LightJSON Documentation

Release 0.1.0

Shenggan

Dec 03, 2017

Contents

1 APIs **1**
1.1 lightjson.h 1

CHAPTER 1

APIs

1.1 lightjson.h

The head file of LightJSON.

Author Shenggan

namespace `ljson`

Typedefs

typedef struct *ljson_value* `ljson_value`

typedef struct *ljson_member* `ljson_member`

Enums

enum `ljson_type`

the basic type of the json struct

Values:

`LJSON_NULL`

`LJSON_FALSE`

`LJSON_TRUE`

`LJSON_NUMBER`

`LJSON_STRING`

`LJSON_ARRAY`

`LJSON_OBJECT`

enum **ljson_state**
the type of the results or error

Values:

LJSON_PARSE_OK = 0

LJSON_STRINGIFY_OK

LJSON_PARSE_EXPECT_VALUE

LJSON_PARSE_INVALID_VALUE

LJSON_PARSE_ROOT_NOT_SINGULAR

LJSON_PARSE_NUMBER_TOO_BIG

LJSON_PARSE_MISS_QUOTATION_MARK

LJSON_PARSE_INVALID_STRING_ESCAPE

LJSON_PARSE_INVALID_STRING_CHAR

LJSON_PARSE_INVALID_UNICODE_HEX

LJSON_PARSE_INVALID_UNICODE_SURROGATE

LJSON_PARSE_MISS_COMMA_OR_SQUARE_BRACKET

LJSON_PARSE_MISS_KEY

LJSON_PARSE_MISS_COLON

LJSON_PARSE_MISS_COMMA_OR_CURLY_BRACKET

Functions

void **ljson_init**(*ljson_value* **v*)
initailize a *ljson_value*, use it after declaration

Parameters

- v: the pointer of *ljson_value* you want to initailize

void **ljson_free**(*ljson_value* **v*)
free the memory of a *ljson_value*, use it if you will not use it or initailize it again

Parameters

- v: the pointer of *ljson_value* you want to free

void **ljson_set_null**(*ljson_value* **v*)
the same as *ljson_free(ljson_value* v)*

int **ljson_parse**(*ljson_value* **v*, **const** char **json*)
parse a string to get the *ljson_value*

Return ljson_state

Parameters

- v: the pointer of *ljson_value* you want to store the result of parse

- `json`: the string you want to parse

int **ljson_parse**(*ljson_value* *v, **const** std::string &*json*)
parse a string to get the *ljson_value*

Return ljson_state

Parameters

- `v`: the pointer of *ljson_value* you want to store the result of parse
- `json`: the string you want to parse

int **ljson_stringify**(**const** *ljson_value* *v, std::string &*json*)
ljson_value v to get the string os the json

Return ljson_state

Parameters

- `v`: the pointer of *ljson_value* you want to stringify
- `json`: the string you want to store the result

int **ljson_stringify**(**const** *ljson_value* *v, char **json*)
ljson_value v to get the string os the json

Return ljson_state

Parameters

- `v`: the pointer of *ljson_value* you want to stringify
- `json`: the string you want to store the result

ljson_type **ljson_get_type**(**const** *ljson_value* *v)

void **ljson_set_number**(*ljson_value* *v, double *n*)

double **ljson_get_number**(**const** *ljson_value* *v)

void **ljson_set_boolean**(*ljson_value* *v, int *b*)

int **ljson_get_boolean**(**const** *ljson_value* *v)

void **ljson_set_string**(*ljson_value* *v, **const** char *s, size_t *len*)

void **ljson_set_string**(*ljson_value* *v, **const** std::string &*s*)

const char ***ljson_get_string**(**const** *ljson_value* *v)

size_t **ljson_get_string_length**(**const** *ljson_value* *v)

void **ljson_set_array**(*ljson_value* *v, std::vector<*ljson_value*> &*vec*)

ljson_value ***ljson_get_array_element**(**const** *ljson_value* *v, size_t *index*)

size_t **ljson_get_array_size**(**const** *ljson_value* *v)

void **ljson_set_object**(*ljson_value* *v, std::vector<*ljson_member*> &*vec*)

size_t **ljson_get_object_size**(**const** *ljson_value* *v)

const char ***ljson_get_object_key**(**const** *ljson_value* *v, size_t *index*)

size_t **ljson_get_object_key_length**(**const** *ljson_value* *v, size_t *index*)

ljson_value ***ljson_get_object_value**(**const** *ljson_value* *v, size_t *index*)

struct ljson_member
#include <lightjson.h> the struct of the member of json object

Public Members

std::string **key**

ljson_value **value**

struct ljson_object
#include <lightjson.h> the struct of the json object

Public Members

char ***name**

ljson_value **value**

struct ljson_value
#include <lightjson.h> the inner struct of a json, can present all kind of ljson_type

Public Members

union *ljson*::*ljson_value*::*__data* **data**
data part of *ljson_value*

ljson_type **type**
typr of this *ljson_value*

union __data
#include <lightjson.h>

Public Members

std::vector<*ljson_member*> ***mobject**
object

std::string ***mstring**
string

std::vector<*ljson_value*> ***marray**
array

double **mdouble**
number

Index

L

ljson (C++ type), 1
ljson::LJSON_ARRAY (C++ enumerator), 1
ljson::LJSON_FALSE (C++ enumerator), 1
ljson::ljson_free (C++ function), 2
ljson::ljson_get_array_element (C++ function), 3
ljson::ljson_get_array_size (C++ function), 3
ljson::ljson_get_boolean (C++ function), 3
ljson::ljson_get_number (C++ function), 3
ljson::ljson_get_object_key (C++ function), 4
ljson::ljson_get_object_key_length (C++ function), 4
ljson::ljson_get_object_size (C++ function), 3
ljson::ljson_get_object_value (C++ function), 4
ljson::ljson_get_string (C++ function), 3
ljson::ljson_get_string_length (C++ function), 3
ljson::ljson_get_type (C++ function), 3
ljson::ljson_init (C++ function), 2
ljson::ljson_member (C++ class), 4
ljson::ljson_member (C++ type), 1
ljson::ljson_member::key (C++ member), 4
ljson::ljson_member::value (C++ member), 4
ljson::LJSON_NULL (C++ enumerator), 1
ljson::LJSON_NUMBER (C++ enumerator), 1
ljson::ljson_object (C++ class), 4
ljson::LJSON_OBJECT (C++ enumerator), 1
ljson::ljson_object::name (C++ member), 4
ljson::ljson_object::value (C++ member), 4
ljson::ljson_parse (C++ function), 2, 3
ljson::LJSON_PARSE_EXPECT_VALUE (C++ enumerator), 2
ljson::LJSON_PARSE_INVALID_STRING_CHAR (C++ enumerator), 2
ljson::LJSON_PARSE_INVALID_STRING_ESCAPE (C++ enumerator), 2
ljson::LJSON_PARSE_INVALID_UNICODE_HEX (C++ enumerator), 2
ljson::LJSON_PARSE_INVALID_UNICODE_SURROGATE (C++ enumerator), 2
ljson::LJSON_PARSE_INVALID_VALUE (C++ enumerator), 2
ljson::LJSON_PARSE_MISS_COLON (C++ enumerator), 2
ljson::LJSON_PARSE_MISS_COMMA_OR_CURLY_BRACKET (C++ enumerator), 2
ljson::LJSON_PARSE_MISS_COMMA_OR_SQUARE_BRACKET (C++ enumerator), 2
ljson::LJSON_PARSE_MISS_KEY (C++ enumerator), 2
ljson::LJSON_PARSE_MISS_QUOTATION_MARK (C++ enumerator), 2
ljson::LJSON_PARSE_NUMBER_TOO_BIG (C++ enumerator), 2
ljson::LJSON_PARSE_OK (C++ enumerator), 2
ljson::LJSON_PARSE_ROOT_NOT_SINGULAR (C++ enumerator), 2
ljson::ljson_set_array (C++ function), 3
ljson::ljson_set_boolean (C++ function), 3
ljson::ljson_set_null (C++ function), 2
ljson::ljson_set_number (C++ function), 3
ljson::ljson_set_object (C++ function), 3
ljson::ljson_set_string (C++ function), 3
ljson::ljson_state (C++ type), 1
ljson::LJSON_STRING (C++ enumerator), 1
ljson::ljson_stringify (C++ function), 3
ljson::LJSON_STRINGIFY_OK (C++ enumerator), 2
ljson::LJSON_TRUE (C++ enumerator), 1
ljson::ljson_type (C++ type), 1
ljson::ljson_value (C++ class), 4
ljson::ljson_value (C++ type), 1
ljson::ljson_value::__data (C++ type), 4
ljson::ljson_value::__data::marray (C++ member), 4
ljson::ljson_value::__data::mdouble (C++ member), 4
ljson::ljson_value::__data::mobject (C++ member), 4
ljson::ljson_value::__data::mstring (C++ member), 4
ljson::ljson_value::data (C++ member), 4
ljson::ljson_value::type (C++ member), 4