# Water Level Monitoring Subsystem

Generated by Doxygen 1.10.0

# Chapter 1

# Hierarchical Index

## 1.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

# 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

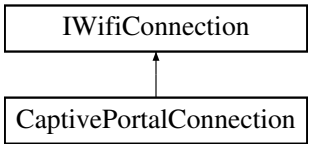
# Class Documentation

## 4.1 CaptivePortalConnection Class Reference

CaptivePortalConnection class. Connect to a free wifi with Captive Portal authentication.

`#include <CaptivePortalConnection.h>`

Inheritance diagram for CaptivePortalConnection:




### Public Member Functions

- CaptivePortalConnection ()
  *Construct a new CaptivePortalConnection object.*
- CaptivePortalConnection (const char ∗ssid, const char ∗captivePortalDomain, const char ∗query)
  *Construct a new CaptivePortalConnection object.*
- void setup_wifi () override
  *Initialize WiFi connection.*
- void setSSID (const char ∗ssid) override
  *Set the SSID of the WiFi network.*
- const char ∗ getSSID () override
  *Get the SSID of the WiFi network.*
- void setCaptive (const char ∗captivePortalDomain)
  *Set the captive portal domain.*
- const char ∗ getCaptive ()
  *Get the captive portal domain.*
- void setQuery (const char ∗query)
  *Set the query for the radius server.*
- const char ∗ getQuery ()
  *Get the query for the radius server.*
- int status () override
  *Check connection status (a WiFi.status() wrapper).*

**Private Attributes**

- const char * **captivePortalDomain**
- const char * **query**

**Additional Inherited Members**

## Protected Attributes inherited from IWifiConnection

- const char * **ssid**

### 4.1.1 Detailed Description

CaptivePortalConnection class. Connect to a free wifi with Captive Portal authentication.

### 4.1.2 Constructor & Destructor Documentation

#### 4.1.2.1 CaptivePortalConnection() [1/2]

```
CaptivePortalConnection::CaptivePortalConnection ( )
```

Construct a new CaptivePortalConnection object.

Default values for ssid, captivePortalDomain and query are used (see ../config.h).

#### 4.1.2.2 CaptivePortalConnection() [2/2]

```
CaptivePortalConnection::CaptivePortalConnection (
            const char * ssid,
            const char * captivePortalDomain,
            const char * query )
```

Construct a new CaptivePortalConnection object.

**Parameters**

| | |
|---|---|
| *ssid* | WiFi network SSID. |
| *captivePortalDomain* | URL or IP of the radius server. |
| *query* | Usually a POST request. |

### 4.1.3 Member Function Documentation

#### 4.1.3.1 getCaptive()

```
const char * CaptivePortalConnection::getCaptive ( )
```

Get the captive portal domain.

**Returns**

char* captivePortalDomain: URL or IP of the radius server.

#### 4.1.3.2 getQuery()

```
const char * CaptivePortalConnection::getQuery ( )
```

Get the query for the radius server.

**Returns**

char* query: Usually a POST request.

#### 4.1.3.3 getSSID()

```
const char * CaptivePortalConnection::getSSID ( )  [override], [virtual]
```

Get the SSID of the WiFi network.

**Returns**

char* ssid: WiFi network SSID.

Implements IWifiConnection.

#### 4.1.3.4 setCaptive()

```
void CaptivePortalConnection::setCaptive (
            const char * captivePortalDomain )
```

Set the captive portal domain.

**Parameters**

| | |
|---|---|
| *captivePortalDomain* | URL or IP of the radius server. |

#### 4.1.3.5 setQuery()

```
void CaptivePortalConnection::setQuery (
            const char * query )
```

Set the query for the radius server.

**Parameters**

| | |
|---|---|
| *query* | Usually a POST request. |

#### 4.1.3.6 setSSID()

```
void CaptivePortalConnection::setSSID (
            const char * ssid )  [override], [virtual]
```

Set the SSID of the WiFi network.

**Parameters**

| | |
|---|---|
| *ssid* | WiFi network SSID. |

Implements IWifiConnection.

#### 4.1.3.7 setup_wifi()

```
void CaptivePortalConnection::setup_wifi ( )  [override], [virtual]
```

Initialize WiFi connection.

Implements IWifiConnection.

#### 4.1.3.8 status()

```
int CaptivePortalConnection::status ( )  [override], [virtual]
```

Check connection status (a WiFi.status() wrapper).

**Returns**

int status: WiFi network status.

Implements IWifiConnection.

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

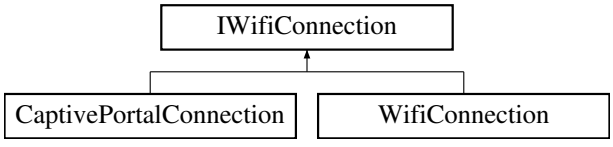
- src/utils/CaptivePortalConnection.h
- src/utils/CaptivePortalConnection.cpp

## 4.2 IWifiConnection Class Reference

Interface for WiFi connection.

```
#include <IWifiConnection.h>
```

Inheritance diagram for IWifiConnection:



**Public Member Functions**

- virtual void setup_wifi ()=0
- virtual void setSSID (const char *ssid)=0
- virtual const char * getSSID ()=0
- virtual int status ()=0

**Protected Attributes**

- const char * **ssid**

### 4.2.1 Detailed Description

Interface for WiFi connection.

### 4.2.2 Member Function Documentation

#### 4.2.2.1 getSSID()

```
virtual const char * IWifiConnection::getSSID ( )  [pure virtual]
```

Implemented in CaptivePortalConnection, and WifiConnection.

#### 4.2.2.2 setSSID()

```
virtual void IWifiConnection::setSSID (
            const char * ssid )  [pure virtual]
```

Implemented in CaptivePortalConnection, and WifiConnection.

#### 4.2.2.3 setup_wifi()

```
virtual void IWifiConnection::setup_wifi ( )  [pure virtual]
```

Implemented in CaptivePortalConnection, and WifiConnection.

#### 4.2.2.4 status()

```
virtual int IWifiConnection::status ( )  [pure virtual]
```

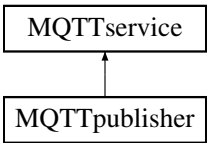
Implemented in CaptivePortalConnection, and WifiConnection.

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

- src/utils/IWifiConnection.h

## 4.3 MQTTpublisher Class Reference

Inheritance diagram for MQTTpublisher:




#### Public Member Functions

- MQTTpublisher (const char ∗server, const char ∗client_id, const char ∗topic)

  *Construct a new MQTTpublisher object.*
- MQTTpublisher (const char ∗server, const char ∗client_id)

  *Construct a new MQTTpublisher object.*
- void publish (const char ∗topic, const char ∗payload)

  *Publish a message to a topic (a PubSubClient::publish wrapper).*
- void publish (const char ∗payload)

  *Publish a message to the default topic (a PubSubClient::publish wrapper).*
- void publishJSON (const char ∗topic, const char ∗JSONfield, const char ∗JSONvalue)

  *Publish a JSON message to a topic.*

### Public Member Functions inherited from MQTTservice

- MQTTservice (const char ∗server, const char ∗client_id, const char ∗topic)

  *Construct a new MQTTservice object.*
- MQTTservice (const char ∗server, const char ∗client_id)

  *Construct a new MQTTservice object.*
- void **connect** ()

  *Connect to MQTT broker.*
- void **loop** ()

  *Loop function (a PubSubClient::loop wrapper).*
- bool connected ()

  *Check if the client is connected to the MQTT broker (a PubSubClient::connected wrapper).*

#### Additional Inherited Members

### Protected Attributes inherited from MQTTservice

- const char ∗ **mqtt_server**
- const char ∗ **mqtt_client_id**
- const char ∗ **mqtt_topic**
- WiFiClient **espClient**
- PubSubClient **mqttClient**

### 4.3.1 Constructor & Destructor Documentation

#### 4.3.1.1 MQTTpublisher() [1/2]

```
MQTTpublisher::MQTTpublisher (
            const char * server,
            const char * client_id,
            const char * topic )  [inline]
```

Construct a new MQTTpublisher object.

**Parameters**

| | |
|---|---|
| *server* | The MQTT broker server. |
| *client↩ _id* | The MQTT client ID. |
| *topic* | The MQTT topic. |

#### 4.3.1.2 MQTTpublisher() [2/2]

```
MQTTpublisher::MQTTpublisher (
            const char * server,
            const char * client_id )  [inline]
```

Construct a new MQTTpublisher object.

**Parameters**

| | |
|---|---|
| *server* | The MQTT broker server. |
| *client↩ _id* | The MQTT client ID. |

### 4.3.2 Member Function Documentation

#### 4.3.2.1 publish() [1/2]

```
void MQTTpublisher::publish (
            const char * payload )
```

Publish a message to the default topic (a PubSubClient::publish wrapper).

**Parameters**

| | |
|---|---|
| *payload* | The payload to publish. |

#### 4.3.2.2 publish() [2/2]

```
void MQTTpublisher::publish (
            const char * topic,
            const char * payload )
```

Publish a message to a topic (a PubSubClient::publish wrapper).

**Parameters**

| | |
|---|---|
| *topic* | The topic to publish to. |
| *payload* | The payload to publish. |

#### 4.3.2.3 publishJSON()

```
void MQTTpublisher::publishJSON (
            const char * topic,
            const char * JSONfield,
            const char * JSONvalue )
```

Publish a JSON message to a topic.

**Parameters**

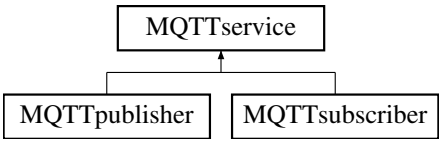
| | |
|---|---|
| *topic* | The topic to publish to. |
| *JSONfield* | The field of the JSON message. |
| *JSONvalue* | The value of the JSON message. |

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

- src/api/MQTTpublisher.h
- src/impl/MQTTpublisher.cpp

## 4.4 MQTTservice Class Reference

Inheritance diagram for MQTTservice:




### Public Member Functions

- MQTTservice (const char ∗server, const char ∗client_id, const char ∗topic)
  *Construct a new MQTTservice object.*
- MQTTservice (const char ∗server, const char ∗client_id)
  *Construct a new MQTTservice object.*
- void **connect** ()
  *Connect to MQTT broker.*
- void **loop** ()
  *Loop function (a PubSubClient::loop wrapper).*
- bool connected ()
  *Check if the client is connected to the MQTT broker (a PubSubClient::connected wrapper).*

### Protected Attributes

- const char ∗ **mqtt_server**
- const char ∗ **mqtt_client_id**
- const char ∗ **mqtt_topic**
- WiFiClient **espClient**
- PubSubClient **mqttClient**

### 4.4.1 Constructor & Destructor Documentation

#### 4.4.1.1 MQTTservice() [1/2]

```
MQTTservice::MQTTservice (
            const char * server,
            const char * client_id,
            const char * topic )  [inline]
```

Construct a new MQTTservice object.

**Parameters**

| | |
|---|---|
| *server* | The MQTT broker server. |
| *client_id* | The MQTT client ID. |
| *topic* | The MQTT topic. |

#### 4.4.1.2 MQTTservice() [2/2]

```
MQTTservice::MQTTservice (
            const char * server,
            const char * client_id )  [inline]
```

Construct a new MQTTservice object.

**Parameters**

| | |
|---|---|
| *server* | The MQTT broker server. |
| *client_id* | The MQTT client ID. |

### 4.4.2 Member Function Documentation

#### 4.4.2.1 connected()

```
bool MQTTservice::connected ( )
```

Check if the client is connected to the MQTT broker (a PubSubClient::connected wrapper).

**Returns**

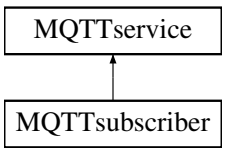
true if connected, false otherwise.

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

- src/api/MQTTservice.h
- src/impl/MQTTservice.cpp

## 4.5 MQTTsubscriber Class Reference

Inheritance diagram for MQTTsubscriber:




### Public Member Functions

- MQTTsubscriber (const char *server, const char *client_id)

  *Construct a new MQTTsubscriber object.*
- void subscribe (const char *topic)

  *Subscribe to a topic.*
- void callback (char *topic, byte *payload, unsigned int length)

  *Callback function for MQTT messages (see PubSubClient::callback).*
- void subscribeJSON (const char *topic)

  *Subscribe to a topic and parse the payload as JSON (a PubSubClient::subscribe wrapper).*
- void callbackJSON (char *topic, byte *payload, unsigned int length)

  *Callback function for MQTT messages (see PubSubClient::callback).*
- void setSavedPayload (byte *payload)

  *Set the local payload.*
- byte * getSavedPayload ()

  *Get the local payload.*
- void setSavedPayload (int payload)

  *Set the local int payload.*
- int getSavedPayloadInt ()

  *Get the local int payload.*

### Public Member Functions inherited from MQTTservice

- MQTTservice (const char *server, const char *client_id, const char *topic)

  *Construct a new MQTTservice object.*
- MQTTservice (const char *server, const char *client_id)

  *Construct a new MQTTservice object.*
- void **connect** ()

  *Connect to MQTT broker.*
- void **loop** ()

  *Loop function (a PubSubClient::loop wrapper).*
- bool connected ()

  *Check if the client is connected to the MQTT broker (a PubSubClient::connected wrapper).*

### Private Attributes

- byte * **saved_payload**
- int **saved_payload_int**

**Additional Inherited Members**

## Protected Attributes inherited from MQTTservice

- const char * **mqtt_server**
- const char * **mqtt_client_id**
- const char * **mqtt_topic**
- WiFiClient **espClient**
- PubSubClient **mqttClient**

### 4.5.1 Constructor & Destructor Documentation

#### 4.5.1.1 MQTTsubscriber()

```
MQTTsubscriber::MQTTsubscriber (
            const char * server,
            const char * client_id )  [inline]
```

Construct a new MQTTsubscriber object.

**Parameters**

| | |
|---|---|
| *server* | The MQTT broker server. |
| *client_id* | The MQTT client ID. |

### 4.5.2 Member Function Documentation

#### 4.5.2.1 callback()

```
void MQTTsubscriber::callback (
            char * topic,
            byte * payload,
            unsigned int length )
```

Callback function for MQTT messages (see PubSubClient::callback).

**Parameters**

| | |
|---|---|
| *topic* | The topic of the message. |
| *payload* | The payload of the message. |
| *length* | The length of the payload. |

#### 4.5.2.2 callbackJSON()

```
void MQTTsubscriber::callbackJSON (
            char * topic,
```

```
            byte * payload,
            unsigned int length )
```

Callback function for MQTT messages (see PubSubClient::callback).

**Parameters**

| | |
|---|---|
| *topic* | The topic of the message. |
| *payload* | The payload of the message. |
| *length* | The length of the payload. |

#### 4.5.2.3 getSavedPayload()

```
byte * MQTTsubscriber::getSavedPayload ( )
```

Get the local payload.

**Returns**

byte∗ : The local payload.

#### 4.5.2.4 getSavedPayloadInt()

```
int MQTTsubscriber::getSavedPayloadInt ( )
```

Get the local int payload.

**Returns**

int : The local int payload.

#### 4.5.2.5 setSavedPayload() [1/2]

```
void MQTTsubscriber::setSavedPayload (
            byte * payload )
```

Set the local payload.

**Parameters**

| | |
|---|---|
| *payload* | The payload to save. |

#### 4.5.2.6 setSavedPayload() [2/2]

```
void MQTTsubscriber::setSavedPayload (
            int payload )
```

Set the local int payload.

**Parameters**

| | |
|---|---|
| *payload* | The payload to save. |

#### 4.5.2.7 subscribe()

```
void MQTTsubscriber::subscribe (
            const char * topic )
```

Subscribe to a topic.

**Parameters**

| | |
|---|---|
| *topic* | The topic to subscribe to. |

#### 4.5.2.8 subscribeJSON()

```
void MQTTsubscriber::subscribeJSON (
            const char * topic )
```

Subscribe to a topic and parse the payload as JSON (a PubSubClient::subscribe wrapper).

**Parameters**

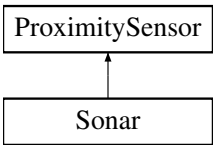
| | |
|---|---|
| *topic* | The topic to subscribe to. |

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

- src/api/MQTTsubscriber.h
- src/impl/MQTTsubscriber.cpp

## 4.6 ProximitySensor Class Reference

Inheritance diagram for ProximitySensor:



#### Public Member Functions

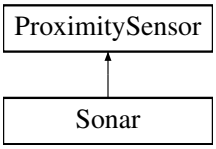
- virtual float **getDistance** ()=0

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

- src/api/ProximitySensor.h

## 4.7 Sonar Class Reference

Inheritance diagram for Sonar:




#### Public Member Functions

- **Sonar** (int echoPin, int trigPin, long maxTime)
- float getDistance ()
- void **setTemperature** (float temp)

#### Private Member Functions

- float **getSoundSpeed** ()

#### Private Attributes

- const float **vs** = 331.5 + 0.6∗20
- float **temperature**
- int **echoPin**
- int **trigPin**
- long **timeOut**

### 4.7.1 Member Function Documentation

#### 4.7.1.1 getDistance()

```
float Sonar::getDistance ( )  [virtual]
```

Implements ProximitySensor.

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

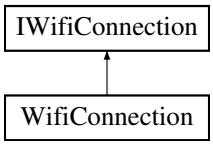
- src/api/Sonar.h
- src/impl/Sonar.cpp

## 4.8 WifiConnection Class Reference

WifiConnection class Connect to a wifi network, with or without password.

`#include <WifiConnection.h>`

Inheritance diagram for WifiConnection:




### Public Member Functions

- WifiConnection ()
  *Construct a new WifiConnection object.*
- WifiConnection (const char *ssid)
  *Construct a new WifiConnection object.*
- WifiConnection (const char *ssid, const char *password)
  *Construct a new WifiConnection object.*
- void setup_wifi () override
  *Initialize WiFi connection.*
- void setSSID (const char *ssid) override
  *Set the SSID object.*
- const char * getSSID () override
  *Get the SSID object.*
- void setPassword (const char *password)
  *Set the Password object.*
- int status () override
  *Check connection status (a WiFi.status() wrapper).*

### Private Attributes

- const char * **password**

### Additional Inherited Members

### Protected Attributes inherited from IWifiConnection

- const char * **ssid**

### 4.8.1 Detailed Description

WifiConnection class Connect to a wifi network, with or without password.

## 4.8.2 Constructor & Destructor Documentation

### 4.8.2.1 WifiConnection() [1/3]

```
WifiConnection::WifiConnection ( )
```

Construct a new WifiConnection object.

Default values for ssid are used (see ../config.h)

### 4.8.2.2 WifiConnection() [2/3]

```
WifiConnection::WifiConnection (
            const char * ssid )
```

Construct a new WifiConnection object.

**Parameters**

| | |
|---|---|
| *ssid* | WiFi network SSID |

### 4.8.2.3 WifiConnection() [3/3]

```
WifiConnection::WifiConnection (
            const char * ssid,
            const char * password )
```

Construct a new WifiConnection object.

**Parameters**

| | |
|---|---|
| *ssid* | WiFi network SSID |
| *password* | WiFi network password |

## 4.8.3 Member Function Documentation

### 4.8.3.1 getSSID()

```
const char * WifiConnection::getSSID ( )  [override], [virtual]
```

Get the SSID object.

**Returns**

char* ssid: WiFi network SSID.

Implements IWifiConnection.

#### 4.8.3.2 setPassword()

```
void WifiConnection::setPassword (
            const char * password )
```

Set the Password object.

**Parameters**

| | |
|---|---|
| *password* | WiFi network password. |

#### 4.8.3.3 setSSID()

```
void WifiConnection::setSSID (
            const char * ssid )  [override], [virtual]
```

Set the SSID object.

**Parameters**

| | |
|---|---|
| *ssid* | WiFi network SSID. |

Implements IWifiConnection.

#### 4.8.3.4 setup_wifi()

```
void WifiConnection::setup_wifi ( )  [override], [virtual]
```

Initialize WiFi connection.

Implements IWifiConnection.

#### 4.8.3.5 status()

```
int WifiConnection::status ( )  [override], [virtual]
```

Check connection status (a WiFi.status() wrapper).

**Returns**

int status: WiFi network status.

Implements IWifiConnection.

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

- src/utils/WifiConnection.h
- src/utils/WifiConnection.cpp

# Chapter 5

# File Documentation

## 5.1 MQTTpublisher.h

```
00001 #include "MQTTservice.h"
00002
00003 class MQTTpublisher : public MQTTservice
00004 {
00005 public:
00006
00013     MQTTpublisher(const char *server, const char *client_id, const char *topic)
00014         : MQTTservice(server, client_id, topic) {}
00015
00021     MQTTpublisher(const char *server, const char *client_id)
00022         : MQTTservice(server, client_id) {}
00023
00029     void publish(const char *topic, const char *payload);
00030
00035     void publish(const char *payload);
00036
00043     void publishJSON(const char *topic, const char *JSONfield, const char *JSONvalue);
00044 };
```

## 5.2 MQTTservice.h

```
00001 #ifndef MQTT_SERVICE_H
00002 #define MQTT_SERVICE_H
00003
00004 #include <PubSubClient.h>
00005 #include <WiFiClient.h>
00006
00007 class MQTTservice
00008 {
00009 protected:
00010     const char *mqtt_server;
00011     const char *mqtt_client_id;
00012     const char *mqtt_topic;
00013
00014     WiFiClient espClient;
00015     PubSubClient mqttClient;
00016
00017 public:
00018
00025     MQTTservice(const char *server, const char *client_id, const char *topic)
00026         : mqtt_server(server), mqtt_client_id(client_id), mqtt_topic(topic), mqttClient(espClient) {}
00027
00033     MQTTservice(const char *server, const char *client_id)
00034         : mqtt_server(server), mqtt_client_id(client_id), mqttClient(espClient) {}
00035
00039     void connect();
00040
00044     void loop();
00045
00050     bool connected();
00051 };
00052
00053 #endif // MQTT_SERVICE_H
```

## 5.3 MQTTsubscriber.h

```
00001 #include "MQTTservice.h"
00002
00003 class MQTTsubscriber : public MQTTservice
00004 {
00005 public:
00006
00012     MQTTsubscriber(const char *server, const char *client_id)
00013         : MQTTservice(server, client_id) {}
00014
00019     void subscribe(const char *topic);
00020
00027     void callback(char *topic, byte *payload, unsigned int length);
00028
00033     void subscribeJSON(const char *topic);
00034
00041     void callbackJSON(char *topic, byte *payload, unsigned int length);
00042
00047     void setSavedPayload(byte *payload);
00048
00053     byte *getSavedPayload();
00054
00059     void setSavedPayload(int payload);
00060
00065     int getSavedPayloadInt();
00066
00067 private:
00068     byte *saved_payload;
00069     int saved_payload_int;
00070 };
```

## 5.4 ProximitySensor.h

```
00001 #ifndef __PROXIMITYSENSOR__
00002 #define __PROXIMITYSENSOR__
00003
00004 class ProximitySensor {
00005
00006 public:
00007   virtual float getDistance() = 0;
00008
00009 };
00010
00011 #endif
```

## 5.5 Sonar.h

```
00001 #ifndef __SONAR__
00002 #define __SONAR__
00003
00004 #include "ProximitySensor.h"
00005
00006 #define NO_OBJ_DETECTED -1
00007
00008 class Sonar: public ProximitySensor {
00009
00010 public:
00011   Sonar(int echoPin, int trigPin, long maxTime);
00012   float getDistance();
00013   void setTemperature(float temp);
00014
00015 private:
00016     const float vs = 331.5 + 0.6*20;
00017     float getSoundSpeed();
00018
00019     float temperature;
00020     int echoPin, trigPin;
00021     long timeOut;
00022 };
00023
00024 #endif
```

## 5.6 config.h

```
00001 #ifndef CONFIG_H
```

```
00002 #define CONFIG_H
00003
00004 /* LEDs PINs */
00005 #define GREEN_LED 19
00006 #define RED_LED 21
00007
00008 /* sonar sensor info */
00009 #define ECHO_PIN 5
00010 #define TRIG_PIN 18
00011 #define SONAR_TIMER 10000
00012
00013 /* mqtt buffers dimensions*/
00014 #define FREQ_MSG_SIZE 16
00015 #define SONAR_MSG_SIZE 16
00016 //#define __CAPTIVE_PORTAL /* Comment this line to use a regular WiFi network */
00017
00018 extern const char *captive_ssid;
00019 extern const char *captive_PortalDomain;
00020 extern const char *captive_query;
00021
00022
00023 extern const char *ssid;
00024 extern const char *password;
00025
00026 /* MQTT server address */
00027 extern const char *default_mqtt_server; /* MQTT server IP address/hostname */
00028
00029 /* MQTT topics */
00030 extern const char *freq_topic;
00031 extern const char *wl_topic;
00032
00033 /* JSON fields */
00034 extern const char *frequency_field;
00035 extern const char *water_level_field;
00036
00037 #endif /* CONFIG_H */
```

## 5.7 CaptivePortalConnection.h

```
00001 // CaptivePortalConnection.h
00002 #ifndef CAPTIVE_PORTAL_CONNECTION_H
00003 #define CAPTIVE_PORTAL_CONNECTION_H
00004
00005 #include "IWifiConnection.h"
00006
00011 class CaptivePortalConnection : public IWifiConnection
00012 {
00013 public:
00021     CaptivePortalConnection();
00022
00023     ~CaptivePortalConnection() = default;
00024
00033     CaptivePortalConnection(const char *ssid, const char *captivePortalDomain, const char *query);
00034
00039     void setup_wifi() override;
00040
00046     void setSSID(const char *ssid) override;
00047
00052     const char *getSSID() override;
00053
00060     void setCaptive(const char *captivePortalDomain);
00061
00066     const char *getCaptive();
00067
00073     void setQuery(const char *query);
00074
00079     const char *getQuery();
00080
00085     int status() override;
00086
00087 private:
00088     const char *captivePortalDomain;
00089     const char *query;
00090 };
00091
00092 #endif // CAPTIVE_PORTAL_CONNECTION_H
```

## 5.8 IWifiConnection.h

```
00001 // IWifiConnection.h
```

```
00002 #ifndef IWIFI_CONNECTION_H
00003 #define IWIFI_CONNECTION_H
00004
00009 class IWifiConnection
00010 {
00011 public:
00012   virtual ~IWifiConnection() = default;
00013   virtual void setup_wifi() = 0;
00014   virtual void setSSID(const char *ssid) = 0;
00015   virtual const char *getSSID() = 0;
00016   virtual int status() = 0;
00017
00018 protected:
00019   const char *ssid;
00020 };
00021
00022 #endif // IWIFI_CONNECTION_H
```

## 5.9 WifiConnection.h

```
00001 // WifiConnection.h
00002 #ifndef WIFI_CONNECTION_H
00003 #define WIFI_CONNECTION_H
00004
00005 #include "IWifiConnection.h"
00006
00011 class WifiConnection : public IWifiConnection
00012 {
00013 public:
00014
00022     WifiConnection();
00023
00028     WifiConnection(const char *ssid);
00029
00035     WifiConnection(const char *ssid, const char *password);
00036     ~WifiConnection() = default;
00037
00042     void setup_wifi() override;
00043
00048     void setSSID(const char *ssid) override;
00049
00054     const char *getSSID() override;
00055
00060     void setPassword(const char *password);
00061
00066     int status() override;
00067
00068 private:
00069     const char *password;
00070 };
00071
00072 #endif // WIFI_CONNECTION_H
```

# Index