Gtk2-perl 编程: 控件编程

CN Alexe
Opensource创业者

2006 年 4 月 13 日

这篇文章是"Gtk2-perl 编程"的后续，上篇文章只是简单的介绍了 Gtk2-perl 编程的概念与大方面的问题，从本篇开始，我们来详细的介绍一下 Gtk2-perl 的各个部分。本文主要向大家介绍 Gtk2-perl 中的控件。

一. 控件的介绍

在上一篇中，我们讲述过 Gtk2-perl 的内部控件，并将大的分类简单介绍了一下。在这里，我们会详细的说明一下某些控件，让大家能较为清楚的了解这些控件。注意，每个控件内部都有很多的调用函数，如果要讲解每个控件的所有函数将要耗费大量的篇章，所以这里只是讲解一些基本的或者重要的函数。

我们先来讲一下这些控件的用途：

Gtk2-perl 中的控件有很多，但是无外乎这几种类型，一种是用来显示数据的，一种用来操作数据的，一种是用来将其他控件布局的，还有一种是特殊目的的控件。

1、显示数据的控件一般可以显示文字数据，图片数据或者其他特殊数据。这类控件的特点是主要用来显示数据，显示的数据一般是不可编辑或操作的。用户一般可以通过这类控件得到某些信息，但是不能操作这些控件。更具体的：

显示文字数据的控件我们一般使用 Gtk2::Label，这个控件主要是用来显示一小段文字从而表现一个标题或者一点提示的作用，也就是说是用来点缀程序的。

显示图片数据的控件我们一般是用Gtk2::Image，这个控件的目的就是用来显示一个图片，与上面的 Gtk2::Label 的作用基本相似。

我们还有显示进度条的控件Gtk2::ProgressBar，这个控件可以显示一个进度条，来指示用户当前某个程序的进展情况，在某些长时间运行的程序中我们经常可以见到。

Gtk2::Statusbar 显示一个状态栏。这个控件一般用在程序的底部，显示某些提示信息给用户。

Gtk2::Frame 一个装饰性的控件，显示一个外边框并带有一个标题。它的内部可以再嵌入一个子控件。

Gtk2::Hseparator 另一个装饰性的控件，显示一个一条水平分割线。

Gtk2::Vseparator 同上，显示一个垂直分割线。

Gtk2::List 一个列表形的控件。就是将数据以垂直排列的方式显示出来。经常用来罗列很多相似的数据。可以按照数据的值来排序显示。

Gtk2::TreeView 树形显示控件。主要用来以树形方式显示数据。例如：类似我们经常使用的win32下的资源管理器就是一个典型的树形控件的例子。在这个控件中你可以显示一个树形列表，可以展开或者收缩这个列表，可以编辑这个列表中的每一项，可以在这个列表中使用简单的按钮等等。这个控件通常与Gtk2::TreeStore 等其他控件一同使用。这个控件的概念也与操作类型的控件有些重复，它不仅显示，还可以让用户操作。

2、操作数据的控件一般可以让用户来输入或编辑文字，或者选择某个选项，或者触发某个事件，例如：点击按钮来运行某个函数。

Gtk2::Button 一个普通的按钮控件，当点击时我们可以触发某个事件。

Gtk2::CheckButton 一个check（打对勾）类型的按钮，一般有两种状态：选择或者没有选择。

Gtk2::RadioButton 一个radio（小圆点）类型的按钮，一般是多个radio类型的按钮，然后我们在其中选择一个，经常用于互斥选择。

Gtk2::ToggleButton 一个toggle（和开关类似）类型的按钮，按下后会保持按下的状态直到用户再次按下才显示按钮松开。

Gtk2::Entry 一个单行文字的输入框。用户可以在这里输入一行文字

。

Gtk2::TextView 强大的文字控件，不仅可以让用户输入、显示、编辑多行文字，而且可以显示静态图片或者简单的动画图片，甚至可以嵌入其他的控件，例如，嵌入按钮控件等等。这个控件的使用在后面我们会详细的讲述。

Gtk2::Combo 一个选择控件。很像html中的select下拉菜单，点击后出现一个下拉菜单，用户可以在下拉菜单的多个选项中选择一个选项。一般常用在工具栏中，例如：word中我们可以在工具栏里直接选择一个字体，或者字体的大小时，我们都会用到该类型的控件。

Gtk2::SpinButton 一个数字选择控件。这个控件主要用来在一定的范围内选择一个数字，控件的左边是上下选择的两个按钮，可以让用户将所选择的数字加1或减1，用户也可以直接在控件中输入自己需要的数字。

Gtk2::Hscale 和 Gtk2::Vscale 水平和竖直输入框。这个框水平或者竖直显示一个移动条，用户移动这个条就可以在一定的范围内选择一个中间值来作为输入的数据，不太常见。

3、将其他控件布局的控件一般并不明确的显示在用户的面前，它们只是负责将其他的控件在主窗口中摆放整齐。在这里一般有两种布局的思想，一种是使用 table 的方式将主窗口横竖的划分，然后将所需的控件按照自己的要求放入划分出来的每个单元中；另一种是使用画布式的方式，你可以随意的摆放所有的控件，只要你准确的说出每个控件在主窗口中的坐标就可以。

按照table方式布局的控件有：

Gtk2::HBox 一个用来水平分割的控件，目的是在不同水平上摆放多个控件。

Gtk2::VBox 一个竖直分割的控件。目的类上。

Gtk2::Table 一个按照表格的方式来分割的控件，也就是同时既有水平分割也有竖直分割，是上面两个控件的结合体。

按照画布方式布局的控件有：

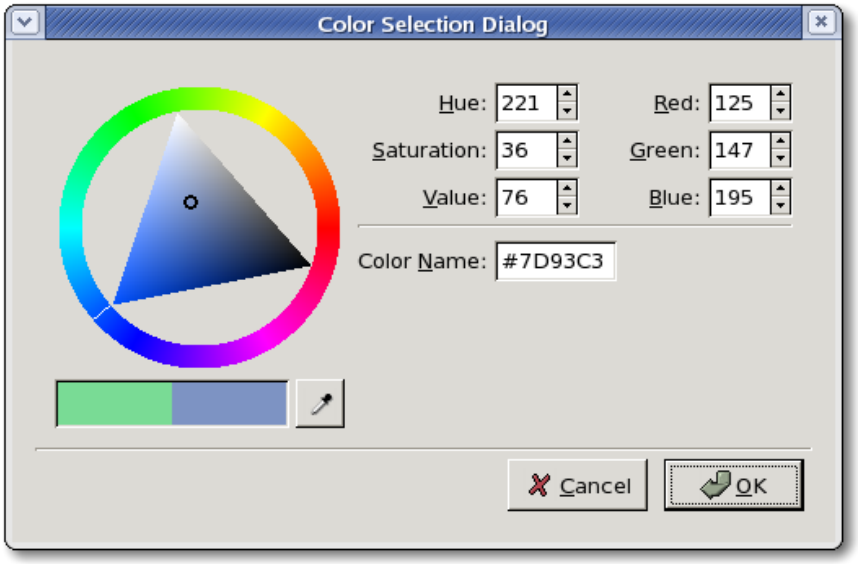
Gtk2::Fixed 这个控件可以让你在它内部的任意位置放置固定的子控件。不过，这个控件有些缺点：例如可能会导致文字的被截断，或者多个子控件的重叠等显示问题。所以这里并不太推荐使用该控件。

Gtk2::Layout这个控件可以让你在它内部自由放置子控件，但是它支持滚动窗口，可以有无限的滚动区域。这个控件是为了弥补Gtk2::Fixed中的缺陷而编写的，它与Gtk2::Fixed的用法基本相同，因此推荐使用这个。

为了达到某些复杂布局的目的，很多时候布局控件都是混合使用的。不过，总的来说，按照table方式布局更加容易。

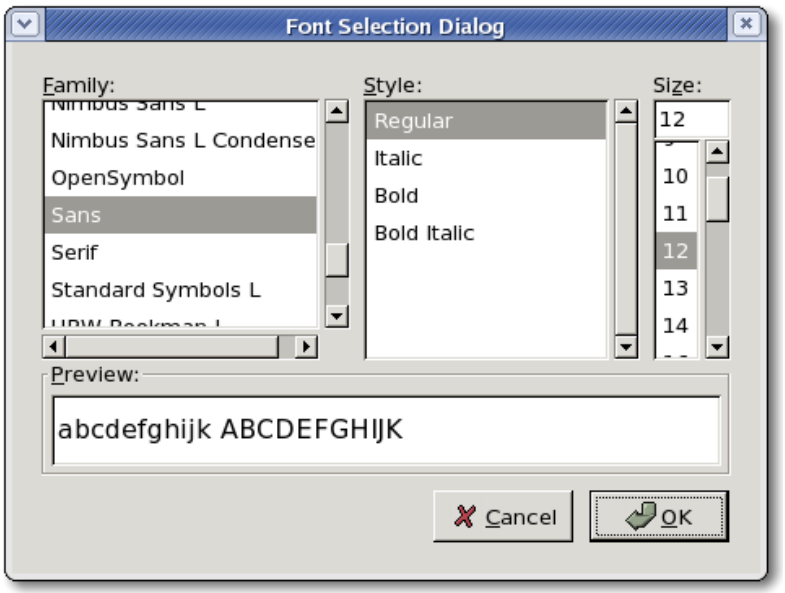
4、特殊目的的控件在Gtk2-perl中有很多，它们都有自己特殊的目的，而我也将大多数无法以上面的形式分类的控件都归结到这里了。

- 首先这样一类控件是选择类控件，这类控件的目的是跳出一个新窗口，来完成某个在很多程序中都通用的功能。
  Gtk2::ColorSelection 颜色选择控件。跳出一个通用的颜色选择窗口，用户可以按照红、绿、蓝的方式来选择一个自己要求的颜色，用户还可以调节颜色的不透明度、饱和值、色调，然后返回用户所选择的颜色值。
  Gtk2::ColorSelection图例：



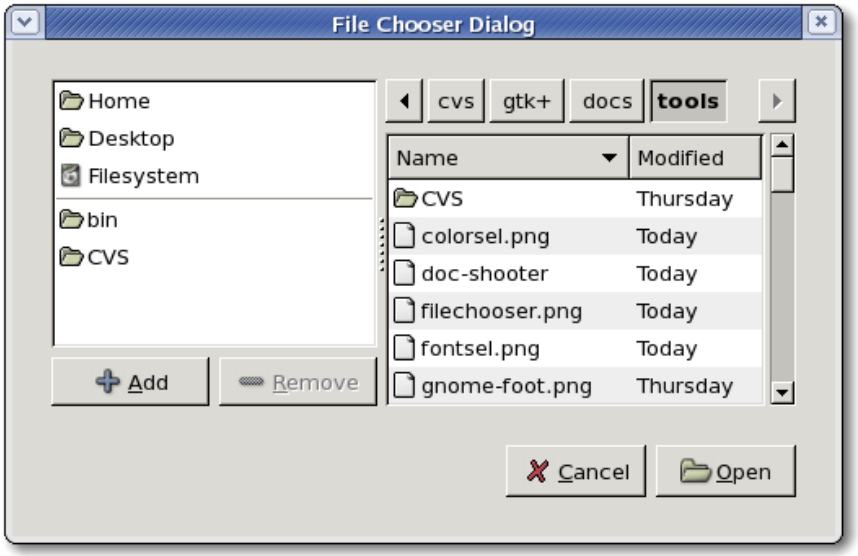
Gtk2::FontSelection 字体选择控件。跳出一个通用的字体选择窗口（该窗口中可以选择的字体都是读取的当前系统所拥有的字体），用户可以选择字体，字体大小以及样式，并伴有当前选择字体的一个预览。

Gtk2::FontSelection图例：



Gtk2::FileSelection 文件选择控件。跳出一个文件选择窗口，可以让用户单选或多选文件，还可以选择目录、建立文件或建立目录。用户可以自由的更换目录，不过这个文件选择框与windows平台的文件选择框的样式不同，应该说它们都有各自的优势吧。

Gtk2::FileSelection图例：



Gtk2::Calendar 日历控件。显示一个标准的公历日历，用户可以查看当前的年、月、日，或者选择一个当前的日期。

Gtk2::Dialog 一个简单的对话框，上面可以有几个简单的按钮选择，例如：确定、取消、应用。这个控件一般用来处理某些用户的简单选择，例如：提示用户程序出错，提示用户是否存储文件等等。

- 其次，菜单类的控件。这类控件都是用来在程序的顶部显示一个通常意义上的菜单或者工具栏。

  Gtk2::Menu 一个标准的菜单控件。这个控件可以让我们产生下拉菜单并在这个下拉菜单中包含多个菜单选项或者更深的级联菜单。每个菜单项可以对应于一个触发事件。菜单上可以添加图标，还可以添加快捷键。

  Gtk2::Toolbar 标准的工具栏控件。我们都清楚，现在的GUI程序一般在菜单下面会有一个快捷工具栏，这个工具栏内放入一些特别常用的按钮，来方便用户的迅速使用。Gtk2::Toolbar 就是这样的标准工具栏。它允许插入文字、图标按钮。
- 再次，还有一些控件的目的就更加特殊了。

  Gtk2::Scrollbar 滚动窗口控件。这个控件相当的重要，很多时候如果我们要显示的内容在程序的主窗口中无法完全显示，我们就需要加入这个控件，更准确的说是将显示数据的控件加入到Gtk2::Scrollbar 控件中，这样我们就可以有水平与垂直的滚动条来滚动显示多出窗口的数据。

  Gtk2::DrawingArea 相当重要的控件，可以称之为绘图控件。这个控件可以帮助我们建立自己的控件或者是不使用任何现有的控件直接调用底层的绘图指令。我们都很清楚，现有的控件也许可以建立标准的一个GUI程序，但是我们在写GUI程序的时候，很有可能有自己的特殊需求，需要的控件并不能在标准控件中找到，这时我们就需要用Gtk2::DrawingArea。要使用该控件就必须了解Glib、pango等模块。在下一篇中我们将讲解这个控件的使用。

  Gtk2::Tooltips 提示控件。这个控件专门用来对于GUI界面上的一些简单标签做更详细的解释，例如：当我们将鼠标停放在界面的某个标签上时，过一会可以出现一个文字框来对于这个标签做更详细解释。

  Gtk2::Notebook 纪事本控件。这个控件也是相当强大，就像我们在firefox里最常使用的tab窗口一样，我们可以通过这个控件来在一个主窗口中嵌入多个控件，而每个控件通过控件上方的标签来切换显示。微软在IE7中也采用了类似的控件。

基本上，上述的控件就是我认为在 Gtk2-perl 编程中经常要使用到的，还有一些较为偏僻的控件，在这里就暂不介绍了。下面，我们来针对几个较为重要的控件详细的说明一下。

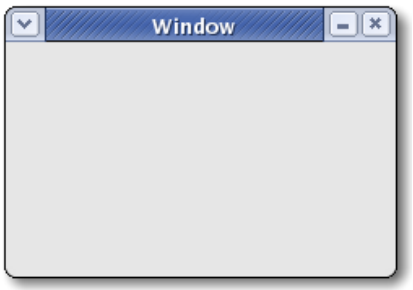
二. Gtk2::Window

这个可以说是最重要的控件了， Gtk2::Window 决定了你所显示的主窗口的样式以及相应的一系列设置。

我们这里通过代码来说明：

```
my $win = Gtk2::Window -> new ( $type ) ; ## ##########
$win -> set_title ( "Test for window" ) ; ## #########
$win -> set_default_icon_from_file ( $filename ); ## ########
$win->set_size_request ( "800" , "600" ) ; ## #########
$win -> set_position ( 'center' ) ; ## #########
$win -> set_type_hint ( $hint ); ## ##########
$win -> signal_connect ( 'destroy' , sub { Gtk2->main_quit } ) ; ## ###########
```

一个新窗口的图例：



第一行里面的 $type 值可以为：toplevel 或 popup。 toplevel 代表了这个窗口是最顶层的窗口，这个窗口可以包含其他的窗口，是我们最常用的窗口类型；popup代表这个窗口只是弹出窗口，这个窗口一般是菜单窗口或者工具提示窗口，通常就是一个弹出式的提示窗口。缺省的话就是toplevel。

第二行，我们可以设置窗口的标题，注意这里的标题文字必须是 utf8 编码，所以当你输入中文时一般需要转换编码。我们可以用$win -> get_title 来取回当前窗口的标题，例如：当我们想要在当前窗口标题的基础上添加某些文字时就常用这个。

第三行，我们设置了一个图标，这个图标是直接从硬盘上的文件取出，文件的格式可以为 png、jpg、gif、bmp，图标的大小一般为16x16像素，如果太大则只会裁去16x16范围以外的部分。

第四行，我们设置了窗口大小，我们还可以固定窗口的大小：$win -> set_resizable (0);或者随时改变窗口的大小：$win -> resize ($width , $height); 或者取回当前窗口大小：($w,$h) = $win -> get_size。当然，我们还可以将窗口全屏显示：$win -> fullscreen; 或者取消全屏：$win -> unfullscreen; 最大化也是如此：$win -> maximize; $win -> unmaximize; 。

第五行，我们设置的是窗口刚被创建时所出现的位置。这里可以有：none、center、center-always、center-on-parent、mouse。None的话代表随机出现，mouse的话代表窗口的右上角出现在鼠标指针的位置上。

第六行，我们设置窗口的暗示类型。可以设置的值有：
normal、dialog、menu、toolbar、splashscreen、utility、dock、desktop。

Splashscreen可以称之为闪屏，就是我们在启动程序时常见的一个显示窗口，随着程序的启动完成这个窗口也会随之消失而变为正式的程序操作窗口，这样的窗口在现代GUI程序中十分常见，主要是为了在用户等待程序的全部读入时给用户一个较好的印象。这个Splashscreen 窗口没有最大最小化、关闭的按钮。

Desktop称为桌面。也就是说这个窗口会作为一个桌面出现，并没有任何的标题或者最大最小化、关闭等按钮。在这里，必须强调一点：win32与X11的不同。例如：在win32环境下Desktop 暗示就无法起作用，它只能作用在X11环境中。所以当你试图编写一个跨平台运行的程序时，就要避免使用这种选项。

dock 船坞选项。这个选项是使窗口作为一个小面板来出现的。该选项同样只能在X11环境中使用。具体的使用可以参考x11的内部协议规定。

注意：在win32下很多暗示选项是不起作用的。

这里还要介绍一个函数：$win -> set_decorated，它的作用是将窗口的装饰去掉，也就是去掉窗口的标题、最大化最小化关闭按钮所在的那一行，窗口作为一个裸窗口显示。当我们需要在win32下使用这种窗口时，可以先设置一个特殊的窗口暗示类型，然后再$win -> set_decorated (0); 就可以。

也许大家都会有疑问，我们为什么要使用这些特殊的暗示类型。在x11下，大家可以参考x11的内部协议规定，里面有描述。在win32下，我想主要是要使用裸窗口。裸窗口的使用在各个平台下还是十分常见的，例如：我们常见的msn信件到来通知，就是这样一个窗口；或者当我们需要写一个比较怪异的窗口时，例如：windows media player里面的那些怪异窗口，这时我们就需要去掉原有的标准窗口装饰，并自己的用图片描绘出窗口的样式。

除了以上我们介绍的这些关于窗口的api以外，还有$win -> present; 设置窗口显示在用户面前；$win -> set_gravity ($gravity); 这个也是用来设置窗口最初出现时的位置，$gravity可以为north-west、north、east等。

三. Gtk2::TextView

强大的多行文字控件，可以显示太多种的数据，基本上如果你是想要纯粹的显示数据，或者编辑文字与图片组成的数据流，都可以直接使用这个控件来完成。

这个控件包含几个部分：

i. Gtk2::TextBuffer
ii. Gtk2::TextChildAnchor
iii. Gtk2::TextIter
iv. Gtk2::TextMark
v. Gtk2::TextTag
vi. Gtk2::TextTagTable
vii. Gtk2::TextView

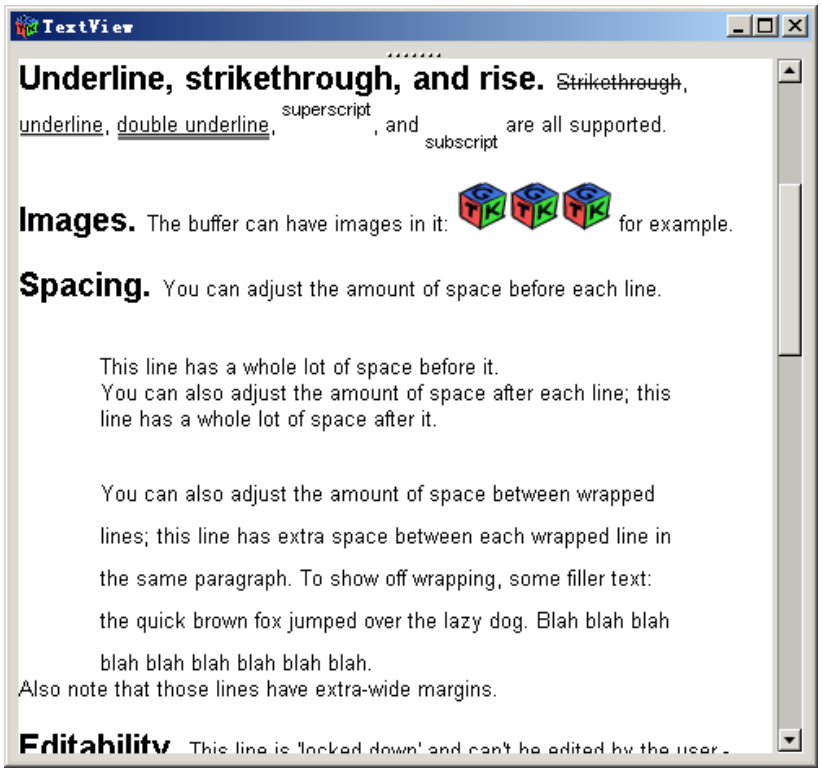
Gtk2::TextBuffer 的目的是保存所有要显示的数据在一个文字缓冲中。
Gtk2::TextChildAnchor 的目的是在文字缓冲中插入一个控件。
Gtk2::TextIter 的目的是操作当前缓冲所处的位置。
Gtk2::TextMark 的目的是在文字缓冲中保存一个位置，比较像我们在Html中的书签功能，我们可以随意的回到这个位置。
Gtk2::TextTag 的目的是为文字缓冲中的文字添加一个显示标签，例如：改变文字的显示颜色，改变文字的字体，或者文字大小。
Gtk2::TextTagTable 的目的是保存所有 Gtk2::TextTag 标签。每个 Gtk2::TextBuffer 缓冲只对应于一个 Gtk2::TextTagTable 。
Gtk2::TextView 的目的是显示文字缓冲中的数据。

我们看到了，要实现一个多行文字控件，需要多个模块配合起来使用，我们大概来描述一下这个过程：一个Gtk2::TextView 控件，它的显示数据（文字与图片）是要存储在Gtk2::TextBuffer 建立的缓冲中，而这个缓冲中的数据我们可以为其添加Gtk2::TextTag，从而为缓冲数据（文字）设定显示格式，或者我们用 Gtk2::TextChildAnchor 可以在缓冲中添加其他的控件来显示。当我们想要操纵我们在缓冲中的位置时，我们可以使用 Gtk2::TextIter，例如：向前移动一行，向后移动一个字母。当我们想要在缓冲中标记一个位置，然后可以快速的移动到这个位置去显示，我们可以使用 Gtk2::TextMark 。

OK，我们现在来个实例来说明一下：

```
my $sw = Gtk2::ScrolledWindow -> new ();
$sw -> set_policy ( 'automatic' , 'automatic' );
my $textview = Gtk2::TextView -> new ();
$textview -> can_focus (1);
$textview -> set_editable (0);
$textview -> set_left_margin (10);
$textview -> set_right_margin (10);
$textview -> set_wrap_mode ( 'GTK_WRAP_WORD_CHAR' );
my $tab_table = Gtk2::TextTagTable -> new ();
my $text_buffer = Gtk2::TextBuffer -> new ( $tab_table );
$textview -> set_buffer ($text_buffer );
$sw -> add ( $textview );
```

一个textview控件的实例：



这个实例里我们首先建立了一个 Gtk2::ScrolledWindow 控件，这个控件的目的是为了产生滚动条。当我们要显示的数据比 Gtk2::TextView 的窗口大时，我们就需要滚动条。这里我们设置$sw -> set_policy ('automatic' , 'automatic'); 代表了水平滚动条与垂直滚动条都是自动（automatic）出现的，也就是如果要显示的数据超出了窗口，这才出现滚动条，否则将不显示滚动条。还有一点要注意的是：$sw -> add ($textview); 滚动条控件要包含文字控件。

然后，我们建立了一个 Gtk2::TextView 控件，并设置它的属性。

- $textview -> can_focus (1); 表示这个文字控件可以接受输入。
- $textview -> set_editable (0); 表示这个文字控件的内容不能被编辑，也就是说我们不能随意的插入文字。如果设置为1，那么我们就可以编辑文字控件的内容，就像一个记事本那样。
- $textview -> set_left_margin (10); 设置文字控件左面的空白大小为10像素，这个空白是说该控件与窗口边界（或者另一个控件）之间的距离，主要是美观的需要。
- $textview -> set_wrap_mode ('GTK_WRAP_WORD_CHAR'); 设置文字换行方式为遇到窗口边界就自动换行。其他的选项还有：none，不自动换行；

然后，我们又建立了一个 Gtk2::TextTagTable 表。接着建立了一个Gtk2::TextBuffer 并将该Gtk2::TextTagTable 表赋予文字缓冲。（这里其实有更简单的方式，我们可以直接使用 Gtk2::TextBuffer -> new ();，这样会自动建立一个tagtable。）

最后，我们将 Gtk2::TextView 的文字缓冲设置为刚建立的那个缓冲。至此，初始化的工作我们就全部完成了。接下来，我们要来看看如何向这样一个文字控件中插入数据。

看例子：

```
use Gtk2::Pango;
$text_buffer -> set_text ( "" );
my $iter = $text_buffer -> get_iter_at_offset (0);
$text_buffer -> insert_txt ($iter , "test test test\n" );
$text_buffer -> create_tag ( "bold" , weight => PANGO_WEIGHT_BOLD );
$text_buffer -> insert_with_tags_by_name ( $iter , "test test" , "bold" );
my $pixbuf = Gtk2::Gdk::Pixbuf -> new_from_file ( 'test.jpg' );
$text_buffer -> insert_pixbuf ( $iter , $pixbuf );
my $input = Gtk2::Entry -> new;
my $anchor = $text_buffer -> create_child_anchor ( $iter );
$textview -> add_child_at_anchor ( $input , $anchor );
$textview -> show_all ();
```

首先，我们 use Gtk2::Pango; ，这是为了使用下面的 PANGO_WEIGHT_BOLD 这样的常量定义。

$text_buffer -> set_text (""); 先将缓冲清空。my $iter = $text_buffer -> get_iter_at_offset (0); 将我们的iter移动到缓冲的开头处。这两步都是一个文字缓冲开始插入数据前最好执行的两步，因为一般一个文字缓冲都要重复的使用，所以在开始一次新的使用前做一下这些工作可以保证缓冲的正确运行。

这里要说明一下iter。iter这里代表的是一个在缓冲中的当前位置，当插入任何数据时，都要从这个iter的当前位置开始插入。由于可以在缓冲中插入文字、图片、控件等多种数据，所以iter的计算是由gtk+的内部完成的，我们只需要关心如何使用iter就可以了。

接下来，$text_buffer -> insert_txt ($iter , "test test test\n"); 我们插入一段文字："test test test \n"。这里还有点说明：这里的$iter值当插入完成时会被自动的增加到插入的结尾处，以便于下一次数据的插入。所以，下一次插入数据时我们还可以直接使用 $text_buffer -> insert_txt ($iter , "test test test\n"); 就可以了，而不用管$iter值，因为每次插入后$iter值都被自动更新了。

接着，我们 $text_buffer -> create_tag ("bold" , weight => PANGO_WEIGHT_BOLD); 建立了一个粗体的显示标签，将这个标签的名称标为："bold"。这里我们可以创建的标签有很多种，例如：文字的颜色，字体，大小，风格，粗体，斜体，或者文字的下滑线，文字背景色，文字的背景位图，文字的对齐方式等等。

$text_buffer -> insert_with_tags_by_name ($iter , "test test" , "bold"); 我们插入一段文字" test test"，并将标签名称为"bold"的标签赋给这段文字，这样这段文字就会以粗体的方式显示。

接下来，my $pixbuf = Gtk2::Gdk::Pixbuf -> new_from_file ('test.jpg'); 我们读取了一个图像文件，建立了一个新的pixbuf。$text_buffer -> insert_pixbuf ($iter , $pixbuf); 我们将这个图片插入到了文字缓冲中。

最后，my $input = Gtk2::Entry -> new; 建立了一个新的单行文字输入控件。 my $anchor = $text_buffer -> create_child_anchor ($iter); 在缓冲的当前位置下建立了一个子锚。 $textview -> add_child_at_anchor ($input , $anchor); 在$textview 文字显示控件中添加新建立的文字输入控件到子锚上。 $textview -> show_all (); 显示所有的数据。当在缓冲中插入一个控件时，最好调用一下，否则控件可能会不显示出来。

Gtk2::TextView文字控件的功能很强，使用起来也比较的方便，所以可以用来实现一些较为常规的功能，如：编写一个文字编辑器，一个简单的显示软件等等。笔者的perl web browser项目"shellweb"最初就是使用这个控件来显示html数据的。如果大家感兴趣的话可以参考其中的用法。

四. 小结

由于篇幅所限，而每个控件的内容又都很多，关于 gtk2-perl 控件的介绍就先写到这里。我会在后面的文章中，不断的补充介绍其他的控件。希望大家能继续支持。

参考资料

- Shellweb: http://www.sourceforge.net/projects/shellweb
- Gtk2-perl: http://gtk2-perl.sourceforge.net
- Gtk+: http://www.gtk.org

关于作者

CN Alexe

Alexe, 一个Opensource创业者， 正在编写shellweb中。通过 Alexe at Alexe.cn 可以跟他联系。