Digital Output Functions

Digital output is accomplished in two steps. You must configure the digital pin as an output, and you must set the state of the digital pin.

`pinMode()`

In order to use `digitalWrite()`, you must configure the digital pin as an output. To do this, call the `pinMode()` function, which also takes two parameters:

1. Pin Number
2. Pin Direction

Like `digitalWrite()`, the pin number is the ID of the pin on the Arduino. The pin direction is a special value that is defined in the Arduino code, and is either INPUT or OUTPUT (there is actually a 3rd mode, but we will discuss that later).

Example code:

```cpp
pinMode(13, OUTPUT);
```

`digitalWrite()`

Digital output on an Arduino is accomplished using the `digitalWrite()` function. `digitalWrite()` takes two parameters:

1. Pin Number
2. Output State

The pin number is just what it sounds like; it is the ID of the pin on the Arduino, e.g. 13. The output state is a special value that is defined in the Arduino core and is either HIGH or LOW. Setting the value LOW will set the pin to 0V, while setting the value to HIGH will set the pin to Vcc, which is either 5V or 3.3V, depending on the specific Arduino hardware being used (Arduino Unos and Megas are both 5V).

Example code:

```cpp
digitalWrite(13, HIGH);
```

`digitalWrite()` Example

The simplest example for `digitalWrite()` is to blink a small LED. Any hobby LED can be used for this example. Alternatively, don't use any external components! The Arduino has an LED already attached to pin 13.

If you wish to use an external LED, simply connect the positive terminal of the LED to the digital pin being used (13 in the example below), and connect a 1kΩ resistor between the negative terminal of the LED and GND. 1kΩ is a conservative value for the resistor. If you want more information on selecting the proper resistor value, or are driving an array of LEDs, please see the [LED (Light Emitting Diode)](#) page.

**NOTE:** Do not try to connect high-power devices (e.g. motors, solenoids, relays, or high-power LEDs) this way. Arduino pins are only capable of handling up to 40mA of current. Trying to pass more current than this will damage your Arduino. If you need digital output with higher current, consider using a [Transistor](#) as a switch.
These schematics are attached and can be viewed with Fritzing.

**Blinking Light Arduino Sketch**

```cpp
#define LED_PIN 13

// Global Variables
bool g_fLedOn = true;

void setup()
{
    // Initialize LED pin as an output
    pinMode(LED_PIN, OUTPUT);
}

void loop()
{
    if (g_fLedOn)
    {
        // Turn LED on
        digitalWrite(LED_PIN, HIGH);
    }
}
```

digitalWrite() Example
else
{
    // Turn LED off
    digitalWrite(LED_PIN, LOW);
}

// Flip LED state
g_fLedOn = !g_fLedOn;

// Wait 1 second
delay( 1000 );

This code can be downloaded [here](#)