Edje Reference Manual

Generated by Doxygen 1.5.1

Wed Mar 28 00:01:43 2007

Contents

1 Edje Library Documentation **1**

1.1 What is Edje? . . . 1

1.2 What does Edje require? . . . 2

1.3 How to compile and test Edje . . . 3

1.4 So how does this all work? . . . 3

2 Edje File Index **5**

2.1 Edje File List . . . 5

3 Edje Page Index **7**

3.1 Edje Related Pages . . . 7

4 Edje File Documentation **9**

4.1 edje.c File Reference . . . 9

5 Edje Page Documentation **31**

5.1 Todo List . . . 31

Chapter 1

Edje Library Documentation

Version:

0.5.0

Author:

Carsten Haitzler <raster@rasterman.com>

Date:

2003-2004

1.1 What is Edje?

Edje is a complex graphical design & layout library.

It's purpose is to be a sequel to "Ebits" which to date has serviced the needs of Enlightenment development for version 0.17. The original design parameters under which Ebits came about were a lot more restricted than the resulting use of them, thus Edje was born.

Edje is a more complex layout engine compared to Ebits. It doesn't pretend to do containing and regular layout like a widget set. It still inherits the more simplistic layout ideas behind Ebits, but it now does them a lot more cleanly, allowing for easy expansion, and the ability to cover much more ground than Ebits ever could. For the purposes of Enlightenment 0.17, Edje should serve all the purposes of creating visual elements (borders of windows, scrollbars, etc.) and allow the designer the ability to animate, layout and control the look and feel of any program using Edje as its basic GUI constructor. This library allows for multiple collections of Layouts in one file, sharing the same image database and thus allowing a whole theme to be conveniently packaged into 1 file and shipped around.

Edje, unlike Ebits, separates the layout and behavior logic. Edje files ship with an image database, used by all the parts in all the collections to source graphical data. It has a directory of logical part names pointing to the part collection entry ID in the file (thus allowing for multiple logical names to point to the same part collection, allowing for the sharing of data between display elements). Each part collection consists of a list of visual parts, as well as a list of programs. A program is

a conditionally run program that if a particular event occurs (a button is pressed, a mouse enters or leaves a part) will trigger an action that may affect other parts. In this way a part collection can be "programmed" via its file as to hilight buttons when the mouse passes over them or show hidden parts when a button is clicked somewhere etc. The actions performed in changing from one state to another are also allowed to transition over a period of time, allowing animation.

This separation and simplistic event driven style of programming can produce almost any look and feel one could want for basic visual elements. Anything more complex is likely the domain of an application or widget set that may use Edje as a convenient way of being able to configure parts of the display.

1.2 What does Edje require?

Edje requires fairly little on your system. to use the Edje runtime library you need:

- Evas (library)
- Ecore (library)
- Eet (library)

Evas needs to be build with the PNG and EET image loaders enabled at a minimum. Edje uses X for the test program, so you will need the SOFTWARE_X11 engine built into Evas as well. A suggested configure list is below in the "cheat sheet" for Evas.

Ecore needs the ECORE, ECORE_EVAS and ECORE_X modules built at a minimum. It's suggested to build all the Ecore modules, but the ECORE_FB modules is definitely optional.

Eet has no options so just build and install it.

It is suggested right now that you get the latest CVS versions of the required libraries. You also need to build them in the right order and make sure the right options are enabled in the required libraries. Here is a quick "cheat sheet" on how to get started.

```
1. You need Eet from the HEAD cvs branch (must be up-to-date)

  cvs co e17/libs/eet
  cd e17/libs/eet
  ./autogen.sh
  ./configure
  make
  sudo make install
  cd

2. You need Evas from the HEAD branch built with eet loader support.

  cvs co e17/libs/evas
  cd e17/libs/evas
  ./autogen.sh
  ./configure
  make
  sudo make install
  cd

3. You need Ecore from the HEAD cvs branch

  cvs co e17/libs/ecore
  cd e17/libs/ecore
  ./autogen.sh
  ./configure
```

```
  make
  sudo make install
  cd

4. You need embryo from the HEAD cvs branch

  cvs co e17/libs/embryo
  cd e17/libs/embryo
  ./autogen.sh
  ./configure
  make
  sudo make install
  cd
```

1.3 How to compile and test Edje

Now you need to compile and install Edje.

```
  ./configure
  make
  sudo make install
```

You now have it installed and ready to go, but you are missing data files. In data/ there are data sets for you to look at as examples. To try one out do:

```
   cd data
   ./e_logo.sh

   edje ./e_logo.eet
```

The Edje test program/viewer is able to view multiple Edje data sets. The following will view 3 of them at once in the one window (which you can resize to give you more space to move and resize the Edje data sets around):

```
  edje ./e_logo.eet ./e_logo.eet ./e_logo.eet
```

1.4 So how does this all work?

Edje internally holds a geometry state machine and state graph of what is visible, not, where, at what size, with what colors etc. This is described to Edje from an Edje .eet file containing this information. These files can be produced by using edje_cc to take a text file (a .edc file) and "compile" an output .eet file that contains this information, images and any other data needed.

Todo

See src/lib/edje_private.h for a list of FIXME's

Todo

Complete documentation of API

Todo

Bytecode language for extending programs... but what/how?

Chapter 2

Edje File Index

2.1 Edje File List

Here is a list of all documented files with brief descriptions:

edje.c (Edje Graphical Design Library) 9

Chapter 3

Edje Page Index

3.1 Edje Related Pages

Here is a list of all related documentation pages:

Todo List 31

Chapter 4

Edje File Documentation

4.1 edje.c File Reference

Edje Graphical Design Library.

Functions

- EAPI int edje_init (void)
  Initialize the EDJE library.
- EAPI int edje_shutdown (void)
  Shutdown the EDJE library.
- EAPI void edje_frametime_set (double t)
  Set the frametime.
- EAPI double edje_frametime_get (void)
  Get the frametime.
- EAPI void edje_freeze (void)
  Freeze all objects in the Edje.
- EAPI void edje_thaw (void)
  Thaw all objects in Edje.
- EAPI Evas_List * edje_file_collection_list (const char *file)
  Get the collection list from the edje file ?
- EAPI void edje_file_collection_list_free (Evas_List *lst)
  Free file collection.
- EAPI int edje_file_group_exists (const char *file, const char *glob)

Determine whether a group matching glob exists in an edje file.

- EAPI char * edje_file_data_get (const char *file, const char *key)

  Get edje file data.

- EAPI void edje_color_class_set (const char *color_class, int r, int g, int b, int a, int r2, int g2, int b2, int a2, int r3, int g3, int b3, int a3)

  Set Edje color class.

- EAPI void edje_color_class_del (const char *color_class)

  Parameters:

  color_class

- EAPI Evas_List * edje_color_class_list (void)

  Lists all color classes known about by the current process.

- EAPI void edje_text_class_set (const char *text_class, const char *font, Evas_Font_Size size)

  Set the Edje text class.

- EAPI void edje_text_class_del (const char *text_class)

  Parameters:

  text_class

- EAPI Evas_List * edje_text_class_list (void)

  Lists all text classes known about by the current process.

- EAPI void edje_extern_object_min_size_set (Evas_Object *obj, Evas_Coord minw, Evas_Coord minh)

  Set the object minimum size.

- EAPI void edje_extern_object_max_size_set (Evas_Object *obj, Evas_Coord maxw, Evas_Coord maxh)

  Set the object maximum size.

- EAPI void edje_extern_object_aspect_set (Evas_Object *obj, Edje_Aspect_Control aspect, Evas_Coord aw, Evas_Coord ah)

  Set the object aspect size.

- EAPI Evas_Object * edje_object_add (Evas *evas)

  Constructs the Edje object.

- EAPI const char * edje_object_data_get (Evas_Object *obj, const char *key)

  Get Edje object data.

- EAPI int edje_object_file_set (Evas_Object *obj, const char *file, const char *part)

  Sets the EET file to be used.

- EAPI void edje_object_file_get (Evas_Object *obj, const char **file, const char **part)

  Get the EET location and group for the Evas Object.

- EAPI int edje_object_load_error_get (Evas_Object *obj)

  Gets the Edje load error.

- EAPI void edje_object_signal_emit (Evas_Object *obj, const char *emission, const char *source)

  Send a signal to the Edje.

- EAPI void edje_object_play_set (Evas_Object *obj, int play)

  Set the Edje to play or pause.

- EAPI int edje_object_play_get (Evas_Object *obj)

  Get the Edje play/pause state.

- EAPI void edje_object_animation_set (Evas_Object *obj, int on)

  Set Animation state.

- EAPI int edje_object_animation_get (Evas_Object *obj)

  Get the animation state.

- EAPI int edje_object_freeze (Evas_Object *obj)

  Freeze object.

- EAPI int edje_object_thaw (Evas_Object *obj)

  Thaw object.

- EAPI void edje_object_color_class_set (Evas_Object *obj, const char *color_class, int r, int g, int b, int a, int r2, int g2, int b2, int a2, int r3, int g3, int b3, int a3)

  Sets the object color class.

- EAPI void edje_object_color_class_del (Evas_Object *obj, const char *color_class)

  Parameters:

  color_class

- EAPI void edje_object_text_class_set (Evas_Object *obj, const char *text_class, const char *font, Evas_Font_Size size)

  Sets Edje text class.

- EAPI void edje_object_size_min_get (Evas_Object *obj, Evas_Coord *minw, Evas_Coord *minh)

  Get the minimum size for an object.

- EAPI void edje_object_size_max_get (Evas_Object *obj, Evas_Coord *maxw, Evas_Coord *maxh)

  Get the maximum size for an object.

- EAPI void edje_object_calc_force (Evas_Object *obj)

  Force a Size/Geometry calculation.

- EAPI void edje_object_size_min_calc (Evas_Object *obj, Evas_Coord *minw, Evas_Coord *minh)

  Calculate minimum size.

- EAPI int edje_object_part_exists (Evas_Object *obj, const char *part)

  Check if Edje part exists.

- EAPI Evas_Object * edje_object_part_object_get (Evas_Object *obj, const char *part)

  Gets the Evas_Object corresponding to a given part.

- EAPI void edje_object_part_geometry_get (Evas_Object *obj, const char *part, Evas_Coord *x, Evas_Coord *y, Evas_Coord *w, Evas_Coord *h)

  Get Edje part geometry.

- EAPI void edje_object_part_text_set (Evas_Object *obj, const char *part, const char *text)

  Sets the text for an object part.

- EAPI const char * edje_object_part_text_get (Evas_Object *obj, const char *part)

  Returns the text of the object part.

- EAPI void edje_object_part_swallow (Evas_Object *obj, const char *part, Evas_Object *obj_swallow)

  Swallows an object into the edje.

- EAPI void edje_object_part_unswallow (Evas_Object *obj, Evas_Object *obj_swallow)

  Unswallow an object.

- EAPI Evas_Object * edje_object_part_swallow_get (Evas_Object *obj, const char *part)

  Get the swallowed part ?!

- EAPI const char * edje_object_part_state_get (Evas_Object *obj, const char *part, double *val_ret)

  Returns the state of the Edje part.

- EAPI int edje_object_part_drag_dir_get (Evas_Object *obj, const char *part)

  Determine dragable directions.

- EAPI void edje_object_part_drag_value_set (Evas_Object *obj, const char *part, double dx, double dy)

  Set the dragable object location.

- EAPI void edje_object_part_drag_value_get (Evas_Object *obj, const char *part, double *dx, double *dy)

  Get the dragable object location.

- EAPI void edje_object_part_drag_size_set (Evas_Object *obj, const char *part, double dw, double dh)

Set the dragable object size.

- EAPI void edje_object_part_drag_size_get (Evas_Object *obj, const char *part, double *dw, double *dh)

  Get the dragable object size.

- EAPI void edje_object_part_drag_step_set (Evas_Object *obj, const char *part, double dx, double dy)

  Sets the drag step increment.

- EAPI void edje_object_part_drag_step_get (Evas_Object *obj, const char *part, double *dx, double *dy)

  Gets the drag step increment values.

- EAPI void edje_object_part_drag_page_set (Evas_Object *obj, const char *part, double dx, double dy)

  Sets the page step increments.

- EAPI void edje_object_part_drag_page_get (Evas_Object *obj, const char *part, double *dx, double *dy)

  Gets the page step increments.

- EAPI void edje_object_part_drag_step (Evas_Object *obj, const char *part, double dx, double dy)

  Steps the dragable x,y steps.

- EAPI void edje_object_part_drag_page (Evas_Object *obj, const char *part, double dx, double dy)

  Pages x,y steps.

- EAPI void edje_object_signal_callback_add (Evas_Object *obj, const char *emission, const char *source, void(*func)(void *data, Evas_Object *o, const char *emission, const char *source), void *data)

  Adds a callback for the object.

- EAPI void * edje_object_signal_callback_del (Evas_Object *obj, const char *emission, const char *source, void(*func)(void *data, Evas_Object *o, const char *emission, const char *source))

  Delete an object's callback.

4.1.1 Detailed Description

Edje Graphical Design Library.

These routines are used for Edje.

4.1.2 Function Documentation

4.1.2.1 void edje_color_class_del (const char * *color_class*)

Parameters:

color_class

Deletes any values at the process level for the specified color class.

4.1.2.2 Evas_List * edje_color_class_list (void)

Lists all color classes known about by the current process.

Returns:

A list of color class names (strings). These strings and the list must be free()'d by the caller.

4.1.2.3 EAPI void edje_color_class_set (const char * *color_class*, int *r*, int *g*, int *b*, int *a*, int *r2*, int *g2*, int *b2*, int *a2*, int *r3*, int *g3*, int *b3*, int *a3*)

Set Edje color class.

Parameters:

color_class

r Object Red value

g Object Green value

b Object Blue value

a Object Alpha value

r2 Outline Red value

g2 Outline Green value

b2 Outline Blue value

a2 Outline Alpha value

r3 Shadow Red value

g3 Shadow Green value

b3 Shadow Blue value

a3 Shadow Alpha value

Sets the color values for a process level color class. This will cause all edje parts in the current process that have the specified color class to have their colors multiplied by these values. (Object level color classes set by edje_object_color_class_set() will override the values set by this function).

The first color is the object, the second is the text outline, and the third is the text shadow. (Note that the second two only apply to text parts)

4.1.2.4 EAPI void edje_extern_object_aspect_set (Evas_Object * *obj*, Edje_Aspect_Control *aspect*, Evas_Coord *aw*, Evas_Coord *ah*)

Set the object aspect size.

Parameters:

obj A valid Evas_Object handle

aspect The aspect control axes

aw The aspect radio width

ah The aspect ratio height

This sets the desired aspect ratio to keep an object that will be swallowed by Edje. The width and height define a preferred size ASPECT and the object may be scaled to be larger or smaller, but retaining the relative scale of both aspwct width and height.

4.1.2.5 EAPI void edje_extern_object_max_size_set (Evas_Object * *obj*, Evas_Coord *maxw*, Evas_Coord *maxh*)

Set the object maximum size.

Parameters:

obj A valid Evas_Object handle

maxw The maximum width

maxh The maximum height

This sets the maximum size restriction for the object.

4.1.2.6 EAPI void edje_extern_object_min_size_set (Evas_Object * *obj*, Evas_Coord *minw*, Evas_Coord *minh*)

Set the object minimum size.

Parameters:

obj A valid Evas_Object handle

minw The minimum width

minh The minimum height

This sets the minimum size restriction for the object.

4.1.2.7 EAPI Evas_List * edje_file_collection_list (const char * *file*)

Get the collection list from the edje file ?

Parameters:

file The file path?

Returns:

The Evas_List of files

4.1.2.8 EAPI void edje_file_collection_list_free (Evas_List * *lst*)

Free file collection.

Parameters:

lst The Evas_List of files

Frees the file collection.

4.1.2.9 EAPI char * edje_file_data_get (const char * *file*, const char * *key*)

Get edje file data.

Parameters:

file The file

key The data key

Returns:

The file data string

4.1.2.10 EAPI int edje_file_group_exists (const char * *file*, const char * *glob*)

Determine whether a group matching glob exists in an edje file.

Parameters:

file The file path

glob A glob to match on

Returns:

1 if a match is found, 0 otherwise

4.1.2.11 EAPI double edje_frametime_get (void)

Get the frametime.

Returns:

The frametime

Returns the frametime in seconds, by default this is 1/30.

4.1.2.12 EAPI void edje_frametime_set (double *t*)

Set the frametime.

Parameters:

t The frametime

Sets the frametime in seconds, by default this is 1/30.

4.1.2.13 EAPI int edje_init (void)

Initialize the EDJE library.

Returns:

The new init count.

4.1.2.14 EAPI Evas_Object * edje_object_add (Evas * *evas*)

Constructs the Edje object.

Parameters:

evas A valid Evas handle

Returns:

The Evas_Object pointer.

Creates the Edje smart object, returning the Evas_Object handle.

4.1.2.15 EAPI int edje_object_animation_get (Evas_Object * *obj*)

Get the animation state.

Parameters:

obj A valid Evas_Object handle

Returns:

0 on Error or if not animated
1 if animated

4.1.2.16 EAPI void edje_object_animation_set (Evas_Object * *obj*, int *on*)

Set Animation state.

Parameters:

obj A valid Evas_Object handle

on Animation State

Stop or start an Edje animation.

4.1.2.17 EAPI void edje_object_calc_force (Evas_Object * *obj*)

Force a Size/Geometry calculation.

Parameters:

obj A valid Evas_Object handle

Forces the object `obj` to recalculation layout regardless of freeze/thaw.

4.1.2.18 void edje_object_color_class_del (Evas_Object * *obj*, const char * *color_class*)

Parameters:

color_class

Deletes any values at the object level for the specified object and color class.

4.1.2.19 EAPI void edje_object_color_class_set (Evas_Object * *obj*, const char * *color_class*, int *r*, int *g*, int *b*, int *a*, int *r2*, int *g2*, int *b2*, int *a2*, int *r3*, int *g3*, int *b3*, int *a3*)

Sets the object color class.

Parameters:

obj A valid Evas_Object handle

color_class

r Object Red value

g Object Green value

b Object Blue value

a Object Alpha value

r2 Outline Red value

g2 Outline Green value

b2 Outline Blue value

a2 Outline Alpha value

r3 Shadow Red value

g3 Shadow Green value

b3 Shadow Blue value

a3 Shadow Alpha value

Sets the color values for an object level color class. This will cause all edje parts in the specified object that have the specified color class to have their colors multiplied by these values.

The first color is the object, the second is the text outline, and the third is the text shadow. (Note that the second two only apply to text parts)

4.1.2.20 EAPI const char * edje_object_data_get (Evas_Object * *obj*, const char * *key*)

Get Edje object data.

Parameters:

obj A valid Evas_Object handle

key The data key

Returns:

The data string

4.1.2.21 EAPI void edje_object_file_get (Evas_Object * *obj*, const char ** *file*, const char ** *part*)

Get the EET location and group for the Evas Object.

?! Assuming eet file

Parameters:

obj A valid Evas_Object handle

file The EET file location pointer

part The EET part pointer

This gets the EET file location and group for the given Evas_Object.

4.1.2.22 EAPI int edje_object_file_set (Evas_Object * *obj*, const char * *file*, const char * *part*)

Sets the EET file to be used.

Parameters:

obj A valid Evas_Object handle

file The path to the EET file

part The group name in the Edje

Returns:

0 on Error
1 on Success and sets EDJE_LOAD_ERROR_NONE

This loads the EET file and sets up the Edje.

4.1.2.23 EAPI int edje_object_freeze (Evas_Object * *obj*)

Freeze object.

Parameters:

obj A valid Evas_Object handle

Returns:

The frozen state
0 on Error

This puts all changes on hold. Successive freezes will nest, requiring an equal number of thaws.

4.1.2.24 EAPI int edje_object_load_error_get (Evas_Object * *obj*)

Gets the Edje load error.

Parameters:

obj A valid Evas_Object handle

Returns:

The Edje load error:
0: No Error
1: Generic Error
2: Does not Exist
3: Permission Denied
4: Resource Allocation Failed
5: Corrupt File
6: Unknown Format
7: Incompatible File

4.1.2.25 EAPI int edje_object_part_drag_dir_get (Evas_Object * *obj*, const char * *part*)

Determine dragable directions.

Parameters:

obj A valid Evas_Object handle

part The part name

Returns:

0: Not dragable
1: Dragable in X direction
2: Dragable in Y direction
3: Dragable in X & Y directions

4.1.2.26 EAPI void edje_object_part_drag_page (Evas_Object * *obj*, const char * *part*, double *dx*, double *dy*)

Pages x,y steps.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The x step

dy The y step

Pages x,y where the increment is defined by edje_object_part_drag_page_set.

WARNING: Paging is bugged!

4.1.2.27 EAPI void edje_object_part_drag_page_get (Evas_Object * *obj*, const char * *part*, double * *dx*, double * *dy*)

Gets the page step increments.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The dx page increment pointer

dy The dy page increment pointer

Gets the x,y page step increments for the dragable object.

4.1.2.28 EAPI void edje_object_part_drag_page_set (Evas_Object * *obj*, const char * *part*, double *dx*, double *dy*)

Sets the page step increments.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The x page step increment

dy The y page step increment

Sets the x,y page step increment values.

4.1.2.29 EAPI void edje_object_part_drag_size_get (Evas_Object * *obj*, const char * *part*, double * *dw*, double * *dh*)

Get the dragable object size.

Parameters:

obj A valid Evas_Object handle

part The part name

dw The drag width pointer

dh The drag height pointer

Gets the dragable object size.

4.1.2.30 EAPI void edje_object_part_drag_size_set (Evas_Object * *obj*, const char * *part*, double *dw*, double *dh*)

Set the dragable object size.

Parameters:

obj A valid Evas_Object handle

part The part name

dw The drag width

dh The drag height

Sets the size of the dragable object.

4.1.2.31 EAPI void edje_object_part_drag_step (Evas_Object * *obj*, const char * *part*, double *dx*, double *dy*)

Steps the dragable x,y steps.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The x step

dy The y step

Steps x,y where the step increment is the amount set by edje_object_part_drag_step_set.

4.1.2.32 EAPI void edje_object_part_drag_step_get (Evas_Object * *obj*, const char * *part*, double * *dx*, double * *dy*)

Gets the drag step increment values.

Parameters:

obj A valid Evas_Object handle

part The part

dx The x step increment pointer

dy The y step increment pointer

Gets the x and y step increments for the dragable object.

4.1.2.33 EAPI void edje_object_part_drag_step_set (Evas_Object * *obj*, const char * *part*, double *dx*, double *dy*)

Sets the drag step increment.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The x step ammount

dy The y step ammount

Sets the x,y step increments for a dragable object.

4.1.2.34 EAPI void edje_object_part_drag_value_get (Evas_Object * *obj*, const char * *part*, double * *dx*, double * *dy*)

Get the dragable object location.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The X value pointer

dy The Y value pointer

Gets the drag location values.

4.1.2.35 EAPI void edje_object_part_drag_value_set (Evas_Object * *obj*, const char * *part*, double *dx*, double *dy*)

Set the dragable object location.

Parameters:

obj A valid Evas_Object handle

part The part name

dx The x value

dy The y value

Places the dragable object at the given location.

4.1.2.36 EAPI int edje_object_part_exists (Evas_Object * *obj*, const char * *part*)

Check if Edje part exists.

Parameters:

obj A valid Evas_Object handle

part The part name to check

Returns:

0 on Error
1 if Edje part exists

4.1.2.37 EAPI void edje_object_part_geometry_get (Evas_Object * *obj*, const char * *part*, Evas_Coord * *x*, Evas_Coord * *y*, Evas_Coord * *w*, Evas_Coord * *h*)

Get Edje part geometry.

Parameters:

obj A valid Evas_Object handle

part The Edje part

x The x coordinate pointer

y The y coordinate pointer

w The width pointer

h The height pointer

Gets the Edje part geometry

4.1.2.38 EAPI Evas_Object * edje_object_part_object_get (Evas_Object * *obj*, const char * *part*)

Gets the Evas_Object corresponding to a given part.

You should never modify the state of the returned object (with evas_object_move() or evas_object_hide() for example), but you can safely query infos about its current state (with evas_object_visible_get() or evas_object_color_get() for example)

Parameters:

obj A valid Evas_Object handle

part The Edje part

Returns:

Returns the Evas_Object corresponding to the given part, or NULL on failure (if the part doesn't exist)

4.1.2.39 EAPI const char * edje_object_part_state_get (Evas_Object * *obj*, const char * *part*, double * *val_ret*)

Returns the state of the Edje part.

Parameters:

obj A valid Evas_Objectart handle

part The part name

val_ret

Returns:

The part state:
"default" for the default state
"" for other states

4.1.2.40 EAPI void edje_object_part_swallow (Evas_Object * *obj*, const char * *part*, Evas_Object * *obj_swallow*)

Swallows an object into the edje.

Parameters:

obj A valid Evas_Object handle

part The part name

obj_swallow The object to swallow

Swallows the object into the edje part so that all geometry changes for the part affect the swallowed object. (e.g. resize, move, show, raise/lower, etc.).

4.1.2.41 EAPI Evas_Object * edje_object_part_swallow_get (Evas_Object * *obj*, const char * *part*)

Get the swallowed part ?!

Parameters:

obj A valid Evas_Object handle

part The part name

Returns:

The swallowed object

4.1.2.42 EAPI const char * edje_object_part_text_get (Evas_Object * *obj*, const char * *part*)

Returns the text of the object part.

Parameters:

obj A valid Evas_Object handle

part The part name

Returns:

The text string

4.1.2.43 EAPI void edje_object_part_text_set (Evas_Object * *obj*, const char * *part*, const char * *text*)

Sets the text for an object part.

Parameters:

obj A valid Evas Object handle

part The part name

text The text string

4.1.2.44 EAPI void edje_object_part_unswallow (Evas_Object * *obj*, Evas_Object * *obj_swallow*)

Unswallow an object.

Parameters:

obj A valid Evas_Object handle

obj_swallow The swallowed object

Causes the edje to regurgitate a previously swallowed object. :)

4.1.2.45 EAPI int edje_object_play_get (Evas_Object * *obj*)

Get the Edje play/pause state.

Parameters:

obj A valid Evas_Object handle

Returns:

0 if Edje not connected, Edje delete_me, or Edje paused
1 if Edje set to play

4.1.2.46 EAPI void edje_object_play_set (Evas_Object * *obj*, int *play*)

Set the Edje to play or pause.

Parameters:

obj A valid Evas_Object handle

play Play instruction (1 to play, 0 to pause)

This sets the Edje to play or pause depending on the parameter. This has no effect if the Edje is already in that state.

4.1.2.47 EAPI void edje_object_signal_callback_add (Evas_Object * *obj*, const char * *emission*, const char * *source*, void(*)(void *data, Evas_Object *o, const char *emission, const char *source) *func*, void * *data*)

Adds a callback for the object.

Parameters:

obj A valid Evas_Object handle

emission Signal to activate callback FIXDOC: Naming Convention?

source Source of signal

func The function to be executed when the callback is signaled

data ? FIXDOC

Creates a callback for the object to execute the given function.

4.1.2.48 EAPI void* edje_object_signal_callback_del (Evas_Object * *obj*, const char * *emission*, const char * *source*, void(*)(void *data, Evas_Object *o, const char *emission, const char *source) *func*)

Delete an object's callback.

Parameters:

obj A valid Evas_Object handle

emission ? FIXDOC

source ? FIXDOC

func ? FIXDOC

Deletes an existing callback

4.1.2.49 EAPI void edje_object_signal_emit (Evas_Object * *obj*, const char * *emission*, const char * *source*)

Send a signal to the Edje.

Parameters:

obj A vaild Evas_Object handle

emission The signal

source The signal source

This sends a signal to the edje. These are defined in the programs section of an edc.

4.1.2.50 EAPI void edje_object_size_max_get (Evas_Object * *obj*, Evas_Coord * *maxw*, Evas_Coord * *maxh*)

Get the maximum size for an object.

Parameters:

obj A valid Evas_Object handle

maxw Maximum width pointer

maxh Maximum height pointer

Gets the object's maximum size values from the Edje. These are set to zero if no Edje is connected to the Evas Object.

4.1.2.51 EAPI void edje_object_size_min_calc (Evas_Object * *obj*, Evas_Coord * *minw*, Evas_Coord * *minh*)

Calculate minimum size.

Parameters:

obj A valid Evas_Object handle

minw Minimum width pointer

minh Minimum height pointer

Calculates the object's minimum size ?!

4.1.2.52 EAPI void edje_object_size_min_get (Evas_Object * *obj*, Evas_Coord * *minw*, Evas_Coord * *minh*)

Get the minimum size for an object.

Parameters:

obj A valid Evas_Object handle

minw Minimum width pointer

minh Minimum height pointer

Gets the object's minimum size values from the Edje. These are set to zero if no Edje is connected to the Evas Object.

4.1.2.53 EAPI void edje_object_text_class_set (Evas_Object * *obj*, const char * *text_class*, const char * *font*, Evas_Font_Size *size*)

Sets Edje text class.

Parameters:

obj A valid Evas_Object handle

text_class The text class name

font Font name

size Font Size

Sets the text class for the Edje.

4.1.2.54 EAPI int edje_object_thaw (Evas_Object * *obj*)

Thaw object.

Parameters:

obj A valid Evas_Object handle

Returns:

The frozen state
0 on Error

This allows frozen changes to occur.

4.1.2.55 EAPI int edje_shutdown (void)

Shutdown the EDJE library.

Returns:

The new init count.

4.1.2.56 void edje_text_class_del (const char * *text_class*)

Parameters:

text_class

Deletes any values at the process level for the specified text class.

4.1.2.57 Evas_List * edje_text_class_list (void)

Lists all text classes known about by the current process.

Returns:

A list of text class names (strings). These strings are stringshares and the list must be free()'d by the caller.

4.1.2.58 EAPI void edje_text_class_set (const char * *text_class*, const char * *font*, Evas_Font_Size *size*)

Set the Edje text class.

Parameters:

text_class The text class name ?!
font The font name
size The font size

This sets updates all edje members which belong to this text class with the new font attributes.

Chapter 5

Edje Page Documentation

5.1 Todo List

page Edje Library Documentation See src/lib/edje_private.h for a list of FIXME's

page Edje Library Documentation Complete documentation of API

page Edje Library Documentation Bytecode language for extending programs... but what/how?

Index

edje.c, 9
    edje_color_class_del, 13
    edje_color_class_list, 14
    edje_color_class_set, 14
    edje_extern_object_aspect_set, 14
    edje_extern_object_max_size_set, 15
    edje_extern_object_min_size_set, 15
    edje_file_collection_list, 15
    edje_file_collection_list_free, 15
    edje_file_data_get, 16
    edje_file_group_exists, 16
    edje_frametime_get, 16
    edje_frametime_set, 16
    edje_init, 17
    edje_object_add, 17
    edje_object_animation_get, 17
    edje_object_animation_set, 17
    edje_object_calc_force, 18
    edje_object_color_class_del, 18
    edje_object_color_class_set, 18
    edje_object_data_get, 19
    edje_object_file_get, 19
    edje_object_file_set, 19
    edje_object_freeze, 19
    edje_object_load_error_get, 20
    edje_object_part_drag_dir_get, 20
    edje_object_part_drag_page, 20
    edje_object_part_drag_page_get, 21
    edje_object_part_drag_page_set, 21
    edje_object_part_drag_size_get, 21
    edje_object_part_drag_size_set, 22
    edje_object_part_drag_step, 22
    edje_object_part_drag_step_get, 22
    edje_object_part_drag_step_set, 22
    edje_object_part_drag_value_get, 23
    edje_object_part_drag_value_set, 23
    edje_object_part_exists, 23
    edje_object_part_geometry_get, 24
    edje_object_part_object_get, 24
    edje_object_part_state_get, 24
    edje_object_part_swallow, 25
    edje_object_part_swallow_get, 25
    edje_object_part_text_get, 25
    edje_object_part_text_set, 25
    edje_object_part_unswallow, 26
    edje_object_play_get, 26
    edje_object_play_set, 26
    edje_object_signal_callback_add, 26
    edje_object_signal_callback_del, 27
    edje_object_signal_emit, 27
    edje_object_size_max_get, 27
    edje_object_size_min_calc, 28
    edje_object_size_min_get, 28
    edje_object_text_class_set, 28
    edje_object_thaw, 28
    edje_shutdown, 29
    edje_text_class_del, 29
    edje_text_class_list, 29
    edje_text_class_set, 29
edje_color_class_del
    edje.c, 13
edje_color_class_list
    edje.c, 14
edje_color_class_set
    edje.c, 14
edje_extern_object_aspect_set
    edje.c, 14
edje_extern_object_max_size_set
    edje.c, 15
edje_extern_object_min_size_set
    edje.c, 15
edje_file_collection_list
    edje.c, 15
edje_file_collection_list_free
    edje.c, 15
edje_file_data_get
    edje.c, 16
edje_file_group_exists
    edje.c, 16
edje_frametime_get
    edje.c, 16
edje_frametime_set
    edje.c, 16
edje_init
    edje.c, 17
edje_object_add
    edje.c, 17
edje_object_animation_get
    edje.c, 17
edje_object_animation_set

edje.c, 17
edje_object_calc_force
edje.c, 18
edje_object_color_class_del
edje.c, 18
edje_object_color_class_set
edje.c, 18
edje_object_data_get
edje.c, 19
edje_object_file_get
edje.c, 19
edje_object_file_set
edje.c, 19
edje_object_freeze
edje.c, 19
edje_object_load_error_get
edje.c, 20
edje_object_part_drag_dir_get
edje.c, 20
edje_object_part_drag_page
edje.c, 20
edje_object_part_drag_page_get
edje.c, 21
edje_object_part_drag_page_set
edje.c, 21
edje_object_part_drag_size_get
edje.c, 21
edje_object_part_drag_size_set
edje.c, 22
edje_object_part_drag_step
edje.c, 22
edje_object_part_drag_step_get
edje.c, 22
edje_object_part_drag_step_set
edje.c, 22
edje_object_part_drag_value_get
edje.c, 23
edje_object_part_drag_value_set
edje.c, 23
edje_object_part_exists
edje.c, 23
edje_object_part_geometry_get
edje.c, 24
edje_object_part_object_get
edje.c, 24
edje_object_part_state_get
edje.c, 24
edje_object_part_swallow
edje.c, 25
edje_object_part_swallow_get
edje.c, 25
edje_object_part_text_get
edje.c, 25
edje_object_part_text_set
edje.c, 25
edje_object_part_unswallow
edje.c, 26
edje_object_play_get
edje.c, 26
edje_object_play_set
edje.c, 26
edje_object_signal_callback_add
edje.c, 26
edje_object_signal_callback_del
edje.c, 27
edje_object_signal_emit
edje.c, 27
edje_object_size_max_get
edje.c, 27
edje_object_size_min_calc
edje.c, 28
edje_object_size_min_get
edje.c, 28
edje_object_text_class_set
edje.c, 28
edje_object_thaw
edje.c, 28
edje_shutdown
edje.c, 29
edje_text_class_del
edje.c, 29
edje_text_class_list
edje.c, 29
edje_text_class_set
edje.c, 29