Application development with Edje: From the very basics

Andres Blanc

Abstract

An introduction to GUI based application development. Starts from the most basic concepts and introduces the EFL libraries that deal with each of them. It covers Edje, Ecore, Evas and EWL.

This book is intended intended conc97a2nt. intended conc97a2nt. conc97a2nt. intended intended EWL.

Chapter 2. Structure of a graphical application

Someday someone might want to create a new application and he might feel overwhelmed by the large number of development libraries available jtie available available available the

Figure 3.1. Edje Workflow

Except for the usage of nested blocks, the sintax of an EDC file is similar to CSS. What really sets them appart is that with EDC the designer it's free to create and layout design elements as he sees

Evas optimises the rendering pipeline to minimise effort in redrawing changes made to the canvas and so takes this work out of the programmers hand, saving a lot of time and energT.

It's small and lean, designed to work on embedded systems all the way to large and powerful multi-cpu workstations. It can be compiled to only have the features you need for your target platform if you so wish, thus keeping it small and lean. It has several display back-ends, letting it display on several display systems, making it portable for cross-device and cross-platform development.

When using the Evas API 2.8 rectly, the Evaelopmener usinge( ) 2.8 (APIfunon) ( ) (to (APIc) (( ) 2.8 (APIlik( ) 2.8 (E

Figure 3.2. The necessary libraries

Chapter 4. The foundations at practice

practi9e

```
...
Ecore_Evas  *ecore_evas = NULL;
...
ecore_evas = ecore_evas_software_x11_new(NULL, 0, 0, 0, 800, 600);
if (!ecore_evas) return EXIT_FAILURE;
...
```

While the software_x11 engine is being used in this example, its possible to use any other supported engine by simply changing the second line to `ecore_evas_enginename_new()`. A list of supported engines and their parameters can be found in the Official API reference.

Once the canvas wrapper has been setup we need to change its state to visible:

```
...
ecore_evas_title_set(ecore_evas, "Example Application");
ecore_evas_name_class_set(ecore_evas, "testapp", "Testapp");
ecore_evas_show(ecore_evas);
...
```

While the first two lines are optional, it is useful to see the way we set the title name and class of the

. . .

Working with the canvas

Now that we know how to setup the enviroment to display the Edje objects we will review how to include and interact with the objects themselves. The following code is platform independent and can

```
    evas_pointer_canvas_xy_get(evas,&x,&y);
    if(x > 255) x = 255;
    if(y > 255) y = 255;             //R,   G,   B,   A,
    edje_color_class_set("main color", 190, x,   y,   255,
                                       255, 255, 255, 255,
                                       255, 255, 255, 255);
}
...
int main() {
    ...
    edje_object_signal_callback_add(edje, "mou     i{
```