Telegram Bot Api Client 0.6.1

Generated by Doxygen 1.8.14

Contents

1	Clas	s Index			1
	1.1	Class I	_ist		1
2	File	Index			3
	2.1	File Lis	st		3
3	Clas	s Docu	mentatior	1	5
	3.1	JsonW	ebClient C	Class Reference	5
		3.1.1	Construc	ctor & Destructor Documentation	6
			3.1.1.1	JsonWebClient()	6
		3.1.2	Member	Function Documentation	6
			3.1.2.1	fire()	6
			3.1.2.2	loop()	7
			3.1.2.3	processHeader()	7
			3.1.2.4	processJson()	7
			3.1.2.5	reConnect()	7
			3.1.2.6	state()	8
			3.1.2.7	stop()	8
		3.1.3	Member	Data Documentation	8
			3.1.3.1	CallBackObject	8
			3.1.3.2	ContentLength	8
			3.1.3.3	Host	8
			3.1.3.4	HttpStatusOk	9
			3135	JWC CALLBACK ERROR SIGNATURE	q

ii CONTENTS

		3.1.3.6	JWC_CALLBACK_MESSAGE_SIGNATURE	9
		3.1.3.7	NetClient	9
		3.1.3.8	Port	9
		3.1.3.9	State	9
3.2	JwcCli	entState C	Class Reference	9
3.3	JwcPro	ocessError	Class Reference	10
3.4	Messa	ge Struct F	Reference	10
	3.4.1	Member	Data Documentation	10
		3.4.1.1	ChatFirstName	10
		3.4.1.2	Chatld	10
		3.4.1.3	ChatLastName	11
		3.4.1.4	ChatType	11
		3.4.1.5	Date	11
		3.4.1.6	FromFirstName	11
		3.4.1.7	FromId	11
		3.4.1.8	FromIsBot	11
		3.4.1.9	FromLanguageCode	11
		3.4.1.10	FromLastName	11
		3.4.1.11	MessageId	12
		3.4.1.12	Text	12
		3.4.1.13	UpdateId	12
3.5	TBCK	eyBoard Cl	lass Reference	12
	3.5.1	Construc	tor & Destructor Documentation	13
		3.5.1.1	TBCKeyBoard()	13
		3.5.1.2	~TBCKeyBoard()	13
	3.5.2	Member	Function Documentation	13
		3.5.2.1	get()	13
		3.5.2.2	getOneTime()	14
		3.5.2.3	getResize()	14
		3.5.2.4	length() [1/2]	14

CONTENTS

		3.5.2.5	length() [2/2]	15
		3.5.2.6	push()	15
	3.5.3	Member	Data Documentation	15
		3.5.3.1	OneTime	15
		3.5.3.2	Resize	16
3.6	TBCKe	eyBoardRo	w Struct Reference	16
3.7	Telegra	amBotClier	nt Class Reference	16
	3.7.1	Construc	tor & Destructor Documentation	17
		3.7.1.1	TelegramBotClient() [1/3]	17
		3.7.1.2	TelegramBotClient() [2/3]	18
		3.7.1.3	TelegramBotClient() [3/3]	18
		3.7.1.4	~TelegramBotClient()	19
	3.7.2	Member	Function Documentation	19
		3.7.2.1	begin()	19
		3.7.2.2	loop()	19
		3.7.2.3	pollError()	19
		3.7.2.4	pollSuccess()	20
		3.7.2.5	postError()	20
		3.7.2.6	postMessage() [1/2]	21
		3.7.2.7	postMessage() [2/2]	21
		3.7.2.8	postSuccess()	22
		3.7.2.9	setCallbacks()	22
		3.7.2.10	startPolling()	23
		3.7.2.11	startPosting()	23
	3.7.3	Member	Data Documentation	23
		3.7.3.1	LastUpdateId	23
		3.7.3.2	Parallel	24
		3.7.3.3	SslPollClient	24
		3.7.3.4	SslPostClient	24
		3.7.3.5	TBC_CALLBACK_ERROR_SIGNATURE	24
		3.7.3.6	TBC_CALLBACK_RECEIVE_SIGNATURE	24
		3.7.3.7	Token	24
3.8	Telegra	amProcess	Error Class Reference	24

iv CONTENTS

4	File	Docume	entation	25
	4.1	JsonW	/ebClient.cpp File Reference	25
		4.1.1	Detailed Description	25
	4.2	JsonW	/ebClient.h File Reference	25
		4.2.1	Detailed Description	26
		4.2.2	Enumeration Type Documentation	26
			4.2.2.1 JwcClientState	26
			4.2.2.2 JwcProcessError	27
	4.3	Telegra	amBotClient.h File Reference	27
		4.3.1	Detailed Description	28
		4.3.2	Enumeration Type Documentation	29
			4.3.2.1 TelegramProcessError	29
Inc	dex			31

Chapter 1

Class Index

1.1 Class List

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

JsonWebClient							 																	Ę
JwcClientState													 						 					9
JwcProcessError	•												 						 					10
Message													 						 					10
TBCKeyBoard .							 						 						 					12
TBCKeyBoardRo	w						 						 						 					16
TelegramBotClier	nt						 						 						 					16
TelegramProcess	sΕ	rro	or				 						 						 					24

2 Class Index

Chapter 2

File Index

2.1 File List

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

JsonWeb	bClient.cpp	
	Implementation of a simple web client receiving json uses an underlying implementation of Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data	25
JsonWeb	bClient.h	
	Header of a simple web client receiving json uses an underlying implementation of Client inter-	
	face. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data	25
TBCDeb	oug.h	??
Telegram	nBotClient.h	
	Header of a simple client sending and receiving message via Telegram's Bot API. Uses one or two underlying objects implementing the Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data	27

File Index

Chapter 3

Class Documentation

3.1 JsonWebClient Class Reference

Public Member Functions

- JsonWebClient (Client *netClient, String host, int port, void *callBackObject, JWC_CALLBACK_MESSAGE_SIGNATURE, JWC_CALLBACK_ERROR_SIGNATURE)
- bool fire (String commands[], int count)

Executes a list of commands.

• JwcClientState state ()

Current state of the client.

bool loop ()

Method to poll client processing.

• bool stop ()

Stops the client.

Private Member Functions

• void reConnect ()

Reconnects to host.

• bool processHeader ()

Process a header.

• bool processJson ()

Process JSON.

Private Attributes

- JwcClientState State = JwcClientState::Unconnected
- Client * NetClient
- String Host
- int Port
- long ContentLength = JWC_BUFF_SIZE
- bool HttpStatusOk = false
- void * CallBackObject
- JWC CALLBACK MESSAGE SIGNATURE
- JWC_CALLBACK_ERROR_SIGNATURE

3.1.1 Constructor & Destructor Documentation

3.1.1.1 JsonWebClient()

```
JsonWebClient::JsonWebClient (
        Client * netClient,
        String host,
        int port,
        void * callBackObject,
        JWC_CALLBACK_MESSAGE_SIGNATURE ,
        JWC_CALLBACK_ERROR_SIGNATURE )
```

Constructor, initializing all members

Parameters

netClient	a object implementing Client interface to access the network. Using a Client implementing ssl feature will result in https otherwise http.
host	Host to connect to
port	Port to connect to
callBackObject	Object passed to the callbacks, shall not be 0
JWC_CALLBACK_MESSAGE_SIGNATURE	Callback called on receiving a message / valid json data
JWC_CALLBACK_ERROR_SIGNATURE	Callback called on error while receiving

3.1.2 Member Function Documentation

3.1.2.1 fire()

Executes a list of commands.

Parameters

in	commands[]	list of commands
in	count	of commands

Returns

Return true on success

Sends a list of commands to the server by calling println() for each command and flush() at the end of list. The commands shall follow the http protocol.

3.1.2.2 loop()

```
bool JsonWebClient::loop ( )
```

Method to poll client processing.

Returns

True is an internal action was executed.

Method to poll client processing, shall be called in each main loop()

3.1.2.3 processHeader()

```
bool JsonWebClient::processHeader ( ) [private]
```

Process a header.

Returns

Returns true while headers found in underlying Client

Read a header from NetClient and process it.

3.1.2.4 processJson()

```
bool JsonWebClient::processJson ( ) [private]
```

Process JSON.

Returns

Returns true on success

Reads data from underlying Client and process it by ArduinoJSON

3.1.2.5 reConnect()

```
void JsonWebClient::reConnect ( ) [private]
```

Reconnects to host.

Returns

Return nothing

Reconnects to host, skips open connection

3.1.2.6 state()

```
JwcClientState JsonWebClient::state ( )
```

Current state of the client.

Returns

The current state as a JwcClientState

Make the current state of the client public accessible.

3.1.2.7 stop()

```
bool JsonWebClient::stop ( )
```

Stops the client.

Returns

True

Stops the underlying client connection and reset client state to JwcClientState::unconnected

3.1.3 Member Data Documentation

3.1.3.1 CallBackObject

```
void* JsonWebClient::CallBackObject [private]
```

Object passed to the callbacks

3.1.3.2 ContentLength

```
long JsonWebClient::ContentLength = JWC_BUFF_SIZE [private]
```

Content length stored during header processing

3.1.3.3 Host

```
String JsonWebClient::Host [private]
```

Host to connect to

3.1.3.4 HttpStatusOk

```
bool JsonWebClient::HttpStatusOk = false [private]
```

Indicate if Http 200 Ok header was found

3.1.3.5 JWC_CALLBACK_ERROR_SIGNATURE

```
JsonWebClient::JWC_CALLBACK_ERROR_SIGNATURE [private]
```

Callback called on error while receiving

3.1.3.6 JWC_CALLBACK_MESSAGE_SIGNATURE

```
JsonWebClient::JWC_CALLBACK_MESSAGE_SIGNATURE [private]
```

Callback called on receiving a message / valid json data

3.1.3.7 NetClient

```
Client* JsonWebClient::NetClient [private]
```

Client used to access the net (depends on hardware)

3.1.3.8 Port

```
int JsonWebClient::Port [private]
```

Port to connect to

3.1.3.9 State

```
JwcClientState JsonWebClient::State = JwcClientState::Unconnected [private]
```

Current state of the client

The documentation for this class was generated from the following files:

- JsonWebClient.h
- JsonWebClient.cpp

3.2 JwcClientState Class Reference

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

• JsonWebClient.h

3.3 JwcProcessError Class Reference

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

· JsonWebClient.h

3.4 Message Struct Reference

Public Attributes

- long UpdateId
- · long MessageId
- long FromId
- bool FromIsBot
- String FromFirstName
- String FromLastName
- String FromLanguageCode
- long Chatld
- String ChatFirstName
- String ChatLastName
- String ChatType
- · String Text
- long Date

3.4.1 Member Data Documentation

3.4.1.1 ChatFirstName

String Message::ChatFirstName

chat_first_name: chat/first_name Optional. First name of the other party in a private chat

3.4.1.2 Chatld

long Message::ChatId

chat_id: chat/id

Used to identify chat while posting a message Unique identifier for this chat. This number may be greater than 32 bits and some programming languages may have difficulty/silent defects in interpreting it. But it is smaller than 52 bits, so a signed 64 bit integer or double-precision float type are safe for storing this identifier.

3.4.1.3 ChatLastName

String Message::ChatLastName

chat_last_name: chat/last_name Optional. Last name of the other party in a private chat

3.4.1.4 ChatType

String Message::ChatType

chat_type: chat/type Type of chat, can be either "private", "group", "supergroup" or "channel"

3.4.1.5 Date

long Message::Date

date: date Date the message was sent in Unix time

3.4.1.6 FromFirstName

String Message::FromFirstName

from_first_name: from/first_name User's or bot's first name

3.4.1.7 FromId

long Message::FromId

from_id : from/id Unique identifier for this user or bot

3.4.1.8 FromIsBot

bool Message::FromIsBot

from is bot: from/is bot True, if this user is a bot

3.4.1.9 FromLanguageCode

String Message::FromLanguageCode

from_language_code: from/language_code Optional. IETF language tag of the user's language

3.4.1.10 FromLastName

String Message::FromLastName

from_last_name: from/last_name Optional. User's or bot's last name

3.4.1.11 Messageld

```
long Message::MessageId
```

message id : message id Unique message identifier inside this chat

3.4.1.12 Text

```
String Message::Text
```

text: text Optional. For text messages, the actual UTF-8 text of the message, 0-4096 characters.

3.4.1.13 Updateld

```
long Message::UpdateId
```

update_id The update's unique identifier. Update identifiers start from a certain positive number and increase sequentially. This ID becomes especially handy if you're using Webhooks, since it allows you to ignore repeated updates or to restore the correct update sequence, should they get out of order. If there are no new updates for at least a week, then identifier of the next update will be chosen randomly instead of sequentially.

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

· TelegramBotClient.h

3.5 TBCKeyBoard Class Reference

Public Member Functions

• TBCKeyBoard (uint count, bool oneTime=false, bool resize=false)

Constructor.

∼TBCKeyBoard ()

Destructor.

• TBCKeyBoard & push (uint count, const String buttons[])

Adds a row to the keyboard.

• const String get (const uint row, const uint col)

Gets a button text.

• const int length (const uint row)

Length of row.

• const int length ()

Length of keyboard.

const bool getOneTime ()

Gets value of OneTime.

• const bool getResize ()

Gets value of Resize.

Private Attributes

- uint Count
- uint Counter
- TBCKeyBoardRow * Rows
- bool OneTime = false
- bool Resize = false

3.5.1 Constructor & Destructor Documentation

3.5.1.1 TBCKeyBoard()

Constructor.

Constructor, initializing all members

Parameters

count	The number of rows in keyboard.
oneTime	value for OneTime
resize	value for Resize

3.5.1.2 \sim TBCKeyBoard()

```
{\tt TBCKeyBoard::}{\sim}{\tt TBCKeyBoard} ( )
```

Destructor.

Destructor

3.5.2 Member Function Documentation

3.5.2.1 get()

Gets a button text.

Parameters

	in	row	Index of row to fetch button text from
ſ	in	col	Index of column to fetch button text from

Returns

button text

Gets the text of a button in given row and column

```
3.5.2.2 getOneTime()
```

```
const bool TBCKeyBoard::getOneTime ( ) [inline]
```

Gets value of OneTime.

Returns

Value of OneTime

See OneTime, this methods makes it read only.

3.5.2.3 getResize()

```
const bool TBCKeyBoard::getResize ( ) [inline]
```

Gets value of Resize.

Returns

Value of Resize

See Resize, this methods makes it read only.

```
3.5.2.4 length() [1/2]
```

Length of row.

Parameters

in	row	Index of row to get length
----	-----	----------------------------

Returns

return length of row

Gets the length of the row at the given index The length of a row is the number of buttons in this row.

```
3.5.2.5 length() [2/2]

const int TBCKeyBoard::length ( )

Length of keyboard.
```

Returns

return length of keyboard

Gets the length of the keyboard The length of a keyboard is the number of rows in this keyboard.

3.5.2.6 push()

Adds a row to the keyboard.

Parameters

in	count	Number of buttons passend in buttons
in	buttons	Button to be displayed in this row

Returns

The keyboard itself

Adds a row to the keyboard containing buttons displaying the string passed in buttons[]

3.5.3 Member Data Documentation

3.5.3.1 OneTime

```
bool TBCKeyBoard::OneTime = false [private]
```

Requests clients to hide the keyboard as soon as it's been used. The keyboard will still be available, but clients will automatically display the usual letter-keyboard in the chat – the user can press a special button in the input field to see the custom keyboard again.

Defaults to false.

https://core.telegram.org/bots/api#replykeyboardmarkup

3.5.3.2 Resize

```
bool TBCKeyBoard::Resize = false [private]
```

Requests clients to resize the keyboard vertically for optimal fit (e.g., make the keyboard smaller if there are just two rows of buttons). Defaults to false, in which case the custom keyboard is always of the same height as the app's standard keyboard.

Defaults to false.

https://core.telegram.org/bots/api#replykeyboardmarkup

The documentation for this class was generated from the following files:

- · TelegramBotClient.h
- TelegramBotClient.cpp

3.6 TBCKeyBoardRow Struct Reference

Public Attributes

- · uint Count
- String * Buttons

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

• TelegramBotClient.h

3.7 TelegramBotClient Class Reference

Public Member Functions

 TelegramBotClient (String token, Client &sslPollClient, Client &sslPostClient, TBC_CALLBACK_RECEIVE_SIGNATURE, TBC_CALLBACK_ERROR_SIGNATURE)

Constructor.

• TelegramBotClient (String token, Client &sslPollClient, Client &sslPostClient)

Constructor

• TelegramBotClient (String token, Client &sslPollClient)

Constructor.

∼TelegramBotClient ()

Destructor.

• void begin (TBC_CALLBACK_RECEIVE_SIGNATURE, TBC_CALLBACK_ERROR_SIGNATURE)

Alias for setCallbacks following Arduino convention.

• void setCallbacks (TBC_CALLBACK_RECEIVE_SIGNATURE, TBC_CALLBACK_ERROR_SIGNATURE)

Sets callbacks.

• bool loop ()

Handles client background tasks.

void postMessage (long chatld, String text, TBCKeyBoard &keyBoard)

Post a message.

void postMessage (long chatld, String text)

Post a message.

• void pollSuccess (JwcProcessError err, JsonObject &json)

Callback called by JSONWebClient.

void pollError (JwcProcessError err, Client *client)

Callback called by JSONWebClient.

void postSuccess (JwcProcessError err, JsonObject &json)

Callback called by JSONWebClient.

void postError (JwcProcessError err, Client *client)

Callback called by JSONWebClient.

Static Public Member Functions

- static void callbackPollSuccess (void *obj, JwcProcessError err, JsonObject &json)
- static void callbackPollError (void *obj, JwcProcessError err, Client *client)
- static void callbackPostSuccess (void *obj, JwcProcessError err, JsonObject &json)
- static void callbackPostError (void *obj, JwcProcessError err, Client *client)

Private Member Functions

· void startPolling ()

Starts polling.

· void startPosting (String Message)

Starts posting a message.

Private Attributes

- long LastUpdateId = 0
- · String Token
- bool Parallel = false
- JsonWebClient * SslPollClient
- JsonWebClient * SslPostClient
- TBC_CALLBACK_RECEIVE_SIGNATURE
- TBC_CALLBACK_ERROR_SIGNATURE

3.7.1 Constructor & Destructor Documentation

3.7.1.1 TelegramBotClient() [1/3]

Constructor.

Constructor, initializing all members including callbacks using different clients for posting and polling

Parameters

token	secure token for your bot provided by BotFather.	
sslPollClient	SSL client used for polling messages from remote server	
sslPostClient	SSL client used for posting messages to remote server	
TBC_CALLBACK_RECEIVE_SIGNATURE	Callback called on receiving a message	
TBC_CALLBACK_ERROR_SIGNATURE	Callback called on error while receiving	

3.7.1.2 TelegramBotClient() [2/3]

Constructor.

Constructor, initializing only members no callbacks using different clients for posting and polling

Parameters

token	secure token for your bot provided by BotFather.
sslPollClient	SSL client used for polling messages from remote server
sslPostClient	SSL client used for posting messages to remote server

3.7.1.3 TelegramBotClient() [3/3]

Constructor.

Constructor, initializing only members no callbacks using the same client for posting and polling

Parameters

token	secure token for your bot provided by BotFather.
sslPollClient	SSL client used for polling messages from remote server
sslPostClient	SSL client used for posting messages to remote server

3.7.1.4 ∼TelegramBotClient()

```
{\tt TelegramBotClient::} {\sim} {\tt TelegramBotClient} \ \ (\ )
```

Destructor.

Destructor

3.7.2 Member Function Documentation

3.7.2.1 begin()

Alias for setCallbacks following Arduino convention.

Parameters

in	TBC_CALLBACK_RECEIVE_SIGNATURE	Callback called on receiving a message
in	TBC_CALLBACK_ERROR_SIGNATURE	Callback called on error while receiving

Returns

Nothing

Alias for setCallbacks following Arduino convention sets callbacks

3.7.2.2 loop()

```
bool TelegramBotClient::loop ( )
```

Handles client background tasks.

Returns

Return true is an action was needed and performed

Handles client background tasks, shall be calles in every main loop()

3.7.2.3 pollError()

Callback called by JSONWebClient.

Parameters

in	err	Error Code from JwcProcessError	
in	client	Client that causes the problem.	

Returns

Nothing

This is an internal method called by underlying JSONWebClient

Note

Do not call this method.

3.7.2.4 pollSuccess()

Callback called by JSONWebClient.

Parameters

in	err	Error Code from JwcProcessError
in	json	JsonObject generated by ArduinoJSON

Returns

Nothing

This is an internal method called by underlying JSONWebClient

Note

Do not call this method.

3.7.2.5 postError()

Callback called by JSONWebClient.

Parameters

in	err	Error Code from JwcProcessError	
in	client	Client that causes the problem.	

Returns

Nothing

This is an internal method called by underlying JSONWebClient

Note

Do not call this method.

3.7.2.6 postMessage() [1/2]

Post a message.

Parameters

in	chatld	ld of the chat the message shall be sent to.	
in	text	Text of the message	
in	keyBoard	Optional. Keyboard to be send with this message.	

Returns

Nothing

Post a message to a given chat. (Only text messages and custom keyboards are supported, yet.)

3.7.2.7 postMessage() [2/2]

Post a message.

Parameters

in	chat← Id	Id of the chat the message shall be sent to.
in	text	Text of the message

Returns

Nothing

Post a message to a given chat. (Only text messages and custom keyboards are supported, yet.)

3.7.2.8 postSuccess()

Callback called by JSONWebClient.

Parameters

in	err	Error Code from JwcProcessError	
in	json	JsonObject generated by ArduinoJSON	

Returns

Nothing

This is an internal method called by underlying JSONWebClient

Note

Do not call this method.

3.7.2.9 setCallbacks()

Sets callbacks.

Parameters

	in	TBC_CALLBACK_RECEIVE_SIGNATURE	Callback called on receiving a message
ĺ	in	TBC_CALLBACK_ERROR_SIGNATURE	Callback called on error while receiving

Returns

Nothing

sets callbacks for receiving message and error handling

```
3.7.2.10 startPolling()
```

```
void TelegramBotClient::startPolling ( ) [private]
```

Starts polling.

Returns

Nothing

Starts the polling by open a http long call

3.7.2.11 startPosting()

Starts posting a message.

Parameters

```
in The Message to post as json string
```

Returns

Nothing

Start the posting of a message by open a http post call

3.7.3 Member Data Documentation

3.7.3.1 LastUpdateId

```
long TelegramBotClient::LastUpdateId = 0 [private]
```

Id of last update, used to generate a call returning only messages more recent than the last received.

3.7.3.2 Parallel

```
bool TelegramBotClient::Parallel = false [private]
```

Indicates if the client uses two underlying client objects allowing posting while keeping the poll call open in parallel.

3.7.3.3 SslPollClient

```
JsonWebClient* TelegramBotClient::SslPollClient [private]
```

Underlying client for polling.

3.7.3.4 SslPostClient

```
JsonWebClient* TelegramBotClient::SslPostClient [private]
```

Underlying client for posting. In case of parallel mode it uses the same Client object than SsIPollClient

3.7.3.5 TBC_CALLBACK_ERROR_SIGNATURE

```
TelegramBotClient::TBC_CALLBACK_ERROR_SIGNATURE [private]
```

Callback called on error

3.7.3.6 TBC_CALLBACK_RECEIVE_SIGNATURE

```
TelegramBotClient::TBC_CALLBACK_RECEIVE_SIGNATURE [private]
```

Callback called on receiving a message

3.7.3.7 Token

```
String TelegramBotClient::Token [private]
```

Secure Token provided by BotFather

The documentation for this class was generated from the following files:

- · TelegramBotClient.h
- TelegramBotClient.cpp

3.8 TelegramProcessError Class Reference

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

· TelegramBotClient.h

Chapter 4

File Documentation

4.1 JsonWebClient.cpp File Reference

Implementation of a simple web client receiving json uses an underlying implementation of Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

```
#include "JsonWebClient.h"
```

4.1.1 Detailed Description

Implementation of a simple web client receiving json uses an underlying implementation of Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

Part of TelegramBotClient (https://github.com/schlingensiepen/TelegramBotClient) Jörn Schlingensiepen joern@schlingensiepen.com

4.2 JsonWebClient.h File Reference

Header of a simple web client receiving json uses an underlying implementation of Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

```
#include "TBCDebug.h"
#include "Arduino.h"
#include <Client.h>
#include <ArduinoJson.h>
```

Classes

· class JsonWebClient

26 File Documentation

Macros

- · #define JsonWebClient h
- #define JWC BUFF SIZE 10000
- #define JWC_CALLBACK_MESSAGE_SIGNATURE void (*callbackSuccess)(void*, JwcProcessError, JsonObject&)
- #define JWC CALLBACK ERROR SIGNATURE void (*callbackError)(void*, JwcProcessError, Client*)

Enumerations

```
    enum JwcProcessError::int { JwcProcessError::Ok = 0, JwcProcessError::HttpErr = -1, JwcProcessError::MsgTooBig = -2, JwcProcessError::MsgJsonErr = -3 }
    enum JwcClientState: int {
        JwcClientState::Unconnected = 0, JwcClientState::Connected = 1, JwcClientState::Waiting = 2, JwcClientState::Headers = 3, JwcClientState::Json = 4 }
```

4.2.1 Detailed Description

Header of a simple web client receiving json uses an underlying implementation of Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

JSONWebClient (netClient, "www.example.com", 80, CallBackObject, callBackMessage, callBackError);.

JwcClientState state = JwcClientState::Unconnected;.

JwcProcessError state = JwcProcessError::Ok;.

Part of TelegramBotClient (https://github.com/schlingensiepen/TelegramBotClient) Jörn Schlingensiepen joern@schlingensiepen.com

Enumeration to indicate internal process state of JsonWebClient.

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/← TelegramBotClient)

Author

```
Jörn Schlingensiepen joern@schlingensiepen.com
```

This class implements a minimum http client to receive json data from a host. It uses an underlying implementation of Client interface and can be used with raw client or ssl client.

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/ \leftarrow TelegramBotClient)

Author

```
Jörn Schlingensiepen joern@schlingensiepen.com
```

4.2.2 Enumeration Type Documentation

4.2.2.1 JwcClientState

```
enum JwcClientState : int [strong]
```

Enumerator

Unconnected	Client is not connected
Connected	Client is connected but no command was sent.
Waiting	Client is waiting for response from server.
Headers	Client is processing headers.
Json	Client is processing json from response

4.2.2.2 JwcProcessError

```
enum JwcProcessError : int [strong]
```

Enumerator

Ok	Everything Ok, no error
HttpErr	Not found HTTP 200 Header -> Server Error
MsgTooBig	Message bigger than JWC_BUFF_SIZE adjust JWC_BUFF_SIZE to avoid this, beware ArduinoJSON still needs to fit to your device's memory
MsgJsonErr	ArduinoJSON was not able to parse the message

4.3 TelegramBotClient.h File Reference

Header of a simple client sending and receiving message via Telegram's Bot API. Uses one or two underlying objects implementing the Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

```
#include "TBCDebug.h"
#include "Arduino.h"
#include <Client.h>
#include <ArduinoJson.h>
#include "JsonWebClient.h"
```

Classes

- struct Message
- struct TBCKeyBoardRow
- class TBCKeyBoard
- class TelegramBotClient

28 File Documentation

Macros

- #define TelegramBotClient_h
- #define TELEGRAMHOST F("api.telegram.org")
- #define TELEGRAMPORT 443
- #define POLLINGTIMEOUT 600
- #define USERAGENTSTRING F("telegrambotclient /0.1")
- #define TBC_CALLBACK_RECEIVE_SIGNATURE void (*callbackReceive)(TelegramProcessError, JwcProcessError, Message*)
- #define TBC CALLBACK ERROR SIGNATURE void (*callbackError)(TelegramProcessError, JwcProcessError)

Enumerations

enum TelegramProcessError : int {
 TelegramProcessError::Ok = 0, TelegramProcessError::JcwPollErr = -1, TelegramProcessError::JcwPostErr = -2, TelegramProcessError::RetPollErr = -3,
 TelegramProcessError::RetPostErr = -4 }

4.3.1 Detailed Description

Header of a simple client sending and receiving message via Telegram's Bot API. Uses one or two underlying objects implementing the Client interface. It implements a pseudo background behavior by providing a loop() method that can be polled and calls callback on receiving valid data.

Telegram Bot Client.

Class to represent a keyboard used in Telegram chat.

Row in a keyboard.

Telegram Message.

TelegramProcessError state = TelegramProcessError::Ok;.

Part of TelegramBotClient (https://github.com/schlingensiepen/TelegramBotClient) Jörn Schlingensiepen joern@schlingensiepen.com

Enumeration to indicate error or success of processing by TelegramBotClient.

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/ \leftarrow TelegramBotClient)

Author

Jörn Schlingensiepen joern@schlingensiepen.com

Struct to store elements of a Telegram Message (https://core.telegram.org/bots/api#message) and the update_id provided by each callback (https://core.telegram.org/bots/api#getting-updates)

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/← TelegramBotClient)

Author

Jörn Schlingensiepen joern@schlingensiepen.com

Struct to store elements of a Telegram key board

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/← TelegramBotClient)

Author

Jörn Schlingensiepen joern@schlingensiepen.com

This class represents a keyboard that can be displayed in a Telegram chat. Keyboards can be assembled by Rows including buttons. To add a row to a keyboard use push().

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/← TelegramBotClient)

Author

Jörn Schlingensiepen joern@schlingensiepen.com

Client to access Telegram's Bot API

Note

Should only be used as a part of TelegramBotClient (https://github.com/schlingensiepen/ \leftarrow TelegramBotClient)

Author

Jörn Schlingensiepen joern@schlingensiepen.com

4.3.2 Enumeration Type Documentation

4.3.2.1 TelegramProcessError

```
enum TelegramProcessError : int [strong]
```

30 File Documentation

Enumerator

Ok	Everything Ok, no error
JcwPollErr	JSONWebClient host returns error while polling
JcwPostErr	JSONWebClient host returns error while posting
RetPollErr	Telegram host returns error while polling
RetPostErr	Telegram host returns error while posting

Index

\sim TBCKeyBoard	JWC_CALLBACK_MESSAGE_SIGNATURE
TBCKeyBoard, 13	JsonWebClient, 9
\sim TelegramBotClient	JsonWebClient, 5
TelegramBotClient, 18	CallBackObject, 8
	ContentLength, 8
begin	fire, 6
TelegramBotClient, 19	Host, 8
	HttpStatusOk, 8
CallBackObject	JWC_CALLBACK_ERROR_SIGNATURE, 9
JsonWebClient, 8	JWC_CALLBACK_MESSAGE_SIGNATURE, 9
ChatFirstName	JsonWebClient, 6
Message, 10	loop, 7
Chatld	NetClient, 9
Message, 10	Port, 9
ChatLastName	processHeader, 7
Message, 10	processJson, 7
ChatType	•
Message, 11	reConnect, 7
ContentLength	State, 9
JsonWebClient, 8	state, 7
	stop, 8
Date	JsonWebClient.cpp, 25
Message, 11	JsonWebClient.h, 25
	JwcClientState, 26
fire	JwcProcessError, 27
JsonWebClient, 6	JwcClientState, 9
FromFirstName	JsonWebClient.h, 26
Message, 11	JwcProcessError, 10
FromId	JsonWebClient.h, 27
Message, 11	
FromIsBot	LastUpdateId
	TelegramBotClient, 23
Message, 11	length
FromLanguageCode	TBCKeyBoard, 14, 15
Message, 11	loop
FromLastName	JsonWebClient, 7
Message, 11	TelegramBotClient, 19
ant	.o.og.ao.o.i., .o
get	Message, 10
TBCKeyBoard, 13	ChatFirstName, 10
getOneTime	Chatld, 10
TBCKeyBoard, 14	ChatLastName, 10
getResize	ChatType, 11
TBCKeyBoard, 14	
	Date, 11
Host	FromFirstName, 11
JsonWebClient, 8	FromId, 11
HttpStatusOk	FromIsBot, 11
JsonWebClient, 8	FromLanguageCode, 11
	FromLastName, 11
JWC_CALLBACK_ERROR_SIGNATURE	Messageld, 11
JsonWebClient, 9	Text, 12

32 INDEX

Updateld, 12	getOneTime, 14
MessageId	getResize, 14
Message, 11	length, 14, 15
NetClient	OneTime, 15
JsonWebClient, 9	push, 15
	Resize, 15
OneTime	TBCKeyBoard Pow 16
TBCKeyBoard, 15	TBCKeyBoardRow, 16
	TelegramBotClient, 16 ∼TelegramBotClient, 18
Parallel	•
TelegramBotClient, 23	begin, 19 LastUpdateId, 23
pollError	•
TelegramBotClient, 19	loop, 19 Parallel, 23
pollSuccess	pollError, 19
TelegramBotClient, 20	pollSuccess, 20
Port	postError, 20
JsonWebClient, 9	postMessage, 21
postError	postSuccess, 22
TelegramBotClient, 20	setCallbacks, 22
postMessage	SslPollClient, 24
TelegramBotClient, 21	SslPostClient, 24
postSuccess	startPolling, 23
TelegramBotClient, 22	startPosting, 23
processHeader	TBC_CALLBACK_ERROR_SIGNATURE, 24
JsonWebClient, 7	TBC_CALLBACK_RECEIVE_SIGNATURE, 24
processJson	TelegramBotClient, 17, 18
JsonWebClient, 7	Token, 24
push	TelegramBotClient.h, 27
TBCKeyBoard, 15	TelegramProcessError, 29
roConnect	TelegramProcessError, 24
reConnect	TelegramBotClient.h, 29
JsonWebClient, 7	Text
Resize	Message, 12
TBCKeyBoard, 15	Token
setCallbacks	TelegramBotClient, 24
TelegramBotClient, 22	rologiam Botoliom, 21
SsIPollClient	UpdateId
TelegramBotClient, 24	Message, 12
SslPostClient	•
TelegramBotClient, 24	
startPolling	
TelegramBotClient, 23	
startPosting	
TelegramBotClient, 23	
State	
JsonWebClient, 9	
state	
JsonWebClient, 7	
stop	
JsonWebClient, 8	
TBC_CALLBACK_ERROR_SIGNATURE	
TelegramBotClient, 24	
TBC_CALLBACK_RECEIVE_SIGNATURE	
TelegramBotClient, 24	
TBCKeyBoard, 12	
∼TBCKeyBoard, 13	
get, 13	