Arduino Programming Part 4: Flow Control

EAS 199B, Winter 2010

Gerald Recktenwald
Portland State University
gerry@me.pdx.edu

Goal

- Make choices based on conditions in the environment
- Logical expressions: Formulas that are T or F
- Different kinds of choices
  - ❖ Act on a single condition
  - ❖ Choose one course of action from several

Conventional on/off switch

Basic light switch or rocker switch

- Makes or breaks connection to power
- Switch stays in position: On or Off
- Toggle position indicates the state
- NOT in the Arduino Inventors Kit

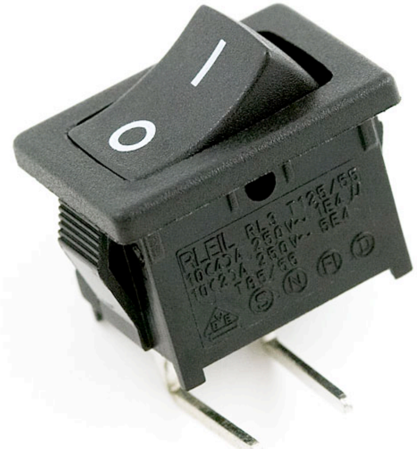
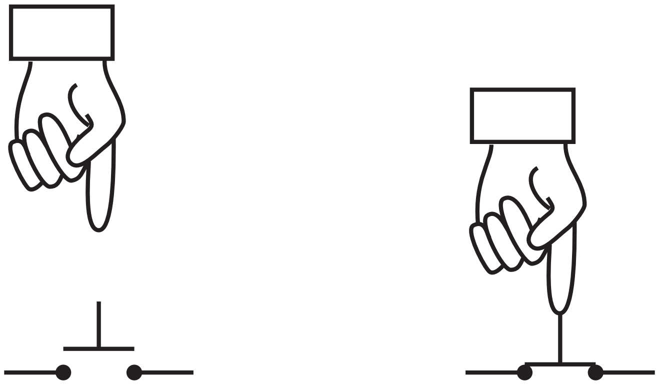


Image from sparkfun.com

Image from lowes.com

Momentary or push-button switches

- Temporary "click" input
- Normally open
  - electrical contact is made when button is pressed
- Normally closed
  - electrical contact is broken when button is pressed
- Internal spring returns button to its un-pressed state



Open

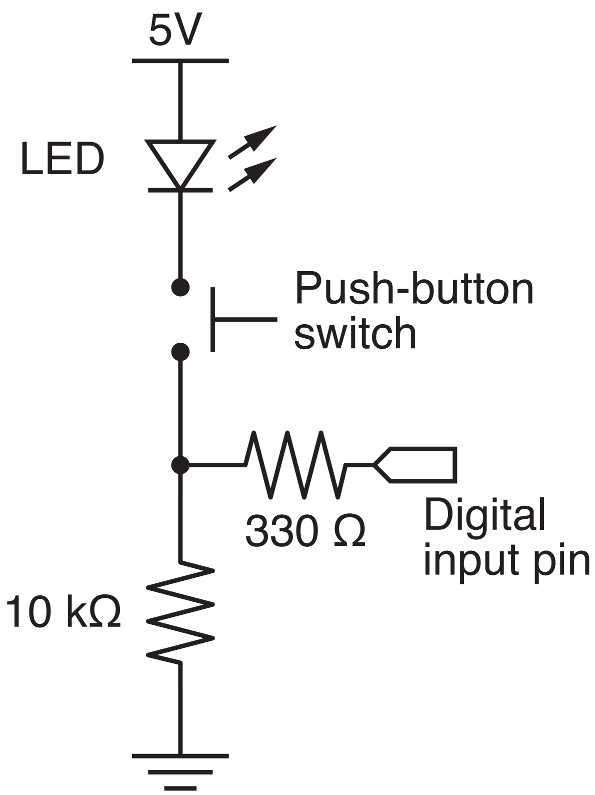
Closed

Image from sparkfun.com

Momentary Button and LED Circuit

Digital input with a *pull-down resistor*

- When the switch is open:
  - Digital input pin is tied to ground
  - No current flows, so there is no voltage difference from input pin to ground
  - Reading on digital input is LOW
- When the switch is closed:
  - Current flows from 5V to ground, causing LED to light up.
  - The 330Ω resistor limits the current draw by the input pin.
  - The 10k resistor causes a large voltage drop between 5V and ground, which causes the digital input pin to be closer to 5V.
  - Reading on digital input is HIGH



Programs for the LED/Button Circuit

1. Continuous monitor of button state
   - Program is completely occupied by monitoring the button
   - Used as a demonstration — not practically useful
2. Wait for button input
   - Blocks execution while waiting
   - May be useful as a start button
3. Interrupt Handler
   - Most versatile
   - Does not block execution
   - Interrupt is used to change a flag that indicates state

All three programs use the same circuit

Continuous monitor of button state

This program does not control the LED

```
int  button_pin = 4;           // pin used to read the button

void setup() {
  pinMode( button_pin, INPUT);
  Serial.begin(9600);          // Button state is sent to host
}

void loop() {
  int button;

  button = digitalRead( button_pin );
  if ( button == HIGH ) {
    Serial.println("on");
  } else {
    Serial.println("off");
  }
}
```

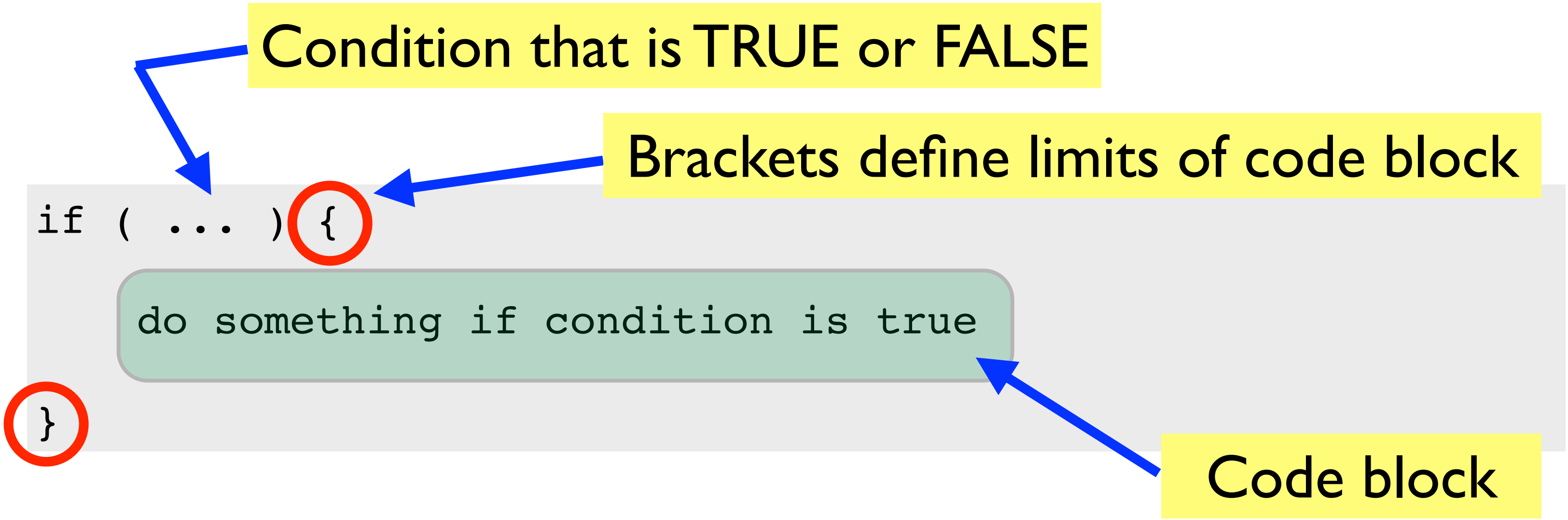
Continuous monitor of button state

This program does not control the LED

```
int  button_pin = 4;              // pin used to read the button

void setup() {
  pinMode( button_pin, INPUT);
  Serial.begin(9600);             // Button state is sent to host
}

void loop() {
  int button;

  button = digitalRead( button_pin );
  if ( button == HIGH ) {
    Serial.println("on");
  } else {
    Serial.println("off");
  }
}
```

Serial monitor shows a continuous stream of "on" or "off"

Basic "if" construct



Condition that is TRUE or FALSE

Brackets define limits of code block

```
if ( ... ) {
    do something if condition is true
}
```

Code block

Example:

```
button_value = digitalRead( button_pin );
if ( button_value == HIGH ) {
  digitalWrite( LED_pin, HIGH);
}
```

Comparison operators

Symbol	Meaning
<	Is less than
>	Is greater than than
==	Is equal to
>=	Is greater than or equal to
<=	Is less than or equal to
!=	Is not equal to

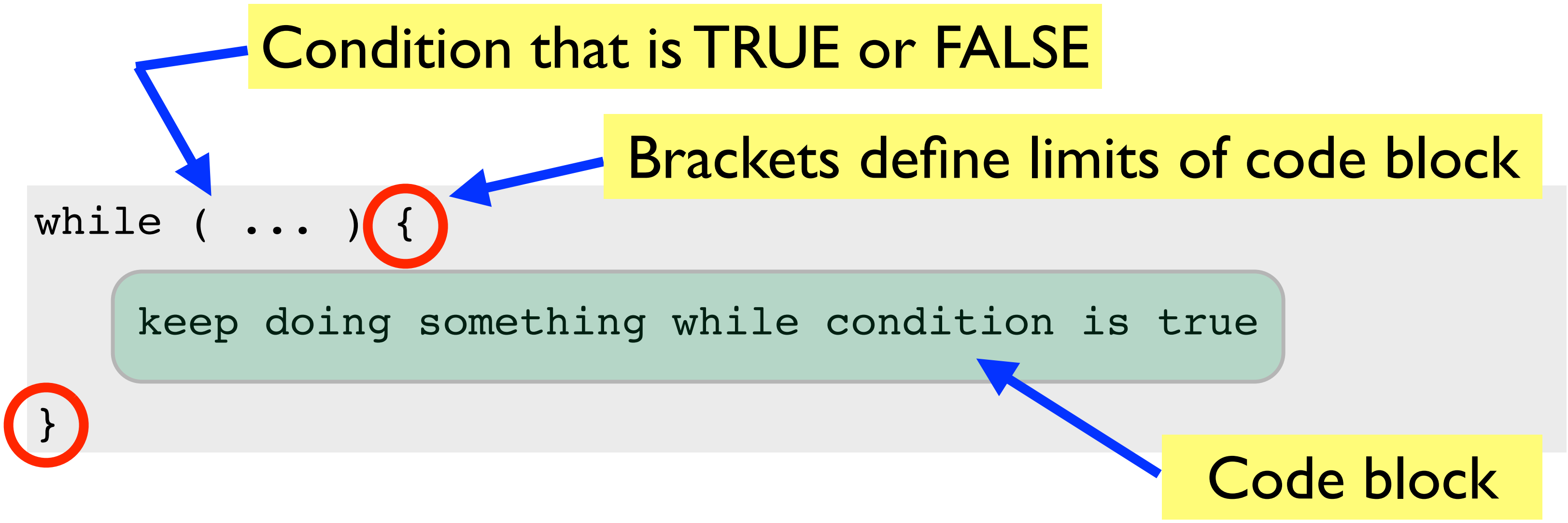
Practice

What is the value z?

```
x = 2;
y = 5;
if ( x < y ) {
  z = y - x;
}
```

```
x = 2;
y = 5;
if ( x > y ) {
  z = y - x;
}
```

Basic “while” construct



Condition that is TRUE or FALSE

Brackets define limits of code block

```
while ( ... ) {
    keep doing something while condition is true
}
```

Code block

Example:

```
int x;
x = analogRead(2);

while ( x > 355 ) {
  digitalWrite( LED_pin, OFF);
  x = analogRead(2);
}
```

Wait for button input

```
int  button_pin = 4;           // pin used to read the button

void setup() {
  int start_click = LOW;        //  Initial state: no click yet

  pinMode( button_pin, INPUT);
  Serial.begin(9600);
  while ( !start_click ) {
    start_click = digitalRead( button_pin );
    Serial.println("Waiting for button press");
  }
}

void loop() {
  int button;

  button = digitalRead( button_pin );
  if ( button == HIGH ) {
    Serial.println("on");
  } else {
    Serial.println("off");
  }
}
```

while loop continues as long as start_click is FALSE

Same loop() function as before

Other references

Ladyada tutorial

- Excellent and detailed
- http://www.ladyada.net/learn/arduino/lesson5.html

Arduino reference

- Minimal explanation
  - http://www.arduino.cc/en/Tutorial/Button
- Using interrupts
  - http://www.uchobby.com/index.php/2007/11/24/arduino-interrupts/
  - http://www.arduino.cc/en/Reference/AttachInterrupt