GSM Test GPRS

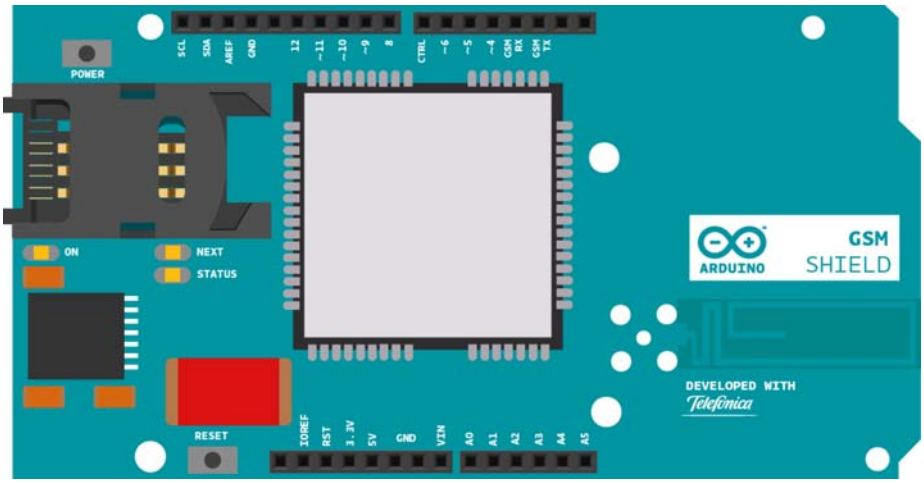
This sketch tests the GPRS data connection on the GSM shield. It tries to connect to arduino.cc.

To use a data connection with the GSM shield, you'll need your provider's Access Point Name (APN), login, and password. To obtain this information, contact the network provider for the most up to date information. This page (http://wiki.apnchanger.org/) has some information about various carrier settings, but it may not be current.

Hardware Required

- Arduino or Genuino Board
- Arduino + Telefonica GSM/GPRS Shield (//www.arduino.cc/en/Main/ArduinoGSMShield)
- SIM card enable for Data

Circuit



(//www.arduino.cc/en/uploads/Tutorial/GSMShield_ArduinoUno.jpg)
image of the Arduino GSM Shield on top of an Arduino or Genuino 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 the GSM, GPRS, and GSMClient classes.

```
GSMClient client;
GPRS gprsAccess;
GSM gsmAccess;
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=2)

Create some status messages to send to the serial monitor:

```
String oktext = "OK";
String errortext = "ERROR";
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=3)

Create some variables to hold the server, path, and proxy you wish to connect to.

```
char url[] = "arduino.cc";
char path[] = "/";
char urlproxy[] = "http://arduino.cc";
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=4)

Create a variable to hold a response from the server, and a flag to indicate f you wil be using a proxy or not.

```
String response = "";

boolean use_proxy = false;
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=5)

In `setup` , open a serial connection to the computer.

```
void setup(){
  Serial.begin(9600);
}
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=6)

You're going to create a custom function to handle serial input from the serial monitor. make a named function that accepts a `char` array as an argument.

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

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=7)

make a variable to act as a counter. While there is serial information available, read it into the `char` array. If a newline character is encountered, terminate the array and return to the main program.

```
int i = 0;
  while(1)
  {
    while (Serial.available() > 0)
    {
      char inChar = Serial.read();
      if (inChar == '\n')
      {
        result[i] = '\0';
        return 0;
      }
      if(inChar!='\r')
      {
        result[i] = inChar;
        i++;
      }
    }
  }
}
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=8)

In `loop()` , set the proxy flag to false

```
void loop()
{
  use_proxy = false;
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=9)

Start the connection to the GSM network by passing the PIN number (if applicable) to `gsmAccess.begin()` .

```
Serial.print("Connecting GSM network...");
  if(gsmAccess.begin(PINNUMBER)!=GSM_READY)
  {
    Serial.println(errortext);
    while(true);
  }
  Serial.println(oktext);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=10)

Create a `char` array to hold the APN. Use the `readSerial()` function you created to get the bytes from the serial monitor

```
char apn[50];
  Serial.print("Enter your APN: ");
  readSerial(apn);
  Serial.println(apn);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=11)

Create a `char` array to hold the APN login. Use the `readSerial()` function you created to get the bytes from the serial monitor

```
char login[50];
  Serial.print("Now, enter your login: ");
  readSerial(login);
  Serial.println(login);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=12)

Create a `char` array to hold the APN password. Use the `readSerial()` function you created to get the bytes from the serial monitor

```
char password[20];
  Serial.print("Finally, enter your password: ");
  readSerial(password);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=13)

Connect to the GPRS network using `gprs.attachGPRS()` . This requires the APN, login, and password you just entered.

When the modem has attached itself to the GPRS network, `gsmAccess()` will return `GSM_READY` .

```
Serial.println("Attaching to GPRS with your APN...");
  if(gprsAccess.attachGPRS(apn, login, password)!=GPRS_READY)
  {
    Serial.println(errortext);
  }
  else{

    Serial.println(oktext);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=14)

Create a `char` array to hold any proxy information you may need. Use the `readSerial()` function you created to get the bytes from the serial monitor.

```
char proxy[100];
    Serial.print("If your carrier uses a proxy, enter it, if not press enter: ");
    readSerial(proxy);
    Serial.println(proxy);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=15)

If a proxy was indicated, ask for the port number, and set the proxy flag to `true`

```
int pport;
    if(proxy[0] != '\0'){
      // read proxy port introduced by user
      char proxyport[10];
      Serial.print("Enter the proxy port: ");
      readSerial(proxyport);
      // cast proxy port introduced to integer
      pport = (int) proxyport;
      use_proxy = true;
      Serial.println(proxyport);
    }
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=16)

Create a variable to indicate if you are connected to the server or not. With `client.connect()`, make a connection to the server. How you make the connection will depend on if you are using a proxy or not.

```
Serial.print("Connecting and sending GET request to arduino.cc...");
    int res_connect;

    if(use_proxy)
      res_connect = client.connect(proxy, pport);
    else
      res_connect = client.connect(url, 80);
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=17)

if you have connected, make a HTTP GET request using `client.print()`.

```
if (res_connect)
    {
      client.print("GET ");

      if(use_proxy)
        client.print(urlproxy);
      else
        client.print(path);

      client.println(" HTTP/1.0");
      client.println();
      Serial.println(oktext);
    }
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=18)

If no connection was made, print out an error

```
else
    {
      // if you didn't get a connection to the server
      Serial.println(errortext);
    }
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=19)

Check to see if the server returned any bytes using `client.available()`. If there are, read them into the `response` String, then cast them into a char array. Check the array for the substring "200 OK", which indicates a valid response from arduino.cc.

```
Serial.print("Receiving response...");

    boolean test = true;
    while(test)
    {
      if (client.available())
      {
        char c = client.read();
        response += c;

        char responsechar[response.length()+1];
        response.toCharArray(responsechar, response.length()+1);

        if(strstr(responsechar, "200 OK") != NULL){
          Serial.println(oktext);
          Serial.println("TEST COMPLETE!");
```

```
          test = false;
        }
      }
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=20)

If the server has disconnected, stop the client and close `loop()`

```
if (!client.connected())
      {
        Serial.println();
        Serial.println("disconnecting.");
        client.stop();
        test = false;
      }
    }
  }
}
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=21)

Once your code is uploaded, open the serial monitor to see the status of the connection.

The complete sketch is below.

```
/*

 This sketch test the GSM shield's ability to connect to a
 GPERS network. It asks for APN information through the
 serial monitor and tries to connect to arduino.cc.

 Circuit:
 * GSM shield attached
 * SIM card with data plan

 Created 18 Jun 2012
 by David del Peral

 This example code is part of the public domain

 http://www.arduino.cc/en/Tutorial/GSMToolsTestGPRS

 */

// libraries
#include <GSM.h>

// PIN Number
#define PINNUMBER ""

// initialize the library instance
GSM gsmAccess;        // GSM access: include a 'true' parameter for debug enabled
GPRS gprsAccess;  // GPRS access
GSMClient client;  // Client service for TCP connection

// messages for serial monitor response
String oktext = "OK";
String errortext = "ERROR";

// URL and path (for example: arduino.cc)
char url[] = "arduino.cc";
char urlproxy[] = "http://www.arduino.cc";
char path[] = "/";

// variable for save response obtained
String response = "";

// use a proxy
boolean use_proxy = false;

void setup() {
  // initialize serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo only
```

```
  }
}

void loop() {
  use_proxy = false;

  // start GSM shield
  // if your SIM has PIN, pass it as a parameter of begin() in quotes
  Serial.print("Connecting GSM network...");
  if (gsmAccess.begin(PINNUMBER) != GSM_READY) {
    Serial.println(errortext);
    while (true);
  }
  Serial.println(oktext);

  // read APN introduced by user
  char apn[50];
  Serial.print("Enter your APN: ");
  readSerial(apn);
  Serial.println(apn);

  // Read APN login introduced by user
  char login[50];
  Serial.print("Now, enter your login: ");
  readSerial(login);
  Serial.println(login);

  // read APN password introduced by user
  char password[20];
  Serial.print("Finally, enter your password: ");
  readSerial(password);

  // attach GPRS
  Serial.println("Attaching to GPRS with your APN...");
  if (gprsAccess.attachGPRS(apn, login, password) != GPRS_READY) {
    Serial.println(errortext);
  } else {

    Serial.println(oktext);

    // read proxy introduced by user
    char proxy[100];
    Serial.print("If your carrier uses a proxy, enter it, if not press enter: ");
    readSerial(proxy);
    Serial.println(proxy);

    // if user introduced a proxy, asks him for proxy port
    int pport;
    if (proxy[0] != '\0') {
      // read proxy port introduced by user
      char proxyport[10];
      Serial.print("Enter the proxy port: ");
      readSerial(proxyport);
      // cast proxy port introduced to integer
      pport = (int) proxyport;
      use_proxy = true;
      Serial.println(proxyport);
    }

    // connection with arduino.cc and realize HTTP request
    Serial.print("Connecting and sending GET request to arduino.cc...");
    int res_connect;

    // if use a proxy, connect with it
    if (use_proxy) {
      res_connect = client.connect(proxy, pport);
    } else {
      res_connect = client.connect(url, 80);
    }

    if (res_connect) {
      // make a HTTP 1.0 GET request (client sends the request)
      client.print("GET ");

      // if use a proxy, the path is arduino.cc URL
      if (use_proxy) {
```

```
        client.print(urlproxy);
      } else {
        client.print(path);
      }

      client.println(" HTTP/1.0");
      client.println();
      Serial.println(oktext);
    } else {
      // if you didn't get a connection to the server
      Serial.println(errortext);
    }
    Serial.print("Receiving response...");

    boolean test = true;
    while (test) {
      // if there are incoming bytes available
      // from the server, read and check them
      if (client.available()) {
        char c = client.read();
        response += c;

        // cast response obtained from string to char array
        char responsechar[response.length() + 1];
        response.toCharArray(responsechar, response.length() + 1);

        // if response includes a "200 OK" substring
        if (strstr(responsechar, "200 OK") != NULL) {
          Serial.println(oktext);
          Serial.println("TEST COMPLETE!");
          test = false;
        }
      }

      // if the server's disconnected, stop the client:
      if (!client.connected()) {
        Serial.println();
        Serial.println("disconnecting.");
        client.stop();
        test = false;
      }
    }
  }
}

/*
  Read input serial
 */
int readSerial(char result[]) {
  int i = 0;
  while (1) {
    while (Serial.available() > 0) {
      char inChar = Serial.read();
      if (inChar == '\n') {
        result[i] = '\0';
        return 0;
      }
      if (inChar != '\r') {
        result[i] = inChar;
        i++;
      }
    }
  }
}
```

[Get Code] (//www.arduino.cc/en/Tutorial/GSMToolsTestGPRS?action=sourceblock&num=1)

See Also

- Arduino GSM Shield (//www.arduino.cc/en/Main/ArduinoGSMShield) – Complete product description.
- Getting started with the GSM Shield (//www.arduino.cc/en/Guide/ArduinoGSMShield) – Get everything set up in minutes.
- GSM library (//www.arduino.cc/en/Reference/GSM) – Your reference for the GSM Library.

- GPRS (//www.arduino.cc/en/Reference/GPRSConstructor) Constructor
- GSMClient (//www.arduino.cc/en/Reference/GSMClientConstructor) Constructor
- attachGPRS (//www.arduino.cc/en/Reference/AttachGPRS)()

- GSMToolsBandManagement (//www.arduino.cc/en/Tutorial/GSMToolsBandManagement) - Checks the band currently configured in the modem and allows you to change it.
- GSMToolsGsmScanNetworks (//www.arduino.cc/en/Tutorial/GSMToolsGsmScanNetworks) - Scans the available networks and prints informations about IMEI and number of the SIM card.
- GSMToolsPinManagement (//www.arduino.cc/en/Tutorial/GSMToolsPinManagement) - How to to change or remove the PIN number.
- GSMToolsTestModem (//www.arduino.cc/en/Tutorial/GSMToolsTestModem) - Tests to see if the modem of the GSM shield is working correctly.
- GSMToolsTestWebServer (//www.arduino.cc/en/Tutorial/GSMToolsTestWebServer) - A simple web server that replies with nothing, but prints the client's request and the server IP address.
- GSMExamplesMakeVoiceCall (//www.arduino.cc/en/Tutorial/GSMExamplesMakeVoiceCall) - How to make a voice call with mic and speaker.

Last revision 2015/08/17 by SM

Share

NEWSLETTER

Enter your email to sign up →

©2017 Arduino Copyright Notice (//www.arduino.cc/en/Main/CopyrightNotice) Contact us (//www.arduino.cc/en/Main/ContactUs)
About us (//www.arduino.cc/en/Main/AboutUs) Careers (//www.arduino.cc/Careers)

(https://twitter.com/arduino) (https://www.facebook.com/official.arduino) (https://plus.google.com/+Arduino)
(https://www.flickr.com/photos/arduino_cc) (https://youtube.com/arduinoteam)