Sortiermaschiene

Generated by Doxygen 1.8.17

Chapter 1

Main Page

1.1 Einleitung

Hallo, dies ist die Dokumentation für den Code der Sortiermaschiene

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

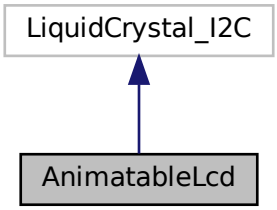
Chapter 5

Class Documentation

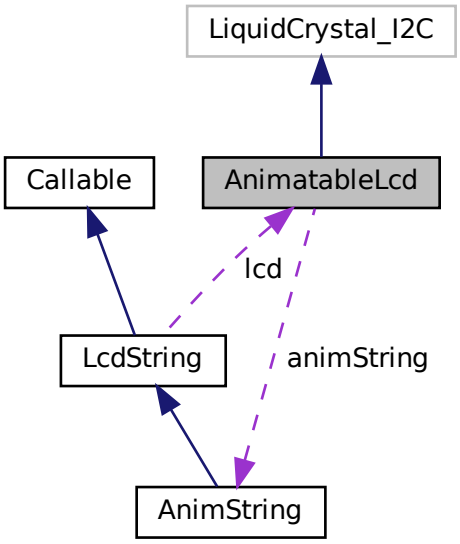
5.1 AnimatableLcd Class Reference

```
#include <animLcd.h>
```

Inheritance diagram for AnimatableLcd:



Collaboration diagram for AnimatableLcd:



Public Member Functions

- void setAnimation (AnimString *_animString)

  setzt die aktuelle animation
- void printCentered (String text, int length=-1, int row=0)

  Gibt einen String zentriert auf dem Lcd-Display aus.
- void printPretty (String text)

  gibt den Text "schön" aus, das heißt zentriert und mit automatischen Zeilenumbrüchen
- void update ()

  wird immer wieder von loop aufgerufen um die Animationen zu updaten
- void init ()

  Überschreibt die normale lcd init function.
- void print (const String &text)

  Eigene Lcd-print funktion, die die Möglichkeit bietet eigene Characters in den Text einzufügen.

Public Attributes

- bool doAnimation = false

Private Attributes

- AnimString * animString

5.1.1 Member Function Documentation

5.1.1.1 init()

```
void AnimatableLcd::init ( )
```

Überschreibt die normale lcd init function.

5.1.1.2 print()

```
void AnimatableLcd::print (
            const String & text )
```

Eigene Lcd-print funktion, die die Möglichkeit bietet eigene Characters in den Text einzufügen.

für eigene Character einfach die nummer des Characters in den Text einfügen **(\1n für den nten Character)**, \1 für leerzeichen, das nicht in Zeilenumbruch resultiert

Parameters

text	

5.1.1.3 printCentered()

```
void AnimatableLcd::printCentered (
            String text,
            int length = -1,
            int row = 0 )
```

Gibt einen String zentriert auf dem Lcd-Display aus.

Parameters

text	
length	Länge des Textes, wird neu berechnet wenn nicht angegeben
row	Zeile in der der Text ausgegeben werden soll

5.1.1.4 printPretty()

```
void AnimatableLcd::printPretty (
            String text )
```

gibt den Text "schön" aus, das heißt zentriert und mit automatischen Zeilenumbrüchen

Parameters

text	

5.1.1.5 setAnimation()

```
void AnimatableLcd::setAnimation (
            AnimString * _animString )
```

setzt die aktuelle animation

Parameters

_animString	

5.1.1.6 update()

```
void AnimatableLcd::update ( )
```

wird immer wieder von loop aufgerufen um die Animationen zu updaten

5.1.2 Member Data Documentation

5.1.2.1 animString

```
AnimString* AnimatableLcd::animString  [private]
```

5.1.2.2 doAnimation

```
bool AnimatableLcd::doAnimation = false
```

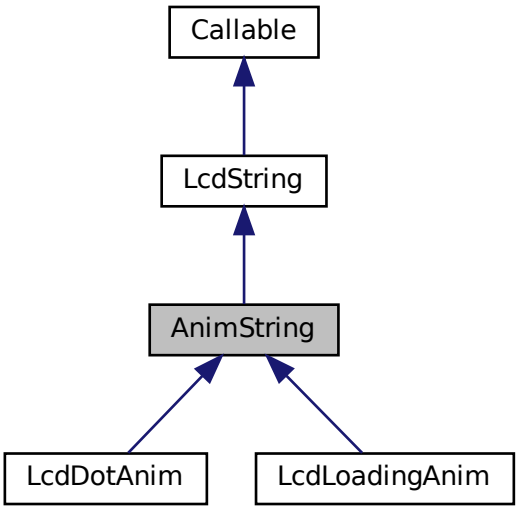
The documentation for this class was generated from the following files:

- animLcd.h
- animLcd.ino

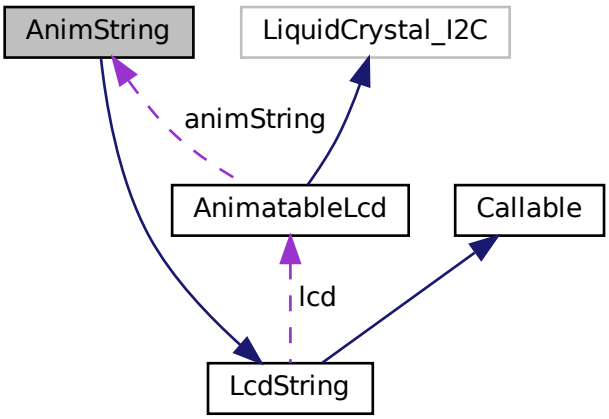
5.2 AnimString Class Reference

`#include <animString.h>`

Inheritance diagram for AnimString:



Collaboration diagram for AnimString:



Public Member Functions

- virtual ~AnimString ()
- virtual void init ()
- void run ()

  gibt den String auf dem Lcd-Display aus
- LcdString (String text, AnimatableLcd *lcd, time_t duration=0)

Protected Attributes

- time_t stepDuration
- time_t animStart
- time_t lastRefresh

Additional Inherited Members

5.2.1 Constructor & Destructor Documentation

5.2.1.1 ~AnimString()

```
virtual AnimString::~AnimString ( )  [inline], [virtual]
```

5.2.2 Member Function Documentation

5.2.2.1 init()

```
virtual void AnimString::init ( )  [inline], [virtual]
```

Reimplemented in LcdDotAnim, and LcdLoadingAnim.

5.2.2.2 LcdString()

```
LcdString::LcdString  [inline]
```

5.2.2.3 run()

```
void AnimString::run ( )  [virtual]
```

gibt den String auf dem Lcd-Display aus

Reimplemented from LcdString.

5.2.3 Member Data Documentation

5.2.3.1 animStart

```
time_t AnimString::animStart  [protected]
```

5.2.3.2 lastRefresh

```
time_t AnimString::lastRefresh  [protected]
```

5.2.3.3 stepDuration

```
time_t AnimString::stepDuration  [protected]
```

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

- animString.h
- animString.ino

5.3 ButtonHandler Class Reference

Public Member Functions

- ButtonHandler ()
- ButtonHandler (int pin, void(∗onclick)())
- void update ()

Public Attributes

- void(∗ onclick)()

Private Attributes

- int pin
- bool isPressed = false

5.3.1 Constructor & Destructor Documentation

5.3.1.1 ButtonHandler() [1/2]

```
ButtonHandler::ButtonHandler ( )  [inline]
```

5.3.1.2 ButtonHandler() [2/2]

```
ButtonHandler::ButtonHandler (
            int pin,
            void(*)() onclick )  [inline]
```

5.3.2 Member Function Documentation

5.3.2.1 update()

```
void ButtonHandler::update ( )  [inline]
```

5.3.3 Member Data Documentation

5.3.3.1 isPressed

```
bool ButtonHandler::isPressed = false  [private]
```

5.3.3.2 onclick

```
void(* ButtonHandler::onclick) ()
```

5.3.3.3 pin

```
int ButtonHandler::pin  [private]
```

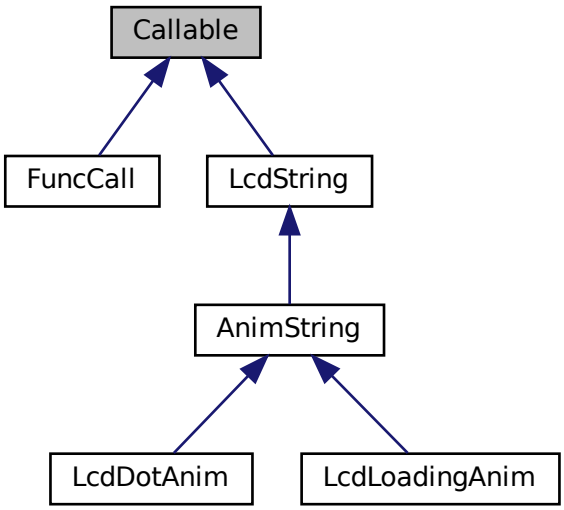
The documentation for this class was generated from the following file:

- sketch.ino

5.4 Callable Struct Reference

```
#include <animString.h>
```

Inheritance diagram for Callable:



Public Member Functions

- virtual void run ()
- virtual bool isDone ()
- virtual ~Callable ()

5.4.1 Constructor & Destructor Documentation

5.4.1.1 ~Callable()

```
virtual Callable::~Callable ( )  [inline], [virtual]
```

5.4.2 Member Function Documentation

5.4.2.1 isDone()

```
virtual bool Callable::isDone ( )  [inline], [virtual]
```

Reimplemented in LcdString, and FuncCall.

5.4.2.2 run()

```
virtual void Callable::run ( )  [inline], [virtual]
```

Reimplemented in AnimString, LcdString, and FuncCall.

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

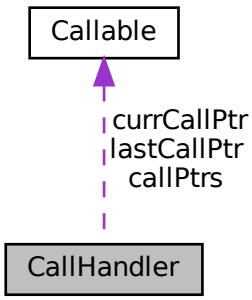
- animString.h

5.5 CallHandler Class Reference

Ermöglicht es Calls wie z.B. Funktionen nacheinander aufzurufen, ohne die delay() Funktion zu verwenden.

```
#include <callHandler.h>
```

Collaboration diagram for CallHandler:



Public Member Functions

- void deleteCalls ()

  setzt den Speicherplatz der von den Calls besetzt wurde frei
- void setCalls (Callable ∗newCallPtrs[ ], size_t nCalls)

  Setzt die neuen Calls, die ausgeführt werden sollen.
- virtual ∼ CallHandler ()
- void update ()

  wird von loop aufgerufen, ermöglicht es zu überprüfen, wechselt zum nächsten Call, wenn der Aktuelle vorbei ist

Public Attributes

- bool running = false

Private Attributes

- Callable ∗∗ callPtrs
- Callable ∗∗ currCallPtr
- Callable ∗∗ lastCallPtr
- time_t lastCallT
- bool callsSet = false

5.5.1 Detailed Description

Ermöglicht es Calls wie z.B. Funktionen nacheinander aufzurufen, ohne die delay() Funktion zu verwenden.

5.5.2 Constructor & Destructor Documentation

5.5.2.1 ∼ CallHandler()

```
virtual CallHandler::~ CallHandler ( )  [virtual]
```

5.5.3 Member Function Documentation

5.5.3.1 deleteCalls()

```
void CallHandler::deleteCalls ( )
```

setzt den Speicherplatz der von den Calls besetzt wurde frei

5.5.3.2 setCalls()

```
void CallHandler::setCalls (
            Callable * newCallPtrs[],
            size_t nCalls )
```

Setzt die neuen Calls, die ausgeführt werden sollen.

Parameters

newCallPtrs	
nCalls	

5.5.3.3 update()

```
void CallHandler::update ( )
```

wird von loop aufgerufen, ermöglicht es zu überprüfen, wechselt zum nächsten Call, wenn der Aktuelle vorbei ist

5.5.4 Member Data Documentation

5.5.4.1 callPtrs

```
Callable** CallHandler::callPtrs  [private]
```

5.5.4.2 callsSet

```
bool CallHandler::callsSet = false  [private]
```

5.5.4.3 currCallPtr

```
Callable** CallHandler::currCallPtr  [private]
```

5.5.4.4 lastCallPtr

```
Callable** CallHandler::lastCallPtr  [private]
```

5.5.4.5 lastCallT

```
time_t CallHandler::lastCallT  [private]
```

5.5.4.6 running

```
bool CallHandler::running = false
```

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

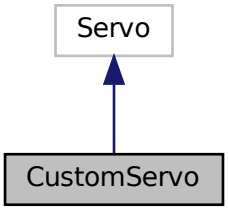
- callHandler.h
- callHandler.ino

5.6 CustomServo Class Reference

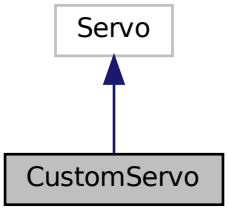
Eine Eigene Servo-Klasse, die es ermöglicht den Servo mit verschiedenen Geschwindigkeiten zu bewegen.

```
#include <customServo.h>
```

Inheritance diagram for CustomServo:



Collaboration diagram for CustomServo:



Public Member Functions

- void write (short newAngle)

  bewegt den Servo mit einer vorher spezifizierten Geschwindigkeit
- void write (short newAngle, time_t duration)

  *bewegt den Servo in **duration** ms*
- void writeDirect (short angle)
- void setSpeed (float newSpeed)
- void updatePos ()
- void stop ()

  stoppt den Servo
- void start ()

  lässt den Servo weiterlaufen
- bool isDone ()

  Gibt an, ob der Servo angekommen ist.

Public Attributes

- bool done

Private Member Functions

- void startMove ()

  Setzt Variabeln, die benötigt werden um den Servo zu bewegen.

Private Attributes

- short startAngle
- short targetAngle
- float speed
- time_t startTime
- Servo servo

5.6.1 Detailed Description

Eine Eigene Servo-Klasse, die es ermöglicht den Servo mit verschiedenen Geschwindigkeiten zu bewegen.

5.6.2 Member Function Documentation

5.6.2.1 isDone()

```
bool CustomServo::isDone ( )
```

Gibt an, ob der Servo angekommen ist.

Returns

true
false

5.6.2.2 setSpeed()

```
void CustomServo::setSpeed (
            float newSpeed )
```

5.6.2.3 start()

```
void CustomServo::start ( )
```

lässt den Servo weiterlaufen

5.6.2.4 startMove()

```
void CustomServo::startMove ( )  [private]
```

Setzt Variabeln, die benötigt werden um den Servo zu bewegen.

5.6.2.5 stop()

```
void CustomServo::stop ( )
```

stoppt den Servo

5.6.2.6 updatePos()

```
void CustomServo::updatePos ( )
```

5.6.2.7 write() [1/2]

```
void CustomServo::write (
            short newAngle )
```

bewegt den Servo mit einer vorher spezifizierten Geschwindigkeit

Parameters

newAngle	

5.6.2.8 write() [2/2]

```
void CustomServo::write (
            short newAngle,
            time_t duration )
```

bewegt den Servo in **duration** ms

Parameters

newAngle	
duration	

5.6.2.9 writeDirect()

```
void CustomServo::writeDirect (
            short angle )
```

5.6.3 Member Data Documentation

5.6.3.1 done

```
bool CustomServo::done
```

5.6.3.2 servo

```
Servo CustomServo::servo  [private]
```

5.6.3.3 speed

```
float CustomServo::speed  [private]
```

5.6.3.4 startAngle

```
short CustomServo::startAngle  [private]
```

5.6.3.5 startTime

```
time_t CustomServo::startTime  [private]
```

5.6.3.6 targetAngle

```
short CustomServo::targetAngle  [private]
```

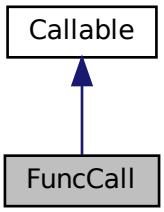
The documentation for this class was generated from the following files:

- customServo.h
- customServo.ino

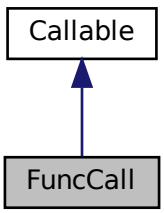
5.7 FuncCall Struct Reference

```
#include <animString.h>
```

Inheritance diagram for FuncCall:



Collaboration diagram for FuncCall:



Public Member Functions

- FuncCall (func_t< void > call, func_t< bool > _isDone)
- FuncCall (func_t< void > call)
- virtual ~FuncCall ()
- void run ()

  ruft die angegebene Funktion auf
- bool isDone ()

  Gibt zurück, ob der nächste Call ausgeführt werden sollte.

Public Attributes

- func_t< void > call
- func_t< bool > _isDone

5.7.1 Constructor & Destructor Documentation

5.7.1.1 FuncCall() [1/2]

```
FuncCall::FuncCall (
            func_t< void > call,
            func_t< bool > _isDone )  [inline]
```

5.7.1.2 FuncCall() [2/2]

```
FuncCall::FuncCall (
            func_t< void > call )  [inline]
```

5.7.1.3 ~FuncCall()

```
virtual FuncCall::~FuncCall ( )  [inline], [virtual]
```

5.7.2 Member Function Documentation

5.7.2.1 isDone()

```
bool FuncCall::isDone ( )  [virtual]
```

Gibt zurück, ob der nächste Call ausgeführt werden sollte.

Returns

true

false

Reimplemented from Callable.

5.7.2.2 run()

```
void FuncCall::run ( )  [virtual]
```

ruft die angegebene Funktion auf

Reimplemented from Callable.

5.7.3 Member Data Documentation

5.7.3.1 _isDone

```
func_t<bool> FuncCall::_isDone
```

5.7.3.2 call

```
func_t<void> FuncCall::call
```

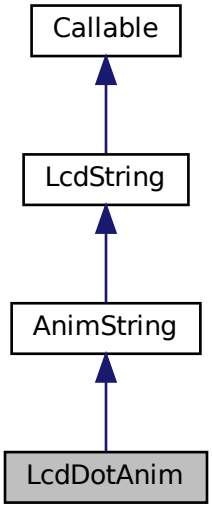
The documentation for this struct was generated from the following files:

- animString.h
- animString.ino

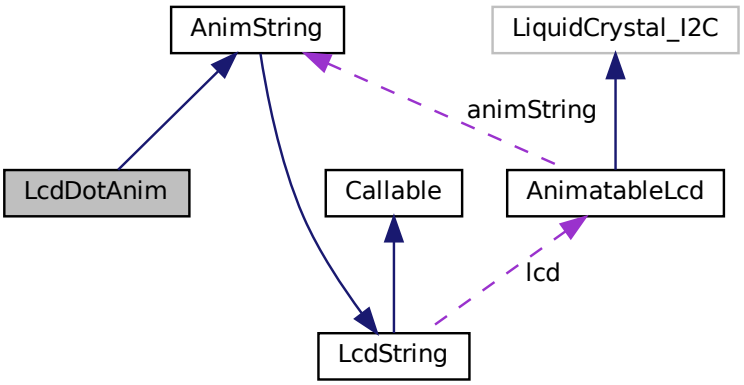
5.8 LcdDotAnim Class Reference

`#include <animString.h>`

Inheritance diagram for LcdDotAnim:



Collaboration diagram for LcdDotAnim:



Public Member Functions

- LcdDotAnim (String text, AnimatableLcd *lcd, time_t duration=0, time_t _stepDuration=500)
- void init ()
- void update ()

Additional Inherited Members

5.8.1 Constructor & Destructor Documentation

5.8.1.1 LcdDotAnim()

```
LcdDotAnim::LcdDotAnim (
            String text,
            AnimatableLcd * lcd,
            time_t duration = 0,
            time_t _stepDuration = 500 )  [inline]
```

5.8.2 Member Function Documentation

5.8.2.1 init()

```
void LcdDotAnim::init ( )  [virtual]
```

Reimplemented from AnimString.

5.8.2.2 update()

```
void LcdDotAnim::update ( )  [virtual]
```

Reimplemented from LcdString.

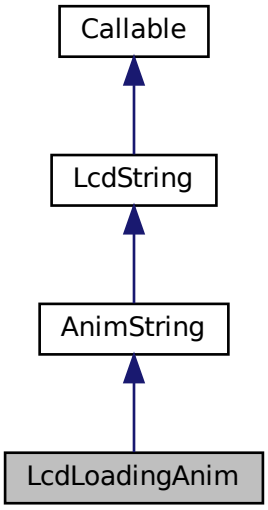
The documentation for this class was generated from the following files:

- animString.h
- animString.ino

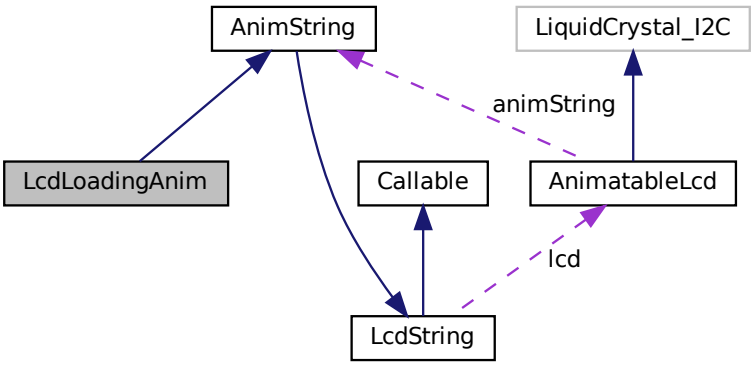
5.9 LcdLoadingAnim Class Reference

`#include <animString.h>`

Inheritance diagram for LcdLoadingAnim:



Collaboration diagram for LcdLoadingAnim:



Public Member Functions

- void init ()
- void update ()

Additional Inherited Members

5.9.1 Member Function Documentation

5.9.1.1 init()

```
void LcdLoadingAnim::init ( )  [virtual]
```

Reimplemented from AnimString.

5.9.1.2 update()

```
void LcdLoadingAnim::update ( )  [virtual]
```

Reimplemented from LcdString.

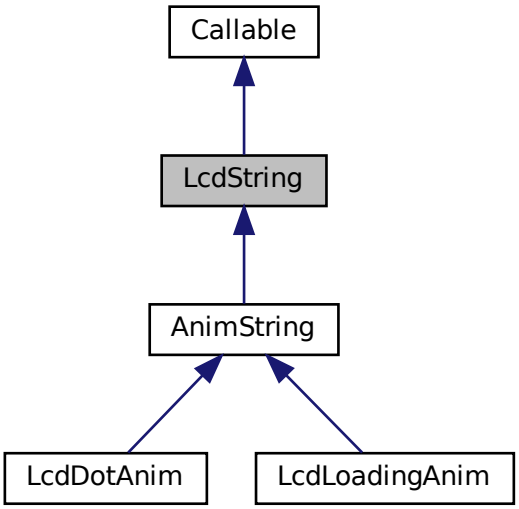
The documentation for this class was generated from the following files:

- animString.h
- animString.ino

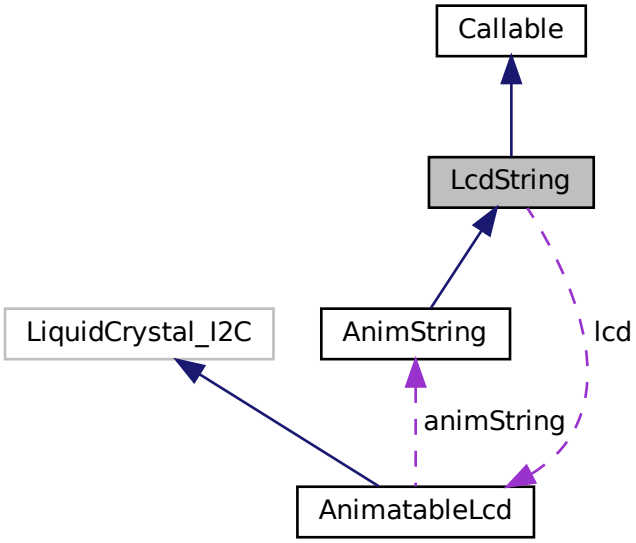
5.10 LcdString Struct Reference

```
#include <animString.h>
```

Inheritance diagram for LcdString:



Collaboration diagram for LcdString:



Public Member Functions

- LcdString (String text, AnimatableLcd *lcd, time_t duration=0)
- virtual ~LcdString ()
- bool isDone ()

  Gibt zurück, ob die duration überschritten ist.
- virtual void run ()

  gibt den String auf dem Lcd-Display aus
- virtual void update ()

Public Attributes

- String text
- AnimatableLcd * lcd
- time_t duration
- time_t callStart

5.10.1 Constructor & Destructor Documentation

5.10.1.1 LcdString()

```
LcdString::LcdString (
            String text,
            AnimatableLcd * lcd,
            time_t duration = 0 )  [inline]
```

5.10.1.2 ~LcdString()

```
virtual LcdString::~LcdString ( )  [inline], [virtual]
```

5.10.2 Member Function Documentation

5.10.2.1 isDone()

```
bool LcdString::isDone ( )  [virtual]
```

Gibt zurück, ob die duration überschritten ist.

Returns

true
false

Reimplemented from Callable.

5.10.2.2 run()

```
void LcdString::run ( )  [virtual]
```

gibt den String auf dem Lcd-Display aus

Reimplemented from Callable.

Reimplemented in AnimString.

5.10.2.3 update()

```
virtual void LcdString::update ( )  [inline], [virtual]
```

Reimplemented in LcdDotAnim, and LcdLoadingAnim.

5.10.3 Member Data Documentation

5.10.3.1 callStart

```
time_t LcdString::callStart
```

5.10.3.2 duration

```
time_t LcdString::duration
```

5.10.3.3 lcd

```
AnimatableLcd* LcdString::lcd
```

5.10.3.4 text

```
String LcdString::text
```

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

- animString.h
- animString.ino

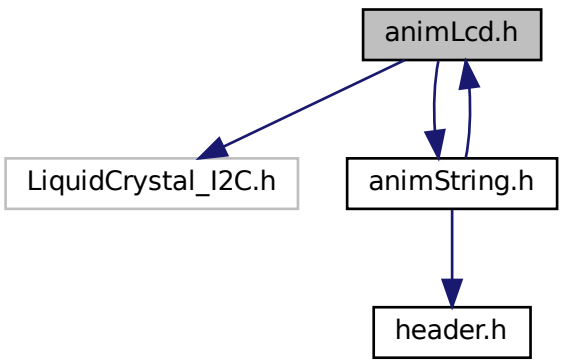
Chapter 6

File Documentation

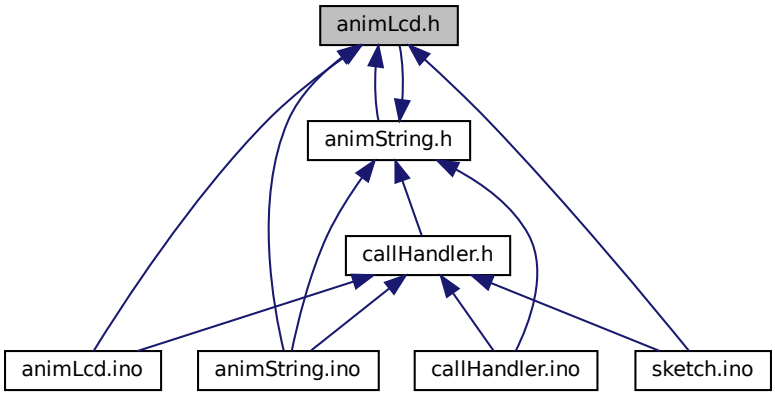
6.1 animLcd.h File Reference

```
#include <LiquidCrystal_I2C.h>
#include "animString.h"
```

Include dependency graph for animLcd.h:



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



Classes

- class AnimatableLcd

Variables

- const int LOADING_BAR_OFFSET = 2

6.1.1 Variable Documentation

6.1.1.1 LOADING_BAR_OFFSET

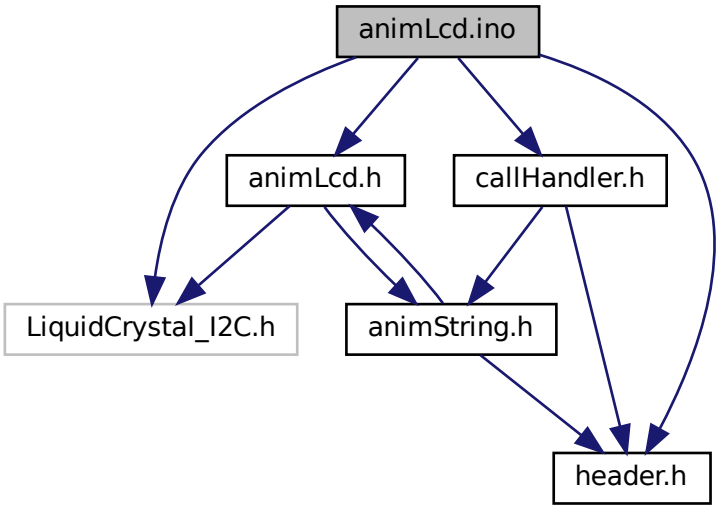
```
const int LOADING_BAR_OFFSET = 2
```

6.2 animLcd.ino File Reference

```
#include <LiquidCrystal_I2C.h>
#include "header.h"
#include "animLcd.h"
```

```
#include "callHandler.h"
```

Include dependency graph for animLcd.ino:



Variables

- const byte loading_empty_c [8]
- const byte loading_full_c [8]

6.2.1 Detailed Description

Author

Arne de Borman

Version

0.1

Date

2022-05-26

6.2.2 Variable Documentation

6.2.2.1 loading_empty_c

```
const byte loading_empty_c[8]
```

Initial value:

```
= {
  B11111,
  B10001,
  B10001,
  B10001,
  B10001,
  B10001,
  B10001,
  B11111
}
```

6.2.2.2 loading_full_c

```
const byte loading_full_c[8]
```

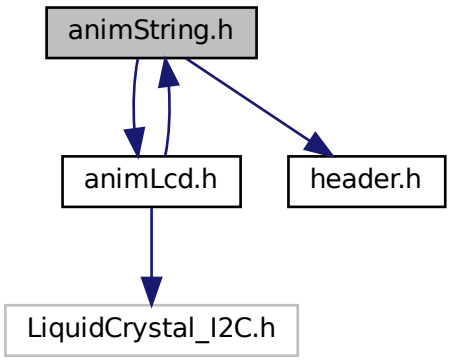
Initial value:

```
= {
  B11111,
  B11111,
  B11111,
  B11111,
  B11111,
  B11111,
  B11111,
  B11111
}
```

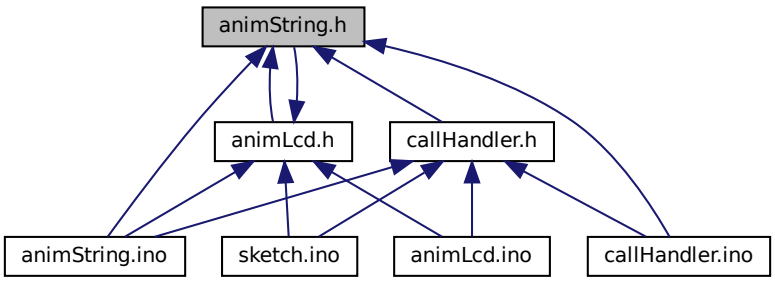
6.3 animString.h File Reference

```
#include "animLcd.h"
#include "header.h"
```

Include dependency graph for animString.h:



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



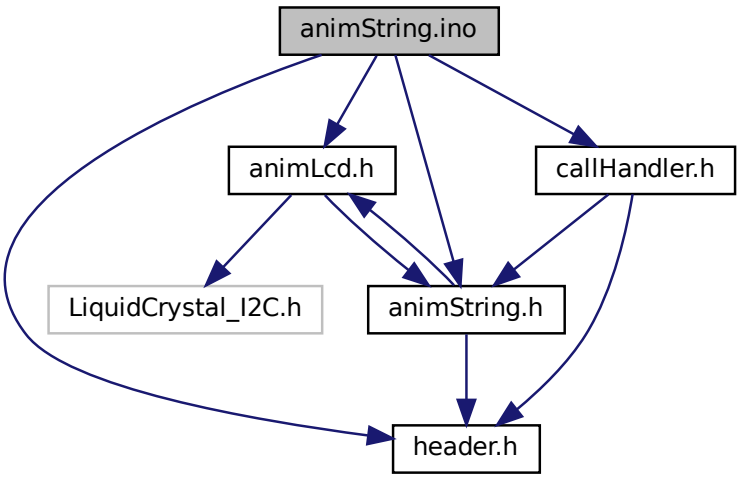
Classes

- struct Callable
- struct FuncCall
- struct LcdString
- class AnimString
- class LcdLoadingAnim
- class LcdDotAnim

6.4 animString.ino File Reference

```
#include "animLcd.h"
#include "callHandler.h"
#include "header.h"
#include "animString.h"
```

Include dependency graph for animString.ino:



6.4.1 Detailed Description

Author

Arne de Borman

Version

0.1

Date

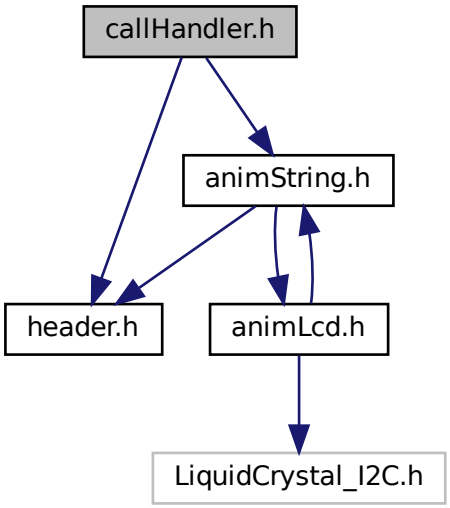
2022-05-26

6.5 callHandler.h File Reference

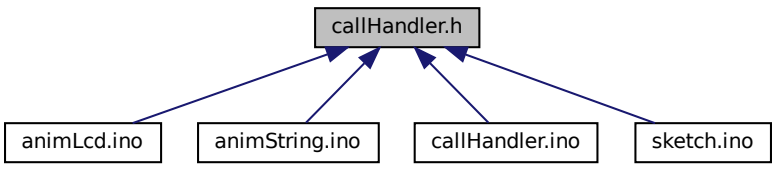
header datei für den CallHandler

```
#include "header.h"
#include "animString.h"
```

Include dependency graph for callHandler.h:



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



Classes

- class CallHandler

  Ermöglicht es Calls wie z.B. Funktionen nacheinander aufzurufen, ohne die delay() Funktion zu verwenden.

6.5.1 Detailed Description

header datei für den CallHandler

Author

your name (you@domain.com)

Version

0.1

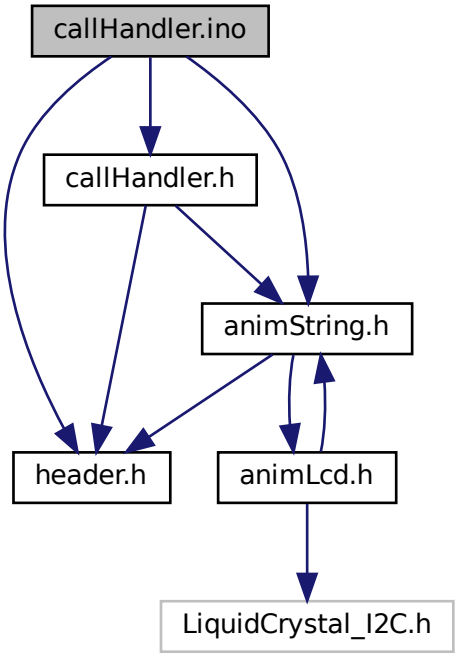
Date

2022-05-26

6.6 callHandler.ino File Reference

```
#include "header.h"
#include "animString.h"
#include "callHandler.h"
```

Include dependency graph for callHandler.ino:



6.6.1 Detailed Description

Author

Arne de Borman

Version

0.1

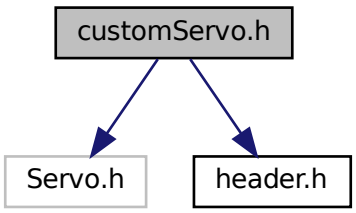
Date

2022-05-26

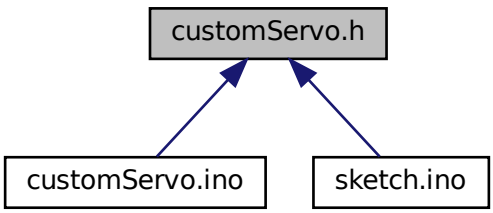
6.7 customServo.h File Reference

```
#include <Servo.h>
#include "header.h"
```

Include dependency graph for customServo.h:



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



Classes

- class CustomServo

  Eine Eigene Servo-Klasse, die es ermöglicht den Servo mit verschiedenen Geschwindigkeiten zu bewegen.

6.7.1 Detailed Description

Version

0.1

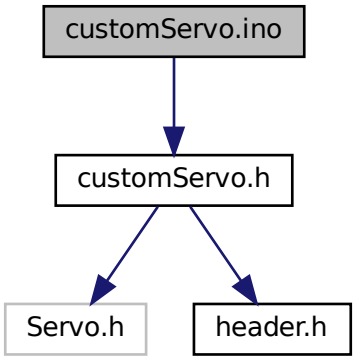
Date

2022-05-26

6.8 customServo.ino File Reference

```
#include "customServo.h"
```

Include dependency graph for customServo.ino:



6.8.1 Detailed Description

Author

Arne de borman

Version

0.1

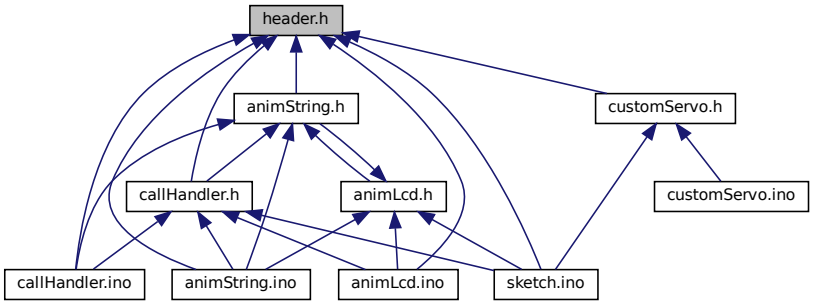
Date

2022-05-26

6.9 header.h File Reference

definiert variablen-types die überall im Programm benutzt werden

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



Typedefs

- using time_t = unsigned long

  Ein Zeit typ.
- template<typename ReturnT = void>
  using func_t = ReturnT(∗)()

  ein Funktions-typ

6.9.1 Detailed Description

definiert variablen-types die überall im Programm benutzt werden

Version

0.1

Date

2022-05-26

6.9.2 Typedef Documentation

6.9.2.1 func_t

```
template<typename ReturnT = void>
using func_t = ReturnT(*)()
```

ein Funktions-typ

Template Parameters

ReturnT	der Rückgabewert der Funktion die referenziert wird

6.9.2.2 time_t

```
using time_t = unsigned long
```

Ein Zeit typ.

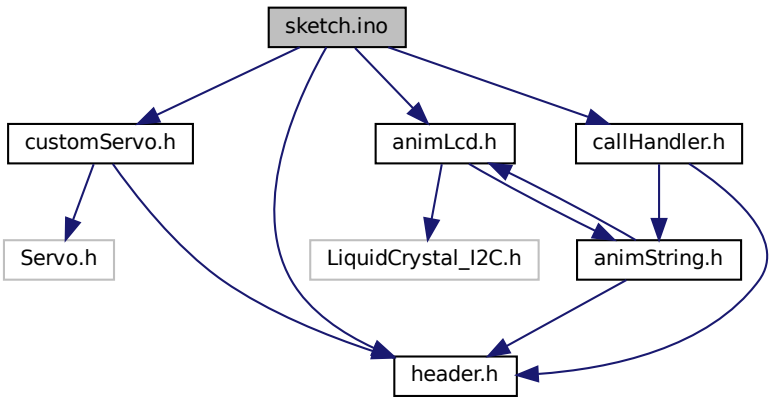
6.10 main.c File Reference

6.11 sketch.ino File Reference

Hauptdatei.

```
#include "customServo.h"
#include "animLcd.h"
#include "callHandler.h"
#include "header.h"
```

Include dependency graph for sketch.ino:



Classes

- class ButtonHandler

Macros

- #define GEH_ZURUECK

Enumerations

- enum Farbe { WHITE, BLACK, ORANGE, NOTHING }

Functions

- Farbe mesureColor ()
- void setLedColor (unsigned char r, unsigned char g, unsigned char b)
- void stopButtonClicked ()

  wird ausgeführt wenn der Stop-Knopf gedrückt wird
- bool servoIsDone ()

  hilf Function, Methoden können nicht als functionsparameter benutzt werden
- void setup ()

  wird am Anfang des Programms aufgerufen
- void loop ()

  wird immer wieder ausgeführt

Variables

- const int LOADING_DURATION = 3000
- const int ANGLE_LEFT_HOLE = 180
- const int ANGLE_RIGHT_HOLE = 90
- const int ANGLE_CENTER = 130
- const int ANGLE_MIN = 45
- const int PIN_SERVO = 6
- const int PIN_STOPBUTTON = 13
- const int PIN_RED = 11
- const int PIN_GREEN = 10
- const int PIN_BLUE = 9
- const float SERVO_SPEED_DEFAULT = 0.01f
- const float SERVO_SPEED_FAST = 0.5f
- AnimatableLcd lcd (0x27, 16, 2)
- CallHandler callHandler
- CustomServo servo
- int nWhite = 0
- int nBlack = 0
- int nOrange = 0
- bool doFlicker =false
- ButtonHandler stopButton

  der Stop Knopf

6.11.1 Detailed Description

Hauptdatei.

Author

Arne de Borman

Version

0.1

Date

2022-05-26

6.11.2 Macro Definition Documentation

6.11.2.1 GEH_ZURUECK

```
#define GEH_ZURUECK
```

Value:

```
  new LcdDotAnim("Gehe zur\365ck",&lcd),\
  new FuncCall([](){\
    servo.setSpeed(SERVO_SPEED_FAST);\
    servo.write(ANGLE_CENTER);\
  },&servoIsDone)
```

6.11.3 Enumeration Type Documentation

6.11.3.1 Farbe

```
enum Farbe
```

Enumerator

WHITE	
BLACK	
ORANGE	
NOTHING	

6.11.4 Function Documentation

6.11.4.1 loop()

```
void loop ( )
```

wird immer wieder ausgeführt

6.11.4.2 mesureColor()

```
Farbe mesureColor ( )
```

6.11.4.3 servoIsDone()

```
bool servoIsDone ( )
```

hilf Function, Methoden können nicht als functionsparameter benutzt werden

Returns

true

false

6.11.4.4 setLedColor()

```
void setLedColor (
            unsigned char r,
            unsigned char g,
            unsigned char b )
```

6.11.4.5 setup()

```
void setup ( )
```

wird am Anfang des Programms aufgerufen

6.11.4.6 stopButtonClicked()

```
void stopButtonClicked ( )
```

wird ausgeführt wenn der Stop-Knopf gedrückt wird

6.11.5 Variable Documentation

6.11.5.1 ANGLE_CENTER

```
const int ANGLE_CENTER = 130
```

6.11.5.2 ANGLE_LEFT_HOLE

```
const int ANGLE_LEFT_HOLE = 180
```

6.11.5.3 ANGLE_MIN

```
const int ANGLE_MIN = 45
```

6.11.5.4 ANGLE_RIGHT_HOLE

```
const int ANGLE_RIGHT_HOLE = 90
```

6.11.5.5 callHandler

```
CallHandler callHandler
```

6.11.5.6 doFlicker

```
bool doFlicker =false
```

6.11.5.7 lcd

```
AnimatableLcd lcd(0x27, 16, 2)
```

6.11.5.8 LOADING_DURATION

```
const int LOADING_DURATION = 3000
```

6.11.5.9 nBlack

```
int nBlack = 0
```

6.11.5.10 nOrange

```
int nOrange = 0
```

6.11.5.11 nWhite

```
int nWhite = 0
```

6.11.5.12 PIN_BLUE

```
const int PIN_BLUE = 9
```

6.11.5.13 PIN_GREEN

```
const int PIN_GREEN = 10
```

6.11.5.14 PIN_RED

```
const int PIN_RED = 11
```

6.11.5.15 PIN_SERVO

```
const int PIN_SERVO = 6
```

6.11.5.16 PIN_STOPBUTTON

```
const int PIN_STOPBUTTON = 13
```

6.11.5.17 servo

```
CustomServo servo
```

6.11.5.18 SERVO_SPEED_DEFAULT

```
const float SERVO_SPEED_DEFAULT = 0.01f
```

6.11.5.19 SERVO_SPEED_FAST

```
const float SERVO_SPEED_FAST = 0.5f
```

6.11.5.20 stopButton

```
ButtonHandler stopButton
```

der Stop Knopf

6.12 wokwi-project.txt File Reference

Index