

class GTK::Glade

Table of Contents

- 1 [Synopsis](#)
- 1.1 [new](#)
- 1.2 [add-gui-file](#)
- 1.3 [add-engine](#)
- 1.4 [add-css](#)
- 1.5 [run](#)

```
unit Gtk::Glade;
```

Synopsis

```
use MyGui::MainEngine;
use MyGui::SecondEngine;
use GTK::Glade;

sub MAIN ( Str:D $glade-xml-file ) {
    my GTK::Glade $gui .= new;
    $gui.add-gui-file($glade-xml-file);
    $gui.add-engine(:engine(MyGui::MainEngine.new()));
    $gui.add-engine(:engine(MyGui::SecondEngine.new()));
    $gui.run;
}
```

new

```
submethod BUILD ( )
```

Initialize Glade interface.

add-gui-file

```
method add-gui-file ( Str $ui-file )
```

Add an XML document saved by the glade user interface designer.

add-engine

```
method add-engine ( GTK::Glade::Engine $engine )
```

Add the user object where callback methods are defined.

add-css

```
method add-css ( Str $css-file )
```

Add a css style file, This is a CSS-like input in order to style widgets. Classes and id's are definable in the glade interface designer. A few are reserved. You need to look up the documents for a particular widget to find that out. E.g. the button knows about the [circular](#) and [flat](#) classes (See also [gnome developer docs](#) section CSS nodes).

run

```
method run ( )
```

Run the glade design. It will enter the main loop and when interacting with the interface, events will call the callbacks defined in one of the added engines.