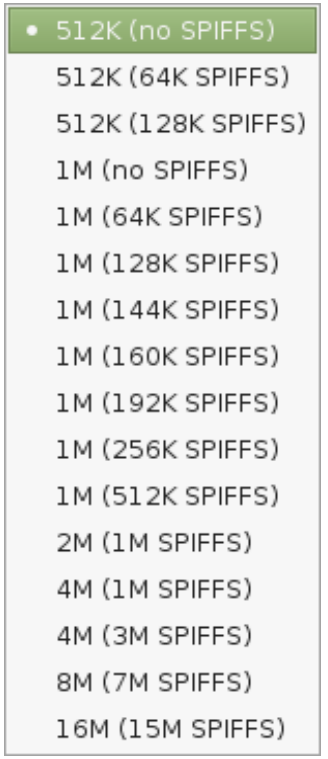
File System Storage

File System Storage

- There are times when storing data in a file can be useful. The data can be collected from sensors or can be configuration data. Whatever it is, the FS library provides a way to allocate some of the SLATE program memory for storing files.
- The SLATE processor board can be set up to support a small file system. Details can be found at
  - https://arduino-esp8266.readthedocs.io/en/2.4.2/filesystem.html
- There are some limitations. Directories are not supported. It can be faked by including slashes in the file name. The other limitation is that a total of 31 characters including slashes can be used in the file name. You have to be careful to make sure you do not exceed the 31 characters as the compiler will not detect that and strange things can happen.
- You program can create, write read, delete and rename files.

File System Storage

- SLATE boards purchased before Fall 2018 have 1Mbyte of Program memory. New SLATES have 4Mbytes. This memory can be divided between the program and the file system.
- In the Tools menu, select the **Flash Size** menu Item. A selection of memory configurations are provided. The file system size is indicated by the SPIFFS in parenthesis. For the 1Mbyte SLATE board, up to half the memory can be allocated for the file system. The 4MByte SLATE can have up to ¾ of the memory allocated for the file system.
- Select one. Start with 128K for the 1MByte SLATE and 1M for the 4MByte slate.



File System Uploader

- There is a tool that can be added to the Arduino IDE to let you upload files. You can create files for your program to use and upload them after uploading your program.
- Go to https://github.com/esp8266/arduino-esp8266fs-plugin/releases/tag/0.3.0

  and download ESP8266FS-0.3.0.zip. It is possible the 0.3.0 may change since the writing of this document.
- In the Arduino sketchbook directory where all your programs are stored, called Arduino, create a directory named **tools**. The Arduino directory should be in the computers Documents directory.
- Move the zip file to the tools directory. Unzip it in that directory. It will create **ESP8266FS/tool** directories.
- Close the Arduino IDE and restart it. In the Tools menu, you should see **ESP8266 Sketch Data Upload**
- Remember, you have to upload your program with the filesystem space configured before uploading files.

Creating a File

- First thing to do is include the SPIFFS library.
- In **setup()**, the file system is initialized with **SPIFFS.begin()**. This must be done before accessing any files.
- This example, the file writing and reading are done in **setup()** so it is only executed once.

```
#include <FS.h>

void setup()
{
  Serial.begin(115200);
  SPIFFS.begin();
  File f = SPIFFS.open(“/file.txt”,”w”);
  f.println(“This is the first line”);
  f.println(“This is the second line”);
  f.close();
  delay(1000);
  f = SPIFFS.open(“/file.txt”,”r”);
  while(f.available()) {
    String line = f.readStringUntil('\n');
    Serial.println(line);
  }
  f.close();
}

void loop()
{
}
```

Creating and Writing to a File

- First, a file is created by opening a file for writing. object **f** is used to access the file. **File** is the object for connecting to the file in the file system.
- The first argument is the file name. The forward slash is required and is used to indicate the top of the file system.
- **“w”** indicates writing to the file.
- **f.println()** writes to the file similar to how **Serial.println()** works to the serial terminal.
- **println(), print(), write()** functions can be used to send data to the file.
- Lastly, the file needs to be closed.

```
#include <FS.h>

void setup()
{
  Serial.begin(115200);
  SPIFFS.begin();
  File f = SPIFFS.open(“/file.txt”,”w”);
  f.println(“This is the first line”);
  f.println(“This is the second line”);
  f.close();
  delay(1000);
  f = SPIFFS.open(“/file.txt”,”r”);
  while(f.available()) {
    String line = f.readStringUntil('\n');
    Serial.println(line);
  }
  f.close();
}

void loop()
{
}
```

Reading From a File

- The second half is reading the file that was just written.
- The file is opened using the same file name and replacing **“w”** with **“r”**.
- Then a while loop is used to read through the whole file one line at a time. While there is data to read from the file, the while loop keeps executing. Once the end of the file is reached, the while loop exits and the file is closed.

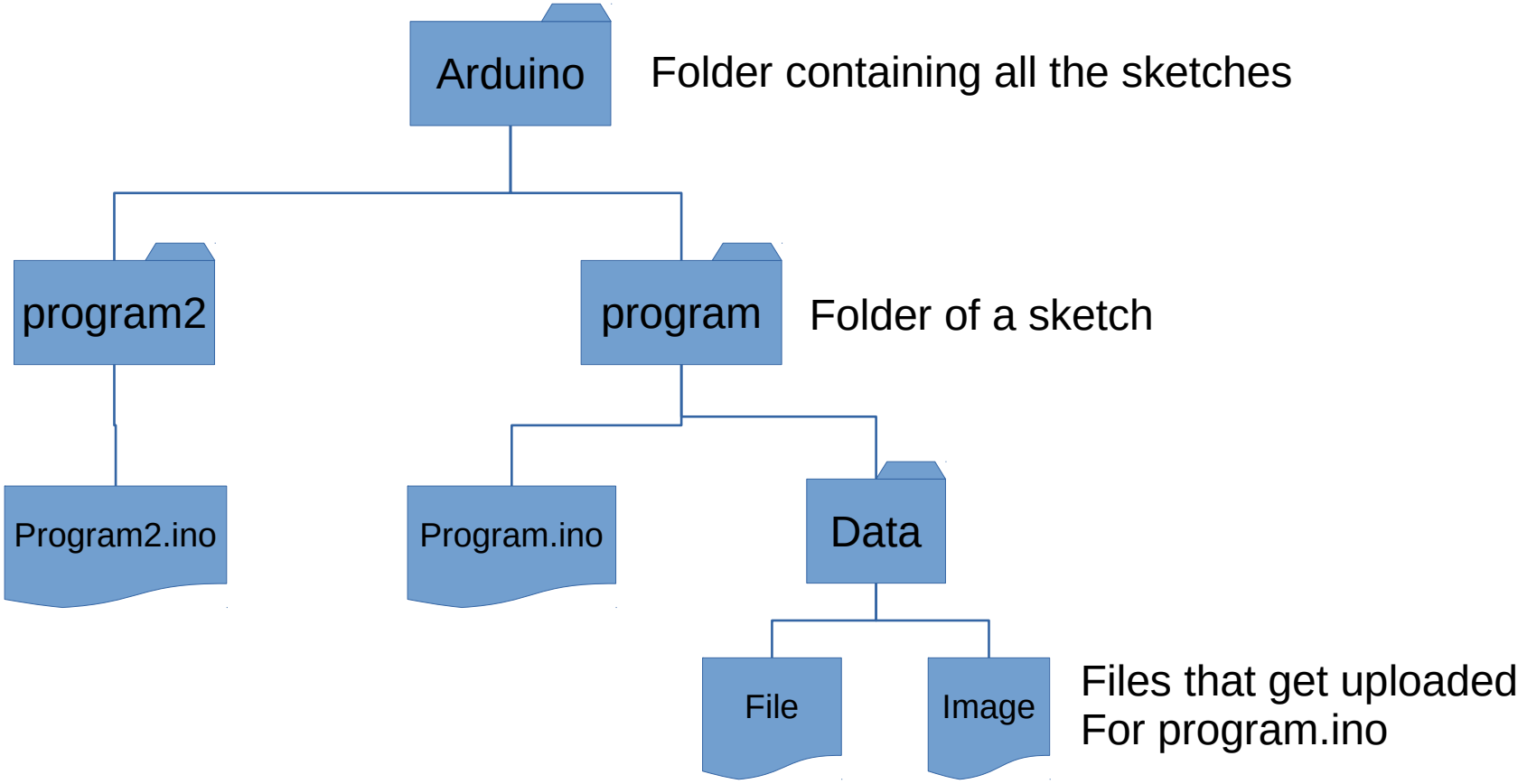
```
#include <FS.h>

void setup()
{
  Serial.begin(115200);
  SPIFFS.begin();
  File f = SPIFFS.open(“/file.txt”,”w”);
  f.println(“This is the first line”);
  f.println(“This is the second line”);
  f.close();
  delay(1000);
  f = SPIFFS.open(“/file.txt”,”r”);
  while(f.available()) {
    String line = f.readStringUntil('\n');
    Serial.println(line);
  }
  f.close();
}

void loop()
{
}
```

File System Upload

- To uploada file, you need to create a directory in the program sketch directory called **data**. Any files placed in this directory will be uploaded when you click on **ESP8266 Sketch Data Upload** menu under **Tools** menu.
- This is useful for uploading html files, images and javascript libraries.
- This will become useful in the Simple Web server lesson.



Other File System Functions

- SPIFFS.exists(“filename”);
  - This returns true if the file exists.
- SPIFFS.remove(“filename”);
  - Deletes the file.
- SPIFFS.rename(“oldfile”,”newfile”);
  - Changes the file name from oldfile to newfile.