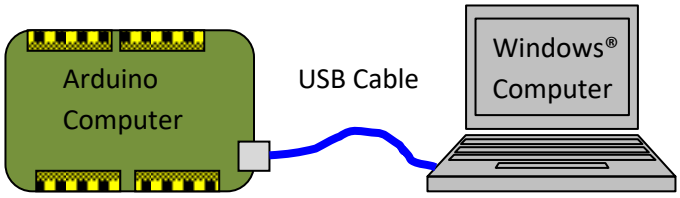
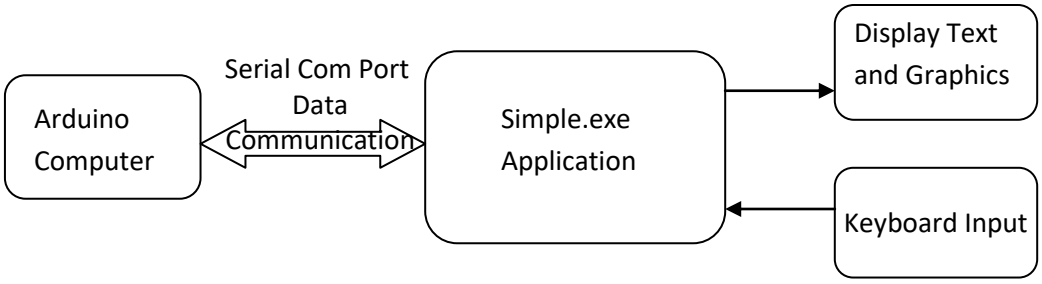
# How to Use the Simple Terminal Application

A terminal consists of a text display and a keyboard. It is used to send data between a computer and a user. Data was sent through a serial link know known as a COM port. Modern computers do not have COM ports. Instead they use a USB (Universal Serial Bus) connection.

The USB ports on your PC can emulate the old COM serial ports. The Arduino computer has a chip that converts the USB data into the old serial format. Digital I/O pins 0 and 1 connect to a serial interface in your Arduino processor chip. These pins connect to the converter chip.

Your PC requires an application to function as a terminal. It turns your computer screen and keyboard into a terminal. It also handles the serial data to and from the USB port.

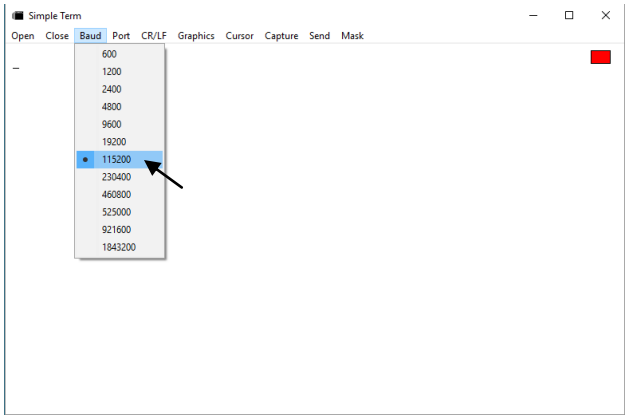


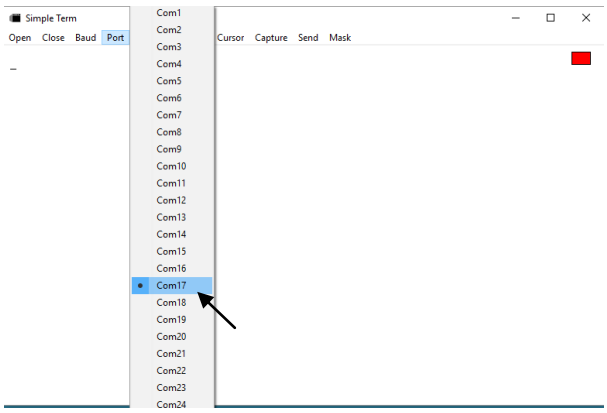

Once the application starts, it needs two parameters before it can connect to your Arduino board. The first is the Baud rate which is in bits per second. Most of our programs will use 115200 Baud. The Baud rate for the terminal must match the rate set on the Arduino. Otherwise, the data gets garbled.

For every byte sent serially, it takes 10-bits. There is a bit to indicate data is starting and a bit to indicate data is stopping. At 115200 Baud, you can send 11520 characters each second.

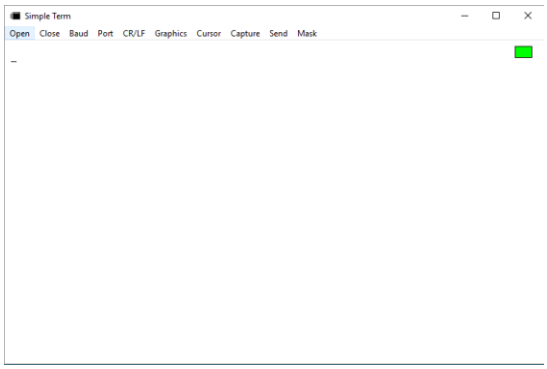
When Simple.exe starts, click on the Baud menu to select your Baud rate.



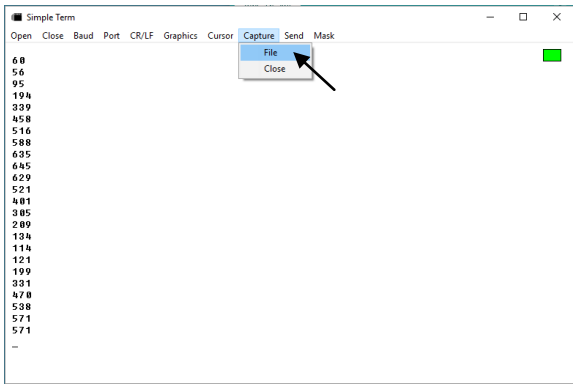

Your computer will assign a unique COM port number to your Arduino. The Arduino IDE shows which port is assigned for programming. Set the same port number in the application. Use the Port menu.



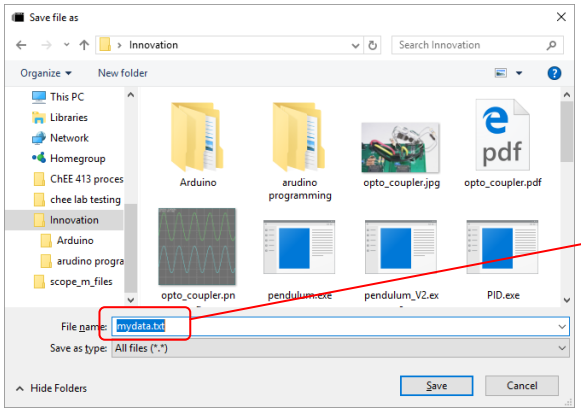

Once Baud and Port are set, click open and the red box will turn to green if successful. If you have the wrong port number or your Arduino is not plugged in, it will remain red.



The application defaults to a text mode terminal. Data sent from your Arduino will display on a screen of 80 characters wide by 25 lines. If you want a record of data sent from the Arduino, use the menu Capture and click on File.

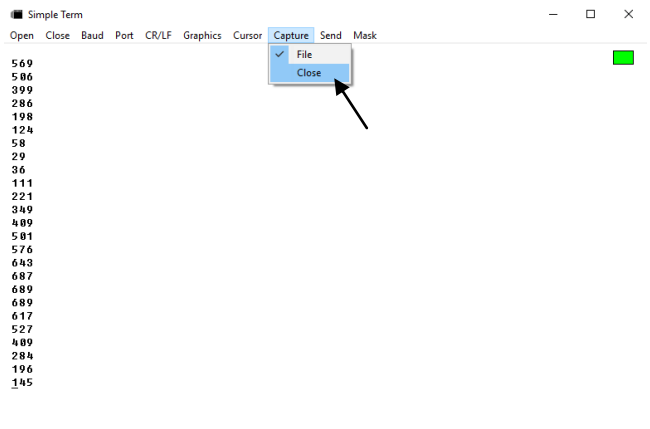



You will be prompted to enter a file name to save the data in. The file will have a .txt extension which identifies it as a text file.




your_file_name.txt

When you are finished collecting text data, use the Capture menu and click Close.




The saved text file can be imported into a spreadsheet and graphed.

The terminal application is not limited to only displaying text characters. There are additional characters for moving the cursor. The carriage return moves it to the beginning of a line. These characters are called control functions. On the keyboard, the Ctrl Key is pressed and held. Then the letter is pressed. The Arduino only has to send the decimal value.

Control Characters:

| Keyboard Character | Decimal Value | Function |
|---|---|---|
| Ctrl-H | 8 | Back Space |
| Ctrl-J | 10 | Line Feed |
| Ctrl-L | 12 | Clear Screen |
| Ctrl-M | 13 | Carriage Return |
| Ctrl-Q | 17 | Cursor Off |
| Ctrl-R | 18 | Cursor On |
| Ctrl-S | 19 | Hide Graphics |
| Ctrl-T | 20 | Show Graphics |
| Esc | 27 | Start Graphic Function |

Graphing uses an escape <Esc> code to initiate a command. Escape is followed by a command character. Next three numbers are sent for the x-position and finally, three more numbers are sent for the y-position.

Basic Form: <Esc>cXXXYYY

Where <Esc> = 27
c = command character (r,g,b,l,C,R,G,B,L and P)
XXX = decimal value
For graphics it is 000 to 719 the x-position left to right
For characters it is 000 to 079 the column position left to right
YYY = decimal value
For graphics it is 000 to 399 the y-position top to bottom
For characters it is 000 to 024 the line position top to bottom

| Command Character | Function |
|---|---|
| C | Move cursor to XXX YYY |
| R | Move red graphic to XXX YYY |
| G | Move green graphic to XXX YYY |
| B | Move blue graphic to XXX YYY |
| L | Move black graphic to XXX YYY |
| r | Red graphic line to XXX YYY |
| g | Green graphic line to XXX YYY |
| b | Blue graphic line to XXX YYY |
| l | Black graphic line to XXX YYY |
| P | Draw point at XXX YYY using last color |
| Q | Draw point at XXX YYY in Red |
| R | Draw point at XXX YYY in Green |
| S | Draw point at XXX YYY in Blue |
| T | Draw point at XXX YYY in Black |

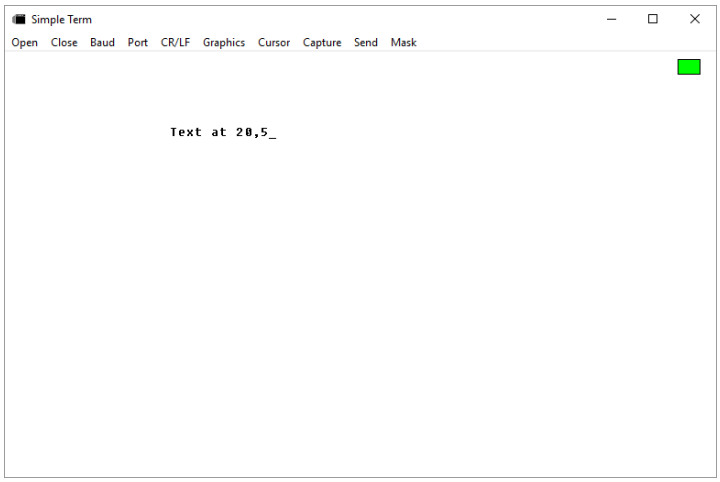
To use graphics with your Arduino Sketch, copy the Crt folder into the libraries folder which is found in the Arduino program folder. Next, include the Crt.h file in your Sketch by placing the following command at the beginning.

```
#include<Crt.h>
```

The Crt.h header file contains functions that simplify drawing graphics on the terminal screen.

Move the cursor to position x, y:

```
crtCursor(x, y);
```



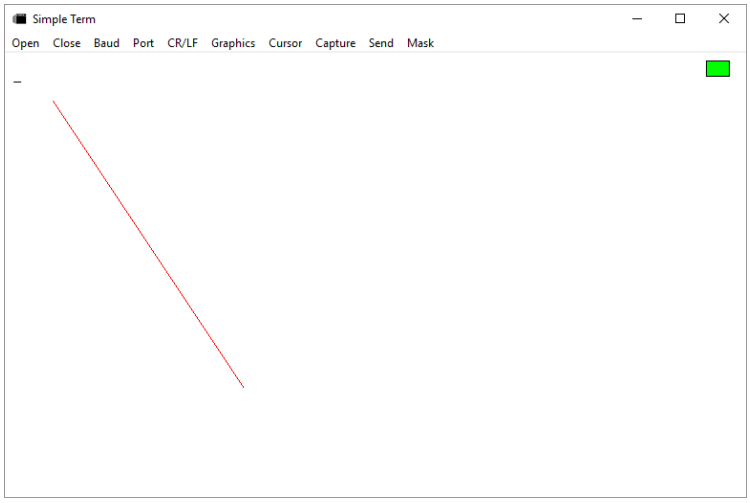

```
#include<Crt.h>

void setup() {
  Serial.begin(115200);
}

void loop(){
  delay(1000);
  Serial.write(12); //clear the screen
  crtCursor(20,5);  //move the cursor
  Serial.print("Text at 20,5"); //print text
}
```

Move the graphics color ‘c’ to position x, y:

crtMoveto( x, y, c); //c is ‘R’,’G’,’B’ or ‘L’

Draw a line in the graphics color ‘c’ from current position to position x and y:

crtLineto( x, y, c); //c is ‘R’,’G’,’B’ or ‘L’



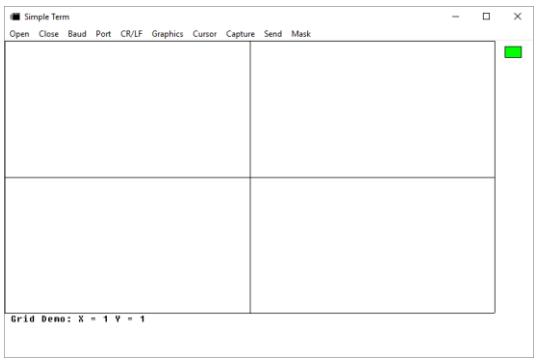

```
#include<Crt.h>

void setup() {
 Serial.begin(115200);
}

void loop(){
 delay(1000);
 Serial.write(20);     //enable graphics
 Serial.write(12);     //clear the screen
 crtMoveto(50,50,'R');  //move the red graphic
 crtLineto(250,350,'R'); //draw a red line
}
```

Draw a square dot of graphics color ‘c’ at position x,y:

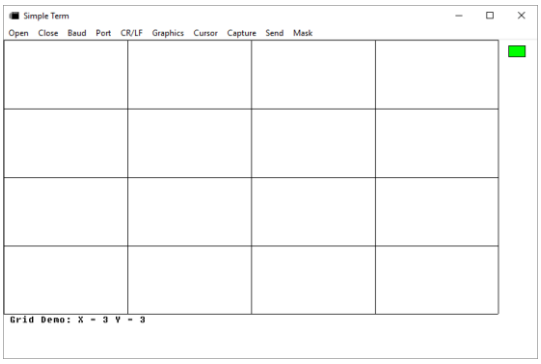
crtPoint( x, y, c); //c is ‘R’,’G’,’B’ or ‘L’

Draw a grid of nx vertical lines and ny horizontal lines

crtGrid(nx, ny);




nx = 1  ny = 1




nx = 3  ny = 3

Place a square dot of color ‘c’ with length mag and angle theta. The origin is at the bottom center of the plot area. Angle 0 is to the right of the origin. Angle 90 is up from the origin and Angle 180 is left of the origin.

```
crtRadar(mag, theta, c);
//mag: 0 to 400
//theta: 0 to 180
//c: ‘R’,’G’,’B’,’L’
```