Creating Classes and Libraries with Arduino

Hans-Petter Halvorsen

https://www.halvorsen.blog
Contents

• We will learn how we can create our own Arduino Libraries from Scratch

• Why create your own Libraries?
  – Better Code structure
  – Reuse your Code in different Applications
  – Distribute to others
Fahrenheit Example

• We will create code that convert from degrees Celsius to degrees Fahrenheit (and the opposite)
void setup()
{
  float Tf;
  float Tc;

  Serial.begin(9600);

  Tc = 0;
  Tf = Tc * 9/5 + 32;
  Serial.println(Tf);

  Tf=32;
  Tc = (Tf-32)*(5/9);
  Serial.println(Tc);
}

void loop()
{
}
Creating Functions

Why Creating Functions?

• In order to structure your code better
• You can reuse your Code
Creating Functions

```cpp
void setup()
{
    float c;
    float f;

    Serial.begin(9600);

    c = 0;
    f = c2f(c);
    Serial.println(f);

    f = 32;
    c = f2c(f);
    Serial.println(c);
}

void loop()
{
}

float c2f(float Tc)
{
    float Tf;
    Tf = Tc * 9/5 + 32;
    return Tf;
}

float f2c(float Tf)
{
    float Tc;
    Tc = (Tf-32)*(5/9);
    return Tc;
}
```
Creating Classes

- Next, I will show how you can group your functions into a Class
- A class is simply a collection of functions and variables that are all kept together in one place
class Fahrenheit
{
    public:

    Fahrenheit()
    {
    }

    float c2f(float Tc){
        float Tf;
        Tf = Tc * 9/5 + 32;
        return Tf;
    }

    float f2c(float Tf){
        float Tc;
        Tc = (Tf-32)*(5/9);
        return Tc;
    }

    void setup()
    {
        float f;
        float c;
        Serial.begin(9600);
        Fahrenheit fahr;
        c = 0;
        f = fahr.c2f(c);
        Serial.println(f);
        f = 32;
        c = fahr.f2c(f);
        Serial.println(c);
    }

    void loop()
    {
    }
};

Creating Classes

The functions and variables can be either private and public

- **public**: they can be accessed by people using your library
- **private**: meaning they can only be accessed from within the class itself

- Each class has a special function known as a **constructor**, which is used to create an instance of the class.
- The constructor has the same name as the class, and no return type.
Running the Program
Arduino Libraries

• Libraries are a collection of code that makes it easy for you to connect to a sensor, display, module, etc.

• There are hundreds of additional libraries available on the Internet for download.

• You can also create your own Libraries from scratch – That’s what we will show here
Arduino Libraries

Why create your own Libraries?

• Better Code structure
• Reuse your Code in different Applications
• Distribute to others
Arduino Libraries

You need at least two files for a library:

• Header file (.h) - The header file has definitions for the library

• Source file (.cpp) – The Functions within the Class

Note the Library Name, Folder name, .h and .cpp files all need to have the same name
Arduino Libraries

Location:

• Windows: C:\Users\hansha\Documents\Arduino\libraries
• macOS: /Users/hansha/Documents/Arduino
/*
Fahrenheit.h - Library converting between Celsius and Fahrenheit.
Created by Hans-Petter Halvorsen, 2018
*/

#ifndef Fahrenheit_h
#define Fahrenheit_h

#include "Arduino.h"

class Fahrenheit{
public:
    Fahrenheit();
    float c2f(float Tc);
    float f2c(float Tf);
};

#endif

/*
Fahrenheit.cpp - Library converting between Celsius and Fahrenheit.
Created by Hans-Petter Halvorsen, 2018
*/

#include "Fahrenheit.h"

Fahrenheit::Fahrenheit(){
}

float Fahrenheit::c2f(float Tc){
    float Tf;
    Tf = Tc * 9/5 + 32;
    return Tf;
}

float Fahrenheit::f2c(float Tf){
    float Tc;
    Tc = (Tf-32)*(5/9);
    return Tc;
}
Creating Libraries

```cpp
/*
  Fahrenheit.h - Library converting between Celsius and Fahrenheit.
  Created by Hans-Petter Halvorsen, 2018
*/

#ifndef FAHRENHEIT_H
#define FAHRENHEIT_H

#include "Arduino.h"

class Fahrenheit{
  public:
    Fahrenheit();
    float c2f(float Tc);
    float f2c(float Tf);
  }

float Fahrenheit::c2f(float Tc){
  float Tf;
  Tf = Tc * 9/5 + 32;
  return Tf;
}

float Fahrenheit::f2c(float Tf){
  float Tc;
  Tc = (Tf-32) * (5/9);
  return Tc;
}
#endif
```
```c
#include <Fahrenheit.h>

Fahrenheit fahr;

void setup()
{
  float f;
  float c;

  Serial.begin(9600);

  c = 0;
  f = fahr.c2f(c);
  Serial.println(f);

  f = 32;
  c = fahr.f2c(f);
  Serial.println(c);
}

void loop()
{
}
```
Deploying the Library

The Arduino Libraries need to be in the following folder (but can be changed from File-Preferences):

C:\Users\hansha\Documents\Arduino\libraries

Here you need to put your .h and .cpp files.
You should also create an “examples” folder where you include one or more examples showing how to use your Library.
Using the Library

When the Library has been installed properly, you should see your Library under “Sketch->Include Library”

Your Library Examples can be found under File->Examples
Using the Library

```c
#include <Fahrenheit.h>

Fahrenheit fahr;

void setup()
{
    float f;
    float c;
    Serial.begin(9600);
    c = 0;
    f = fahr.c2f(c);
    Serial.println(f);
    f = 32;
    c = fahr.f2c(f);
    Serial.println(c);
}

void loop()
{
}
```

Output:
```
32.00
0.00
```

Sketch uses 3326 bytes (10%) of program storage space. Maximum is 32768 bytes.
Global variables use 200 bytes (9%) of dynamic memory, leaving 14482 bytes free.
References


• Writing a Library for Arduino: https://www.arduino.cc/en/Hacking/LibraryTutorial

• How to write libraries for the Arduino? http://playground.arduino.cc/Code/Library
Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no
Web: https://www.halvorsen.blog