Arduino Platform

CSE 132

Arduino Programs

- Community calls them "sketches"
- Composed of the basic structure below

```
void setup() {
    // insert startup code here, will execute once
}

void loop() {
    // insert main code here, will execute over and over
}
```

Hello World

- First complete Arduino program

```
void setup() {
    Serial.begin(9600); //startup comm. link to PC
    Serial.println("Hello world!");
}
void loop() {
}
```

Arduino Timing

- Use delay() library routine
  - Argument is integer number of milliseconds
- Use millis() library routine
  - Returns the number of milliseconds since last reset of Arduino
  - Return type is `long int`, which is 32 bits or 4 bytes
- Later in semester we will use micros()
  - Returns number of microseconds since last reset

Arduino Printing

- Printing goes to Serial Monitor in Arduino IDE
  - Serial.begin(9600) in setup() initializes port and sets baud rate (communication speed)
- How do we print?
  - Use Serial.print() and Serial.println()
  - Argument can be any type
    - Serial.println("String to print");
    - Serial.print(14); // no newline included
    - NOTE: cannot do this – Serial.println("X = " + x); because string concatenation is not supported
    - Do this instead –

```
Serial.print("X = ");
Serial.print(x);
```

Timing in Java

- Use Thread.sleep()
  - Argument is integer number of milliseconds before the method returns

```
for (int i=0; i < endTime; i++) {
    Thread.sleep(1000);
    System.output.println(i + " seconds have elapsed");
}
```

Exceptions

- Deviations from the normal flow of control
- "Old style" error checking:

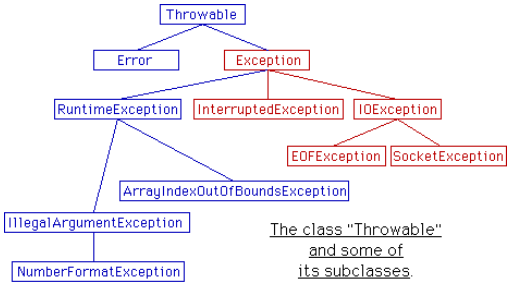
```
if (i < 0 || i >= A.length) {
        // handle out of range index
}
else {
        // access array element A[i]
}
```

- Exceptions allow us to be a bit more general

Try/Catch Block

```
try {
        // arbitrary code that might throw an
        // exception when something goes wrong
}
catch (Exception e) {
        // handle the thrown exception
}
```

Unchecked / Checked



Back to Java Timing

- The Thread.sleep() method can throw the "InterruptedException," so we enclose it in a try/catch block

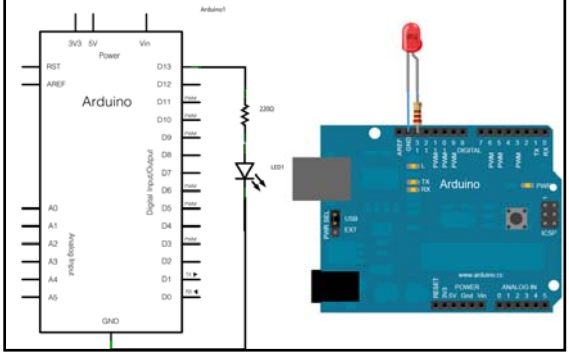
```
for (int i=0; i < endTime; i++) {
    try {
        Thread.sleep(1000);
    } catch (InterruptedException e) {
        // default action
        e.printStackTrace();
    }
    System.output.println(i + " seconds have elapsed");
}
```

Arduino Input/Output

- 20 pins on physical chip can be configured to do digital input, digital output, analog input, analog output (not all pins can do each function)
- We first configure pins at startup, then use them

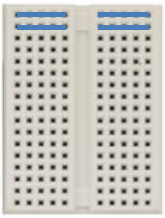
```
const int myPin = 13;
void setup() {
   pinMode(myPin, OUTPUT);
}
void loop() {                    //generates square wave
   digitalWrite(myPin, LOW);
   digitalWrite(myPin, HIGH);
}
```

Digital Output LED



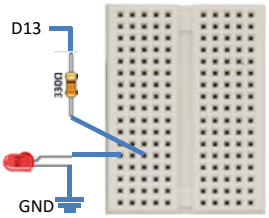
Building Circuits

- 5 horizontal holes are connected:



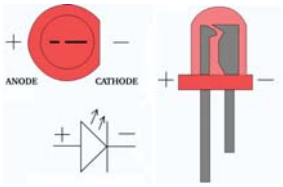
Building Circuits

- Connect components using breadboard:



LEDs

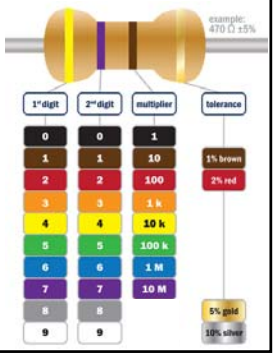
- Anode is "+" side, cathode is "–" side
- Anode has longer lead (assuming not clipped)
- Cathode is the flat side on LED body



Resistor Color Codes

- 1st two digits are values
- 3rd digit is multiplier
- 4th digit is tolerance
- 200 to 500 Ω gives good light out of LED
- We will use 330 Ω, or orange-orange-brown

$33 \times 10^{1}\ \Omega$

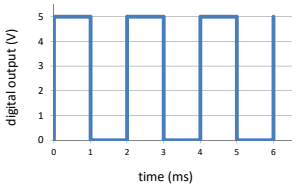


Pulse Width Modulation (PWM)

- Analog output, built using a digital output
- Technique is to exploit the fact that many physical devices are slow, and respond to average of a fast-moving signal
  - E.g., What does our eye do with 30 frames/sec?
  - Our brain smooths out the motion so it looks continuous to us
- Send digital signal up and down quickly, and the "analog output" is the average value

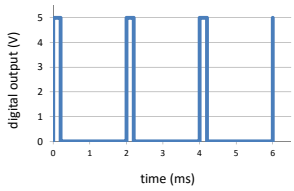
50% Analog Output

- 500 Hz period (2 ms)
- Repeating signal, ½ time 5 V and ½ time 0 V
- Average is 2.5 V



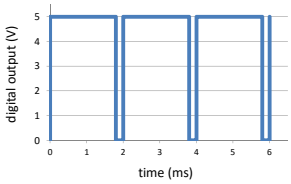
10% Analog Output

- Same 500 Hz period (2 ms)
- In this case, 10% time 5 V and 90% time 0 V
- Average is 0.5V



90% Analog Output

- Same 500 Hz period (2 ms)
- In this case, 90% time 5 V and 10% time 0 V
- Average is 4.5V

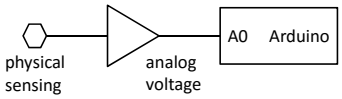


analogWrite()

- analogWrite(pin, value)
  - pin is one that supports PWM outputs
  - value is 8-bit analog value (range is 0 to 255)
- Useful for slow-moving physical devices, e.g.,
  - LEDs (actually, it is our eyes that are slow)
  - 5 V motors (hard to start/stop at 500 Hz)
- Can be used for other devices if averaging is done by circuitry between Arduino and device

Analog to Digital Conversion

- Convert physical property to voltage signal
- A/D converter on Arduino converts voltage signal to digital representation
  - 10-bit A/D converter has range 0 to $2^{10} - 1$ (0 to 1023) for voltage range 0 to V_{REF}



Studio Today

- Come to Urbauer labs
- Form groups of 2 to 4 – we will loan you an Arduino for today
- Do the exercises
  - Hello world
  - Simple heartbeat on Arduino and on PC in Java
- Get signed out by a TA

Arduino Kits Available Wednesday

- Arduino kits will go on sale Wed. (in lab)
- They will be needed for Assignment 2
  - Cost is $90 (payable via cash or check to WU, only)
- After Wed., they will be available in CSE Dept. office (Bryan 509) during normal office hours