Ogród zoologiczny

Generated by Doxygen 1.8.14

Contents

| 1 | Data | Struct | ure Index | | 1 |
|---|------|----------|-------------|------------------------|------|
| | 1.1 | Data S | Structures | | . 1 |
| 2 | File | Index | | | 3 |
| | 2.1 | File Lis | st | | . 3 |
| 3 | Data | Struct | ure Docui | mentation | 5 |
| | 3.1 | HttpRe | esponse S | Struct Reference | . 5 |
| | | 3.1.1 | Detailed | I Description | . 5 |
| | | 3.1.2 | Field Do | ocumentation | . 5 |
| | | | 3.1.2.1 | code | . 6 |
| | | | 3.1.2.2 | data | . 6 |
| | | | 3.1.2.3 | size | . 6 |
| 4 | File | Docum | entation | | 7 |
| | 4.1 | include | e/main_wii | ndow.h File Reference | . 7 |
| | | 4.1.1 | Function | n Documentation | . 8 |
| | | | 4.1.1.1 | main_window_new() | . 8 |
| | 4.2 | include | e/services/ | /http.h File Reference | . 9 |
| | | 4.2.1 | Typedef | Documentation | . 9 |
| | | | 4.2.1.1 | HttpResponse | . 9 |
| | | 4.2.2 | Function | Documentation | . 9 |
| | | | 4.2.2.1 | http_get() | . 9 |
| | | | 4.2.2.2 | write_function() | . 11 |
| | 43 | src/ma | in c File B | Reference | 11 |

ii CONTENTS

| | 4.3.1 | Function | Documentation | . 12 |
|-------|---------|-------------|---------------------|------|
| | | 4.3.1.1 | activate() | . 12 |
| | | 4.3.1.2 | main() | . 13 |
| 4.4 | src/ma | in_window | v.c File Reference | . 13 |
| | 4.4.1 | Function | Documentation | . 14 |
| | | 4.4.1.1 | main_window_new() | . 14 |
| 4.5 | src/sei | vices/http. | .c File Reference | . 15 |
| | 4.5.1 | Function | Documentation | . 16 |
| | | 4.5.1.1 | http_get() | . 16 |
| | | 4.5.1.2 | http_response_new() | . 17 |
| | | 4.5.1.3 | write_function() | . 18 |
| Index | | | | 19 |

Chapter 1

Data Structure Index

| 1 | .1 | Data | Stru | ctu | 29 |
|-----|-----|------|------|-------|----|
| - 1 | - 1 | Data | JULU | ILLUI | |

| Here are the data structures with brief descriptions: | |
|---|---|
| HttpResponse | Ę |

2 Data Structure Index

Chapter 2

File Index

2.1 File List

Here is a list of all files with brief descriptions:

| iclude/main_window.h | |
|-----------------------|----|
| clude/services/http.h | |
| rc/main.c | 1 |
| rc/main_window.c | 10 |
| rc/services/http.c | 1! |

File Index

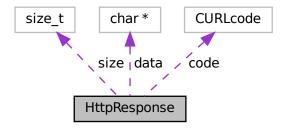
Chapter 3

Data Structure Documentation

3.1 HttpResponse Struct Reference

#include <http.h>

Collaboration diagram for HttpResponse:



Data Fields

- char * data
- size_t size
- CURLcode code

3.1.1 Detailed Description

Definition at line 8 of file http.h.

3.1.2 Field Documentation

3.1.2.1 code

CURLcode HttpResponse::code

Definition at line 11 of file http.h.

Referenced by http_get().

3.1.2.2 data

char* HttpResponse::data

Definition at line 9 of file http.h.

Referenced by http_get(), http_response_new(), and write_function().

3.1.2.3 size

size_t HttpResponse::size

Definition at line 10 of file http.h.

Referenced by http_response_new(), and write_function().

The documentation for this struct was generated from the following file:

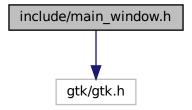
• include/services/http.h

Chapter 4

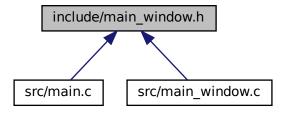
File Documentation

4.1 include/main_window.h File Reference

#include <gtk/gtk.h>
Include dependency graph for main_window.h:



This graph shows which files directly or indirectly include this file:



Functions

GtkWidget * main_window_new (GtkApplication *app)

4.1.1 Function Documentation

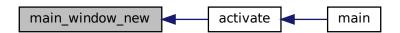
4.1.1.1 main_window_new()

```
\begin{tabular}{ll} $\tt GtkMidget*\ main\_window\_new & ( \\ & & \tt GtkApplication\ *\ app \ ) \end{tabular}
```

Definition at line 7 of file main window.c.

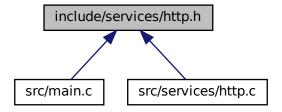
Referenced by activate().

```
8
       GtkWidget *window;
      GtkWidget *mainContainer;
10
        window = gtk_application_window_new (app);
gtk_window_set_title (GTK_WINDOW (window), "Ogród zoologiczny");
11
12
        gtk_window_set_default_size (GTK_WINDOW (window), 500, 500);
13
14
15
        mainContainer = gtk_grid_new();
16
        gtk_container_add(GTK_CONTAINER (window), GTK_WIDGET (mainContainer));
17
18
        GtkWidget *containerTable;
19
        containerTable = gtk scrolled window new(NULL, NULL);
        gtk_widget_set_hexpand(containerTable, 1);
20
21
        gtk_widget_set_vexpand(containerTable, 1);
22
        GtkWidget *table;
23
        table = gtk\_grid\_new();
        gtk_container_add(GTK_CONTAINER(containerTable), GTK_WIDGET(table));
24
        for(int i=0; i<200; ++i){
   GtkWidget *label = gtk_label_new("Test");</pre>
25
26
            gtk_label_set_xalign(GTK_LABEL (label), 0.0);
28
            gtk_widget_set_margin_start(label, 5);
29
             gtk_widget_set_margin_top(label, 5);
30
            gtk_widget_set_hexpand(label, TRUE);
31
            gtk_grid_attach(GTK_GRID (table), label, 0, i, 1, 1);
32
33
34
        \tt gtk\_grid\_attach(GTK\_GRID \ (mainContainer), \ GTK\_WIDGET \ (containerTable), \ 0, \ 1, \ 1, \ 1);
35
36
        GtkWidget *buttonAddAnimal;
        buttonAddAnimal = gtk_button_new();
gtk_button_set_label(GTK_BUTTON (buttonAddAnimal), "Dodaj zwierzę");
37
38
        gtk_grid_attach(GTK_GRID (mainContainer), GTK_WIDGET (buttonAddAnimal), 0, 2, 1, 1); gtk_widget_set_hexpand(buttonAddAnimal, 1);
39
40
        gtk_widget_show_all (window);
42 }
```



4.2 include/services/http.h File Reference

This graph shows which files directly or indirectly include this file:



Data Structures

struct HttpResponse

Typedefs

• typedef struct HttpResponse HttpResponse

Functions

```
    size_t write_function (void *ptr, size_t size, size_t nmemb, HttpResponse *r)
    HttpResponse * http_get (char *url)
    Gets HTTP response for url.
```

4.2.1 Typedef Documentation

4.2.1.1 HttpResponse

```
typedef struct HttpResponse HttpResponse
```

4.2.2 Function Documentation

4.2.2.1 http_get()

Gets HTTP response for url.

Parameters

url Site address

Returns

Filled HttpResponse

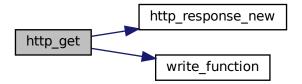
Definition at line 33 of file http.c.

References HttpResponse::code, HttpResponse::data, http_response_new(), and write_function().

Referenced by main().

```
35
        HttpResponse * res = http_response_new();
36
37
        curl = curl_easy_init();
        if(curl) {
38
39
             puts(url);
             /* example.com is redirected, so we tell libourl to follow redirection */
40
41
             curl_easy_setopt(curl, CURLOPT_FOLLOWLOCATION, 1L);
curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, write_function);
curl_easy_setopt(curl, CURLOPT_WRITEDATA, res);
42
43
44
45
46
             /* Perform the request, res will get the return code */
             res->code = curl_easy_perform(curl);
48
             /\star Check for errors \star/
             if(res->code != CURLE_OK)
    fprintf(stderr, "curl_easy_perform() failed: %s\n",
49
50
                            curl_easy_strerror(res->code));
51
                  printf("Data: %s\n", res->data);
55
             /* always cleanup */
56
57
             curl_easy_cleanup(curl);
58
        return res;
60 }
```

Here is the call graph for this function:





4.2.2.2 write_function()

Definition at line 12 of file http.c.

References HttpResponse::data, and HttpResponse::size.

Referenced by http_get().

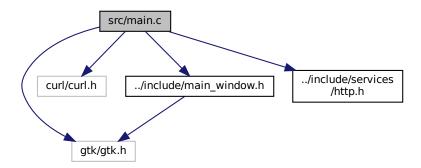
```
12
13     size_t new_len = r->size + size*nmemb;
14     r->data= realloc(r->data, new_len+1);
15     memcpy(r->data+r->size, ptr, size*nmemb);
16     r->data[new_len] = '\0';
17     return size*nmemb;
18 }
```

Here is the caller graph for this function:



4.3 src/main.c File Reference

```
#include <gtk/gtk.h>
#include <curl/curl.h>
#include "../include/main_window.h"
#include "../include/services/http.h"
Include dependency graph for main.c:
```



Functions

```
• static void activate (GtkApplication *app, gpointer user_data) 
Initialize UI.
```

• int main (int argc, char **argv)

4.3.1 Function Documentation

4.3.1.1 activate()

Initialize UI.

Definition at line 10 of file main.c.

References main_window_new().

Referenced by main().

```
11 {
12    main_window_new(app);
13 }
```

Here is the call graph for this function:





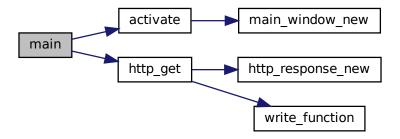
4.3.1.2 main()

```
int main (
          int argc,
          char ** argv )
```

Definition at line 15 of file main.c.

References activate(), and http_get().

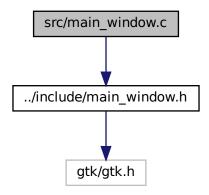
Here is the call graph for this function:



4.4 src/main_window.c File Reference

#include "../include/main_window.h"

Include dependency graph for main_window.c:



Functions

GtkWidget * main window new (GtkApplication *app)

4.4.1 Function Documentation

4.4.1.1 main_window_new()

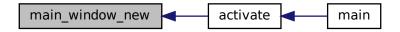
Definition at line 7 of file main_window.c.

Referenced by activate().

```
8
        GtkWidget *window;
        GtkWidget *mainContainer;
10
          window = gtk_application_window_new (app);
gtk_window_set_title (GTK_WINDOW (window), "Ogród zoologiczny");
gtk_window_set_default_size (GTK_WINDOW (window), 500, 500);
11
12
13
14
15
          mainContainer = gtk_grid_new();
16
          \verb|gtk_container_add(GTK_CONTAINER (window), GTK_WIDGET (mainContainer))|;\\
17
          GtkWidget *containerTable;
18
          containerTable = gtk_scrolled_window_new(NULL, NULL);
gtk_widget_set_hexpand(containerTable, 1);
19
20
21
          gtk_widget_set_vexpand(containerTable, 1);
          GtkWidget *table;
          table = gtk_grid_new();
23
          gtk_container_add(GTK_CONTAINER(containerTable), GTK_WIDGET(table));
for(int i=0; i<200; ++i){
    GtkWidget *label = gtk_label_new("Test");</pre>
24
25
26
               gtk_label_set_xalign(GTK_LABEL (label), 0.0);
```

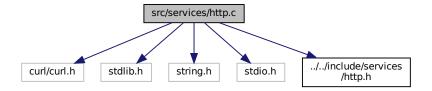
```
28
            gtk_widget_set_margin_start(label, 5);
             gtk_widget_set_margin_top(label, 5);
30
             gtk_widget_set_hexpand(label, TRUE);
31
             {\tt gtk\_grid\_attach(GTK\_GRID\ (table),\ label,\ 0,\ i,\ 1,\ 1);}
32
33
        gtk_grid_attach(GTK_GRID (mainContainer), GTK_WIDGET (containerTable), 0, 1, 1, 1);
34
35
36
        GtkWidget *buttonAddAnimal;
37
        buttonAddAnimal = gtk_button_new();
        gtk_button_set_label(GTK_BUTTON (buttonAddAnimal), "Dodaj zwierze");
gtk_grid_attach(GTK_GRID (mainContainer), GTK_WIDGET (buttonAddAnimal), 0, 2, 1, 1);
38
39
        gtk_widget_set_hexpand(buttonAddAnimal, 1);
40
        gtk_widget_show_all (window);
42 }
```

Here is the caller graph for this function:



4.5 src/services/http.c File Reference

```
#include <curl/curl.h>
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "../../include/services/http.h"
Include dependency graph for http.c:
```



Functions

- size_t write_function (void *ptr, size_t size, size_t nmemb, HttpResponse *r)
- HttpResponse * http_response_new ()
- HttpResponse * http_get (char *url)

Gets HTTP response for url.

4.5.1 Function Documentation

4.5.1.1 http_get()

Gets HTTP response for url.

Parameters

url Site address

Returns

Filled HttpResponse

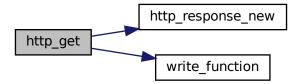
Definition at line 33 of file http.c.

References HttpResponse::code, HttpResponse::data, http_response_new(), and write_function().

Referenced by main().

```
33
34
35
         HttpResponse * res = http_response_new();
37
         curl = curl_easy_init();
         if(curl) {
38
               puts(url);
39
              puts(url);
curl_easy_setopt(curl, CURLOPT_URL, url);
/* example.com is redirected, so we tell libcurl to follow redirection */
curl_easy_setopt(curl, CURLOPT_FOLLOWLOCATION, 1L);
curl_easy_setopt(curl, CURLOPT_WRITEFUNCTION, write_function);
40
41
42
43
               curl_easy_setopt(curl, CURLOPT_WRITEDATA, res);
44
45
               /\star Perform the request, res will get the return code \star/
46
              res->code = curl_easy_perform(curl);
47
              /* Check for errors */
if(res->code != CURLE_OK)
fprintf(stderr, "curl_easy_perform() failed: %s\n",
48
49
50
51
                                curl_easy_strerror(res->code));
               else {
52
                   printf("Data: %s\n", res->data);
56
               /* always cleanup */
               curl_easy_cleanup(curl);
57
58
59
         return res;
```

Here is the call graph for this function:



Here is the caller graph for this function:



4.5.1.2 http_response_new()

```
HttpResponse* http_response_new ( )
```

Definition at line 20 of file http.c.

References HttpResponse::data, and HttpResponse::size.

Referenced by http_get().



4.5.1.3 write_function()

Definition at line 12 of file http.c.

References HttpResponse::data, and HttpResponse::size.

Referenced by http_get().

```
12
13     size_t new_len = r->size + size*nmemb;
14     r->data= realloc(r->data, new_len+1);
15     memcpy(r->data+r->size, ptr, size*nmemb);
16     r->data[new_len] = '\0';
17     return size*nmemb;
18 }
```



Index

http.c, 17 http.h, 11

| activate main.c, 12 |
|---|
| code HttpResponse, 5 |
| data HttpResponse, 6 |
| http.c |
| http_get, 16 http_response_new, 17 write_function, 17 |
| http.h |
| http_get, 9 HttpResponse, 9 write_function, 11 |
| http_get http.c, 16 http.h, 9 |
| http_response_new |
| HttpResponse, 5 code, 5 data, 6 |
| http.h, 9 size, 6 |
| include/main_window.h, 7 include/services/http.h, 9 |
| main |
| main.c, 12 main.c |
| activate, 12 |
| main, 12 |
| main_window.c main_window_new, 14 |
| main_window.h |
| main_window_new, 8 |
| main_window_new main_window.c, 14 |
| main_window.t, 8 |
| size |
| HttpResponse, 6 src/main.c, 11 src/main_window.c, 13 src/services/http.c, 15 |
| write_function |