

# Qt 设计器手册

前言

创建一个 Qt 应用程序

创建含有工具栏和菜单的主窗口

走近设计器

派生类和动态对话框

创建自定义窗口部件

创建数据库应用程序

定制和集成 Qt 设计器

参考：快捷键

参考：菜单选项

参考：工具栏按钮

参考：对话框

参考：向导

参考：窗口

参考：.ui 文件格式

# 前言

## 介绍

这本参考手册是有关于*Qt 设计器*的，*Qt 设计器*是用来设计和实现用户界面并能够在多平台下使用的一种工具。*Qt 设计器*可以使用户界面设计实验变得简单。在任何时候你可以要求所生成的代码去重建*Qt 设计器*产生的用户界面文件，并可以根据你的喜好来改变你的设计。假如你使用的是先前的版本，你将发现在新的版本下自己可以立即进入工作，因为新的版本在界面上基本没有什么变化。但是你将发现根据你们的反馈而开发出的新的部件和新的或者改进的功能。

*Qt 设计器*帮助你使用部局工具在运行时自动的移动和缩放你的部件(Windows中的术语*控件*)来构建用户界面。最终界面是既好用又好看，使最终用户拥有一个舒适的操作环境并且能够方便的进行参数选择。*Qt 设计器*支持信号和槽机制以使部件间能够进行有效的通信。*Qt 设计器*包含一个代码编辑器，使你能够在合成的代码里面嵌入自己定制的槽。那些更喜欢使用手工方法分解合成代码的朋友也能够继续使用基类，因为从第一版的*Qt 设计器*开始就把这些基类移植进去了。

这本手册通过讲述开发例程来向你介绍*Qt 设计器*。一开始的六章是设计指南，而且各自间都尽可能设计成是独立的。接下来要介绍的是除了首章以外的每一章，并假定你已经熟悉了第一章的内容，该章包含使用*Qt 设计器*创建一个Qt应用程序的基础。以下是便各章的简要概述：

- 第一章，[创建一个Qt应用程序](#)，通过带着你一步一步的创建一个小但功能完整的应用程序来介绍*Qt 设计器*的使用。按照着这种方法你将学到如何创建一个窗体并且向窗体中添加部件。在你阅读这一章的过程中你将使用窗体和属性编辑器来定制你的应用程序，并且学习怎样使用部局工具来对一个窗体进行部局。你也将学到如何使用信号和槽机制和*Qt 设计器*的内建代码编辑器来制造应用程序的各种功能。我们也将解释如何使用qmake来生成Makefile，以致于你能够编译和运行应用程序。
- 第二章，[创建含有工具栏和菜单的主窗口](#)，我们将创建一个简单的文本编辑器。通过写这个应用程序你将学到如何使用菜单栏和工具栏来创建一个主窗口。我们将看到如何使用Qt的内建功能来处理一般任务(e.g. 复制粘贴操作)还将看到如何为我们自己的菜单栏选项和工具栏按钮创建我们自己的功能。
- 第三章，[走近设计器](#)，提供有关*Qt 设计器*的信息如相关的开发应用程序，并且还对*Qt 设计器*背后的一些基本原理进行解释。
- 第四章，[派生类和动态对话框](#)，将展示如何派生一个窗体；这将让你清楚的通过执行关键代码的功能来分解用户界面。本章中还附加有关qmake和 uic的信息。本章也将阐述如何使用QWidgetFactory把.ui文件放进你的应用程序从而动态的加载对话框和如何访问这些对话框的部件和派生部件。
- 第五章，[创建自定义窗口部件](#)，告诉你如何才能创建自定义部件。既有在第一版的*Qt 设计器*中就被介绍的简单方法，又有像利用插件这种新的更有效的方法都在这一章里被介绍了。

- 第六章， [创建数据库应用程序](#)介绍了Qt的SQL类并且带着你通过一个实例来演示如何执行查询和如何设置主要关系的细节，深入讲解和处理外关键字。
- 第七章， [定制和集成Qt设计器](#)，聚焦Qt设计器本身，向你展示如何定制设计器，如何使用可视化工作室集成设计器和如何创建一个Makefiles。

这剩下的章节提供了一些参考资料，用以讲述Qt设计器的菜单选项、工具栏、快捷键以及对话框等的细节。

## 你所应该知道的

该手册假定你已经有了一些有关 C++和 Qt 应用程序开发框架的基础。假如你需要学习 C++或者 Qt，这儿有大量的 C++的书可供使用和少量的但是数量却在不断增长的有关 Qt 的书。你可以尝试一下大量伴随着 Qt 的联机文档和许多例程。

企业版的Qt包含了SQL模块。在[创建数据库应用程序](#)一章里我们演示了如何使用Qt设计器来编写SQL应用程序；这一章需要一些SQL和关系数据库的知识。

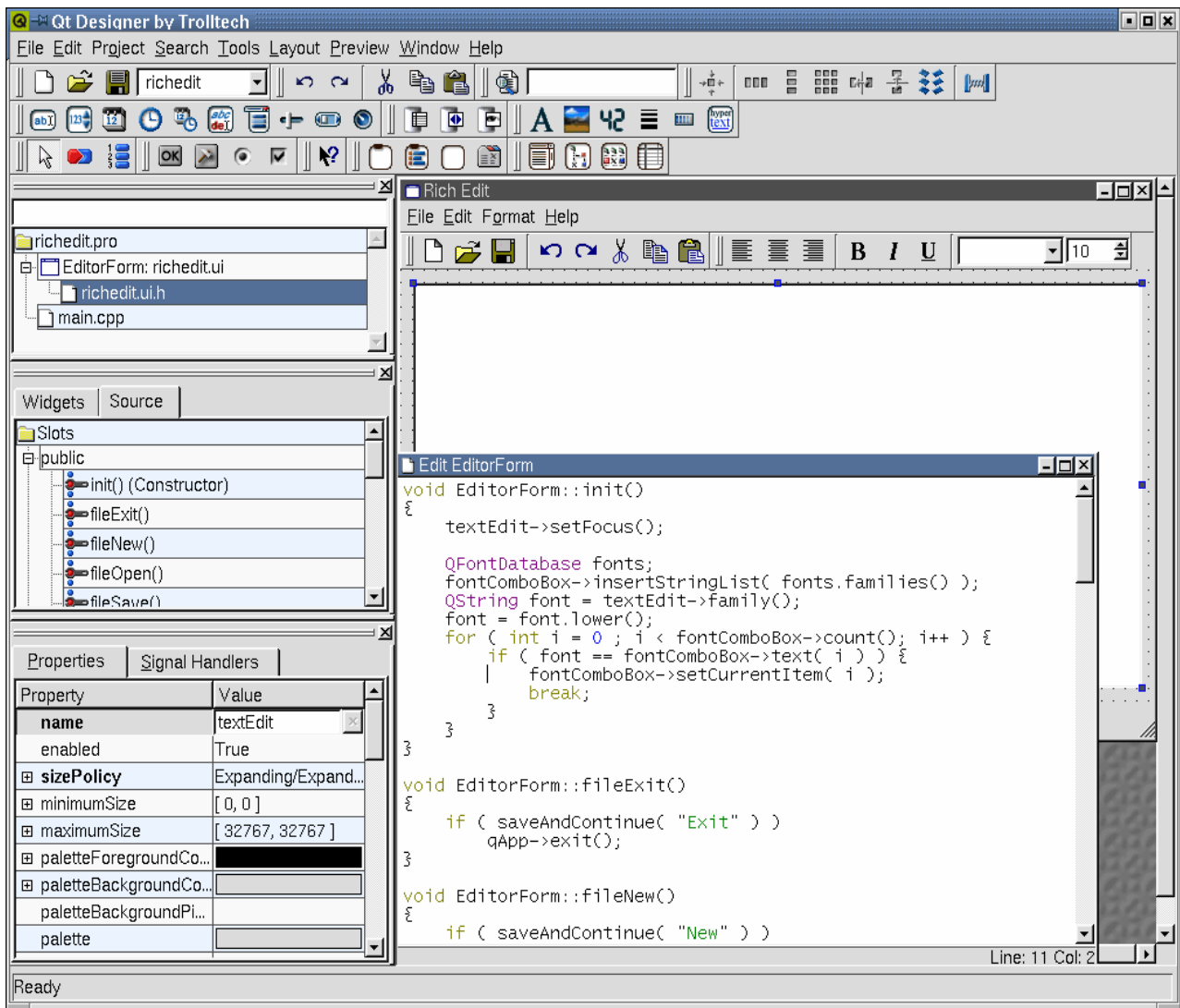
## Qt设计器中的新东西

该版本的Qt设计器较其前一版本来说增加了更加多的功能。自己定制的槽功能代码可以直接在Qt设计器中编辑；操作工具栏、菜单栏就可以创建主窗口了；部局可以结合使用分解器；一些插件还允许你把大量的自定义部件进行打包并且在Qt设计器中可以使用他们。从对用户界面微小的改善到提高效率还有很多其它增强功能被结使在里面，例如在一个应用程序里的所有窗体可以有效的共享像素映射。

该版本的Qt设计器创建的工程文件使得开关在一个应用程序中的所有窗体变得十分简单，而且仍然保持了一个通常的数据库设置和映象。通过对派生类的全面支持，为直接在Qt设计器中编写代码带来了很多的益处，这些知识已经全面涵盖在[走近设计器](#)一章中了。

还介绍了一个新的库libqui，该库允许你在运行时从Qt设计器的.ui文件中自动加载对话框。这允许你提供给你的应用程序用户相当可观的自定义界面自由度，否则就需要使用C++了。

如果你仅仅想要一个简单而功能强大的单对话框可视设计工具，虽然新版本的Qt设计器介绍了新的进阶和技术但你可以忽略这些方面并且正确的使用与Qt 2.x相同的方法。



Qt 设计器

## 反馈

如果你关于这个手册有一些注释、建议、批评或者适当的赞美，请访问[doc@trolltech.com](mailto:doc@trolltech.com)让我们知道。关于Qt或者Qt 设计器的bug报告可以发送至[Qt-bugs@trolltech.com](mailto:Qt-bugs@trolltech.com)。你也许也想要加入专门由开发者阅读和捐献的Qt-interest邮件列表；请访问<http://www.trolltech.com>以了解更多的细节。

# 创建一个 Qt 应用程序

## 启动和退出 Qt 设计器

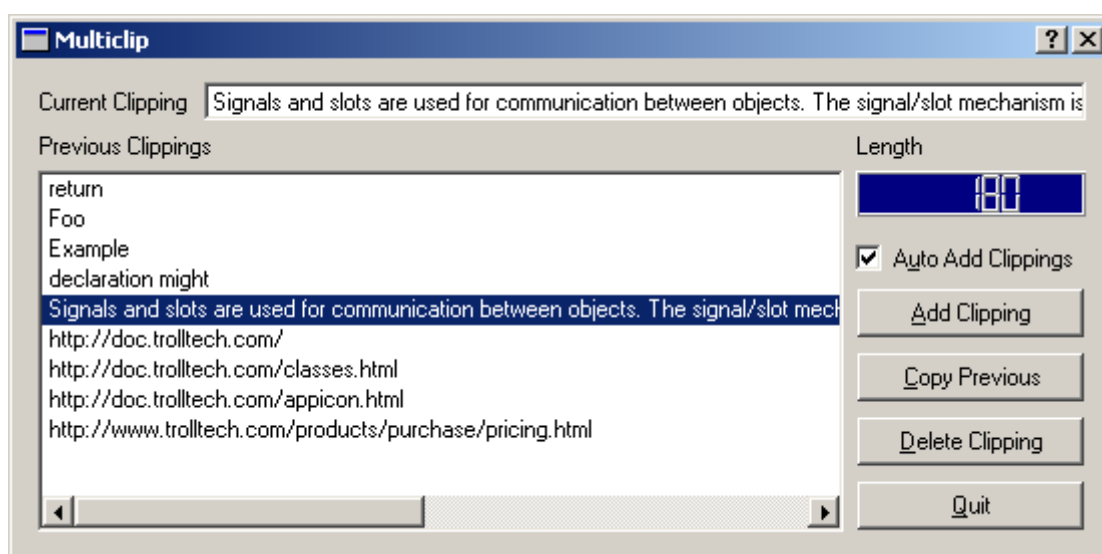
Qt 设计器和任何其他现代桌面应用程序一样已同样的方式被控制着，为了在 Windows 系统下启动 Qt 设计器，点击“开始”按钮，依次点击“程序|Qt X.x.x|Designer”（X.x.x 是 Qt 的版本号，例如 3.0.0）如果你是在 Unix 或者 Linux 操作系统下，你可以双击 Qt 设计器图标，或者在终端输入 designer。

当你完成使用 Qt 设计器，点击 File|Exit 退出时，设计器会提示你保存任何没有保存的改动，按下 F1 键或者点击“Help”菜单，你可以获得帮助。

为了使你从教程中获得最大收益，我们建议你现在开始启动 Qt 设计器，并且当你读到这些例子程序时创建它们，包括使用 Qt 设计器菜单、对话框和编辑器，大部分工作仅仅只需要键入少量的代码。

默认情况下，当你启动 Qt 设计器时，你会看到位于顶部的一个菜单栏和各种工具栏，在左边是三个窗口，第一个是文件（File）窗口，第二个是部件（Widget）和来源（Source）窗口（对象浏览器），第三个是属性（Properties）窗口。文件窗口列举了和该项目相关的文件和图像文件，要想打开任何窗体只需点击文件列表中对应的文件；部件和来源窗口列出了当前窗体的部件和槽，属性窗口用于浏览和改变窗体和部件的属性。当我们创建例子程序时，我们将介绍 Qt 设计器窗体、对话框、菜单选项和工具栏按钮的使用。

在这章我们将创建一个称为“Multiclip”的应用程序，它允许你存储多行文本到剪贴板或者从剪贴板获得多行文本。

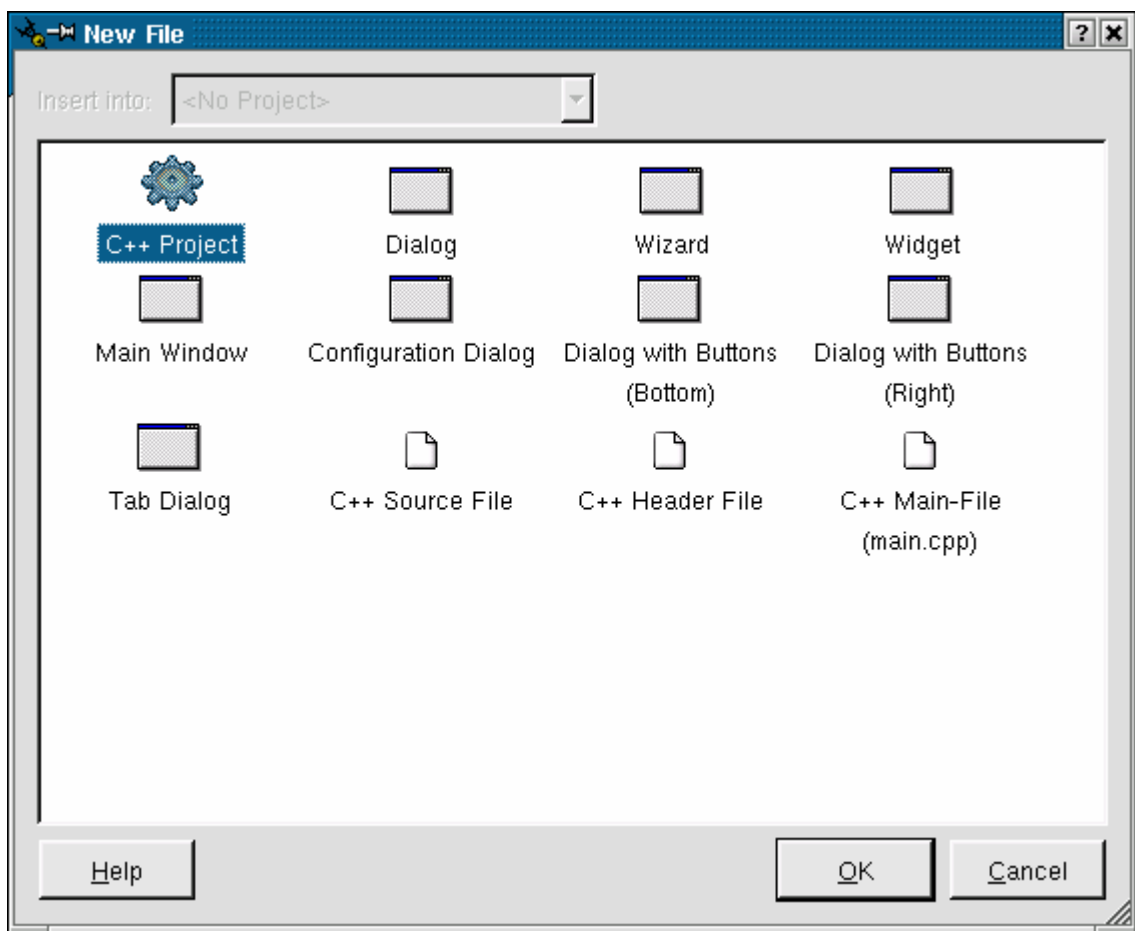


The Multiclip Application

## 创建一个项目

任何时候你创建一个新的应用程序，我们强类推荐你创建一个工程或者打开一个工程文件而不是只是创建一个单独的.ui 文件，使用一个工程有这样的好处：你为这个工程创建的所有窗体(form)可以在文件打开的对话框中仅仅通过鼠标点击加载，使用工程文件的另一个好处是就是，其允许你存贮你所有的图像文件到一个单一的文件而不是复制他们到将出现的每一个窗体中，请查看[走进设计器](#)章节的[工程管理](#)部分以获得更多的使用工程文件的好处的信息。

如果你仍未启动那现在就开始吧，点击 File|New 激活新文件对话框，点击“C++ Project”图标，点击“OK”按钮，激活项目设置对话框，你需要给出项目的文件名，我们推荐你将每一个工程文件放到其自己的子文件夹中，点击“...”按钮，激活 Save As 对话框，并浏览至你需要存放项目文件的地方，点击“Create New Folder”工具栏按钮，创建“multiclip 对话框”，双击“mulitclip”目录是其成为当前工作目录，键入一个文件名“multiclip.pro”，并点击 Save 按钮。项目设置对话框的“Project File”区域将显示你的信项目的路径名和名字，点击”OK”创建这个项目。



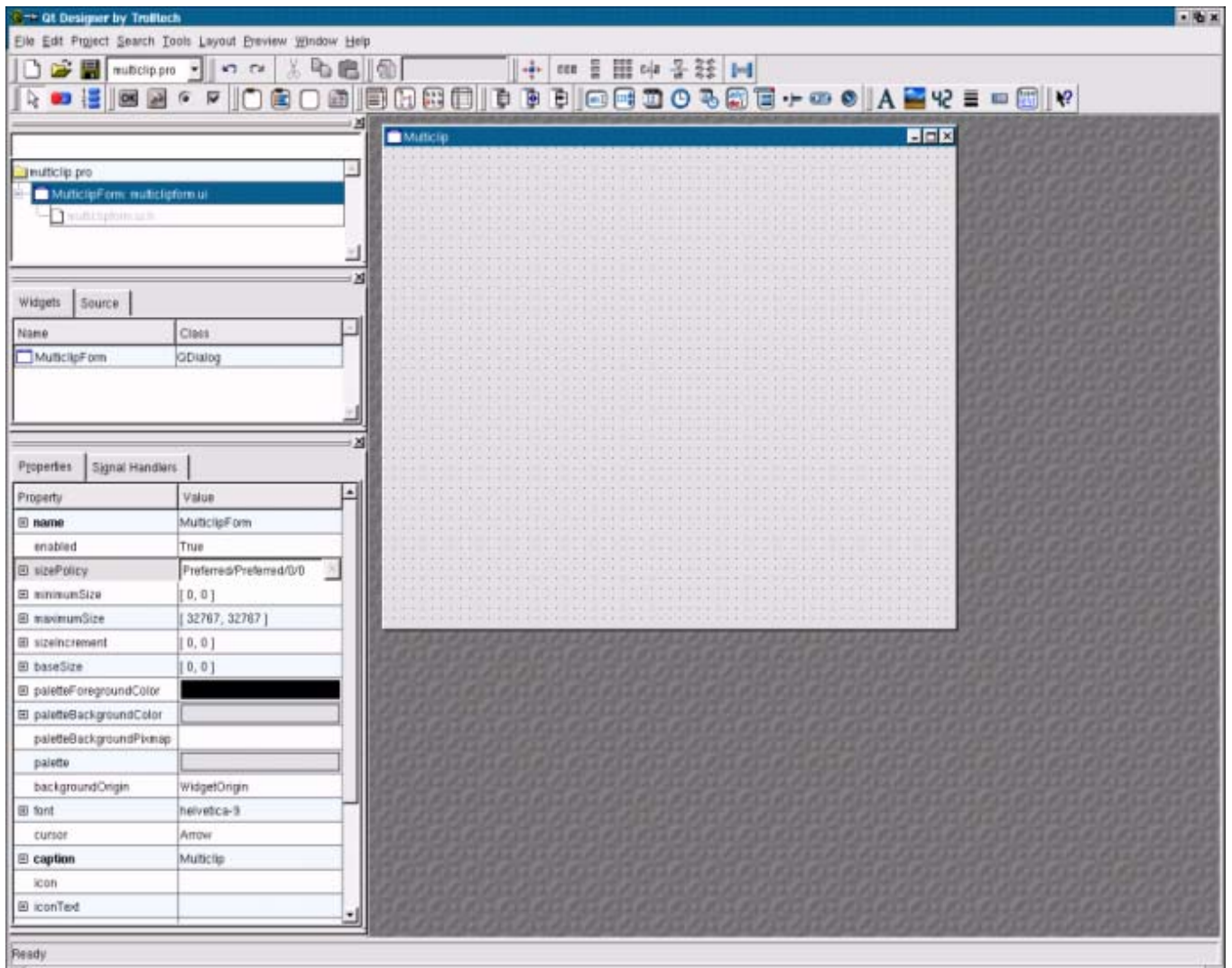
创建一个新项目



当前项目的名字显示在默认条件下左上工具栏中 Files 工具栏里，一旦你有了一个项目我们就能加入窗体并开始创建我们的应用程序，（请参看 Customizing Qt Designer 部分以获取有关改变 Qt 设计器的工具栏和窗口样式以符合你的需要的更多信息）

## 创建一个新窗体

点击 File|New 激活 New File 对话框，有几个默认的窗体出现，但是我们将使用默认的对话框形式，于是仅仅只需点击 OK，一个新的叫做“Form1”的窗体将会显示出来，注意到这个新的窗体被列表在文件列表栏，并且属性窗口显示了窗体的默认属性设置。

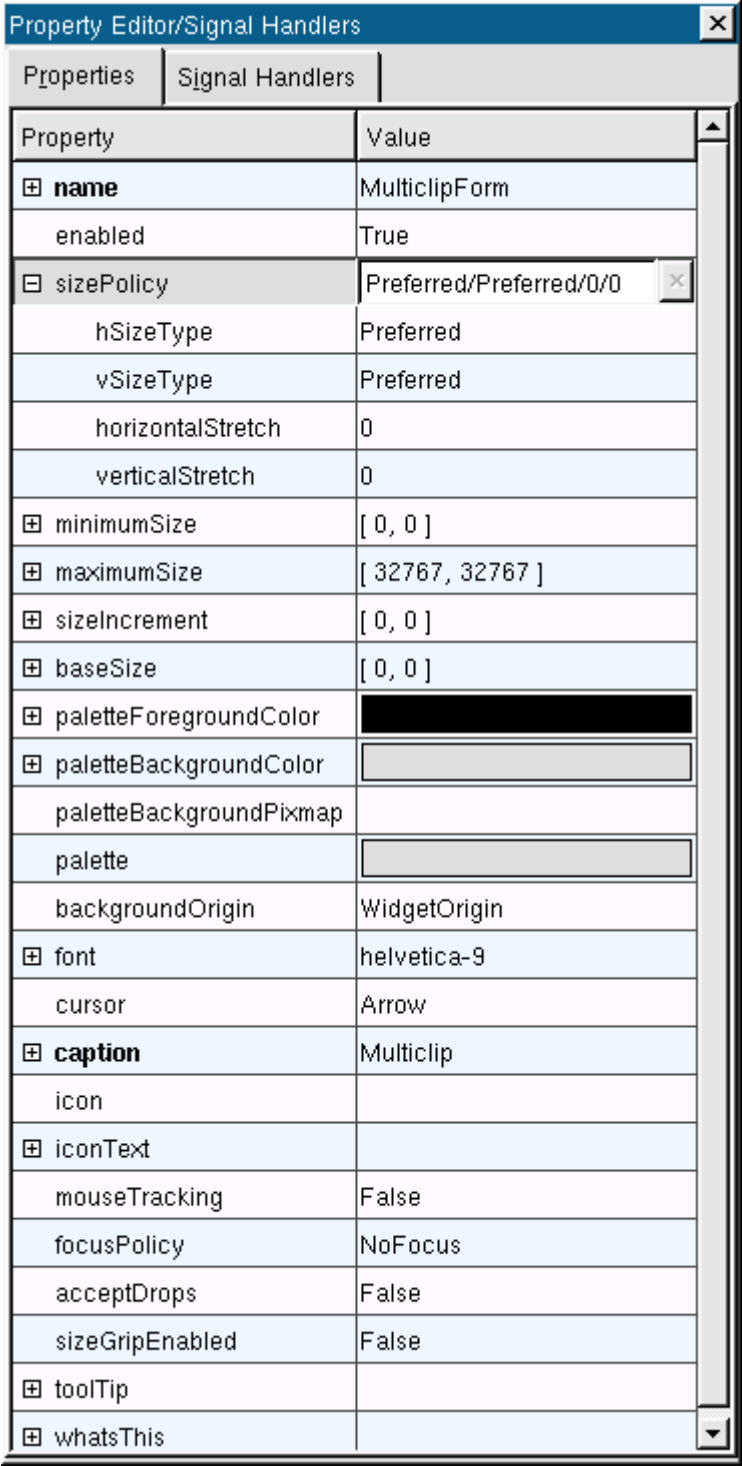


## 创建一个新窗体

点击 name 属性旁边的值，改变窗体的名字为“MulticlipForm”，改变窗体的标题为“MultiClip”，属性依据继承层次排序，标题大约在属性编辑器的中间显示，保存该窗体：点击 File|Save，键入名字“multiclip.ui”，于是点击“Save”按钮保存设置。

## 使用属性编辑器

属性编辑器有两列，属性列列出了属性的名字，值列列出了值（属性的取值），有些属性名字有一个“+”号，在他们左边的区域，这表示这个属性名是一系列相关属性的集合名字，点击窗体让属性编辑器显示窗体的属性，点击 SizePolicy 属性的加号，你将会看到四个属性缩进显示如下：sizePolicy, hSizeType, vSizeType, horizontalStretch 和 verticalStretch. 这些属性和其他属性一样以同样的方式在编辑器中修改。



Property Editor/Signal Handlers	
Properties	Signal Handlers
Property	Value
⊕ name	MulticlipForm
enabled	True
⊖ sizePolicy	Preferred/Preferred/0/0
hSizeType	Preferred
vSizeType	Preferred
horizontalStretch	0
verticalStretch	0
⊕ minimumSize	[ 0, 0 ]
⊕ maximumSize	[ 32767, 32767 ]
⊕ sizeIncrement	[ 0, 0 ]
⊕ baseSize	[ 0, 0 ]
⊕ paletteForegroundColor	
⊕ paletteBackgroundColor	
paletteBackgroundPixmap	
palette	
backgroundOrigin	WidgetOrigin
⊕ font	helvetica-9
cursor	Arrow
⊕ caption	Multiclip
icon	
⊕ iconText	
mouseTracking	False
focusPolicy	NoFocus
acceptDrops	False
sizeGripEnabled	False
⊕ tooltip	
⊕ whatsThis	

属性编辑框



有些属性具有简单是值，例如，名字(name)属性是一个文本值，宽度属性(width)（例如最小尺寸 minimmSize）是一个数字值，为了改变文本值，点击当前文本并键入你的新文本。要改比那数字值，点击改制同时键入新的数字，或者用翻动按钮(Spin button)增加或者减少当前的数字直道达到你所期望的数字，有些属性有一些固定的属性列值，例如，mouseTracking 属性是一个布尔型，能设置为真或者假，cursor 属性也有一列固定的值，如果你点击 cursor 属性，或者 mouseTracking 属性，则其值会显示在下拉组合框中，点击向下的箭头查看可用的属性取值，一些属性有复杂的取值集合，例如字体属性，如果你点击了字体属性后的省略号按钮，将会出现字体选择对话框，你可以在其中改变字体设置。其他有省略号按钮的属性设置依据属性所有的属性值会显示不同的对话框，例如，如果你要为一文本属性键入许多的文本，你可以点击省略号按钮激活多行文本编辑器对话框。

改变了的属性的名字以粗体字显示，如果你改变了属性值但是你又想恢复它到默认值，点击属性值并点击红色的”X”按钮，直到变成正确的值，有些属性有一个初始值，如“TextEdit1”，但不是默认值，如果你想恢复有初始值但是没有默认值的属性（通过点击红色的”X”按钮），如果属性，如名字，允许为空，则属性值会变成空值。

如果选中多个部件，属性编辑器会显示选中的部件的共有的属性，改变其中一个属性会导致所有选中的部件的该属性值的改变。

属性编辑器完全支持撤销和重复操作(Ctrl+Z 和 Ctrl+Y, 同样可以选择编辑菜单 Edit)

## 增加部件

这个 MultiClip 应用程序包含一个文本框显示当前剪贴板的文本（如果有的话），一个列表框显示前一个剪辑，一个长度指示器，一个校验框和按钮，如果你运行这个应用程序并改变它的大小，则所有的部件会按比例变化。

走进 Qt 设计器这一节讲述了设置一个窗体并将所需的部件放置到他们将显示的合适位置，使用布局工具正确的设置他们的大小和位置，现在我们添加 multiclip 窗体的部件

我们从当前的剪贴文本框开始，点击 Text Label 工具栏按钮并将鼠标指针移动到窗体左上方（如果你将鼠标在工具栏按钮上停留，则工具按钮名字会以标签形式显示出来），我们不想因为对这些标签重新命名而自寻麻烦，因为在代码编写过程中不会设计到它，但是我们需要改变它的文本属性，改变其文本属性(text property)为“Current Clipping”。(属性编辑器的解释请看使用属性编辑浮动栏部分)

点击工具栏中的 Line Edit 按钮并在窗体的右上方点击放置该部件，使用属性编辑器改变该部件的名字为“currentLineEdit”。

1. 现在我们将添加另一个标签和列表框，点击 Text Label 工具栏按钮，并在“Current Clipping”标签下方放置，改变其文本属性(text property)为“Previous Clippings”，不要为这些部件放置的位置是否恰到好处而担心，布局工具（下部分会涉及到的）将会帮助你做好这一切。

点击 List Box 工具栏按钮，并点击鼠标放置在“Previous Clippings”标签下方，改变该列表框的名字为“clippingListBox”，默认情况下 Qt 设计器会以一个初始值“New Item”创建列表框，但我们不需要这个值（后面我们会以代码的方式组装我们的列表框），因为我们需要删除该值，在列表框处点击鼠标右键，在弹出的菜单中选中“Edit”菜单子项激活列表框的值编辑对话框，点击“Delete Item”按钮删除默认的项，然后点击“OK”按钮。（注意看值编辑器工具条）

2. 我们想知道当前剪贴文本的长度，所以我们将添加另一个标签和一个 LCD Number 部件。

点击 Text Label 工具栏按钮，并将其放置在 Line Edit 下方，改变其文本属性值为“Length”，点击 LCD Number 工具栏按钮并放置在长度标签下方，改变 LCD Number 的名字为“lengthLCDNumber”。

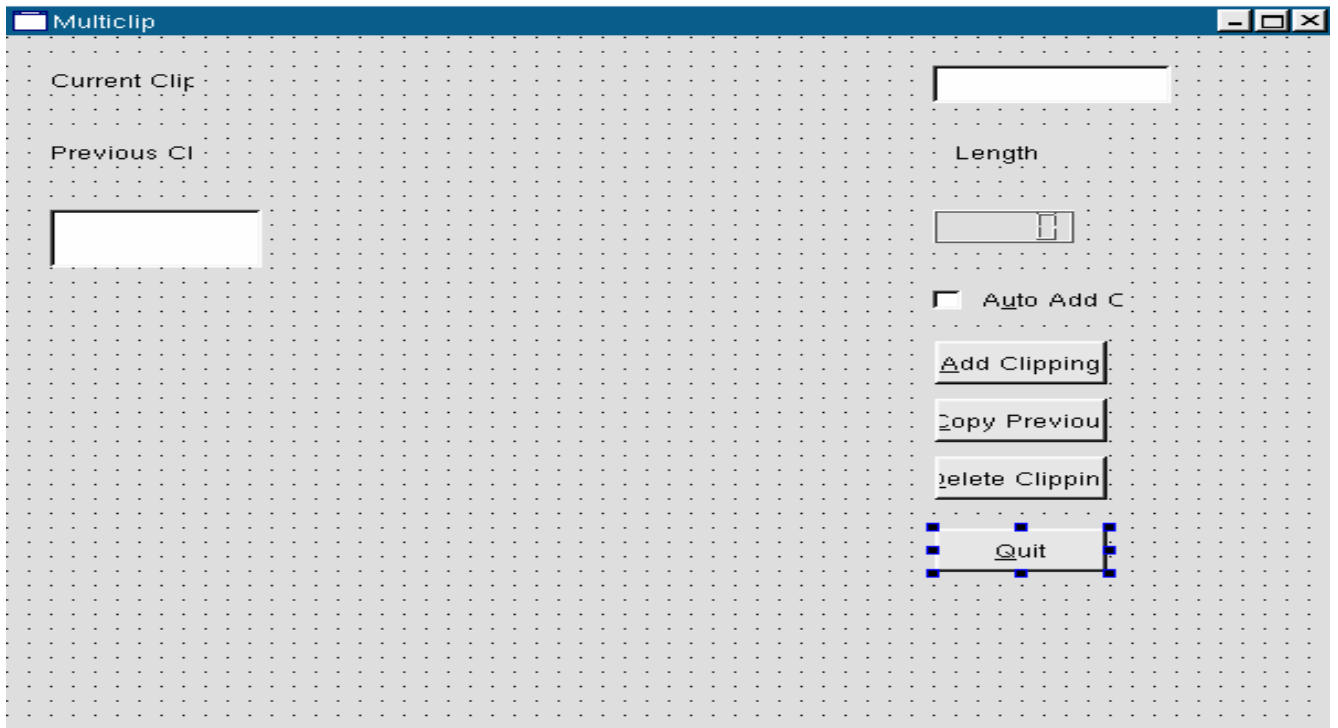
3. 多项剪辑程序能检测到剪贴板的变化并自动增加新的剪辑，无论这是是否发生，我们想让用户控制它，于是我们提供一个检测按钮来指示他们的最优选择。

点击 Check Box 工具栏按钮并放置到 LCD Number 下方，改变检测框的名字为“autoCheckBox”，改变其文本属性值为“A&uto Add Clippings”，注意到加速键属性值自动改变为 Alt+U，因为文本中的符号 & 指明了一个键盘快捷键。

4. 最后我们需要的部件是按钮，添加同种类型部件的一种方式是个添加该部件，复制它并重复的粘贴之，这里我们使用另一种方式。。

双击 Push Button 工具栏按钮，然后在检测框的下方点击放置一个按钮，然后再这个刚刚添加的按钮下方点击鼠标放置第二个按钮，如此这般，依次添加第三个和第四个按钮，点击“Pointer”（箭头型的按钮）工具栏按钮关闭自动添加同种部件的行为，改变第一个按钮的名字为“addPushButton”并将其文本置为“&Add Clipping”，改变第二个按钮的名字为“copyPushButton”并将其文本置为“&Copy Previous”改变第三个按钮的名字和文本值分别为“deletePushButton”和“&Delete Clipping”，类似的第四个按钮改变为“quitPushButton”和“&Quit”。

所有的部件的属性依据我们程序的需要改变后放置在窗体中，下一步我们将使用 Qt 设计器的布局工具正确的设置部件的大小和位置，这样当用户改变窗体的大小时部件能随着适当的变化。



## 向窗体添加部件

### 值编辑器

同时属性编辑器用于定制部件的一般属性，值编辑器用于编辑特定部件的对象值，例如一个 QLineEdit 仅能包含一个单行的文本，但是一个 QListBox 能包含任何多的项，每一项可以是一行文本、一个像素映射或者都是。为了激活一个部件的值编辑器，双击该部件，（你也可以在该部件处右击鼠标出现一个弹出的菜单，如果第一个子菜单选项是“Edit”，你能点击该项访问该部件的值编辑器对话框）不同的部件有不同的值编辑器，参看值编辑器以获得更多细节。

### 布置部件和预览

#### 布局介绍

通过对部件和部件组以垂直、水平和网格形式组合实现布局工作，以垂直和水平方式放置的部件能以布局或者分裂形式组合，唯一的不同是用户能操作这些分裂机。（各组件之间用布局工具放置组合，为了使得位置恰当，有时加入分裂机使得空间布局合理）

如果我们想一个靠着一个的放置一些部件，我们将选中他们并点击 Lay Out Horizontally 工具栏按钮，如果我们想以排列组件一个在另一个上面，我们可以使用 Lay Out Vertically 工具栏按钮，一旦我们对这些组件组合好了我们就能再次用垂直、水平和网格布局工具布置这些组件组合，一旦我们有了一个布局组合集，我们就能点击窗体本身，使用其中一个布局按钮在窗体中来优化放置写写组件组合。

一些组件会在水平方向或者垂直方向或者在这两种方向上填充可用的空间，例如按钮和线编辑为填充水平方向的空间，虽然列表视图会在两种方向上充满整个空间，获得这种布局的最简单的方式是使用 Qt 设计器的布局工具，当你在某些位置上对一些部件使用了一个布局，这些部件的布局可能不如你意，如果一个部件没有充满整个空间，试着改变其大小策略(SizePolicy)扩展之，如果一个部件占用了太多的空间，一种方法是改变其大小策略，另一种方法是使用空间器(Spacer)消耗掉多余的空间。

在运行的窗体上，空间器(Spacers)没有可视化的显示，纯粹用于在部件之间和部件组合之间插入空间，设想你有一个占用了太多空间的部件，你能消除这种布局重新设置部件大小并为空间器留有空间，于是你可以插入空间器并将其和部件一起布局，空间器会消耗掉多余的空间。如果空间器不能有效地利用空间，你可以改变其大小策略以便更好控制。

学习布局 and 空间器工具的最好方法是试着用用他们，用着这布局工具试验是很简单的，如果你对你做的改动不是满意你也可以很简单的通过点击 Edit|Undo 菜单或者按下 Ctrl +Z 键撤销这些改动，下一步我们将一步一步地对我们的多项剪辑程序布局。

## 放置部件

布局工具提供了一个将部件和组件集以水平和垂直和网格方式组合的方法，如果你用布局工具布局了带有部件的窗体，那么当用户改变窗体大小时，这些部件也会随着自动的改变尺寸，这比你使用绝对的尺寸和位置更好，因为你不必写任何代码获得窗体规模大小，并且你的用户也能充分利用他们屏幕的大小，无论他们是一个笔记本或者一个有非常大的屏幕的台式机。布局工具使用标准尺寸设置边界和部件的空间大小，这有助于使你的程序看上去一致和匀称，而这不需要你做更多的工作，比起设置绝对位置而言，布局设计工具更容易也更快上手，你只需要在窗体上合适的位置放置你的部件并使用布局工具正确的设置部件的大小和规模。

## 选择和插入部件

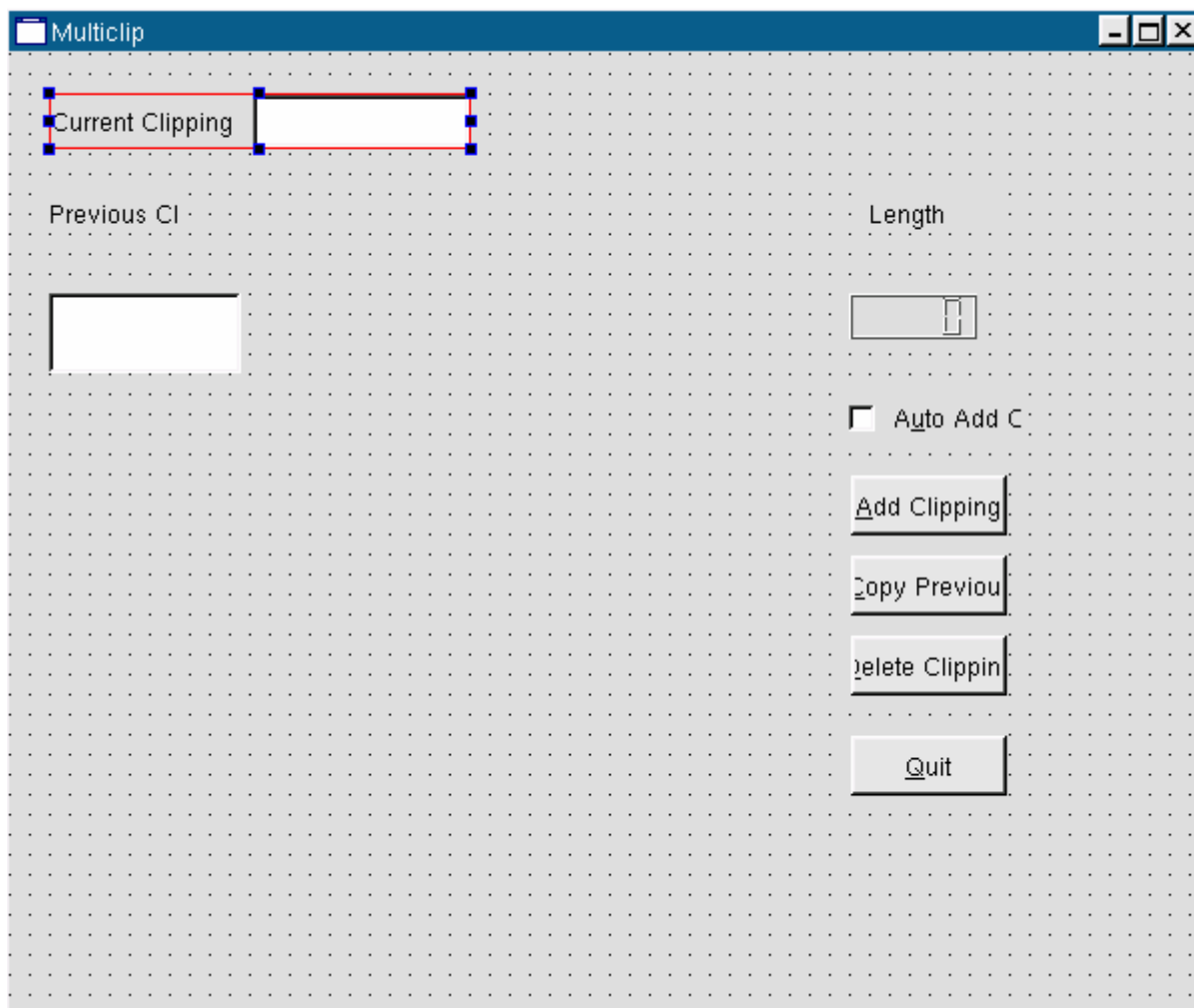
为了选择一个单独的窗口部件，或者点击该部件或者在对象浏览器窗口中点击他的名字，为了选择一组窗口部件或者点击他红色边缘线的一部分或者在对象浏览器窗口中的点击他的名字，为了选择多个部件或者多组部件，点击窗口取消任何选中的部件，然后按住 Ctrl 键，点击一个或者一组部件，然后拖动选中边框橡皮圈的边缘直到选中所有你想选中的部件或者组合部件。这项技术在选择位于另一个部件内部的部件时特别有用。例如选择一按钮组合(button group)中的单选按钮(radio button)，但是不选择该按钮组合本身，你可以点击窗体，然后按住 Ctrl+Click 键(Ctrl+鼠标点击)，选中其中一个单选按钮并拖动橡皮圈选中其他的单选按钮。

如果我们想插入一个部件到一个布局中部件之间的空隙时，我们可以点击工具栏按钮选择需要插入的新按钮，然后点击该空隙处即可。Qt Designer 会询问我们是否想打破这个布局，如果我们点击 Break Layout, 这个布局就会被打破并且我们新部件被插入了。于是我

们可以选择需要布局的部件和一组部件再次布局他们。通过点击这组部件或者点击 Break Layout 工具栏按钮或者按住 Ctrl+B 键可以达到同样的效果。

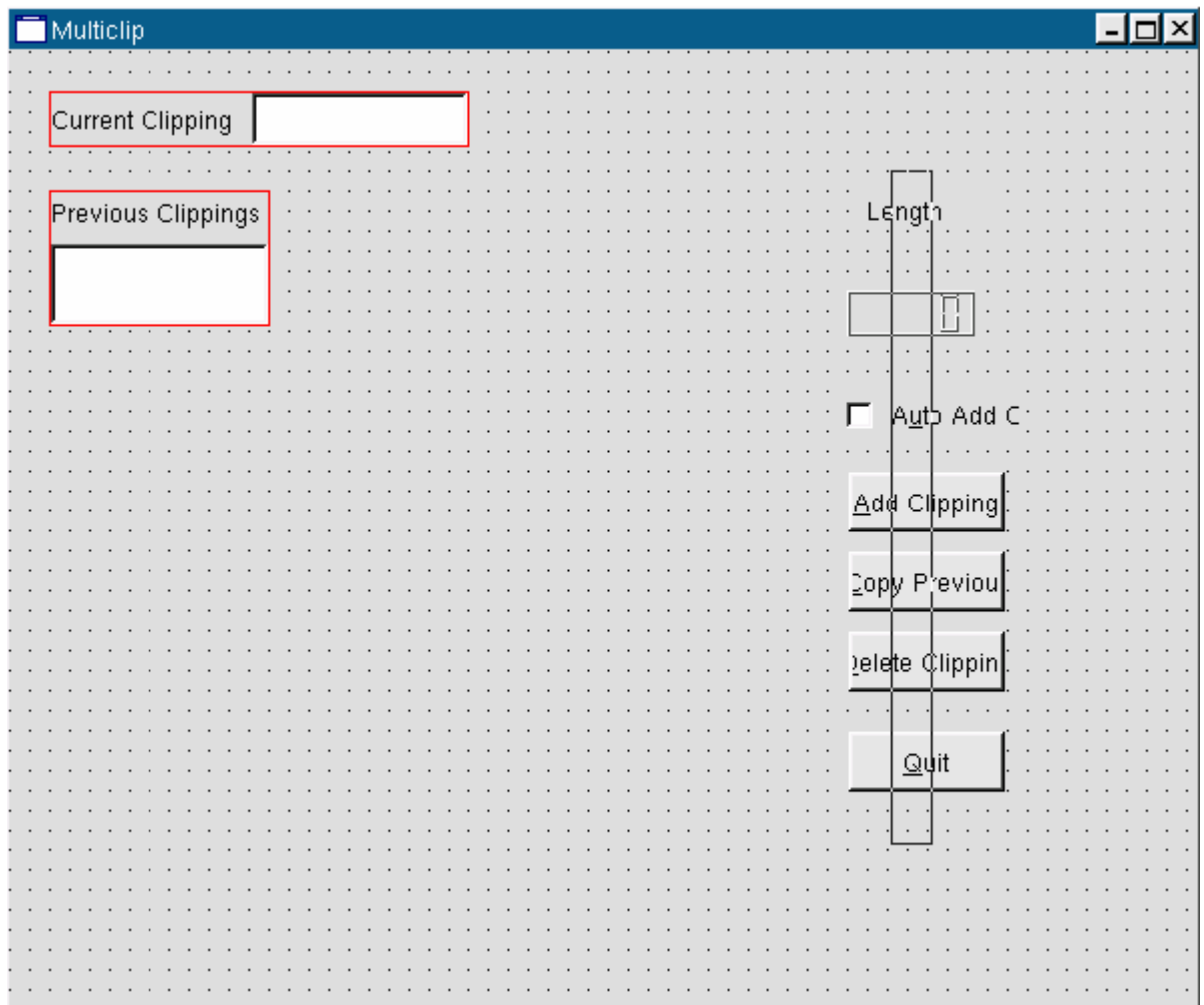
我们想要的布局方式是使当前剪辑标签和当前剪辑文本(currentLineEdit)并列放置于窗体顶部，使前一个剪辑标签和剪辑列表框(clippingListBox)充满左边的窗体，剩下的部件已列的形式放置在右边，我们想要使用一个分裂机将左右分开，并使默认情况下左边的空间大一点，部件得大小交给 Qt 设计器去做，布局控制在布局工具栏中（默认情况下，从左到右四个按钮）现在我们将布局放置在窗体上的部件。

1. 点击当前剪辑标签并按住 Shift 按钮，在点击 currentLineEdit 行编辑（Shift+Click 意思是按住 Shift 键同时点击，这可以使得 Qt 设计器能执行多项选择功能[这样的功能想必大家都会 ☺]）现在大多数布局按钮是可用的，点击 Lay Out Horizontally 水平布局工具栏按钮（如果你将鼠标停留在工具栏上，则会有个标志提示你该按钮的名字），这样这两个部件会被一细细的红色线条包围并可以一起移动，不要担心部件没有恰当的尺寸或者不再正确的位置上，因为我们能用 Qt 设计器正确的设置其大小和位置。



设置当前剪辑变迁和当前行编辑框

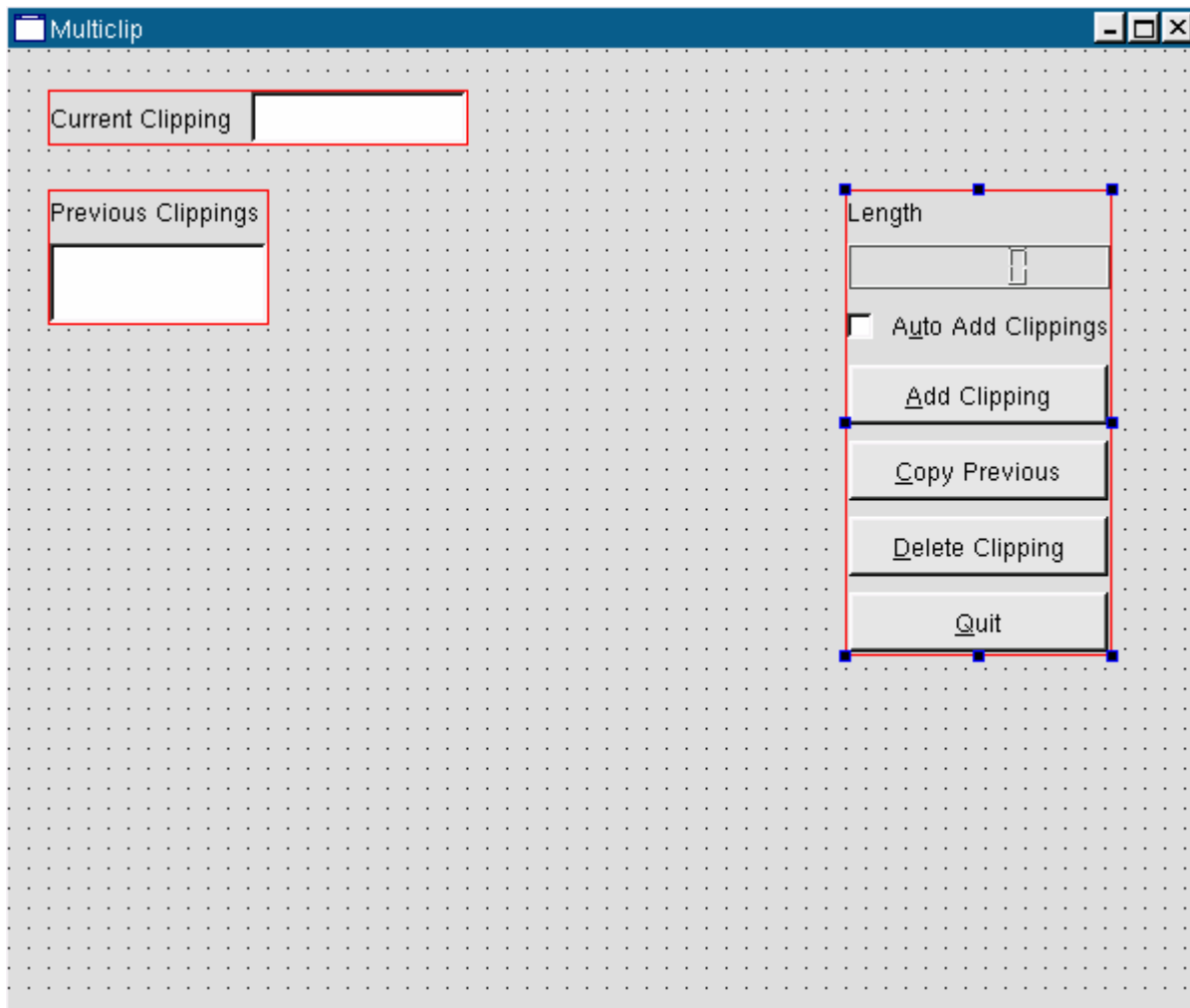
2. 点击前一个剪辑标签并按住 Shift 键，点击 ClippingListBox，点击 Lay Out Vertically 工具栏按钮。
3. 我们要使剩下的组件集合以垂直方式布局，不用按住 Shift 键一个一个点击部件，我们可以从窗体上部开始点击鼠标，拖动鼠标覆盖长度标签，LCD Number、检测框和所有的按钮，使得这些部件位于鼠标滑过的范围之内，释放鼠标，如果你没有做任何按住 Shift 并点击鼠标的操作，所有剩下的部件都会被选中，现在点击 Lay Out Vertically 工具栏按钮。



选中一组部件

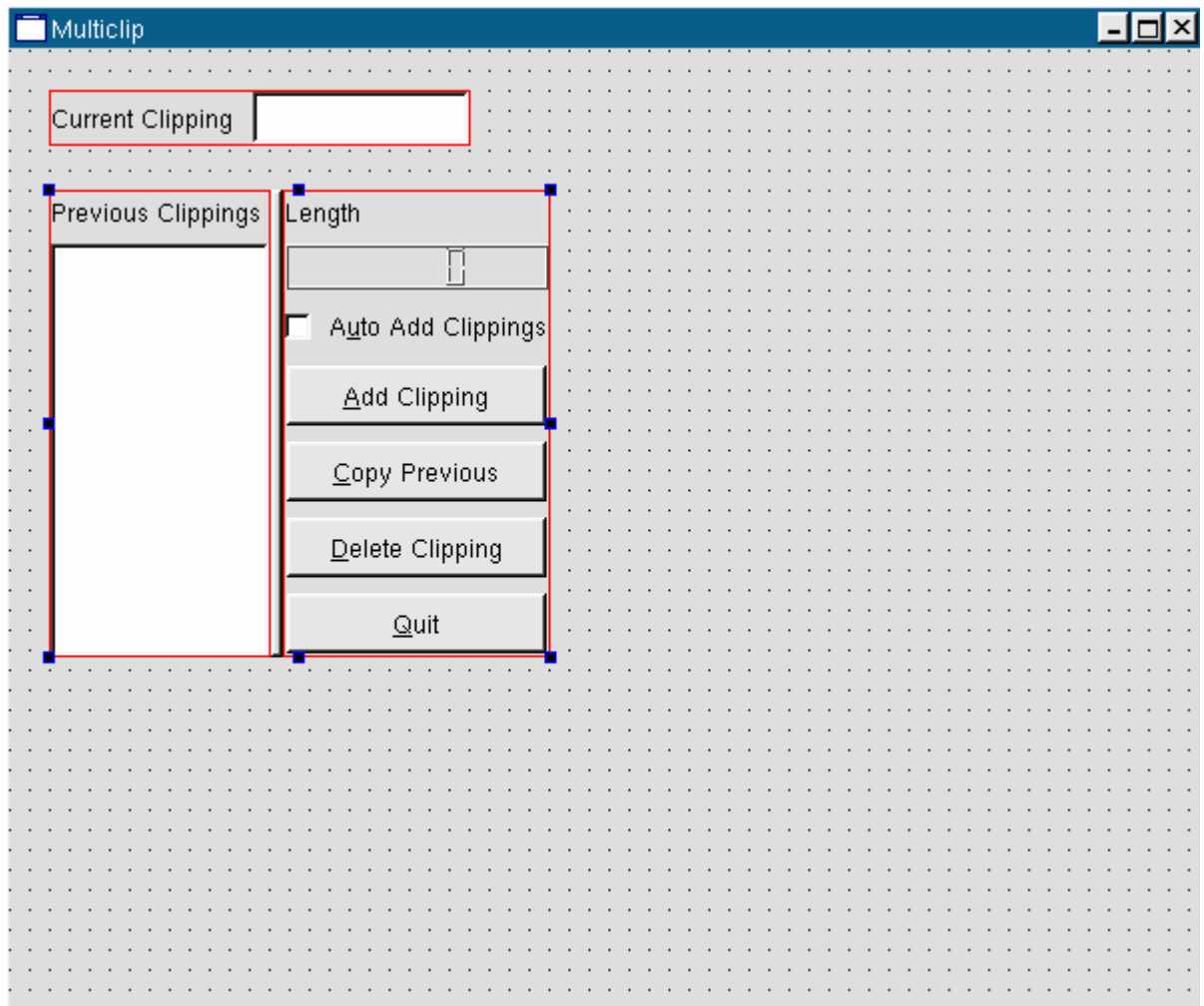
现在我们有三组必须相互联系的部件需要放置，并且各自和窗体本身相关。





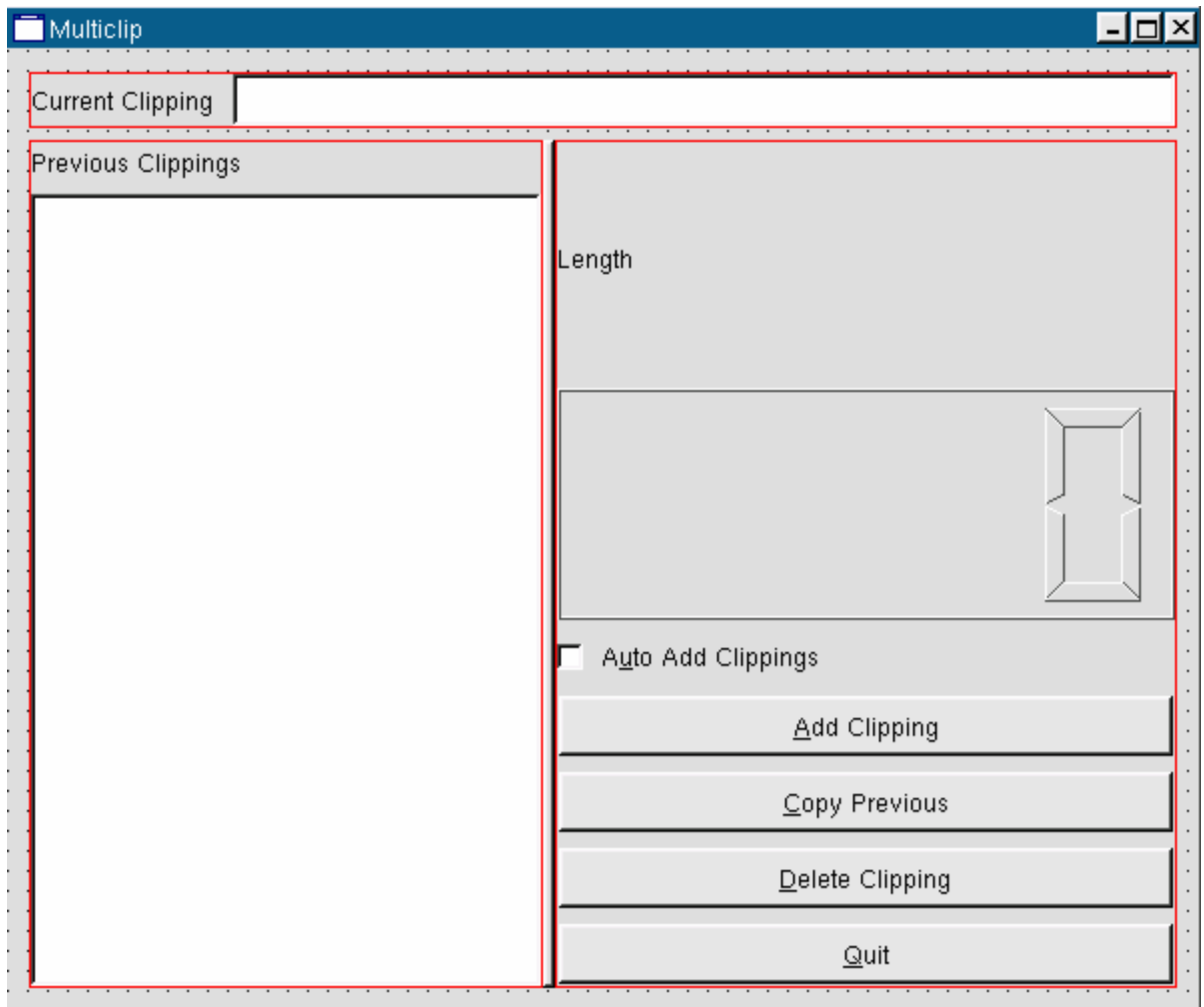
### 布局部件分组

1. 按住 Shift 按钮，点击鼠标 (Shift+Click 组合键) 用于选中单独的部件，为了选中一组部件我们必须点击窗体取消任何选定的部件，然后按住 Ctrl 键，点击该组合知道选中所有你想选中并布局的部件然后释放鼠标，按钮和其他部件都已放置好并选中了，按住 Ctrl 键，点击列表框并拖动鼠标直至选中按钮，然后释放，这样两个组件集合将被选中，点击 Lay Out Horizontally (in Splitter) 工具栏按钮。(我想这部分我翻译得有点晦涩，但是大体意思我想每个热爱编程的人都能明白的，不是么？☺ 那就试试阿)



### 部件设置分组

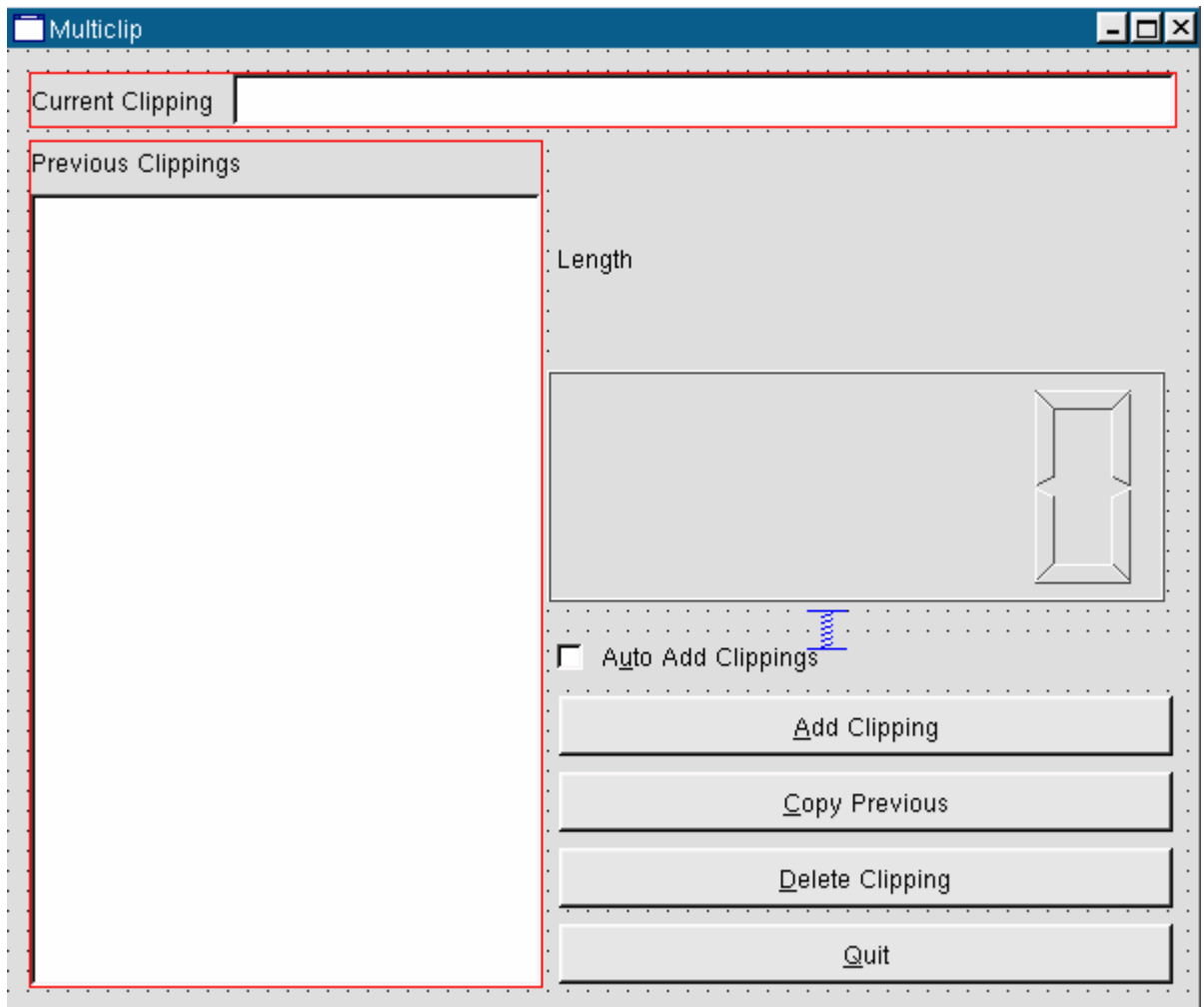
2. 现在我们有两组了部件组合了，最上面的那个有一个当前剪辑标签和行编辑组件，以及我们刚刚创建的又一个列表框和一组按钮和其他部件。现在我们要放置这些部件使得和窗体相关，点击窗体并点击 Lay Out Vertically 工具栏按钮，这些部件将被重新设置大小并充满整个窗体。



### 设置部件与窗体相关

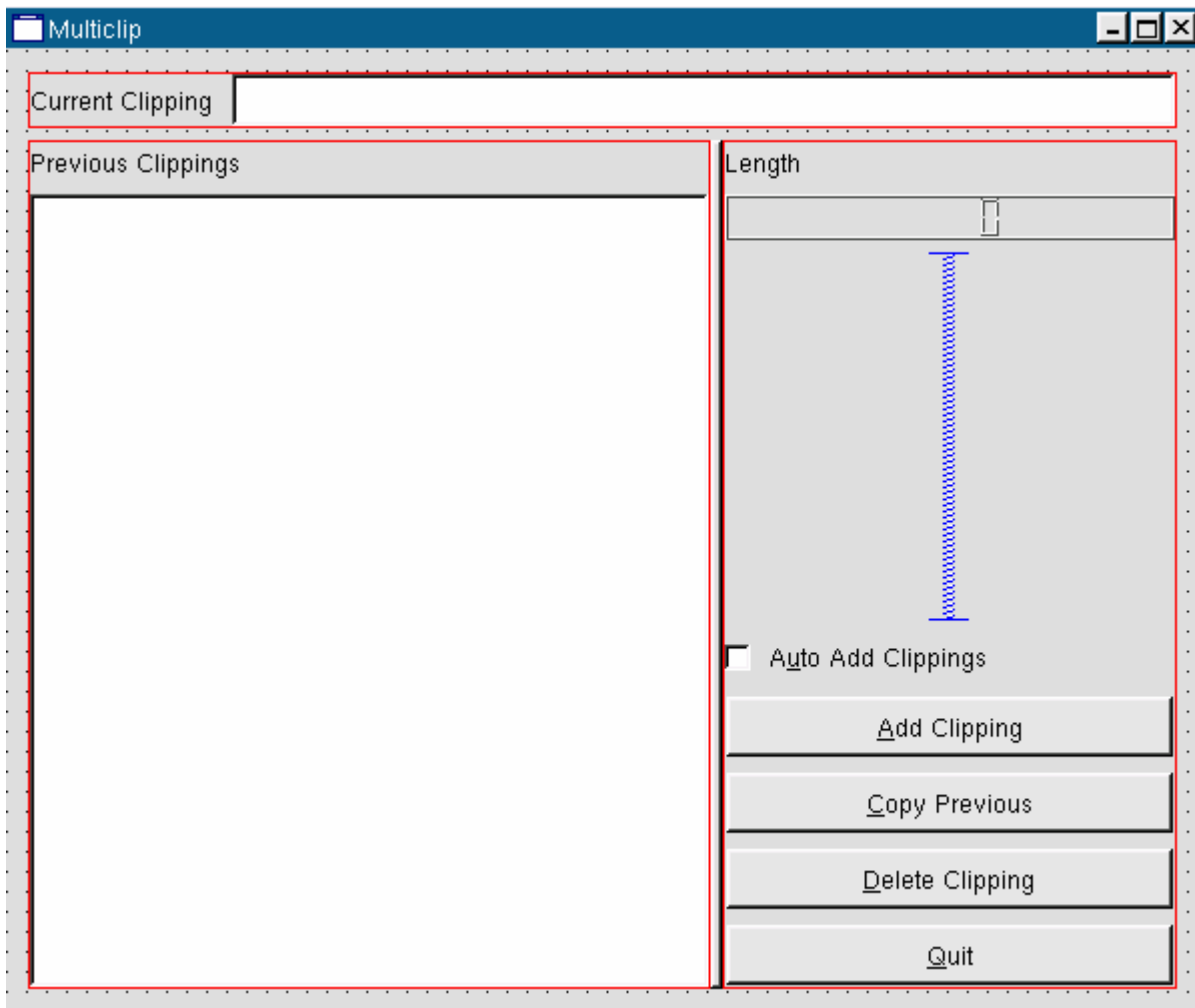
不幸的是，长度标签和 LCD Number（数字标签）占用了太多的空间，于是我们将不得不修订该布局，如果有经验你会发现你不必经常重新布局，我们可以插入空间器消耗掉额外的空间。

1. 首先我们选中需要填充的空间，点击 LCD Number(数字标签)选中它，点击 Break Layout 工具栏按钮，向上移动这个数字标签使其小一点，没必要十分精确，我们将在其下方构建一些空间。
2. 现在我们添加空间器，点击 Spacer 工具栏按钮，于是在数字标签和检测框之间你所创建的空间处点击该窗体，一个有两个选项的菜单弹出，水平和垂直，点击垂直，我们选择垂直是因为我们要空间器在垂直方向消耗多余的空间。



增加一个垂直的空间器

3. 我们需要重新组合这些按钮和其他部件为一个垂直的组件集合。拖动鼠标从窗体的底部，是的鼠标拖动范围可以达到这些按钮、检测框、空间器、数字标签和长度标签，然后释放鼠标，如果你忘记了选中其他组件，点击窗体并在此拖动鼠标直至正确选中所要选中的组件，点击 Lay Out Vertically 工具栏按钮。
4. 现在我们有如前面我们所有的那样的组合，只是添加了一些空间器。通过点击窗体选中列表框和这些按钮，拖动鼠标制止选中这两个部件组合，点击 Lay Out Horizontally(In SPlitter)用一个分裂器重新组合它们。
5. 最后一步是对窗体本身布局设置，点击该窗体并点击 Lay Out Vertically 工具栏按钮，窗体会被正确放置。



设置窗体

在我们得到的布局中有两个小的不足之处，首先是列表框和按钮占用了相等的宽度，尽管我们宁愿使列表框占用四分之三的宽度；其次是长度标签、检测框和按钮在右边延伸到了分裂器，如果在这里有一个小的空间器使它们与分裂器分开，看上去会更有魅力。

延伸列表框到分裂器的一半宽度处，这需要我们添加一个 `Init()` 函数，代码如下：

```
void MulticlipForm::init()
{
    QList< int > sizes;
    sizes << 250 << 40;
    Splitter->setSizes( sizes );
}
```

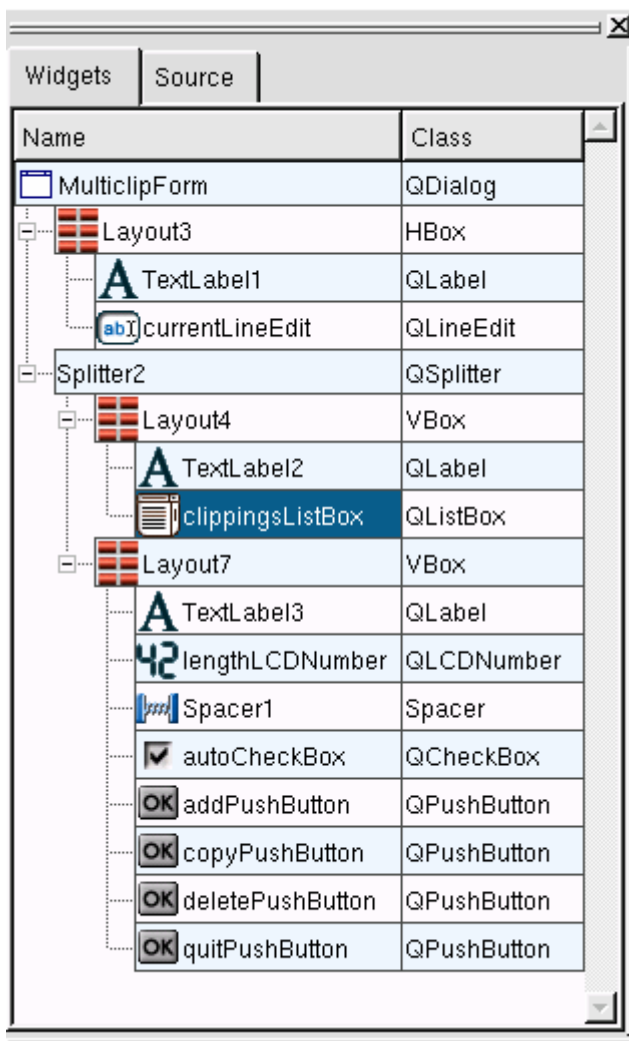
我们不想现在就添加这段代码，因为我们在本章后面将会讲述应用程序函数的实现。

我们通过改变布局组合的边缘，在分裂器周围创建一些空间，点击一个布局或者点击布局顶部红色线的一部分，或者在对象浏览器（部件和来源窗口）中点击布局的名字。（参

看对象浏览器部分，认识对象浏览器）。点击包含有列表框的部件，改变其 `LayoutMargin` 属性值为 6，并按下 `Enter` 键，点击包含有按钮和其他部件的布局，改变它的 `LayoutMargin` 属性值为 6，然后按下 `Enter` 键确认修改。

## 对象浏览器

通过点击 `Window|Views|Object Explorer`. 察看对象浏览器（部件和来源）窗口，对象浏览器有两个标签，组件标签显示了对象的层次，源码标签显示了你添加到窗体的代码，在部件标签下点击一个部件的名字，将会选中该部件，并且在属性编辑框显示其属性，在对象浏览器中查看和选择部件很容易的，特别是对那种有许多部件或者布局的窗体很有用。



对象浏览器

在 Qt 设计器的早期版本中，如果你想为一个窗体提供一份代码，你必须做个窗体的子集，并将你的代码放在子集中，这个版本的 Qt 设计器仍然支持子集的方法，但也提供了另外一种方法：直接将你的代码放在窗体中。在 Qt 设计器中编写代码方法和子集方法不尽相同，例如：你不能直接访问窗体的构造函数和析构函数，如果你的代码需要在构造函数中实现，你需要创建一个名为 `void Init()` 的槽函数；如果它存在了，它将在构造函数中调用，同样，如果你希望在析构函数执行前执行一段代码，你需要添加另外一个槽函数 `v`



`oid destroy()`;你也可以添加增加你自己的类变量，这些变量将被放置在生成的构造函数代码中，你也可以添加前部声明和包含任何你需要的文件，为了增加一个变量或者声明，在源码标签下点击适当的子项，例如：类变量，于是点击新增加按钮并键入你的文本，例如 `QString filename`，如果有一个或多个项存在，鼠标右键点击该项时，将会弹出一个有三个菜单子项的菜单，分别为 New、Edit 和 Delete，要编辑代码只需点击函数的名字激活代码编辑器，代码编辑和创建在后面详述。

如果你是对窗体进行子集，你要创建一个你自己的 .cpp 文件，该文件包含了你自己的构造函数，析构函数，函数，槽，声明和变量，这些都是你的需求指定的。（更多信息参看子集 (Subclassing) 信息）

在例子中我们使用了 Qt 设计器布局工具设置我们的部件，在接下来的章节中的例子中我们将继续适用布局工具，如果你想使用绝对位置，也就是放置窗体和窗体大小以精确的像素尺寸定义，你能很容易办到，要放置一个部件，点击该部件并拖动到预期的位置，要改变其大小，点击该部件，拖拽大小调节框（有蓝色的方框显示）到合适的位置，要使得部件在窗口大小改变时停止部件尺寸改变，需要改变 `hSizeType` 和 `vSizeType`（这些包含在尺寸策略属性中）这两个属性值，为固定值。

## 预览

虽然 Qt 设计器能给出窗体的正确视图，但通常我们想看到在程序运行时窗体的样子，能检测出窗体的一些外貌这也是有用的，例如当窗体尺寸变化是如何比例缩放的，或者分裂器实际上是如何工作的，如果我们在创建一个多平台的应用程序，在不同的环境下预览窗体的样子这也是很有用的。

要想看到预览的样子，点击 `Preview Form` 或者按住 `Ctrl+T` 键，要离开预览模式已当前环境下的标准方式关闭窗口，为了查看应用程序在其他平台下的预览情形，可以点击 `Preview` 菜单并点击其中的菜单子项。

预览多行编辑窗体并试试分裂器和改变窗体的大小，很可能的是如果你将分裂器移动到右边减小按钮的尺寸会使得窗体更好看，这个分裂器看上去是个不错的主意，但实际上，我们需要这些按钮和其他部件在左边占用一个固定的空间，Qt 设计起使得改变这些布局非常容易，所以这样下去不会错的。（我是这么理解的）



点击分裂器然后点击 Break Layout 工具栏按钮，这个分裂器将会被移除，现在在窗体底部点击窗体本身，并拖动鼠标直至选中列表框和一些按钮，然后释放鼠标，这样列表框部件组合按钮部件组被选中了，点击 Lay Out Horizontally 工具栏按钮，点击窗体本身然后点击 Lay Out Vertically 工具栏按钮，这样窗体就按我们的要求布置好了，预览窗体（按下 Ctrl+T 键）并试着重新布置其大小。

既然这些布局工作是可视化的且最好的学习途径是实践，那么当你进一步实验时你会发现该工具很有用。点击 Break Layout 工具栏按钮移除一个布局，选择相关的部件和部件组并点击布局按钮实现一个布局，任何时候你都可以预览窗体的样式并撤销你所做的更改。

让我们做一个实验，来看看网格布局是如何工作的，点击列表框，并按住 Ctrl+B（破坏布局），点击一个按钮并点击 Ctrl+B 按钮，在底部点击窗体并拖动鼠标直至所有的部件被选中，（但是不要选中当前编辑标签和当前线编辑）；然后释放鼠标，点击 Ctrl+G（以网格方式布局）点击窗体，然后按下 Ctrl+L（以垂直方式布局）回到我们最初的设计——但是这次使用的是网格布局。

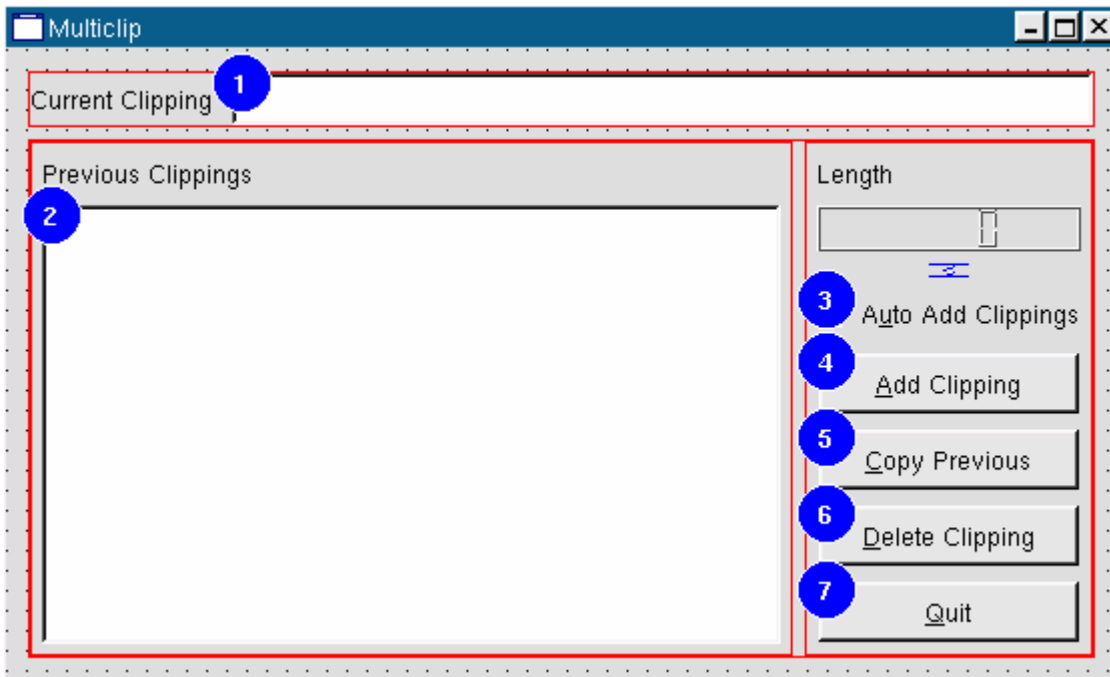
### 改变标签的次序

键盘用户按住 Tab 键然后在部件上一个一个的移动焦点，则焦点移动的顺序就是 Tab 顺序，预览多行编辑程序(按住 Ctrl+T 键)并试着对部件标签，这个标签的顺序可能不是我们所希望的，所以我们将标签模式下改变该标签次序为我们所要的顺序。

当你点击 Tab Order 工具栏按钮是，一个被蓝色圆圈包围的数字将会出现在每一个部件上，这些都能接受键盘焦点，（如果你按住 Ctrl 键同时按你的顺序点击每个部件，可以改变蓝色圆圈中数字的顺序，这一点和 VC 中的控件排序没有太大区别）。这个数字代表了每一个部件的标签顺序，起始值为 1。你可以点击这些部件改变这个标签顺序已满足你所要的新的标签顺序，如果你不小心操作错了，需要重新排序，双击该部件，这个部件的标签号码就会变成 1，于是你可以按需要需要的顺序点击其他部件，当你完成你的标签排序后你可以按下 ESC 键退出标签排序模式。如果你犯了点错，或者更喜欢修改前的标签顺序，你可以退出标签排序模式或者撤销操作（按下 Esc 键并按下 Ctrl+Z 键）来取消你刚才对标签顺序所作的修改。

点击 Tab Order 工具栏按钮，点击部件 当前剪辑线编辑(current clipping Line Edit)——即使该部件的标签号码就是 1，然后点击上次剪辑列表框(ListBox)部件，然后是点击检测框(CheckBox)，按次序从上到下（从增加剪辑到退出按钮）点击每一个按钮，按下 Esc 键完成标签模式下对标签顺序的更改，然后预览窗体并试着修改所有部件标签顺序。

注意到如果所有部件的标签顺序数字是正确的，你可以停止点击部件，只需按下 Esc 键离开标签顺序模式。



设置标签顺序

## 连接信号和槽

Qt 为两个部件之间的通讯提供了信号和槽的机制，当一个特定的事件发生时部件发射信号，我们可以将信号连接到槽，这些槽可以是预先定义的也可以是我们自己创建的槽，在以前的开发包中，这种通讯机制的实现是靠回调函数实现的。（要想知道 Qt 的信号和槽机制的详细解释请参看在线的信号和槽文档）

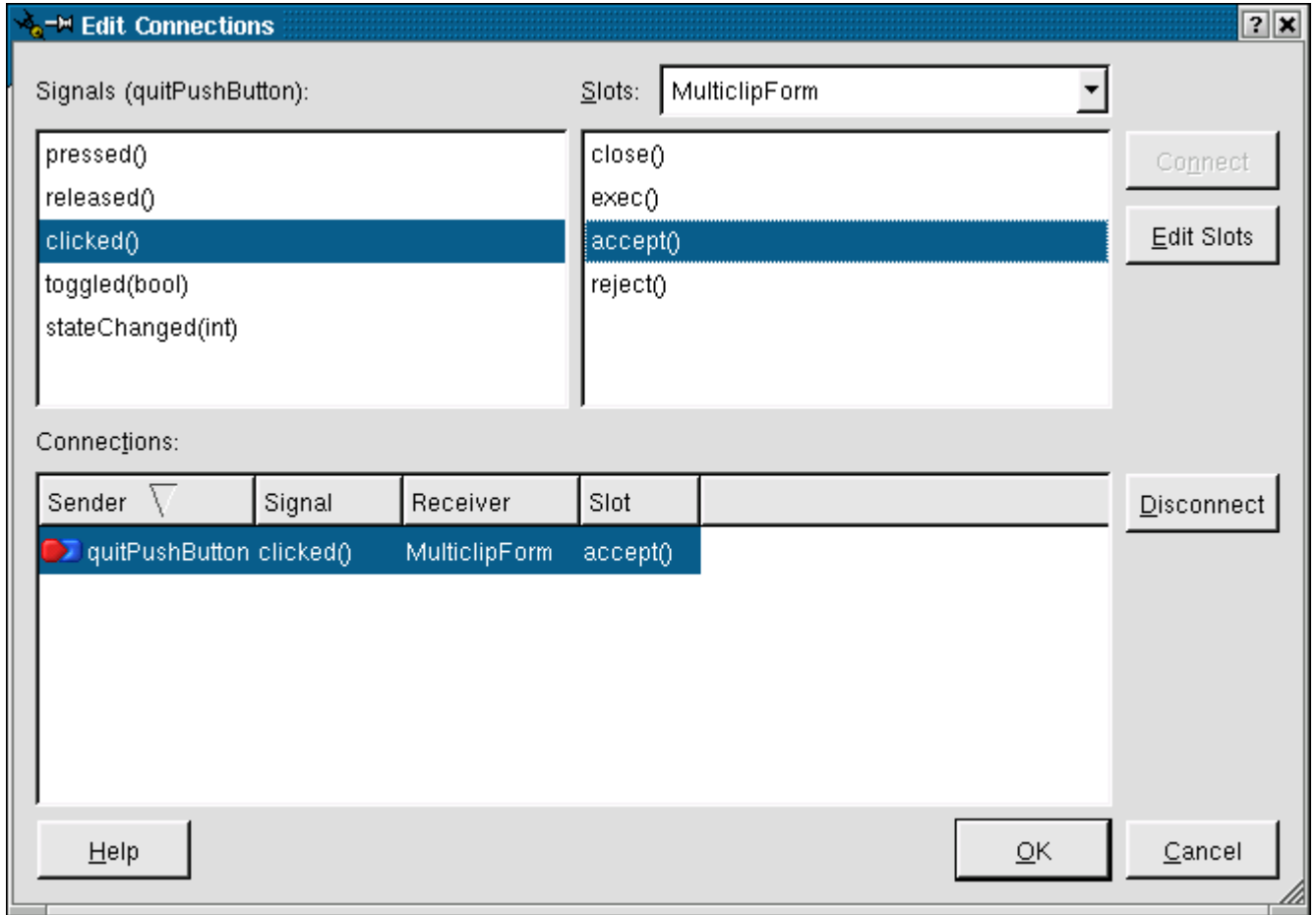
### 连接预先定义的信号和槽

一个应用程序的一些功能可以简单的通过连接预定义的信号和槽来实现。在这个多项剪辑的程序中仅仅只有一个预定义的连接可为我们所用，但是在 richedit 这个应用程序中我们将创建一个带有 Actions, ToolBar 和 Menu 的主窗口，那时我们将用到许多预定义的信号和槽来实现我们所需的大多数功能而不需要些任何代码。

我们将连接退出按钮的 `clicked()` 信号到窗体的 `accept()` 槽，这个 `accept()` 槽通知对话框的调用者（`dialog's caller`）对话框不再需要了，因为我们的对话框是我们程序的主窗口，这样就退出了这个程序，预览该窗体 (Ctrl+T)；点击退出按钮，这个按钮处了显示出来外不任何事情，按下 Esc 键或者关闭窗口退出预览模式。

点击连接信号和槽（Connect Signals/Slots）工具栏按钮，然后点击退出按钮后释放鼠标（点击连接信号和槽工具栏后将鼠标移动到窗体上时，鼠标形状变为十字型，如果点击窗体或者窗体上的某个部件则会出现信号和槽的编辑对话框），（信号和槽的）连接编辑对话框就显示出来了，左上方的列表框列出了我们点击的部件所能发射的信号，右上方的组合框列出了窗体和窗体上的部件，这些都是接收信号的对象，由于我们是在窗体上

而不是在部件上释放鼠标，槽组合框显示的是窗体的名字，“MulticlipForm”，在组合框下方是列表框，显示了窗体上可用的槽或者组合框中是部件时显示的是部件可用的槽。注意到只有这些槽能连接到高亮显示的信号，如果你点击选中了不同的信号，比如是 `toggled()` 信号，则可用的槽列表将会随之改变，点击信号，然后点击在连接列表中将显示的连接，点击 OK 按钮确认。



连接 `clicked()` 信号到 `accept()` 槽

当我们运行这个例子程序是我们将实现多个信号和槽的连接，包括连接我们自定义的槽，信号和槽连接（使用预定义的信号和槽）在预览模式下可以工作，按下 `Ctrl+T` 键预览窗体，点击退出按钮，此时按钮能正确工作了（即关闭对话框退出了程序）

## 创建和连接自定义的槽

在第一版的 Qt 设计器中，你可以创建你自定义的槽的信号并使他们连接起来，但是你不能直接实现你的槽，你不得不予集于该窗体，并在这个子集中对你自定义的槽编码，子集的方法依然有用，在某些情况下仍起作用 (Make sense)，但是现在你可以在 Qt 设计器中直接实现你的槽，在多的对话框和窗体的子集也不再需要了。（Qt 设计器在 `.ui.h` 文件中存贮者槽的实现，具体的细节参看 走近设计器 (*The Designer Approach*) 中的 `.ui.h` 扩展方法）。

多项剪辑应用程序需要四个槽，一个用于每个按钮，因为我们将一个信号连接到预定义的槽使得按下退出按钮时退出程序，所以仅有三个槽需要我们自定义，我们需要增加剪辑按钮实现一个槽，其作用是将当前剪辑添加到列表框中，复制以前的按钮(Copy Previous)需要一个槽实现复制列表框中选中的项到线型编辑框中（也复制到剪贴板上），删除剪辑按钮>Delete Clipping)需要一个槽实现删除党旗那剪辑和列表框中的当前项，我们也需要写一些初始化的代码使得程序启动时，将当前剪贴板上的文本（如果有的话）复制到线型编辑框中，代码可以直接在Qt设计器中编写，声称的已写好的代码片断可以从下面的文件中获取：`qt/tools/designer/examples/multiclip/multiclip.ui.h`

我们需要一个Qt的全局剪贴板对象，在整个代码中有几个地方需要调用其含义是调用 `QApplication::clipboard()` 或者是 `qApp->clipboard()`，比执行这些函数调用更好的方法，我们在窗体本身添加一个指向剪贴板的指针，点击对象浏览器中的源码标签（如果对象浏览器部可见，点击 Windows|Views|Object Explorer [由于Qt设计器没有汉化，再说汉化后似乎失去了其魅力，所以菜单的引用依然是English]）源码标签显示了当前窗体中的函数、类变量、前部的声明和我们需要的一些包含文件的名字。

鼠标右键点击类变量(Class Variable)项(可能在底部，你需要点击滑动条)，然后再弹出的菜单中选择 New 子菜单，(如果存在变量，则弹出来的菜单中会有一个“Delete”删除的选项)键入 `QClipboard * cb` 并按下 Enter 键确认输入，我们将创建一个 `init()` 函数，在这个函数中我们会赋予这个指针一个Qt的全局剪贴板对象的值。我们也需要声明剪贴板对象的头文件，右键点击包含（声明文件）[在Qt设计器中对应的为 Include (in Declaration) 项]，然后再弹出的菜单上选择 New，键入“”，并按下 Enter 键确认输入，因为我们需要一个全局对象，`qApp`，我们就必须包含另一个声明文件，鼠标右键点击 包含（实现）项[在Qt设计器中对应为 Include (in Implementation)]，然后点击 New 菜单项，键入“”并按下 Enter 键确认输入，这个变量和声明文件将被Qt设计器的.ui文件生成后包含（从.ui文件生成的源文件.h和.cpp文件会包含这个变量和声明文件）。

We will invoke *Qt Designer's* code editor and write the code. 现在我们将激活Qt设计器的代码编辑器并编写代码。

首先我们创建一个 `init()` 函数，激活代码编辑器的一种途径是点击 Source 标签，然后点击你想编写代码的函数的名字，如果没有你需要的函数或者你想创建一个在 Source 标签中你可用的新函数，鼠标右键点击 Source 标签的槽(Slots)列表中的“protected”子项，然后点击 New 子菜单，出现一个“Edit Slots”编辑槽的对话框，改变槽的名字为 `Init()`，然后点击 OK，这样你可以点击编辑窗口中出现的函数名字并键入你的代码。

注意到并不是强迫你使用Qt设计器的代码编辑器，在Qt设计器中你可以尽管的增加、删除或者重命名你的槽，你也可以用一个外部的编辑器编写的实现代码，Qt设计器会保存你所写的代码。下面是你需要实现的 `init()` 函数：

```
void MulticlipForm::init()
{
```

```

lengthLCDNumber->setBackgroundColor( darkBlue );
currentLineEdit->setFocus();

cb = qApp->clipboard();
connect( cb, SIGNAL( dataChanged() ), SLOT( dataChanged() ) );
if ( cb->supportsSelection() )
    connect( cb, SIGNAL( selectionChanged() ), SLOT( selectionChanged()
) );

dataChanged();
}

```

函数体中开始的两行改变了数字指示器的背景颜色并使窗体的启动焦点在线型编辑框中，我们使用了一个指向 Qt 全局剪贴板的指针，并保存在我们定义的类变量 cb 中，我们连接剪贴板的 dataChanged() 信号到一个叫做 dataChanged() 的槽，这个操我们马上就会创建，如果剪贴板支持选择（例如在 X Windows 系统中），我们也要连接剪贴板的 selectionChanged() 信号到一个我们将创建的具有相同名字的槽，最后我们将调用我们的 dataChanged() 槽函数，当程序启动时，将当前剪贴板中的文本内容（如果有的话）粘贴但线型编辑框中

既然我们提到了 dataChanged() 和 selectionChanged() 槽，下面我们将对他们编码，从 dataChanged() 开始：

```

void MulticlipForm::dataChanged()
{
    QString text;
    text = cb->text();
    clippingChanged( text );
    if ( autoCheckBox->isChecked() )
        addClipping();
}

```

我们复制了剪贴板的文本并用我们获得文本调用我们自己定义的 clippingChanged() 槽，如果用户选中了自动增加剪辑检测框，我们将调用 addClipping() 槽增加该剪辑到列表框中。

下面的代码只是在 X Windows 系统中可用，微软的 Windows 用户也能包含下面的代码确保该应用程序可以在多平台下运行。



```

void MulticlipForm::selectionChanged()
{
    cb->setSelectionMode( TRUE );
    dataChanged();
    cb->setSelectionMode( FALSE );
}

```

上面的代码中我们首先告诉剪贴板使用选择模式，我们调用 `dataChanged()` 槽获得任何选中的文本，然后设置剪贴板为默认模式。

In the another custom slot, `clippingChanged()`. 在另一个自定义的槽中, `clippingChanged()`:

```

void MulticlipForm::clippingChanged( const QString & clipping )
{
    currentLineEdit->setText( clipping );
    lengthLCDNumber->display( (int)clipping.length() );
}

```

我们用任何传递给 `clippingChanged()` 槽的文本设置当前的线型编辑框, 并用这个新文本的长度更新数字指示器。

我们将要编码实现的下一个槽实现增加剪辑功能, 当用户点击增加剪辑按钮时, 这个槽被我们的代码内部调用 (看上面的 `dataChanged()` 槽), 即使我们让 Qt 设计器能够通过编辑窗口键入代码来创建一个槽与信号的连接, 我们也让 Qt 设计器为我们创建一个结构, (为我们生成程序框架), 点击 Edit|Slots 激活编辑槽 (Edit Slots) 对话框, 点击 New Slot 按钮并把槽的缺省名字 “`new_slot()`” 改为 “`addClipping()`”, 这里不必改变访问标着和返回类型, 现在我们已经创建了我们的槽, 我们能在代码编辑器中出现的地方创建我们的槽并实现之。

增加剪辑按钮用于从当前剪辑线型编辑框中复制文本到列表框中, 同时我们更新了文本长度数字指示器。

```

void MulticlipForm::addClipping()
{
    QString text = currentLineEdit->text();
    if ( ! text.isEmpty() ) {
        lengthLCDNumber->display( (int)text.length() );
        int i = 0;
        for ( ; i < (int)clippingsListBox->count(); i++ ) {
            if ( clippingsListBox->text( i ) == text ) {
                i = -1; // Do not add duplicates
            }
        }
    }
}

```

```

        break;
    }
}
if ( i != -1 )
    clippingsListBox->insertItem( text, 0 );
}
}

```

如果有新文本需要显示，同时改变数字指示器的值为当前文本的长度，然后我们在列表框中逐个比较文本项，看当前输入的文本是否存在列表框中，如果列表框中不存在该剪辑文本项，就在列表中插入该剪辑文本。

为了使增加剪辑按钮工作，我们需要连接这个按钮的 `addClipping()` 槽，点击 `Connect Signal/Slots` 工具栏按钮，然后点击增加剪辑按钮，（将变成十字型的鼠标拖动到窗体上点击该按钮然后释放），（确认你拖向窗体而不是其他部件—拖动过程中窗体会会有一个细红色边界，如果你操作错误你只需在槽组合框中改变其名字就可以了[这里我按原文翻译，可是有些地方和单词觉得上下文不对，如果你发现了正确的请与我联系]）。连接编辑对话框将会显示，点击 `addClipping()` 槽，点击 OK 按钮确认该连接。

复制前一个文本按钮（`Copy Previous`）用于从列表框中复制选中的文本项到线型编辑框中，同时该文本也放到了剪贴板上，剩下的过程和增加剪辑按钮（`Add Clipping`）是一样的：首先我们创建一个槽，然后我们实现该槽，最后我们将按钮点击的信号和槽连接起来。

## 1. 创建槽：

点击 `Edit|Slots` 菜单子项，激活槽编辑（`Edit Slots`）对话框，点击 `New Slots` 并将缺省的槽函数名字“`new_slot()`”改变为“`copyPrevious()`”，点击 OK 确认。

## 2. 实现槽函数

```

3.         void MulticlipForm::copyPrevious()
4.         {
5.             if ( clippingsListBox->currentItem() != -1 ) {
6.                 cb->setText( clippingsListBox->currentText() );
7.                 if ( cb->supportsSelection() ) {
8.                     cb->setSelectionMode( TRUE );
9.                     cb->setText( clippingsListBox->currentText() );
10.                    cb->setSelectionMode( FALSE );
11.                }
12.            }

```

13.            }

这段复制上次剪辑的代码检测列表框中是否有选项选中，如果有选项被选中则该项会被复制到线型编辑框中，如果我们使用了一个支持选择的系统，我们将不得不重复复制，复制的次数和选择模式有关，我们不要显示的更新剪贴板，当线型编辑框的文本改变时，他发射了一个 `dataChanged()` 的信号，这个信号被我们创建的 `dataChanged()` 槽接收，我们的这个 `dataChanged()` 槽同时更新剪贴板。

#### 14. 连接到槽

点击 `Connect Signal/Slots` 工具栏按钮，点击复制上次剪辑 (`Copy Previous`) 按钮，拖动到窗体然后释放鼠标，（光标变成十字型后，点击某个部件 1，如按钮，拖动鼠标到另一个部件 2 或者窗体，则在出现的信号槽连接编辑对话框中，信号的发射者是部件 1，接受信号的槽是部件 2 或者窗体的[试试就知道了 ☺]），在弹出的连接编辑对话框中，点击 `clicked()` 信号和 `copyPrevious()` 槽，点击 OK 确认连接编辑。

用同样的方法我们对删除剪辑 (`Delete Clipping`) 按钮进行信号和槽的连接编辑。

1. 点击 `Edit|Slots` 菜单激活槽编辑对话框，点击 `New Slot` 按钮并替换缺省的槽名字为 “`deleteClipping()`”，点击 OK 按钮。

2. 删除按钮必须删除列表框中的当前项并清除线型编辑框中的文本。

```
3.         void MulticlipForm::deleteClipping()
4.         {
5.             clippingChanged( "" );
6.             clippingsListBox->removeItem( clippingsListBox->currentItem
7.             ( ) );
8.         }
```

我们以一个空字符串调用我们自己创建的 `clippingChanged()` 槽，使用列表框的 `removeItem()` 函数删除当前选项。

8. 连接删除剪辑 (`Delete Clipping`) 按钮的 `clicked()` 信号到我们创建的 `deleteClipping()` 槽，（按下 F3 键——这个点击 `Connect Signals/Slots` 工具栏按钮效果一样，点击删除剪辑按钮并拖动鼠标到窗体上，释放鼠标，出现连接编辑对话框，点击 `clicked()` 信号和 `deleteClipping()` 槽，点击 OK() 确认连接编辑）

## 编译和生成应用程序

至此在 Qt 设计器中，我们写了一个 Qt 整个应用程序大约 99%的代码，为了使应用程序编译和运行我们必须创建一个 main.cpp 文件，在这个文件中我们将显示调用我们的窗体。

创建一个新的源文件的简单方法是点击 File|New 激活 “New File” 新文件对话框，适当的点击 “C++ Source” 或者 “C++ Header” 图标，然后点击 “OK” 按钮确认，一个新的空源文件显示出来，点击 File|Save 激活另存为对话框 (Save As )，键入 “main.cpp” 然后点击 Save 按钮保存该文件。

在 main.cpp 这个 C++编辑窗口，键入以下代码。

```
#include <qapplication.h>
#include "multiclip.h"

int main( int argc, char *argv[] )
{
    QApplication app( argc, argv );

    MulticlipForm clippingForm;
    app.setMainWidget( &clippingForm );
    clippingForm.show();

    return app.exec();
}
```

这个程序创建了一个 QApplication 对象和我们的多项剪辑窗体的实例，并将该窗体设置为主部件并显示该窗体，app.exec() 开始调用程序事件循环。

现在启动控制台命令（命令行格式），并将当前目录切换到 multiclip 程序下，运行 qmake 命令，一个与你的系统兼容的 Makefile 文件将会生成。

```
qmake -o Makefile multiclip.pro
```

现在你可以通过运行 `make` 或者 `nmake` 命令生成该应用程序。试着编译和运行 `Multiplic`，你会取得许多的进步，使用布局工具和编写代码的试验可以帮助你学到关于 Qt 和 Qt 设计器的更多的东西。

这一章向你介绍了使用 Qt 设计器创建一个跨平台的应用程序，我们创建了一个窗体并且用部件对其修饰，这些部件整齐优美的放置在窗体上并可以随着窗体缩放，我们已经使用了 Qt 的信号和槽的机制实现程序的功能，并生成了 `Makefile` 文件，这些增加部件到窗体并用布局工具对其放置，以及创建、编写和连接槽的技术，当你再次用 Qt 设计器创建程序是会用到，下面的章节会给出进一步的例子并介绍使用 Qt 设计器的更多技术。