第三部分　Linux GUI生成器Glade

第17章　Glade：GUI生成器

17.1　安装Glade

17.1.1　Glade简介

毫无疑问，使用Gtk/Gnome构件编程的概念并不难。然而，使用这些函数存在一些困难：首先是创建程序界面的代码是非常繁琐的，特别是在使用不同的布局构件组装界面元素，创建菜单、工具条等时，不能在编写代码时直接看到显示效果；其次是对代码量较大的程序，可能要将代码放在不同的C语言文件中，为它们配置编译选项、写Makefile文件也是一项巨大的工程。特别是对于大型项目的开发，这两点尤为突出。应该有一种工具，它可以将我们从这些工作中解放出来，并让我们能够专注于任务的核心。

有没有像Microsoft Windows平台下的Visual Basic、Delphi、C++ Builder那样快速的开发工具呢？到目前为止，Linux下还没有功能完备的可视化、快速的编程工具，但是已经有了一个非常出色的界面生成工具：Glade。它可以用可视化的方法绘制应用程序界面，设置窗口、构件的外观、设置构件信号的回调函数，然后生成C语言代码。Glade目前的版本号是0.5，也就是说还是发布前版本，不太稳定、成熟。但它的性能已经非常出色。一些专家已经将Glade列为今后最有前途的Linux快速开发工具。将Glade、gcc编译器以及gdb结合起来使用，Linux下的编程将是非常直观的、高效的。

最近，已有一批专家正在致力于开发一个类似于Visual Basic的可视化编程工具gBasic，目的是开发一套可与VB媲美的Basic编译器。Inprise公司（以前的Borland）也可能将要发布Delphi for Linux。毫无疑问，Linux今后将成为大部分程序员的重要目标平台，Linux软件开发将会更加快捷，更加方便。

17.1.2　安装Glade

Glade是由Damon Chaplin创建维护的Gtk用户界面生成器。它是基于GPL许可的自由软件。只要遵从GPL协议，就可以自由地获得其源代码，使用它开发自由软件以及商用的非自由软件。你还可以对它进行修改然后重新发布。所以，如果你对Glade有什么意见和建议，或者从Glade的源代码中发现了bug，最好能够和Glade的维护者联系，以便于对Glade的改进。如果你自己有针对Glade的修改意见，也可以将修改方案、代码提交给维护者。

在http://glade.pn.org网站上能够找到Glade最新源代码。一般下载的文件名是glade-0.5.0.tar.gz。其中的数字代表其版本号。本书出版时，可能会因为版本升级而略有不同。通常将其下载后放在/usr/src目录下。

因为安装文件是打包的压缩文件，所以需要先解压缩，然后再编译、安装。安装方法如下：

1) 在shell提示符下输入以下命令，进入文件所在目录：

```
cd /usr/src
```

2) 在shell提示符下输入以下命令，将其解压缩，生成一个归档文件：

```
gunzip glade-0.5.0.tar.gz
```

3) 将tar归档文件展开为目录结构：

```
tar xvf glade-0.5.0.tar
```

这样会将Glade的源代码解压缩到/usr/src/glade-0.5.0目录下。

4) 进入源代码所在目录，运行configure使用程序，配置编译选项，生成Makefile文件：

```
cd glade-0.5.0
./configure
```

5) 现在，可以输入make命令编译glade：

```
make
```

依赖于计算机性能，可能需要几分钟到一刻钟时间。编译完成后 ,将在glade目录下生成Glade可执行文件。

6) 到此为止，已经成功安装了Glade，可以运行了：

```
cd glade
./glade
```

17.1.3 在Gnome主菜单下为Glade创建菜单项

有时我们可能不想每次都从 xterm下启动Glade，而是希望从Gnome的主菜单下启动该程序。现在，我们为Glade在Gnome的主菜单上创建一个快捷方式。

1) 选“ 主菜单/Gnome设置/菜单编辑器 ”，弹出Gnome菜单编辑器，如图17-1所示。

2) 点击“ 系统菜单 ”，然后选“ 新子菜单 ”按钮，在 Name后输入Glade，在Comment后输入说明性文字“ Gtk GUI Builder ”，最后点击“ 保存 ”按钮。这样，就在 Gnome主菜单上创建了一个文件夹。

3) 点击工具条上的“ 新条目 ”，在Name后输入Glade，并在Comment后输入说明性文字GUI Builder，在Command后输入Glade可执行文件的路径：

```
/usr/src/glade-0.5.0/glade/glade
```

然后点击“ 保存 ”按钮。经过以上步骤 ,在Gnome的主菜单上就创建了一个快捷方式。

图17-1 Gnome菜单编辑器

17.1.4 在Gnome面板上创建快捷按钮

上面我们为Glade创建了一个快捷方式，就像 Windows 95的“ 开始/程序 ”下的快捷方式作用一样。我们还可以将刚才创建的快捷方式放在 Gnome的面板上。

点击“ Gnome主菜单/Glade ”，打开Glade快捷方式文件夹，按住鼠标左键，将里面的Glade拖到Gnome面板上，然后放下。这样就在 Gnome的面板上创建了一个快捷键。点击它可以直接启动Glade。

注意，本节所展示的菜单编辑器图像是在 TurboLinux 4.0中文版下拷屏得到的。不同的Linux发布版本可能会略有不同，请读者留意。

17.2 用Glade生成图形用户接口

17.2.1 Glade的界面简介

如果已经在Gnome中用前述方法为 Glade创建了快捷方式，选择“ Gnome主菜单/Glade ”下的Glade，即可启动Glade应用程序。Glade的主窗口如图17-2所示：

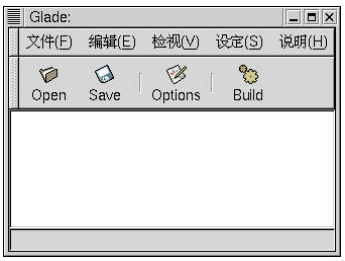

图17-2 Glade主窗口

除主窗口之外， Glade还有构件箱窗口（Palette）、属性编辑器窗口、构件树窗口、剪贴板窗口。下面我们将分别介绍这些窗口的作用。

1. 主窗口

主窗口显示应用程序的最顶层对象，如窗口、弹出菜单、对话框等。要编辑这些对象时，只需在主窗口的列表中双击该对象，即可打开它。选中对象，按 Delete键，就可以将该对象删除。

主窗口上有文件、编辑、检视、设定、说明，分别说明如下：

(1)“ 文件 ”菜单，有以下菜单项：

New Project：创建新的应用程序。因为目前 Glade还不是一个成熟的版本，所以如果有一个应用程序正在使用，它不会提示保存旧文档。这一点一定要留心。

开启旧档：打开已有的应用程序。在Glade中文件是以glade为后缀保存的，实际上它是一个XML格式的文本文件，描述了界面的构件属性以及其他设置，如构件的信号、回调函数等。

储存文件：保存当前应用程序。一般是以glade为后缀的XML格式的文本文件。

Build Source Code：Glade的主要目的就是生成创建应用程序界面的代码。所生成的源文件的结构我们将在后面介绍。目前可以生成基于 glib和Gtk+/Gnome构件库的C语言代码，今后还将支持C++和Ada语言。

Project Option：设置应用程序项目的选项，如所用语言、源文件结构等。

结束：退出Glade。

(2)“ 编辑 ”菜单，有以下菜单项：

剪下：Glade带一个很有意思的剪贴板，可以将窗口上的构件剪切到剪贴板中。剪贴板只在Glade运行器件时起作用。 Glade退出之后，剪贴板中的内容立即消失。剪贴板中可以同时容纳多个构件。最新加到剪贴板的构件是当前构件。

复制：将窗口上的构件复制到剪贴板中。

粘贴：从剪贴板中将构件复制到窗口上。你可以将剪贴板打开，从多个构件中指定一个当前构件，并将其粘贴到窗口上。

清除：删除窗口上的构件。

(3)“检视”菜单，用于打开Glade的其他窗口。它有以下几个菜单项：

Show Palette：打开“构件箱”窗口。

Show Property Editor：打开“属性编辑器”窗口。

Show Widget Tree：打开“构件树”窗口。

Show Clipboard：打开剪贴板窗口。

(4)“设定”菜单，用于设置Glade的选项。它有以下几个菜单：

Show Grid：在应用程序窗口上显示网格，以便于构件对齐。

Snap to Grid：让构件与网格对齐。

Show Widget Tooltips：显示构件的工具提示。

Set Grid Option：设置网格选项。

Set Snap Option：设置“对齐到网格”选项。

(5)“说明”菜单，用于显示版权信息，只有一个菜单项：

关于：显示一个About窗口，显示关于Glade的版权信息。

2. 构件箱窗口（Palette）

选“检视”菜单的Show Palette，可以打开“构件箱”窗口，如图17-3所示。

图17-3 Glade的构件箱

构件箱中容纳了绝大多数Gtk+/Gnome构件。用鼠标点击构件箱上的构件，再在窗口上点击，可以将构件添加到窗口上。要注意的是，在窗口上添加构件也要遵从Gtk的构件组装原则；另外，构件箱中有一些构件还是实验性的，如GtkSpellChecker等，最好不要使用。

构件箱将Gtk+/Gnome构件分为三类：Gtk+基本构件、Gtk+附加构件以及Gnome构件。在构件箱上点击任何一个标签页，都将显示上面三类构件之一。选择某个构件后，点击Selector前的箭头，可以取消前面的选择。

3. 属性编辑器窗口

在主窗口的“检视”菜单中选择Show Property Editor，可以打开“属性编辑器”窗口。属性编辑器类似于VB中的“属性窗口”，用于设置构件的属性。但是，实际上最后生成的源代码是一些设置属性的函数调用，比如窗口的缺省尺寸、窗口的标题等。这里可以设置构件的名称、构件在窗口上的组装位置、构件的加速键，还可以为构件的信号设置回调函数。属性编辑器的选项根据不同的控件会有所不同。

4. 构件树窗口

在主窗口的“检视”菜单中选择Show Widget Tree菜单项，可以打开“构件树”窗口。因为Gtk+/Gnome构件在窗口上的定位是用容器的方法实现的，因而应用程序的每个窗口都是一个按层次组织的构件树，其中窗口在构件树的根部。在使用属性编辑器窗口设置构件的属性时，如果因为其他构件挡住或者由于其他原因不能直接选中控件时，可以从“构件树”窗口中展开构件树，在要设置的构件上点击右键，然后在弹出菜单上选 Select，即可选中该控

件。

5. “剪贴板”窗口

在主窗口的“检视”菜单上选择 Show Clipboard，可以打开“剪贴板”窗口。如果有多个构件剪切或复制到剪贴板，可以在此处选择需要的构件，粘贴到窗口上。

图17-4 剪贴板窗口

图17-4是一个剪贴板窗口示意图，它有三个构件： combo1、label1、button1，其中的combo1是当前构件。当选择“编辑” 菜单的“粘贴”菜单项时，会将combo1贴到窗口上。你可以在这里将其他构件设为当前构件。

17.2.2 用Glade创建应用程序界面

用Glade能以非常直观的方法生成应用程序界面，类似于 Visual Basic和Delphi。不同点在于Visual Basic和Delphi是一个集成开发环境，不仅可以创建界面，还可以创建完整的应用程序，以及调试、编译等。而 Glade仅仅是一个GUI（图形用户接口）生成器，它只能用于生成创建界面的代码，实现应用程序的功能、编译、调试等工作需要使用其他工具。另外，Gtk+/Gnome的构件与Visual Basic/Delphi的控件的定位方法有很大的不同，前者使用一种组装技术实现，而后者使用直接定位在窗口上，类似于 GtkFixed构件。

选“文件/New Project”，可以新建一个应用程序。新建程序里面没有任何对象。为程序设计界面要做以下几个工作：创建新窗口，在窗口中将构件定位，为设置构件的属性，为构件的信号设置回调函数。

1. 创建新窗口

在构件箱窗口（Palette）上点击Gtk+ Basic中的GtkWindow构件，将出现一个，标题为window1的窗口，这就是应用程序的第一个窗口。选择“检视”（或View）菜单中的 Show Property Editor，可以显示属性编辑器。在其中可以设置窗口的属性，如标题、 Border Width、尺寸等，还可以为程序添加新的窗体。另外，使用 GnomeApp构件作为应用程序的主窗口也是一个不错的选择。

2. 在窗口中添加构件

在窗口中添加构件涉及构件的定位方法。可以使用 GtkHBox、GtkVBox、GtkTable等来定位构件，还可以用 GtkFixed定位构件。不过，使用 GtkFixed的潜在问题是如果用户调整窗口尺寸，构件的相对位置和尺寸不会随之变化，窗口可能会看起来很怪。一般情况下，不要使用GtkFixed构件。

(1) 用GtkHBox/GtkVBox定位构件

在构件箱上选GtkHBox或者GtkVBox，然后在窗口上点击，会弹出一个对话框，询问组装盒应该分成几部分。根据界面规划，选定一个值，然后确定，这时窗口被分为几部分。现在可以在其中放置构件了。要注意的是，其实将GtkHBox(或GtkVBox)划分为几部分并不重要，因为只要需要，随时可以很方便地改变这个值。添加组装盒之后，可以在属性编辑器中设置组装盒的参数：Homogenous、Expand、Fill、Spacing、Padding等。这几个参数的含义请参看前面关于GtkBox的内容。

下一步，在组装盒中添加构件。在构件箱上找到想要的构件，先用鼠标左键点击该构件，

然后在组装盒上的适当位置再点击一次，就可以将这个构件添加到窗口上。你可以依次向窗口添加其他构件。GtkHBox和GtkVBox的组装方法是完全一样的。将这两者结合起来就可以创建非常复杂的界面。

使用组装盒必须遵守 Gtk+/Gnome构件的组装规则。下面是几个技巧（也适用于GtkTable）：

1) 在窗口的某个区域放置一个 GtkFrame构件，然后在 GtkFrame构件中放三个GtkRadioButton构件。因为 GtkFrame只能容纳一个子构件，需要采取这样的方法：将GtkFrame定位在窗口上，在GtkFrame上放一个GtkHBox（分为三部分）作为子构件，然后将GtkRadioButton构件组装到这个GtkHBox中。

2) 在窗口上放置一个GtkLabel构件，但是想要让它对鼠标点击响应。因为 GtkLabel没有自己的 X窗口，所以需要使用 GtkEventBox构件：将 GtkEventBox在窗口上定位，并将GtkLabel添加在GtkEventBox上，然后再设置GtkEventBox的事件。

3) 在窗口上放一个GtkList或GtkCList（或其他类似的构件），但是需要这个构件在必要的时候能够自动出现滚动条。因为 GtkList或GtkCList自己并没有滚动条，我们需要使用一个GtkScrolledWindow为它提供滚动条。实现方法是：在窗口上添加一个 GtkScrolledWindow构件，并在这个滚动窗口构件上添加 GtkList或者GtkCList构件，然后再设置GtkScrolledWindow构件的显示滚动条的模式 —— 自动或者总是出现（Automatic 或Always）。

4) 在窗口上放一个GtkNotebook构件，然后在笔记本构件的某一页上放置一个 GtkFrame构件，在GtkFrame上放三个GtkRadioButton。注意到GtkNotebook的每一页都是容器，它的实现方法是：将GtkNotebook在窗口上定位，然后在笔记本构件的某一页上放一个 GtkFrame，并在GtkFrame上添加一个GtkHBox（分为三部分），最后将三个GtkRadioButton组装到GtkHBox上。

(2) 使用表格构件（GtkTable）定位构件

用鼠标在构件箱上选择GtkTable构件，在窗口上点击一下，将弹出一个对话框询问表格划分为几行几列，缺省设置是 3×3。根据规划设定好行数和列数，表格构件就可以添加在窗口上了。实际上以后还可以根据需要随时增加或减少行数和列数。

下一步就是向表格构件中添加构件。在构件箱中选择需要的构件，并在表格上点击，就可以将构件添加到表格上了。注意，即使要让构件占据表格几个单元格，也只能先放在某一个格子上，然后再调整构件的位置。构件在表格上的位置由以下几个参数决定：水平起点坐标、垂直方向起点坐标、跨越的单元格列数、跨越的单元格行数。选中刚添加的构件，在“属性编辑器”窗口上选择 Place标签页，然后设置构件的位置。其中， Cell X对应于构件在表格中的起点坐标， Cell Y指定垂直起点坐标， Col Span指定跨越多少列， Row Span指定跨越多少行。例如，一个2×2的表格构件，一个按钮要占据上面两格，它的 Cell X应为0，Cell Y为0，Col Span为2，Row Span为1；另一个按钮占据左下角的单元格，那么它的 Cell X为0，Cell Y为1，Col Span为1，Row Span为1；第三个按钮占据右下角的单元格，那么，它的 Cell X为1，Cell Y为1，Col Span为1，Row Span为1。你也可以将表格和组装盒结合起来使用。

(3) 用GtkFixed定位构件

虽然不鼓励使用 GtkFixed构件，但是在某些情况下使用它还是很方便的，比如在任何情况窗口的大小都不会改变，窗口内的构件也不会改变位置时。

在构件箱上选择GtkFixed构件，然后点击窗口，可以将GtkFixed添加到窗口上。在构件箱上选择构件，然后在GtkFixed构件上点击，构件就会在上面定位。构件在GtkFixed上的位置由以下四个参数决定：X、Y、Width、Height。X和Y是构件左上角的坐标，Width和Height是构件的宽度和高度。你可以用鼠标直接拖动构件以调整它的位置，或将鼠标放在构件角部按住拖动以改变它的大小。也可以在属性编辑器的Place标签页上设置上面四个参数值。

3. 为窗口添加菜单

为窗口添加菜单是一个很烦琐的工作。Glade提供了一个很直观的方法来创建菜单。在构件箱上选GtkMenubar构件，然后在窗口上点击，为窗口添加一个菜单条。选中菜单条，选择“检视”菜单的Show Properties Editor，在“属性编辑器”窗口上的Widget页，有一个Edit Menus，点击这个按钮，显示如图17-5所示的菜单编辑器。

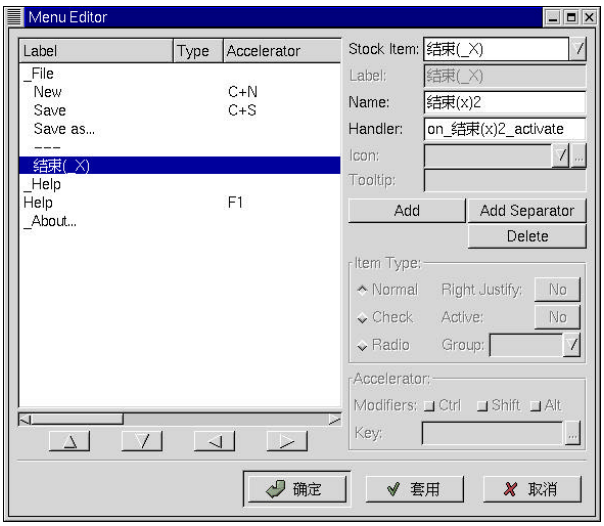

图17-5 Glade菜单编辑器

点击“Add”按钮，添加一个菜单项菜单。在Label后填写菜单项的标题，在Name后写菜单名称，Handle后会自动出现处理函数的名称——这也是后面将连接到菜单项上的回调函数。点击向右的箭头可将菜单项设为前面一个菜单项的子菜单。向左的箭头提升一个层次。用向上和向下的箭头可以调整菜单项在菜单中的位置。

可以为菜单设置图片。在Icon后的下拉组合框中为菜单项选择一个图片；或者点“Icon”后的按钮，选择一个图片。

选择Check无线按钮，可以将菜单项设置成为“检查”菜单项——例如有的程序中的“显示工具条”菜单项，选中表示显示工具条，未选中表示不显示工具条。

选择Radio无线按钮，可以将菜单项设置为“无线按钮”类型的菜单项，这时同组的“无线按钮”类菜单项是互斥的——例如有的程序中的对齐方式设置，不可能既是左对齐，又是居中对齐。

在Accelerator中可以为菜单项设置快捷键。在Modifiers后选中组合键，然后点击Key后面的按钮，选择一个按键。建议遵从常用的快捷键设置，比如“新建”菜单项的快捷键一般为Ctrl+N。

点击Add Separator，在菜单中间添加一条分隔线。点击 Delete按钮，删除一个菜单项。

注意，不要为带子菜单的菜单项设置快捷键，不要为点击菜单设置图片，也不要将带子菜单的菜单项设置为“检查”或“无线”类型的菜单项。另外，在窗口上添加菜单也要遵守构件组装原则：一般在窗口上放一个GtkHBox，然后将菜单条组装到这个GtkHBox中。

做好上面的设置以后，选“确定”按钮，就可以创建一套完整的菜单。

很多程序都提供弹出菜单功能。在窗口上单击鼠标右键，依据具体场合，会弹出一个快捷菜单，给人的感觉是很方便，也很神秘。 Glade也提供了一个弹出菜单的实现方法。在构件箱的Gtk Additional上点击Pop Menu，主窗口中添加一个弹出菜单对象。双击它，出现菜单编辑器，与普通的菜单编辑器完全一样。使用上面的步骤创建菜单， Glade会生成一段独立的创建菜单的代码，在需要弹出的场合调用就可以了。

4. 设置构件的属性

上面已经介绍了一部分构件属性。构件的绝大多数属性都可以在属性编辑器中设置，并且可以立即看到效果。

在窗口上选中构件，在属性编辑器中设置它的各项属性。如果在窗口上不能选中它，可以打开“构件树”窗口（选“检视”菜单的 Show Widget Tree菜单项），然后找到该构件，在上面点击右键，从弹出菜单中选 select，再转到属性编辑器，设置它的属性。有一些属性在Glade中还没有完全实现，比如按钮构件的背景颜色和前景颜色、不活动时的颜色等，在Glade中既能够设置属性值，又能够看到效果，但是实际上，生成代码时并没有写到源代码中；编译后，也没有实际效果。今后的 Glade版本可能会实现这些功能。在使用 Glade时对这一点务必注意。

大多数构件在属性编辑器中都有一项 Tooltips。在其中输入说明性的字符串，设计时即可看到它的效果。实际上它为构件添加了一个 GtkTooltip对象。

5. 为构件的信号设置回调函数

在窗口上选择构件，然后选中“属性编辑器”窗口上的Sig页，如图17-6所示。

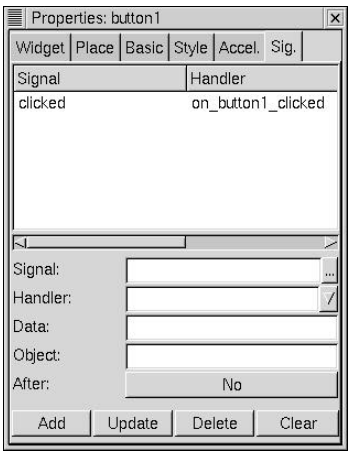

图17-6 为构件的信号设置回调函数

点击Signal后面的按钮，选择一个信号，然后点击Handler后面的下拉箭头，选择一个回调函数。如果需要，在Data后面输入要传入函数的用户数据，然后点击添加，就为构件的信号设置了一个回调函数。在本例中，为一个名为button1的按钮的clicked信号设置了一个回调函数：on_button1_clicked。目前， Glade不能为构件的子构件的信号设置回调函数。例如， GtkCombo包含一个GtkEntry子构件。如果想对GtkCombo构件combo1的显示文本的变化进行响应，应该为 GTK_COMBO(combo1)->entry的changed信号设置回调函数。目前的 Glade还不能做到这一点，所以必须在interface.c、callbacks.c以及callbacks.h中手工添加代码。

6. 生成源代码

完成上面所列的工作后，实际上应用程序的界面设计已经大体完工了。下一步就是生成源代码，然后实现应用程序的特有功能。

在Glade下，选择“文件”菜单下的 Build Source Code菜单项，或者点击主窗口工具条上的“Build”按钮，可以生成C语言的源代码。这些将是我们应用程序的基础，以后的工作就是通过编码实现程序的各项功能。 Glade生成的源代码完全符合 Gnome应用程序的编码规范。

一般Glade会创建一个Macros目录（其中包含了编译需要的宏）、一个po目录（用于容纳国际化文件）、一个src目录（源代码）、一个autogen.sh脚本文件，以及其他设置编译选项时要用到的文件。此外还有 ChangeLog、Readme、News、Authors等文件（空文件）。应用程序的源代码都放在src目录里。其中包含main.c、interface.c、interface.h、callbacks.c、callbacks.h、support.c、support.h几个主要文件。

Main.c是程序的主文件，它包含了 main函数，在它的头部包含了 gnome.h文件（包含了gnome.h之后不再需要包含 gtk.h、gdk.h、glib.h以及所有用到的构件的头文件）。创建应用程序用户界面的函数都放在 interface.c中，interface.h中包含了interface.c中的所有函数声明。所有的回调函数都放在 callbacks.c中，callbacks.h文件中包含了 callbacks.c中所有函数的声明；support.c文件中包含 Glade提供的几个实用函数； support.h包含support.c中的所有函数的声明。

有了上面这些文件之后，下一步就是在这个基础上增加代码以实现应用程序的功能。如果要直接在代码修改界面，可以修改 interface.c文件；如果要添加新的回调函数，可以在callbacks.c和interface.c中添加代码。要注意的是，如果增加了新函数，不要忘了在相应的头文件（interface.h、support.h、callbacks.c）里添加函数声明。

另外，support.c中包含了几个由Glade提供的实用函数，它们是：

```
GtkWidget*  lookup_widget  (GtkWidget       *widget,
                            const gchar     *widget_name);
GtkWidget*  create_pixmap   (GtkWidget       *widget,
                             const gchar     *filename,
                             gboolean         gnome_pixmap);
```

lookup_widget根据提供的构件的名称返回一个构件指针。在代码中调用这个行数来传递指针是非常方便的。create_pixmap用于在interface.c中由文件名创建pixmap图片。

7. 编译用Glade生成的代码

对较大型的程序设置编译选项，以及创建 Makefile是很复杂的。一般要联合使用各种GNU工具，如automake、autoconf等，创建一个configure脚本和Makefile.am文件。然后运行configure脚本设置编译选项生成 Makefile文件。 Glade所生成的C源代码中包含一个名为autogen.sh的脚本。使用它可以轻松完成这一复杂任务。假设用 Glade创建了一个应用程序myapp的界面，并已经通过编程实现了所需要的各种功能。源代码存放在 /root/myapp下。在shell提示符下执行下面的代码：

```
cd /root/myapp
./autogen.sh
```

autogen.sh脚本会搜索源代码的路径、头文件路径、所需库文件的安装路径，然后生成一个Makefile文件。现在，就可以开始编译了。在shell提示符下输入：

```
make
```

如果源代码没有错误，就会生成所需要的可执行文件了。编译结果一般放在 src子目录下。可以尝试运行如下程序：

```
cd src
./myapp
```

如果需要，可以用gdb或者xxgdb调试这个程序。

除了某些功能还没有实现以外，当前 Glade版本中还有一些bug。最好不要使用构件箱中Gnome页上的GnomeMessageBox（Gnome消息框），因为它所生成的关于GnomeMessageBox代码不能正常编译。实际上，创建GnomeMessageBox以及调用它的方法非常简单。

上面仅仅是对Glade用法的一个简单概述。还有一些 Glade功能这里没有介绍。Glade是非常有希望的GUI生成器，Linux社区的专家对它寄予厚望。可以预见，今后的 Glade功能会更加强大。