 (x) 的文件，还有更高级的内容，则要确保其他文件与 xls(x)在同一文件夹中。高级的内容可能是宏，ActiveX 控件，数据连接等。

注意表单的索引参照 Excel 文件以 0 开始，因此第一张表在 Excel 文件中的索引是 0，第二章表的索引是 1，等等。

当您准备在 FarPoint Spread 中打开文件时，如果 Excel 文件被其他应用程序打开，文件就不会被导入，FarPoint Spread 也不会包含任何导入数据。

下图展示了 SpreadWinDemo 工程中使用 OpenExcel 方法，SpreadWinDemo 工程在 C:\Program Files\FarPoint Technologies\Spread.WinForms.v5\v5.x.x.\Samples\Vb\SpreadWinDemo 路径下。



使用代码

您可以使用 FpSpread 类的一种 OpenExcel 方法来打开 Excel 文件。您也可以使用 SheetView 类的 OpenExcel 方法打开的 Excel 文件中一个指定的表单 (使用 Sheets 或者 ActiveSheet)。

示例

下面的代码使用 FpSpread 类来打开全部 Excel 格式文件，然后把 Excel 文件中特定表单的数据加载到 FarpointSpread 组件中指定的表单。

```
[C#]
// 打开 Excel 文件的第三个表单.
fpSpread1.ActiveSheet.OpenExcel("C:\\excelfile.xls", 3);

[Visual Basic]
'打开 Excel 文件的第三个表单.
FpSpread1.ActiveSheet.OpenExcel("C:\\excelfile.xls", 3)
```

使用 Spread 设计器

1. 在“文件”菜单中选择“打开”。
2. 一个对话框将出现提醒您如果您打开一个文件将覆盖您已经存在的设置。如果您想打开一个文件，选择 Yes 继续打开这个文件。
3. 一个打开对话框出现。

4. 在类型框里选择 Excel 文件(*.xls)，为了打开用逗号分隔的文件，在类型框里选择用逗号分隔文件(*.csv)。
5. 选择路径和文件名，然后打开文件。如果一个文件被成功打开，一个对话框将提示指定文件已经被打开。
6. 点击 OK 来关闭 Spread 设计器。

9.2 保存为 Excel 文件

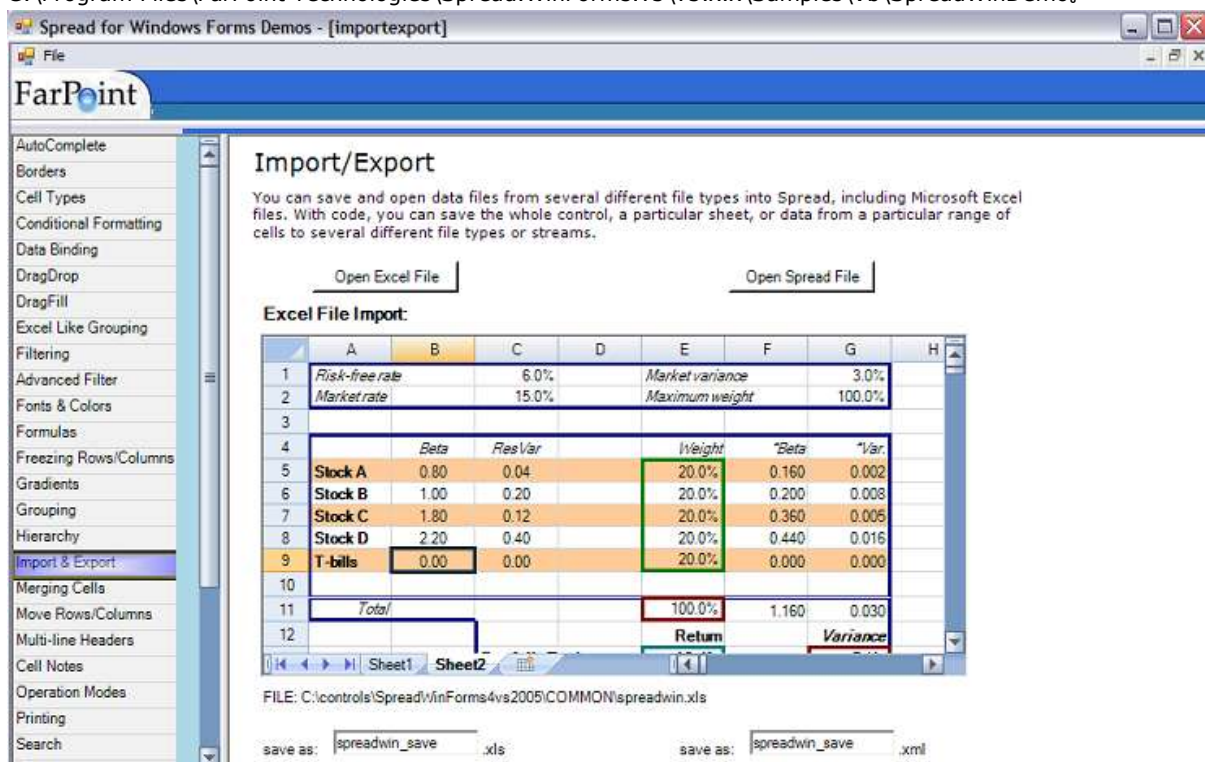
您使用 ExcelSaveFlags 枚举 UseOOXMLFormat 选项可以把数据保存成 Excel 格式的文件 (BIFF8 格式) 或 Excel 2007 中的 XML 格式 (xlsx)。默认情况下，当您保存 Excel，会将 FarPoint Spread 中存储的数据模型导出到一个文件或 BIFF8 格式流。

如果您把一个数字或日期导出到 Excel 单元格并且 Excel 那一列的宽度不够大，Excel 将会显示 # # 在单元格中。请确保 Excel 列的宽度设置足够宽，以显示导出的 Excel 数据格式的文件。

详情请参考 FpSpread 类的 SaveExcel 方法。有许多不同的 SaveExcel 方法。某些方法有 saveFlags 选项。这个 saveFlags 选项允许您指定头和其他选项。头将导出为冻结列和冻结行。

文件缓存选项的 ExcelOpenFlags 或 ExcelSaveFlags 枚举允许用户在不损失高级文件内容和格式的情况下打开，编辑和保存。只有在打开的文件格式类似于保存的文件格式时内容才真正无损。如果高级文件的内容除了 xls 文件 (x)，那么其他高级内容必须确保与 xls (x) 在同一文件夹中。高级的内容可能是宏，ActiveX 控件，数据连接等。

下面的图片是使用 OpenExcel 方法以后的效果。项目 SpreadWinDemo 位于 C 盘以下路径：
C:\Program Files\FarPoint Technologies\Spread.WinForms.v5\v5.x.x\Samples\Vb\SpreadWinDemo。



使用代码

使用 FpSpread 类的一个 SaveExcel 方法可以导出 Excel 文件，使用中需要提供文件名字以及路径和其他依赖于指定方法的额外参数。

示例

下面的示例代码保存了 FarPoint Spread 组件的数据到一个 Excel 格式文件。行头和列头都会被导出。

```
[C#]
// 把数据保存到一个 Excel 格式的文件,包括头。
fpSpread1.SaveExcel("C:\\excelfile.xls", FarPoint.Win.Spread.Model.IncludeHeaders.BothCustomOnly);

[Visual Basic]
'把数据保存到一个 Excel 格式的文件,包括头。
FpSpread1.SaveExcel("C:\\excelfile.xls", FarPoint.Win.Spread.Model.IncludeHeaders.BothCustomOnly)
```

使用 Spread 设计器

1. 在“文件”菜单中选择“保存”，保存对话框出现。
2. 从类型框里选择保存成 Excel 文件格式 (*.xls)。
3. 指定保存的路径和文件名字然后点击“保存”。如果保存成功，一个对话框将显示指定的文件已经被保存。
4. 点击 OK 关闭 Spread 设计器。

9.3 导出 PDF

您使用 PrintInfo 类的 PrintToPdf 方法可以打印电子表单到 PDF 文件格式。使用 PdfFileName 属性可以指定文件名字和保存的路径。因为 PrintInfo 是在单独的电子表单中被调用的，所以这个方法只能打印单个电子表单。

使用代码

调用 PrintInfo 类中的 PrintToPdf 方法来打印指定电子表单。

示例

此示例演示了如何使用代码保存电子表到 PDF 格式文件。

```
[C#]
FarPoint.Win.Spread.PrintInfo printset = new FarPoint.Win.Spread.PrintInfo();
printset.PrintToPdf = true;
printset.PdfFileName = "D:\\results.pdf";
//设置打印机设置然后打印。
fpSpread1.Sheets[0].PrintInfo = printset;
fpSpread1.PrintSheet(0);

[Visual Basic]
Dim printset As New FarPoint.Win.Spread.PrintInfo()
printset.PrintToPdf = True
printset.PdfFileName = "D:\\results.pdf"
'设置打印机设置然后打印。
FpSpread1.Sheets(0).PrintInfo = printset
FpSpread1.PrintSheet(0)
```

使用 Spread 设计器

1. 选择“文件”菜单标签(SpreadButton 图标)。
2. 选择“打印”。
3. 选择“打印 PDF”。
4. 使用保存对话框来指定路径和文件名。

10. Spread Win 5 中英文术语对照

为明确对应的中英文术语，帮助理解文档内容，特建此表。

中文	英文
单元格	Cell
行	Row
列	Column
单元格样式	Cell Style Info
单元格类型	Cell Type
单元格批注	Cell Note
工作表	Work Sheet (In Spread, it's one sheet)
工作簿	Work book(In Spread, it's the whole Spread)
工作表外观	Sheet Skin
公式	Formula
子工作表	Child Sheet View
过滤	Filter
排序	Sort
Excel 样式的行、列组合	Range Group
Outlook 样式的分组	Group by GroupDataModule supported
条件格式	Condition Format
工作表窗口	View Port
操作模式	Operation Mode
头	Header
窗体	Form
图标	Indicator
图形	Shape
图表	Chart
设计器	Designer
系列	Series
选项条	Tab Strip
分组栏	Group Bar
右键菜单	ContextMenu
调整柄	Handle
冻结	Freeze
合并	Span
标题	Title
副标题	Subtitle