Universal Electro Quiz Solver

Amadeusz Drożdż

Chapter 1

project-sem2

Chapter 2

Class Index

2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 3

File Index

3.1 File List

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

Chapter 4

Class Documentation

4.1 CalculationResult Struct Reference

```
#include <MathEquationsParser.h>
```

Public Attributes

- CalculationStatus **status**
- double **result**
- int **errorPosition**

4.1.1 Detailed Description

The results of calculations

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

- MathEquationsParser.h

4.2 cJSON Struct Reference

Public Attributes

- struct cJSON ∗ **next**
- struct cJSON ∗ **prev**
- struct cJSON ∗ **child**
- int **type**
- char ∗ **valuestring**
- int **valueint**
- double **valuedouble**
- char ∗ **string**

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

- cJSON.h

4.3 cJSON_Hooks Struct Reference

Public Member Functions

- void ∗CJSON_CDECL ∗ **malloc_fn** (size_t sz)
- **void** (CJSON_CDECL ∗free_fn)(void ∗ptr)

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

- cJSON.h

4.4 error Struct Reference

Public Attributes

- const unsigned char ∗ **json**
- size_t **position**

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

- cJSON.c

4.5 HTTP_HEADER Struct Reference

Public Attributes

- char **method** [8]
- int **status**
- char **content_type** [H_FIELD_SIZE]
- long **content_length**
- BOOL **chunked**
- BOOL **close**
- char **location** [H_FIELD_SIZE]
- char **referrer** [H_FIELD_SIZE]
- char **cookie** [H_FIELD_SIZE]
- char **boundary** [H_FIELD_SIZE]

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

- https.h

4.6 HTTP_INFO Struct Reference

Public Attributes

- HTTP_URL **url**
- HTTP_HEADER **request**
- HTTP_HEADER **response**
- HTTP_SSL **tls**
- long **length**
- char **r_buf** [H_READ_SIZE]
- long **r_len**
- BOOL **header_end**
- char ∗ **body**
- long **body_size**
- long **body_len**

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

- https.h

4.7 HTTP_SSL Struct Reference

Public Attributes

- BOOL **verify**
- mbedtls_net_context **ssl_fd**
- mbedtls_entropy_context **entropy**
- mbedtls_ctr_drbg_context **ctr_drbg**
- mbedtls_ssl_context **ssl**
- mbedtls_ssl_config **conf**
- mbedtls_x509_crt **cacert**

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

- https.h

4.8 HTTP_URL Struct Reference

Public Attributes

- BOOL **https**
- char **host** [256]
- char **port** [8]
- char **path** [H_FIELD_SIZE]

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

- https.h

4.9 internal_hooks Struct Reference

Public Member Functions

- void ∗CJSON_CDECL ∗ **allocate** (size_t size)
- **void** (CJSON_CDECL ∗deallocate)(void ∗pointer)
- void ∗CJSON_CDECL ∗ **reallocate** (void ∗pointer, size_t size)

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

- cJSON.c

4.10 parse_buffer Struct Reference

Public Attributes

- const unsigned char ∗ **content**
- size_t **length**
- size_t **offset**
- size_t **depth**
- internal_hooks **hooks**

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

- cJSON.c

4.11 printbuffer Struct Reference

Public Attributes

- unsigned char ∗ **buffer**
- size_t **length**
- size_t **offset**
- size_t **depth**
- cJSON_bool **noalloc**
- cJSON_bool **format**
- internal_hooks **hooks**

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

- cJSON.c

4.12 re_cache Struct Reference

Public Attributes

- char ∗ **re**
- pcre2_code ∗ **compiled_re**
- UT_hash_handle **hh**

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

- preg_replace.c

4.13 state Struct Reference

Public Attributes

- const char ∗ **start**
- const char ∗ **next**
- int **type**
-

  union {
  double **value**
  const double ∗ **bound**
  const void ∗ **function**
  };

- void ∗ **context**
- const te_variable ∗ **lookup**
- int **lookup_len**

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

- tinyexpr.c

4.14 Task Struct Reference

Unique data for a given task, such as task content or mathematical formula to get the task results.

```
#include <Task.h>
```

Public Attributes

- char ∗ **content**
- char ∗ **contentSlug**
- char ∗ **variablesRegistered**
- char ∗ **formula**
- char ∗ **unit**
- char ∗ **additionalInformation**
- double **resultValue**

4.14.1 Detailed Description

Unique data for a given task, such as task content or mathematical formula to get the task results.

struct "Task" to represent add data which are always single for a given task

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

- Task.h

4.15 te_expr Struct Reference

Public Attributes

- int **type**
-
```
union {
   double value
   const double * bound
   const void * function
};
```
- void ∗ **parameters** [1]

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

- tinyexpr.h

4.16 te_variable Struct Reference

Public Attributes

- const char ∗ **name**
- const void ∗ **address**
- int **type**
- void ∗ **context**

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

- tinyexpr.h

4.17 VariableObject Struct Reference

```
#include <TaskVariable.h>
```

Public Attributes

- short int **index**
- enum VariableIdentificationMethod **identificationMethod**
- char ∗ **variableName**
- char ∗ **regexInput0**
- char ∗ **regexInput1**
- char ∗ **generatedRegex**
- char ∗ **valueFound**
- double **convertedValue**
- bool **isVaildValue**
- struct VariableObject ∗ **nextVariable**

4.17.1 Detailed Description

Singly linked list “VariableObject” to represent data in the varaibles in the task

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

- TaskVariable.h

Chapter 5

File Documentation

5.1 AddTask.c File Reference

```
#include "AddTask.h"
```

Functions

- static void **DataExchangeInitCb** (JSContextRef ctx, JSObjectRef object)
- static JSObjectRef **DataExchangeConstructorCb** (JSContextRef context, JSObjectRef constructor, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void **DataExchangeDestroyCd** (JSObjectRef object)
- static JSValueRef **validateTaskCb** (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static JSValueRef submitTaskCb (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static JSValueRef **findVariableCb** (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void **JsWindowObjectClearedCb** (WebKitWebView ∗webView, WebKitWebFrame ∗frame, gpointer context, gpointer windowObject, gpointer userData)
- static void **destroy** (GtkWidget ∗widget, gpointer data)
- void initAddTaskWindow (int argc, char ∗argv[ ])

5.1.1 Function Documentation

5.1.1.1 initAddTaskWindow()

```
void initAddTaskWindow (
            int argc,
            char ∗ argv[ ] )
```

Initiate the task adding window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.1.1.2 submitTaskCb()

```
static JSValueRef submitTaskCb (
            JSContextRef context,
            JSObjectRef function,
            JSObjectRef thisObject,
            size_t argumentCount,
            const JSValueRef arguments[],
            JSValueRef * exception )  [static]
```

DataExchange.submitTask method callback implementation

5.2 AddTask.h File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <gtk/gtk.h>
#include <webkit/webkit.h>
#include <JavaScriptCore/JavaScript.h>
#include "Task.h"
#include "TextProcessing.h"
#include "MathEquationsParser.h"
#include "ApiService.h"
#include "preg_replace.h"
#include "JsonHelper.h"
```

Macros

- #define **ADD_TASK_WEB_SOURCE_DIRECTORY** "/forntend_forms/default/"
- #define **ADD_TASK_WEB_SOURCE_NAME** "add_task_form.html"

Functions

- static void DataExchangeInitCb (JSContextRef ctx, JSObjectRef object)
- static JSObjectRef DataExchangeConstructorCb (JSContextRef context, JSObjectRef constructor, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void DataExchangeDestroyCd (JSObjectRef object)
- static JSValueRef validateTaskCb (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static JSValueRef submitTaskCb (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static JSValueRef findVariableCb (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void JsWindowObjectClearedCb (WebKitWebView ∗webView, WebKitWebFrame ∗frame, gpointer context, gpointer windowObject, gpointer userData)
- static void destroy (GtkWidget ∗widget, gpointer data)
- void initAddTaskWindow (int argc, char ∗argv[ ])

Variables

- static const JSStaticFunction DataExchangeStaticFunctions [ ]
- static const JSClassDefinition DataExchangeDef

5.2.1 Function Documentation

5.2.1.1 DataExchangeConstructorCb()

```
static JSObjectRef DataExchangeConstructorCb (
             JSContextRef context,
             JSObjectRef constructor,
             size_t argumentCount,
             const JSValueRef arguments[],
             JSValueRef * exception )  [static]
```

JS class constructor

5.2.1.2 DataExchangeDestroyCd()

```
static void DataExchangeDestroyCd (
             JSObjectRef object )  [static]
```

JS class finalize

5.2.1.3 DataExchangeInitCb()

```
static void DataExchangeInitCb (
             JSContextRef ctx,
             JSObjectRef object )  [static]
```

Add a task window Most of the functions in this file are intended to attain JS <> C data exchange By using JavaScriptCore I can expand JS language by my own C functions. A more detailed discussion of the functions below is available in Final Report To create a user interface, GTK3 + and WebKit were implemented

JS Class initialize

5.2.1.4 destroy()

```
static void destroy (
             GtkWidget * widget,
             gpointer data )  [static]
```

JS destroy callback

5.2.1.5 findVariableCb()

```
static JSValueRef findVariableCb (
             JSContextRef context,
             JSObjectRef function,
             JSObjectRef thisObject,
             size_t argumentCount,
             const JSValueRef arguments[],
             JSValueRef * exception ) [static]
```

DataExchange.findVariable method callback implementation

5.2.1.6 initAddTaskWindow()

```
void initAddTaskWindow (
             int argc,
             char * argv[] )
```

Initiate the task adding window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.2.1.7 JsWindowObjectClearedCb()

```
static void JsWindowObjectClearedCb (
             WebKitWebView * webView,
             WebKitWebFrame * frame,
             gpointer context,
             gpointer windowObject,
             gpointer userData ) [static]
```

Callback - JavaScript window object has been cleared

5.2.1.8 submitTaskCb()

```
static JSValueRef submitTaskCb (
             JSContextRef context,
             JSObjectRef function,
             JSObjectRef thisObject,
             size_t argumentCount,
             const JSValueRef arguments[],
             JSValueRef * exception ) [static]
```

DataExchange.submitTask method callback implementation

5.2.1.9 validateTaskCb()

```
static JSValueRef validateTaskCb (
            JSContextRef context,
            JSObjectRef function,
            JSObjectRef thisObject,
            size_t argumentCount,
            const JSValueRef arguments[],
            JSValueRef * exception )  [static]
```

DataExchange.validateTask method callback implementation

5.2.2 Variable Documentation

5.2.2.1 DataExchangeDef

```
const JSClassDefinition DataExchangeDef  [static]
```

Initial value:

```
= {
    0,
    kJSClassAttributeNone,
    "DataExchange",
    NULL,
    NULL,
    DataExchangeStaticFunctions,
    DataExchangeInitCb,
    DataExchangeDestroyCd,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    DataExchangeConstructorCb,
    NULL,
    NULL
}
```

Definition of javascript class This structure contains properties and callbacks that define a type of object. All fields other than the version field are optional.

5.2.2.2 DataExchangeStaticFunctions

```
const JSStaticFunction DataExchangeStaticFunctions[]  [static]
```

Initial value:

```
= {
    { "validateTask", validateTaskCb, kJSPropertyAttributeReadOnly },
    { "submitTask", submitTaskCb, kJSPropertyAttributeReadOnly },
    { "findVariable", findVariableCb, kJSPropertyAttributeReadOnly },
    { NULL, NULL, 0 }
}
```

Class methods declarations This structure describes a statically declared function property. An annotation that marks a C functions as JavaScript static function

5.3 ApiService.c File Reference

```
#include "ApiService.h"
```

Functions

- void sendTaskToApi (cJSON *task)

  The functions below are used to exchange data between porgram and rest api server.
- char * getTask (cJSON *task)

5.3.1 Function Documentation

5.3.1.1 getTask()

```
char* getTask (
            cJSON * task )
```

The function that sends a content of the task to the rest api server, when user wants to solve his task

Parameters

JOSN	data to send to the server

5.3.1.2 sendTaskToApi()

```
void sendTaskToApi (
            cJSON * task )
```

The functions below are used to exchange data between porgram and rest api server.

The function that sends a task added by the user to the rest rest api server

Parameters

JOSN	data to send to the server

5.4 ApiService.h File Reference

```
#include <string.h>
#include <stdio.h>
```

```
#include <stdlib.h>
#include <gtk/gtk.h>
#include "https.h"
#include "cJSON.h"
```

Functions

- void sendTaskToApi (cJSON ∗task)

  The functions below are used to exchange data between porgram and rest api server.
- char ∗ getTask (cJSON ∗task)

5.4.1 Function Documentation

5.4.1.1 getTask()

```
char* getTask (
            cJSON * task )
```

The function that sends a content of the task to the rest api server, when user wants to solve his task

Parameters

JOSN	data to send to the server

5.4.1.2 sendTaskToApi()

```
void sendTaskToApi (
            cJSON * task )
```

The functions below are used to exchange data between porgram and rest api server.

The function that sends a task added by the user to the rest rest api server

Parameters

JOSN	data to send to the server

5.5 JsonHelper.c File Reference

```
#include "JsonHelper.h"
```

Functions

- char ∗ getValueFromJson (cJSON ∗json, char ∗str)

  Additional functions to parse JSON.

5.5.1 Function Documentation

5.5.1.1 getValueFromJson()

```
char* getValueFromJson (
            cJSON * json,
            char * str )
```

Additional functions to parse JSON.

The function that extracts json value for given index

Parameters

Source	JSON object
Index	of the item whose value had to be extracted

Returns

Variable value

5.6 JsonHelper.h File Reference

```
#include "cJSON.h"
```

Functions

- char ∗ getValueFromJson (cJSON ∗json, char ∗str)

  Additional functions to parse JSON.

5.6.1 Function Documentation

5.6.1.1 getValueFromJson()

```
char* getValueFromJson (
            cJSON * json,
            char * str )
```

Additional functions to parse JSON.

The function that extracts json value for given index

Parameters

Source	JSON object
Index	of the item whose value had to be extracted

Returns

Variable value

5.7 main.c File Reference

```
#include "time.h"
#include "MainMenu.h"
#include "Memd/mem.h"
```

Functions

- int main (int argc, char ∗argv[ ])

5.7.1 Function Documentation

5.7.1.1 main()

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

The main function of the program

Parameters

Number	of console line arguments
Console	line arguments

5.8 MainMenu.c File Reference

```
#include "MainMenu.h"
```

Macros

- #define **ARRAY_SIZE**(a) (sizeof(a) / sizeof(a[0]))
- #define **CTRLD** 4

Functions

- int initMainMenu (int argc, char ∗argv[ ])
- void printInMiddle (WINDOW ∗win, int starty, int startx, int width, char ∗string, chtype color)

5.8.1 Function Documentation

5.8.1.1 initMainMenu()

```
int initMainMenu (
             int argc,
             char * argv[ ] )
```

The function which draws the main menu of the program

Parameters

Number	of console line arguments
Console	line arguments

5.8.1.2 printInMiddle()

```
void printInMiddle (
             WINDOW * win,
             int starty,
             int startx,
             int width,
             char * string,
             chtype color )
```

The function which allows to center the header text in main menu

Parameters

Main	menu window
Line	from which to start drawing
Column	from which to start drawing
Width	of the text
Color	of the text

5.9 MainMenu.h File Reference

```
#include <curses.h>
#include <menu.h>
#include <stdlib.h>
#include <string.h>
#include "TaskSolver.h"
#include "SupportedTasks.h"
#include "AddTask.h"
```

Functions

- int initMainMenu (int argc, char ∗argv[ ])
- void printInMiddle (WINDOW ∗win, int starty, int startx, int width, char ∗string, chtype color)

Variables

- static char ∗ menuChoices [ ]

  Main menu of the program.

5.9.1 Function Documentation

5.9.1.1 initMainMenu()

```
int initMainMenu (
            int argc,
            char * argv[ ] )
```

The function which draws the main menu of the program

Parameters

Number	of console line arguments
Console	line arguments

5.9.1.2 printInMiddle()

```
void printInMiddle (
            WINDOW * win,
            int starty,
            int startx,
            int width,
            char * string,
            chtype color )
```

The function which allows to center the header text in main menu

Parameters

Main	menu window
Line	from which to start drawing
Column	from which to start drawing
Width	of the text
Color	of the text

5.9.2 Variable Documentation

5.9.2.1 menuChoices

```
char* menuChoices[ ]  [static]
```

Initial value:

```
=
{
    "Solve the task                      ",
    "Supported tasks                     ",
    "Add a task                          ",
    "Exit                                ",
    (char *)NULL,
}
```

Main menu of the program.

Char array which contains all the options available in the main menu

5.10 MathEquationsParser.c File Reference

```
#include "MathEquationsParser.h"
```

Functions

- bool isDouble (const char *str)
- double convertToDouble (const char *str)
- CalculationResult * calculateTask (VariableObject *variableListHead, char *taskFormula)

5.10.1 Function Documentation

5.10.1.1 calculateTask()

```
CalculationResult* calculateTask (
            VariableObject * variableListHead,
            char * taskFormula )
```

The function calculates the task using the formula porvided by the user

Parameters

Reference	to the head of “VariableObject” singly linked list
Formula	to solve the task

Returns

CalculationResult struct

5.10.1.2 convertToDouble()

```
double convertToDouble (
             const char * str )
```

The function converts a string to the double-precision data type

Parameters

source	string

Returns

daouble value

5.10.1.3 isDouble()

```
bool isDouble (
             const char * str )
```

Determine whether input is double-precision data type

Parameters

source	string

Returns

true when the DataType property of fi object a is double, and false otherwise.

5.11 MathEquationsParser.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
```

```
#include <stdbool.h>
#include <gtk/gtk.h>
#include "tinyexpr.h"
#include "TaskVariable.h"
#include "preg_replace.h"
```

Classes

- struct CalculationResult

Typedefs

- typedef enum CalculationStatus CalculationStatus

  The functions below are used to perform mathematical calculations.
- typedef struct CalculationResult CalculationResult

Enumerations

- enum CalculationStatus { **CALCULATED_CORRECTLY** = 0, **PARSE_ERROR** = 1 }

  The functions below are used to perform mathematical calculations.

Functions

- bool isDouble (const char ∗str)
- double convertToDouble (const char ∗str)
- CalculationResult ∗ calculateTask (VariableObject ∗variableListHead, char ∗taskFormula)

5.11.1 Typedef Documentation

5.11.1.1 CalculationResult

```
typedef struct CalculationResult CalculationResult
```

The results of calculations

5.11.1.2 CalculationStatus

```
typedef enum CalculationStatus CalculationStatus
```

The functions below are used to perform mathematical calculations.

The status of the calculateTask results 0 - calculated correctly 1 - a calculation error has occurred

5.11.2 Enumeration Type Documentation

5.11.2.1 CalculationStatus

```
enum CalculationStatus
```

The functions below are used to perform mathematical calculations.

The status of the calculateTask results 0 - calculated correctly 1 - a calculation error has occurred

5.11.3 Function Documentation

5.11.3.1 calculateTask()

```
CalculationResult* calculateTask (
            VariableObject * variableListHead,
            char * taskFormula )
```

The function calculates the task using the formula porvided by the user

Parameters

Reference	to the head of “VariableObject” singly linked list
Formula	to solve the task

Returns

CalculationResult struct

5.11.3.2 convertToDouble()

```
double convertToDouble (
            const char * str )
```

The function converts a string to the double-precision data type

Parameters

source	string

Returns

daouble value

5.11.3.3 isDouble()

```
bool isDouble (
             const char * str )
```

Determine whether input is double-precision data type

Parameters

source	string

Returns

true when the DataType property of fi object a is double, and false otherwise.

5.12 SupportedTasks.c File Reference

```
#include "SupportedTasks.h"
```

Functions

- static void **destroy** (GtkWidget ∗widget, gpointer data)
- void initSupportedTasksWindow (int argc, char ∗argv[ ])

5.12.1 Function Documentation

5.12.1.1 initSupportedTasksWindow()

```
void initSupportedTasksWindow (
             int argc,
             char * argv[ ] )
```

Initiate the task adding window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.13 SupportedTasks.h File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <gtk/gtk.h>
#include <webkit/webkit.h>
```

Macros

- #define **SUPPORTED_TASKS_WEB_SOURCE_DIRECTORY** "/forntend_forms/default/"
- #define **SUPPORTED_TASKS_WEB_SOURCE_NAME** "search_database_form.html"

Functions

- static void destroy (GtkWidget ∗widget, gpointer data)
- void initSupportedTasksWindow (int argc, char ∗argv[ ])

5.13.1 Function Documentation

5.13.1.1 destroy()

```
static void destroy (
            GtkWidget ∗ widget,
            gpointer data )  [static]
```

Supported Tasks window To create a user interface, GTK3 + and WebKit were implemented

Destroy callback

5.13.1.2 initSupportedTasksWindow()

```
void initSupportedTasksWindow (
            int argc,
            char ∗ argv[ ] )
```

Initiate the task adding window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.14 Task.c File Reference

```
#include "Task.h"
```

Functions

- void initTask ()
- void freeTask ()
- cJSON ∗ taskToJson ()

5.14.1 Function Documentation

5.14.1.1 freeTask()

```
void freeTask ( )
```

The function frees the memory used for the Task struct

5.14.1.2 initTask()

```
void initTask ( )
```

The function allocates the memory for the Task struct

5.14.1.3 taskToJson()

```
cJSON* taskToJson ( )
```

The function converts the objects contained in the Task struct into JSON format It is used to exchange the date between the program and the rest api server

5.15 Task.h File Reference

```
#include <gtk/gtk.h>
#include "TaskVariable.h"
#include "TextProcessing.h"
#include "cJSON.h"
```

Classes

- struct Task

  Unique data for a given task, such as task content or mathematical formula to get the task results.

Typedefs

- typedef struct Task Task

  Unique data for a given task, such as task content or mathematical formula to get the task results.

Functions

- void initTask ()
- void freeTask ()
- cJSON * taskToJson ()

Variables

- Task * task

5.15.1 Typedef Documentation

5.15.1.1 Task

```
typedef struct Task Task
```

Unique data for a given task, such as task content or mathematical formula to get the task results.

struct “Task” to represent add data which are always single for a given task

5.15.2 Function Documentation

5.15.2.1 freeTask()

```
void freeTask ( )
```

The function frees the memory used for the Task struct

5.15.2.2 initTask()

```
void initTask ( )
```

The function allocates the memory for the Task struct

5.15.2.3 taskToJson()

```
cJSON* taskToJson ( )
```

The function converts the objects contained in the Task struct into JSON format It is used to exchange the date between the program and the rest api server

5.15.3 Variable Documentation

5.15.3.1 task

```
Task* task
```

Varaible stores the reference to the Task struct object

5.16 TaskSolver.c File Reference

```
#include "TaskSolver.h"
```

Functions

- cJSON ∗ **solveTask** (char ∗taskContnet)
- static void **SolveTaskDataExchangeInitCb** (JSContextRef ctx, JSObjectRef object)
- static JSObjectRef **SolveTaskDataExchangeConstructorCb** (JSContextRef context, JSObjectRef constructor, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void **SolveTaskDataExchangeDestroyCd** (JSObjectRef object)
- static JSValueRef **solveTaskCb** (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void **SolveTaskJsWindowObjectClearedCb** (WebKitWebView ∗webView, WebKitWebFrame ∗frame, gpointer context, gpointer windowObject, gpointer userData)
- static void **destroy** (GtkWidget ∗widget, gpointer data)
- void initSolveTaskWindow (int argc, char ∗argv[ ])

5.16.1 Function Documentation

5.16.1.1 initSolveTaskWindow()

```
void initSolveTaskWindow (
            int argc,
            char * argv[ ] )
```

Initiate the task solving window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.17 TaskSolver.h File Reference

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <gtk/gtk.h>
#include <webkit/webkit.h>
#include <JavaScriptCore/JavaScript.h>
#include "Task.h"
#include "TextProcessing.h"
#include "MathEquationsParser.h"
#include "ApiService.h"
#include "preg_replace.h"
#include "JsonHelper.h"
```

Macros

- #define **TASK_SOLVER_WEB_SOURCE_DIRECTORY** "/forntend_forms/default/"
- #define **TASK_SOLVER_WEB_SOURCE_NAME** "solve_task_form.html"

Functions

- cJSON ∗ solveTask ()
- static void SolveTaskDataExchangeInitCb (JSContextRef ctx, JSObjectRef object)
- static JSObjectRef SolveTaskDataExchangeConstructorCb (JSContextRef context, JSObjectRef constructor, size_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void SolveTaskDataExchangeDestroyCd (JSObjectRef object)
- static JSValueRef solveTaskCb (JSContextRef context, JSObjectRef function, JSObjectRef thisObject, size←_t argumentCount, const JSValueRef arguments[ ], JSValueRef ∗exception)
- static void SolveTaskJsWindowObjectClearedCb (WebKitWebView ∗webView, WebKitWebFrame ∗frame, gpointer context, gpointer windowObject, gpointer userData)
- static void destroy (GtkWidget ∗widget, gpointer data)
- void initSolveTaskWindow (int argc, char ∗argv[ ])

Variables

- static const JSStaticFunction SolveTaskDataExchangeStaticFunctions [ ]
- static const JSClassDefinition SolveTaskDataExchangeDef

5.17.1 Function Documentation

5.17.1.1 destroy()

```
static void destroy (
            GtkWidget * widget,
            gpointer data )  [static]
```

JS Destroy callback

5.17.1.2 initSolveTaskWindow()

```
void initSolveTaskWindow (
            int argc,
            char * argv[] )
```

Initiate the task solving window To create a user interface, GTK3 + and WebKit were implemented in this function

Parameters

Number	of console line arguments
Console	line arguments

5.17.1.3 solveTask()

```
cJSON* solveTask ( )
```

Solve the task window Most of the functions in this file are intended to attain JS <> C data exchange By using JavaScriptCore I can expand JS language by my own C functions. A more detailed discussion of the functions below is available in

5.17.1.4 solveTaskCb()

```
static JSValueRef solveTaskCb (
            JSContextRef context,
            JSObjectRef function,
            JSObjectRef thisObject,
            size_t argumentCount,
            const JSValueRef arguments[],
            JSValueRef * exception )  [static]
```

SolveTaskDataExchange.solveTask method callback implementation

5.17.1.5 SolveTaskDataExchangeConstructorCb()

```
static JSObjectRef SolveTaskDataExchangeConstructorCb (
            JSContextRef context,
            JSObjectRef constructor,
            size_t argumentCount,
            const JSValueRef arguments[],
            JSValueRef * exception )  [static]
```

JS class constructor

5.17.1.6 SolveTaskDataExchangeDestroyCd()

```
static void SolveTaskDataExchangeDestroyCd (
            JSObjectRef object )  [static]
```

JS class finalize

5.17.1.7 SolveTaskDataExchangeInitCb()

```
static void SolveTaskDataExchangeInitCb (
            JSContextRef ctx,
            JSObjectRef object )  [static]
```

JS Class initialize

5.17.1.8 SolveTaskJsWindowObjectClearedCb()

```
static void SolveTaskJsWindowObjectClearedCb (
            WebKitWebView * webView,
            WebKitWebFrame * frame,
            gpointer context,
            gpointer windowObject,
            gpointer userData )  [static]
```

Callback - JavaScript window object has been cleared

5.17.2 Variable Documentation

5.17.2.1 SolveTaskDataExchangeDef

```
const JSClassDefinition SolveTaskDataExchangeDef  [static]
```

Initial value:

```
= {
    0,
    kJSClassAttributeNone,
    "SolveTaskDataExchange",
    NULL,
    NULL,
    SolveTaskDataExchangeStaticFunctions,
    SolveTaskDataExchangeInitCb,
    SolveTaskDataExchangeDestroyCd,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    NULL,
    SolveTaskDataExchangeConstructorCb,
    NULL,
    NULL
}
```

Definition of javascript class This structure contains properties and callbacks that define a type of object. All fields other than the version field are optional.

5.17.2.2 SolveTaskDataExchangeStaticFunctions

```
const JSStaticFunction SolveTaskDataExchangeStaticFunctions[]  [static]
```

Initial value:

```
= {
    { "solveTask", solveTaskCb, kJSPropertyAttributeReadOnly },
    { NULL, NULL, 0 }
}
```

Class methods declarations This structure describes a statically declared function property. An annotation that marks a C functions as JavaScript static function

5.18 TaskVariable.c File Reference

```
#include "TaskVariable.h"
```

Functions

- void freeAllVariables (VariableObject ∗head)
- VariableObject ∗ getLastVariableObject (VariableObject ∗head)
- VariableObject ∗ findVariableObjectByIndex (VariableObject ∗head, short int index)
- void printAllVariableObjects (VariableObject ∗head)
- cJSON ∗ variableObjectToJson (VariableObject ∗object)
- cJSON ∗ variablesObjectsToJsonArray (VariableObject ∗head)
- VariableObject ∗ addVariableObjectToList (short int index, short int selectedMode, char ∗variableName, char ∗regexInput0, char ∗regexInput1, char ∗regex, char ∗valueFound)

5.18.1 Function Documentation

5.18.1.1 addVariableObjectToList()

```
VariableObject* addVariableObjectToList (
             short int index,
             short int selectedMode,
             char * variableName,
             char * regexInput0,
             char * regexInput1,
             char * regex,
             char * valueFound )
```

The function adds a new object into the list This function is also used to refresh data in already existing objects.

5.18.1.2 findVariableObjectByIndex()

```
VariableObject* findVariableObjectByIndex (
             VariableObject * head,
             short int index )
```

The function that finds an element in a list through a given index

5.18.1.3 freeAllVariables()

```
void freeAllVariables (
            VariableObject * head )
```

The function frees the memory used for the “VariableObject” list

5.18.1.4 getLastVariableObject()

```
VariableObject* getLastVariableObject (
            VariableObject * head )
```

The function returns the last element in the list

5.18.1.5 printAllVariableObjects()

```
void printAllVariableObjects (
            VariableObject * head )
```

The function displays all items in the list. This function is used for debugging program.

5.18.1.6 variableObjectToJson()

```
cJSON* variableObjectToJson (
            VariableObject * object )
```

The function converts the objects contained in the “VariableObject” list into JSON format It is used to exchange the date between the program and the rest api server

5.18.1.7 variablesObjectsToJsonArray()

```
cJSON* variablesObjectsToJsonArray (
            VariableObject * head )
```

The function converts the one object into a JSON format. This function is used by the “printAllVariableObjects”, it just works for a given object.

5.19 TaskVariable.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <JavaScriptCore/JavaScript.h>
#include <gtk/gtk.h>
#include <pcre2.h>
#include "cJSON.h"
```

Classes

- struct VariableObject

Macros

- #define **PCRE2_CODE_UNIT_WIDTH** 8

Typedefs

- typedef enum VariableIdentificationMethod VariableIdentificationMethod
- typedef struct VariableObject VariableObject

Enumerations

- enum VariableIdentificationMethod { **BETWEEN_PHARSES** = 0, **AFTER_PHARSE** = 1, **REGEX_EXPRE↩ SSION** = 2 }

Functions

- void freeAllVariables (VariableObject ∗head)
- VariableObject ∗ getLastVariableObject (VariableObject ∗head)
- VariableObject ∗ findVariableObjectByIndex (VariableObject ∗head, short int index)
- void printAllVariableObjects (VariableObject ∗head)
- cJSON ∗ variableObjectToJson (VariableObject ∗object)
- cJSON ∗ variablesObjectsToJsonArray (VariableObject ∗head)
- VariableObject ∗ addVariableObjectToList (short int index, short int selectedMode, char ∗variableName, char ∗regexInput0, char ∗regexInput1, char ∗regex, char ∗valueFound)

Variables

- VariableObject ∗ variableListHead

5.19.1 Typedef Documentation

5.19.1.1 VariableIdentificationMethod

```
typedef enum VariableIdentificationMethod VariableIdentificationMethod
```

Variables in the text needed to correctly calculate the task A more detailed discussion of the functions below is available in Final Report

Variable search method selected by user.

5.19.1.2 VariableObject

```
typedef struct VariableObject VariableObject
```

Singly linked list “VariableObject” to represent data in the varaibles in the task

5.19.2 Enumeration Type Documentation

5.19.2.1 VariableIdentificationMethod

```
enum VariableIdentificationMethod
```

Variables in the text needed to correctly calculate the task A more detailed discussion of the functions below is available in Final Report

Variable search method selected by user.

5.19.3 Function Documentation

5.19.3.1 addVariableObjectToList()

```
VariableObject* addVariableObjectToList (
             short int index,
             short int selectedMode,
             char * variableName,
             char * regexInput0,
             char * regexInput1,
             char * regex,
             char * valueFound )
```

The function adds a new object into the list This function is also used to refresh data in already existing objects.

5.19.3.2 findVariableObjectByIndex()

```
VariableObject* findVariableObjectByIndex (
             VariableObject * head,
             short int index )
```

The function that finds an element in a list through a given index

5.19.3.3 freeAllVariables()

```
void freeAllVariables (
            VariableObject * head )
```

The function frees the memory used for the “VariableObject” list

5.19.3.4 getLastVariableObject()

```
VariableObject* getLastVariableObject (
            VariableObject * head )
```

The function returns the last element in the list

5.19.3.5 printAllVariableObjects()

```
void printAllVariableObjects (
            VariableObject * head )
```

The function displays all items in the list. This function is used for debugging program.

5.19.3.6 variableObjectToJson()

```
cJSON* variableObjectToJson (
            VariableObject * object )
```

The function converts the objects contained in the “VariableObject” list into JSON format It is used to exchange the date between the program and the rest api server

5.19.3.7 variablesObjectsToJsonArray()

```
cJSON* variablesObjectsToJsonArray (
            VariableObject * head )
```

The function converts the one object into a JSON format. This function is used by the “printAllVariableObjects”, it just works for a given object.

5.19.4 Variable Documentation

5.19.4.1 variableListHead

```
VariableObject* variableListHead
```

Varaible stores the reference to the head of “VariableObject” singly linked list

5.20 TextProcessing.c File Reference

```
#include "TextProcessing.h"
```

Functions

- char ∗ getStringValue (JSContextRef context, const JSValueRef jsValue)

  Functions functions for file handling.
- char ∗ getContentSlug (char ∗content)
- void toUpperCase (char ∗str)
- char ∗ replace (char ∗str, char ∗a, char ∗b)
- char ∗ getVaraibleFormText (char ∗RegexStr, char ∗source)

5.20.1 Function Documentation

5.20.1.1 getContentSlug()

```
char* getContentSlug (
            char * content )
```

The function deletes all dynamic data from the content of the task A more detailed discussion of the function below is available in Final Report

Parameters

The	source string

Returns

Output string

5.20.1.2 getStringValue()

```
char* getStringValue (
            JSContextRef context,
            const JSValueRef jsValue )
```

Functions functions for file handling.

Get string from JS value

Returns

Output string

5.20.1.3 getVaraibleFormText()

```
char* getVaraibleFormText (
             char * RegexStr,
             char * source )
```

The function searches for the values by the given REGEX expression A more detailed discussion of the function below is available in Final Report

Parameters

REGEX	expression
The	source string

Returns

Output string

5.20.1.4 replace()

```
char* replace (
             char * str,
             char * a,
             char * b )
```

The replace() function converts all characters of the string into a uppercase letter.

Parameters

The	source string
The	value that will be replaced by the new value
The	value to replace the search value with

Returns

Output string

5.20.1.5 toUpperCase()

```
void toUpperCase (
             char * str )
```

The function converts all characters of the string into a uppercase letter.

Parameters

String	to convert to upper case

5.21 TextProcessing.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <stdbool.h>
#include <gtk/gtk.h>
#include <JavaScriptCore/JavaScript.h>
#include <pcre2.h>
#include "cJSON.h"
#include "preg_replace.h"
```

Macros

- #define **PCRE2_CODE_UNIT_WIDTH** 8

Functions

- char ∗ getStringValue (JSContextRef context, const JSValueRef jsValue)

    Functions functions for file handling.
- char ∗ getContentSlug (char ∗content)
- void toUpperCase (char ∗str)
- char ∗ replace (char ∗str, char ∗a, char ∗b)
- char ∗ getVaraibleFormText (char ∗RegexStr, char ∗source)

5.21.1 Function Documentation

5.21.1.1 getContentSlug()

```
char* getContentSlug (
          char * content )
```

The function deletes all dynamic data from the content of the task A more detailed discussion of the function below is available in Final Report

Parameters

The	source string

Returns

Output string

5.21.1.2 getStringValue()

```
char* getStringValue (
            JSContextRef context,
            const JSValueRef jsValue )
```

Functions functions for file handling.

Get string from JS value

Returns

Output string

5.21.1.3 getVaraibleFormText()

```
char* getVaraibleFormText (
            char * RegexStr,
            char * source )
```

The function searches for the values by the given REGEX expression A more detailed discussion of the function below is available in Final Report

Parameters

REGEX	expression
The	source string

Returns

Output string

5.21.1.4 replace()

```
char* replace (
            char * str,
            char * a,
            char * b )
```

The replace() function converts all characters of the string into a uppercase letter.

Parameters

The	source string
The	value that will be replaced by the new value
The	value to replace the search value with

Returns

Output string

5.21.1.5 toUpperCase()

```
void toUpperCase (
             char * str )
```

The function converts all characters of the string into a uppercase letter.

Parameters

String	to convert to upper case

Index