Send SMS

Use the Serial Monitor to type in SMS messages to different phone numbers.

LAST REVISION: **02/04/2022, 09:50 AM**

This sketch send a SMS message from an Arduino board equipped with a GSM shield. Using the serial monitor of the Arduino Software (IDE), you'll enter the number to connect with, and the text message to send.

Hardware Required

- Arduino Board

- Arduino + Telefonica GSM/GPRS Shield
- SIM card

Circuit

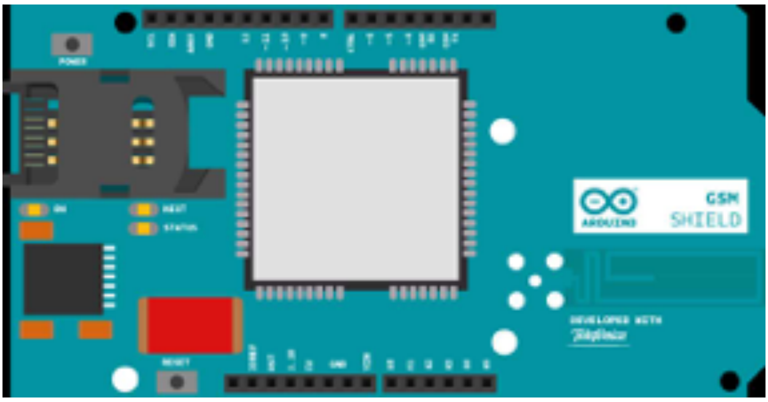

image of the Arduino GSM Shield on top of an Arduino board

Code

First, import the GSM library

```
#include <GSM.h>
```

SIM cards may have a PIN number that enables their functionality. Define the PIN for your SIM. If your SIM has no PIN, you can leave it blank :

```
#define PINNUMBER ""
```

Initialize instances of the classes you're going to use. You're going to need both the GSM and

GSM_SMS class.

COPY

```
GSM gsmAccess;

GSM_SMS sms;
```

In

`setup`, open a serial connection to the computer. After opening the connection, send a message indicating the sketch has started.

COPY

```
void setup(){

  Serial.begin(9600);

  Serial.println("SMS Messages Sen
```

Create a local variable to track the connection status. You'll use this to keep the sketch from starting until the SIM is connected to the network :

COPY

```
boolean notConnected = true;
```

Connect to the network by calling

`gsmAccess.begin()`. It takes the SIM card's PIN as an argument. By placing this inside a `while()` loop, you can continually check the status of the connection. When the modem does connect, `gsmAccess()` will return

`GSM_READY`. Use this as a flag to set the `notConnected` variable to `true` or `false`. Once connected, the remainder of `setup` will run.

COPY

```
while(notConnected)

  {

    if(gsmAccess.begin(PINNUMBER)

      notConnected = false;

    else

    {

      Serial.println("Not connect

      delay(1000);

    }

  }
```

Finish

`setup` with some information to the serial monitor.

COPY

```
Serial.println("GSM initialized.")
}
```

Create a function named

`readSerial` of type `int`. You'll use this to iterate through input from the serial monitor, storing the number you wish to send an SMS to, and the message you'll be sending. It should accept a `char` array as an argument.

COPY

```
int readSerial(char result[])
{
```

Create a variable to count through the items in the serial buffer, and start a

`while` loop that will continually execute.

COPY

```
int i = 0;

  while(1)

  {
```

As long as there is serial information available, read the data into a variable named

`inChar`.

COPY

```
while (Serial.available() > 0)

    {

      char inChar = Serial.read();
```

If the character being read is a newline, terminate the array, clear the serial buffer and exit the function.

COPY

```
if (inChar == '\n')

        {

          result[i] = '\0';

          Serial.flush();

          return 0;

        }
```

If the incoming character is an ASCII character other than a newline or carriage return, add it to the array and increment the index. Close up the

`while` loops and the function.

COPY

```
if(inChar!='\r')

        {

          result[i] = inChar;

          i++;

        }

      }

    }
}
```

In

`loop`, create a `char` array named `remoteNumber` to hold the number you wish to send an SMS to. Invoke the `readSerial` function you just created, and pass `remoteNumber` as the argument. When `readSerial` executes, it will populate `remoteNumber` with the number you wish to send the message to.

COPY

```
Serial.print("Enter a mobile numbe

  char remoteNumber[20];

  readSerial(remoteNumber);

  Serial.println(remoteNumber);
```

Create a new

`char` array named `txtMsg`. This will hold the content of your SMS. Pass `txtMsg` to `readSerial` to populate the array.

COPY

```
Serial.print("Now, enter SMS conte

  char txtMsg[200];

  readSerial(txtMsg);
```

Call

`sms.beginSMS()` and pass it `remoteNumber` to start sending the message, `sms.print()` to send the message, and `sms.endSMS()` to complete the process. Print out some diagnostic information and close the `loop`. Your message is on its way!

COPY

```
Serial.println("SENDING");

  Serial.println();

  Serial.println("Message:");

  Serial.println(txtMsg);

  sms.beginSMS(remoteNumber);

  sms.print(txtMsg);

  sms.endSMS();

  Serial.println("\nCOMPLETE!\n")
}
```

Once your code is uploaded, open the serial monitor. Make sure the serial monitor is set to only send a newline character on return. When prompted to enter the number you wish to call, enter the digits and press return. You'll then be asked to enter your message. Once you've typed that, press return again to send it.

Complete Sketch

The complete sketch is below.

Last revision 2018/08/23 by SM