GTK+ can do *this*?

Matthias Clasen
Guadec 2015
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
- Steppers are not dead yet
Steppers

Add to ~/.config/gtk-3.0/gtk.css

.scrollbar {
  -GtkScrollbar-has-forward-stepper: true;
  -GtkScrollbar-has-secondary-backward-stepper: true;
}
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
- Steppers are not dead yet
- Context menus
Context Menus

Connect to GtkWidget::popup-menu

```c
sb = gtk_scrolled_window_get_vscrollbar (swin);
menu = gtk_menu_new ();
...
g_signal_connect (sb, "popup-menu",
    G_CALLBACK (popup_cb), menu);
```
2. Output-only Windows

X11: events propagate from child to parent
2. Output-only Windows

X11: Input shapes to the rescue
2. Output-only Windows
2. Output-only Windows

Client-side windows emulate X semantics
2. Output-only Windows

- `gdk_window_set_pass_through` implements all-or-nothing input shape
- `GtkOverlay::pass-through child property`
2. Output-only Windows

- gdk_window_set_pass_through implements all-or-nothing input shape
- GtkOverlay::pass-through child property
- Use widget drawing as decorations
2. Output-only Windows

- gdk_window_set_pass_through implements all-or-nothing input shape
- GtkWidget::pass-through child property
- Use widget drawing as decorations
3. Touch Selection

- You can add your own things to these popovers
Context Menus

Use the GtkTextView::populate-popup signal

```c
    g_signal_connect (tv, "populate-popup",
                     G_CALLBACK (populate_cb), NULL);
    g_object_set (tv, "populate-all", TRUE, NULL);

    ...

    static void populate_cb (GtkWidget *tv, GtkWidget *popup)
    {
        if (GTK_IS_MENU (popup))
            ...
        else
            ...
    }
```
3. Touch Selection

- You can add your own things to these popovers.
- You can also add your own things to other context menus: GtkEntry, GtkPlacesSidebar,...
4. Custom Spinbuttons

- Spin buttons use adjustments as model
- Text doesn't have to be numeric
Spinbutton Output

Use the GtkSpinButton::output signal

```c
entry = GTK_ENTRY (button);

adjustment = gtk_spin_button_get_adjustment (button);
value = gtk_adjustment_get_value (adjustment);

for (i = 1; i <= 12; i++)
  if (fabs (value - (double)i) < 1e-5)
    {
      text = gtk_entry_get_text (entry);
      if (strcmp (month[i-1], text) != 0)
        gtk_entry_set_text (entry, month[i-1]);
    }
```
Spinbutton Input

Use the GtkSpinButton::input signal; return TRUE, FALSE or GTK_INPUT_ERROR

```c
for (i = 1; i <= 12 && !found; i++)
{
    text = gtk_entry_get_text (entry);
    if (strstr (month[i - 1], text) == text)
    {
        *new_val = (double)i;
        return TRUE;
    }
}
return GTK_INPUT_ERROR;
```
5. Discrete Scales

- If only 5 values make sense, don't confuse the user by offering him a continuous selection
5. Discrete Scales

- If only 5 values make sense, don't confuse the user by offering him a continuous selection
- You can add marks to show the values
Rounding digits

Use the round-digits property to set the granularity of allowed values

<object class="GtkScale">
  <property name="round-digits">0</property>
</object>
6. Markup in Textviews

- Asked for many times – finally possible
6. Markup in Textviews

- Asked for many times – finally possible
- Use gtk_text_buffer_insert_markup()
6. Markup in Textviews

- Asked for many times – finally possible
- Use `gtk_text_buffer_insert_markup()`
- Markup can do exotic things like colored underlines or letter-spacing or font features
7. Fancy Text

- Pango and Cairo together can do great things
7. Fancy Text

- Pango and Cairo together can do great things
- `pango_cairo_layout_path()` turns the layout into the path of a cairo context
7. Fancy Text

- Pango and Cairo together can do great things
- `pango_cairo_layout_path()` turns the layout into the path of a cairo context
- Not very efficient, use with care
8. Templates for everything

- Templates can set up signal handlers
Signal Handler

In your template, use `<signal>`

```xml
<object class="GtkButton">
  <signal name="clicked" handler="clicked_cb"/>
</object>
```

In your class_init function, bind the callback

```c
gtk_widget_class_bind_template_callback (widget_class, clicked_cb);
```
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
Property Binding

In your template, use bind-source, bind-property, etc on the target <property>

<object class="GtkCheckButton">
  <property name="active"
    bind-source="other-object"
    bind-property="enabled"
    bind-flags="bidirectional|sync-create"/>
</object>
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
- Bindings can refer to the created instance
“this” Reference

Refer to the created instance by its class name

<object class="GtkCheckButton">
  <property name="active"
    bind-source="GtkFileChooserWidget"
    bind-property="enabled"
    bind-flags="bidirectional|sync-create"/>
</object>
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
- Bindings can refer to the created instance
- Create Pango attributes in your .ui file
Pango Attributes

GtkBuilder markup for Pango attributes ≠ Pango markup

<object class="GtkLabel">
  <property name="label">Not confusing at all!</property>
  <attributes>
    <attribute name="weight" value="bold"/>
    <attribute name="scale" value="1.2"/>
  </attributes>
</object>
9. Filter model flexibility

- Filter models can filter, of course
- Filter models can also 'invent' new columns
Questions?

All examples in this presentation are part of gtk3-demo

GTK+ team meeting: Monday morning
GTK+ can do this?

Matthias Clasen
GUADEC 2015
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
- Steppers are not dead yet
Steppers

Add to ~/.config/gtk-3.0/gtk.css

.scrollbar {
  -GtkScrollbar-has-forward-stepper: true;
  -GtkScrollbar-has-secondary-backward-stepper: true;
}

1. Scrolling, Scrolling, Scrolling

- Overlay scrollbars
- Kinetic scrolling
- Smooth scrolling
- Steppers are not dead yet
- Context menus
Context Menus

Connect to GtkWidget::popup-menu

sb = gtk_scrolled_window_get_vscrollbar (swin);
menu = gtk_menu_new ();
...

g_signal_connect (sb, "popup-menu",
                G_CALLBACK (popup_cb), menu);
2. Output-only Windows

X11: events propagate from child to parent
2. Output-only Windows

X11: Input shapes to the rescue
2. Output-only Windows
2. Output-only Windows

Client-side windows emulate X semantics
2. Output-only Windows

- gdk_window_set_pass_through implements all-or-nothing input shape
- GtkOverlay::pass-through child property
2. Output-only Windows

- gdk_window_set_pass_through implements all-or-nothing input shape
- GtkOverlay::pass-through child property
- Use widget drawing as decorations
2. Output-only Windows

- gdk_window_set_pass_through implements all-or-nothing input shape
- GtkOverlay::pass-through child property
- Use widget drawing as decorations
3. Touch Selection

- You can add your own things to these popovers
Context Menus

Use the GtkTextView::populate-popup signal

```c
static void populate_cb (GtkWidget *popup)
{
  if (GTK_IS_MENU (popup))
    ...
  else
    ...
}
```
3. Touch Selection

- You can add your own things to these popovers
- You can also add your own things to other context menus: GtkEntry, GtkPlacesSidebar,...
4. Custom Spinbuttons

- Spin buttons use adjustments as model
- Text doesn't have to be numeric
Spinbutton Output

Use the GtkSpinButton::output signal

```c
entry = GTK_ENTRY(button);

adjustment = gtk_spin_button_get_adjustment(button);
value = gtk_adjustment_get_value(adjustment);

for (i = 1; i <= 12; i++)
    if (fabs(value - (double)i) < 1e-5)
        {
            text = gtk_entry_get_text(entry);
            if (strcmp(month[i-1], text) != 0)
                gtk_entry_set_text(entry, month[i-1]);
        }
```
Spinbutton Input

Use the GtkSpinButton::input signal; return TRUE, FALSE or GTK_INPUT_ERROR

```c
for (i = 1; i <= 12 && !found; i++)
{
    text = gtk_entry_get_text (entry);
    if (strstr (month[i - 1], text) == text)
    {
        *new_val = (double)i;
        return TRUE;
    }
}
return GTK_INPUT_ERROR;
```
5. Discrete Scales

- If only 5 values make sense, don't confuse the user by offering him a continuous selection
5. Discrete Scales

- If only 5 values make sense, don't confuse the user by offering him a continuous selection
- You can add marks to show the values
Rounding digits

Use the round-digits property to set the granularity of allowed values

<object class="GtkScale">
  <property name="round-digits">0</property>
</object>
6. Markup in Textviews

- Asked for many times – finally possible
6. Markup in Textviews

- Asked for many times – finally possible
- Use gtk_text_buffer_insert_markup()
6. Markup in Textviews

- Asked for many times – finally possible
- Use `gtk_text_buffer_insert_markup()`
- Markup can do exotic things like colored underlines or letter-spacing or font features
7. Fancy Text

- Pango and Cairo together can do great things
7. Fancy Text

- Pango and Cairo together can do great things
- `pango_cairo_layout_path()` turns the layout into the path of a cairo context
7. Fancy Text

- Pango and Cairo together can do great things
- `pango_cairo_layout_path()` turns the layout into the path of a cairo context
- Not very efficient, use with care
8. Templates for everything

- Templates can set up signal handlers
Signal Handler

In your template, use `<signal>`

```xml
<object class="GtkButton">
  <signal name="clicked" handler="clicked_cb"/>
</object>
```

In your `class_init` function, bind the callback

```c
gtk_widget_class_bind_template_callback (widget_class, clicked_cb);
```
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
Property Binding

In your template, use bind-source, bind-property, etc on the target <property>

<object class="GtkCheckButton">
  <property name="active"
      bind-source="other-object"
      bind-property="enabled"
      bind-flags="bidirectional|sync-create"/>
</object>
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
- Bindings can refer to the created instance
“this” Reference

Refer to the created instance by its class name

<object class="GtkCheckButton">
<property name="active"
  bind-source="GtkFileChooserWidget"
  bind-property="enabled"
  bind-flags="bidirectional|sync-create"/>
</object>
8. Templates for everything

- Templates can set up signal handlers
- Templates can set up property bindings
- Bindings can refer to the created instance
- Create Pango attributes in your .ui file
Pango Attributes

GtkBuilder markup for Pango attributes ≠

Pango markup

<object class="GtkLabel">
   <property name="label">Not confusing at all!</property>
   <attributes>
      <attribute name="weight" value="bold"/>
      <attribute name="scale" value="1.2"/>
   </attributes>
</object>
9. Filter model flexibility

- Filter models can filter, of course
- Filter models can also 'invent' new columns
Questions ?

All examples in this presentation are part of gtk3-demo

GTK+ team meeting: Monday morning