**GSM INTERFACE PROGRAMS**

1. **WEB CLIENT**

This sketch connects to a website through a GSM shield. Specifically, this example downloads the URL "http://www.arduino.cc/asciilogo.txt" and prints it to the Serial monitor.

[Getting Started with the Arduino GSM Shield | Arduino Documentation](https://docs.arduino.cc/retired/getting-started-guides/ArduinoGSMShield)

[SoftwareSerial Library | Arduino Documentation](https://docs.arduino.cc/learn/built-in-libraries/software-serial)

Circuit:

\* GSM shield attached to an Arduino

\* SIM card with a data plan

**http://www.arduino.cc/en/Tutorial/GSMExamplesWebClient**

#include <GSM.h>

#define PINNUMBER ""

// APN data

#define GPRS\_APN "GPRS\_APN" // replace your GPRS APN

#define GPRS\_LOGIN "login" // replace with your GPRS login

#define GPRS\_PASSWORD "password" // replace with your GPRS password

// initialize the library instance

GSMClient client;

GPRS gprs;

GSM gsmAccess;

// URL, path & port (for example: arduino.cc)

char server[] = "arduino.cc";

char path[] = "/asciilogo.txt";

int port = 80; // port 80 is the default for HTTP

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial) {

;

// wait for serial port to connect. Needed for native USB port only

}

Serial.println("Starting Arduino web client.");

// connection state

bool notConnected = true;

// After starting the modem with GSM.begin()

// attach the shield to the GPRS network with the APN, login and password

while (notConnected) {

if ((gsmAccess.begin(PINNUMBER) == GSM\_READY) &(gprs.attachGPRS(GPRS\_APN, GPRS\_LOGIN, GPRS\_PASSWORD) == GPRS\_READY)) {

notConnected = false;

}

Else

{

Serial.println("Not connected");

delay(1000);

}

}

Serial.println("connecting...");

// if you get a connection, report back via serial:

if (client.connect(server, port)) {

Serial.println("connected");

// Make a HTTP request:

client.print("GET ");

client.print(path);

client.println(" HTTP/1.1");

client.print("Host: ");

client.println(server);

client.println("Connection: close");

client.println();

}

else

{

// if you didn't get a connection to the server:

Serial.println("connection failed");

}

}

void loop() {

// if there are incoming bytes available

// from the server, read them and print them:

if (client.available()) {

char c = client.read();

Serial.print(c);

}

// if the server's disconnected, stop the client:

if (!client.available() && !client.connected()) {

Serial.println();

Serial.println("disconnecting.");

client.stop();

// do nothing forevermore:

for (;;)

;

}

}

**2). WEB SERVER**

A simple web server that shows the value of the analog input pins.

using a GSM shield.

Circuit:

\* GSM shield attached

\* Analog inputs attached to pins A0 through A5 (optional)

#include <GSM.h>

#define PINNUMBER ""

// APN data

#define GPRS\_APN "GPRS\_APN" // replace your GPRS APN

#define GPRS\_LOGIN "login" // replace with your GPRS login

#define GPRS\_PASSWORD "password" // replace with your GPRS password

// initialize the library instance

GPRS gprs;

GSM gsmAccess; // include a 'true' parameter for debug enabled

GSMServer server(80); // port 80 (http default)

// timeout

const unsigned long \_\_TIMEOUT\_\_ = 10 \* 1000;

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial) {

;

// wait for serial port to connect. Needed for native USB port only

}

// connection state

bool notConnected = true;

// Start GSM shield

// If your SIM has PIN, pass it as a parameter of begin() in quotes

while (notConnected)

{

if ((gsmAccess.begin(PINNUMBER) == GSM\_READY) &

(gprs.attachGPRS(GPRS\_APN, GPRS\_LOGIN, GPRS\_PASSWORD) == GPRS\_READY))

{

notConnected = false;

}

else

{

Serial.println("Not connected");

delay(1000);

}

}

Serial.println("Connected to GPRS network");

// start server

server.begin();

//Get IP.

IPAddress LocalIP = gprs.getIPAddress();

Serial.println("Server IP address=");

Serial.println(LocalIP);

}

void loop() {

// listen for incoming clients

GSMClient client = server.available();

if (client) {

while (client.connected()) {

if (client.available())

{

Serial.println("Receiving request!");

bool sendResponse = false;

while (char c = client.read()) {

if (c == '\n') {

sendResponse = true;

}

}

// if you've gotten to the end of the line (received a newline

// character)

if (sendResponse) {

// send a standard http response header

client.println("HTTP/1.1 200 OK");

client.println("Content-Type: text/html");

client.println();

client.println("<html>")

// output the value of each analog input pin

for (int analogChannel = 0; analogChannel < 6; analogChannel++) {

client.print("analog input ");

client.print(analogChannel);

client.print(" is ");

client.print(analogRead(analogChannel));

client.println("<br />");

}

client.println("</html>");

//necessary delay

delay(1000);

client.stop();

}

}

}

}

}

**3). Make Voice Calls**

**Make Voice Call**

This sketch, for the Arduino GSM shield, puts a voice call tom a remote phone number that you enter through the serial monitor.To make it work, open the serial monitor, and when you see the READY message, type a phone number. Make sure the serial monitor is set to send a just newline when you press return.

Circuit:

\* GSM shield

\* Voice circuit.

With no voice circuit the call will send nor receive any sound

#include <GSM.h>

#define PINNUMBER ""

// initialize the library instance

GSM gsmAccess;

// include a 'true' parameter for debug enabled

GSMVoiceCall vcs;

String remoteNumber = ""; // the number you will call

char charbuffer[20];

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial)

{

;

// wait for serial port to connect. Needed for native USB port only

}

Serial.println("Make Voice Call");

// connection state

bool notConnected = true;

// Start GSM shield

// If your SIM has PIN, pass it as a parameter of begin() in quotes

while (notConnected) {

if (gsmAccess.begin(PINNUMBER) == GSM\_READY)

{

notConnected = false;

}

else

{

Serial.println("Not connected");

delay(1000);

}

}

Serial.println("GSM initialized.");

Serial.println("Enter phone number to call.");

}

void loop() {

// add any incoming characters to the String:

while (Serial.available() > 0)

{

char inChar = Serial.read();

// if it's a newline, that means you should make the call:

if (inChar == '\n')

{

// make sure the phone number is not too long:

if (remoteNumber.length() < 20) {

// let the user know you're calling:

Serial.print("Calling to : ");

Serial.println(remoteNumber);

Serial.println();

// Call the remote number

remoteNumber.toCharArray(charbuffer, 20);

// Check if the receiving end has picked up the call

if (vcs.voiceCall(charbuffer))

{

Serial.println("Call Established. Enter line to end");

// Wait for some input from the line

while (Serial.read() != '\n' && (vcs.getvoiceCallStatus() == TALKING));

// And hang up

vcs.hangCall();

}

Serial.println("Call Finished");

remoteNumber = "";

Serial.println("Enter phone number to call.");

}

Else

{

Serial.println("That's too long for a phone number. I'm forgetting it");

remoteNumber = "";

}

}

else

{

// add the latest character to the message to send:

if (inChar != '\r') {

remoteNumber += inChar;

}

}

}

}

**4). RECEIVE VOICE CALL**

This sketch, for the Arduino GSM shield, receives voice calls, displays the calling number, waits a few seconds then hangs up.

Circuit:

\* GSM shield

\* Voice circuit. Refer to to the GSM shield getting started guide

**at http://www.arduino.cc/en/Guide/ArduinoGSMShield**

\* SIM card that can accept voice calls

With no voice circuit the call will connect, but will not send or receive sound

#include <GSM.h>

#define PINNUMBER ""

// initialize the library instance

GSM gsmAccess;

GSMVoiceCall vcs;

// Array to hold the number for the incoming call

char numtel[20];

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial)

{

;

// wait for serial port to connect. Needed for native USB port only

}

Serial.println("Receive Voice Call");

// connection state

bool notConnected = true;

// Start GSM shield

// If your SIM has PIN, pass it as a parameter of begin() in quotes

while (notConnected) {

if (gsmAccess.begin(PINNUMBER) == GSM\_READY)

{

notConnected = false;

}

else

{

Serial.println("Not connected");

delay(1000);

}

}

// This makes sure the modem correctly reports incoming events

vcs.hangCall();

Serial.println("Waiting for a call");

}

void loop() {

// Check the status of the voice call

switch (vcs.getvoiceCallStatus()) {

case IDLE\_CALL: // Nothing is happening

break;

case RECEIVINGCALL: // Yes! Someone is calling us

Serial.println("RECEIVING CALL");// Retrieve the calling number

vcs.retrieveCallingNumber(numtel, 20);// Print the calling number

Serial.print("Number:");

Serial.println(numtel);

// Answer the call, establish the call

vcs.answerCall();

break;

case TALKING: // In this case the call would be established

Serial.println("TALKING. Press enter to hang up.");

while (Serial.read() != '\n') {

delay(100);

}

vcs.hangCall();

Serial.println("Hanging up and waiting for the next call.");

break;

}

delay(1000);

}

**5). SEND SMS**

This sketch, for the Arduino GSM shield,sends an SMS message you enter in the serial monitor. Connect your Arduino with the GSM shield and SIM card, open the serial monitor, and wait for the "READY" message to appear in the monitor. Next, type a message to send and press "return". Make sure the serial

monitor is set to send a newline when you press return.

Circuit:

\* GSM shield

\* SIM card that can send SMS

#include <GSM.h>

#define PINNUMBER ""

// initialize the library instance

GSM gsmAccess;

GSM\_SMS sms;

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial) {

;

// wait for serial port to connect. Needed for native USB port only

}

Serial.println("SMS Messages Sender");

// connection state

bool notConnected = true;

// Start GSM shield

// If your SIM has PIN, pass it as a parameter of begin() in quotes

while (notConnected) {

if (gsmAccess.begin(PINNUMBER) == GSM\_READY)

{

notConnected = false;

}

else

{

Serial.println("Not connected");

delay(1000);

}

}

Serial.println("GSM initialized");

}

void loop() {

Serial.print("Enter a mobile number: ");

char remoteNum[20]; // telephone number to send sms

readSerial(remoteNum);

Serial.println(remoteNum);

// sms text

Serial.print("Now, enter SMS content: ");

char txtMsg[200];

readSerial(txtMsg);

Serial.println("SENDING");

Serial.println();

Serial.println("Message:");

Serial.println(txtMsg);

// send the message

sms.beginSMS(remoteNum);

sms.print(txtMsg);

sms.endSMS();

Serial.println("\nCOMPLETE!\n");

}

/\*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';

Serial.flush();

return 0;

}

if (inChar != '\r') {

result[i] = inChar;

i++;

}

}

}

}

**6). RECEIVING THE SMS**

This sketch, for the Arduino GSM shield, waits for a SMS message and displays it through the Serial port.

Circuit:

\* GSM shield attached to and Arduino

\* SIM card that can receive SMS messages

#include <GSM.h>

#define PINNUMBER ""

// initialize the library instances

GSM gsmAccess;

GSM\_SMS sms;

// Array to hold the number a SMS is retreived from

char senderNumber[20];

void setup() {

// initialize serial communications and wait for port to open:

Serial.begin(9600);

while (!Serial) {

;

// wait for serial port to connect. Needed for native USB port only

}

Serial.println("SMS Messages Receiver");

// connection state

bool notConnected = true;

// Start GSM connection

while (notConnected) {

if (gsmAccess.begin(PINNUMBER) == GSM\_READY)

{

notConnected = false;

}

else

{

Serial.println("Not connected");

delay(1000);

}

}

Serial.println("GSM initialized");

Serial.println("Waiting for messages");

}

void loop() {

char c;

// If there are any SMSs available()

if (sms.available()) {

Serial.println("Message received from:");

// Get remote number

sms.remoteNumber(senderNumber, 20);

Serial.println(senderNumber);

// An example of message disposal

// Any messages starting with # should be discarded

if (sms.peek() == '#') {

Serial.println("Discarded SMS");

sms.flush();

}

// Read message bytes and print them

while (c = sms.read()) {

Serial.print(c);

}

Serial.println("\nEND OF MESSAGE");

// Delete message from modem memory

sms.flush();

Serial.println("MESSAGE DELETED");

}

delay(1000);

}