ChickTECH

SOFT CIRCUITS

Workshop Team

Teresa, TJ, Rena, Helena, Liam, Joli, Rana, Taishi

Agenda

- Circuit basics
- Plan your project
- Gather your materials
- Start sewing your circuit
- LUNCH
- Arduino code basics
- Program your microcontroller
- Showcase your project

Soft Circuit Basics

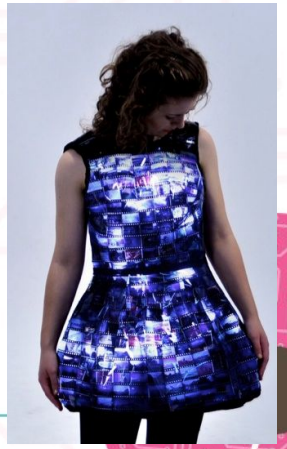
What is a circuit?

- What is a circuit?
- What is a soft circuit?

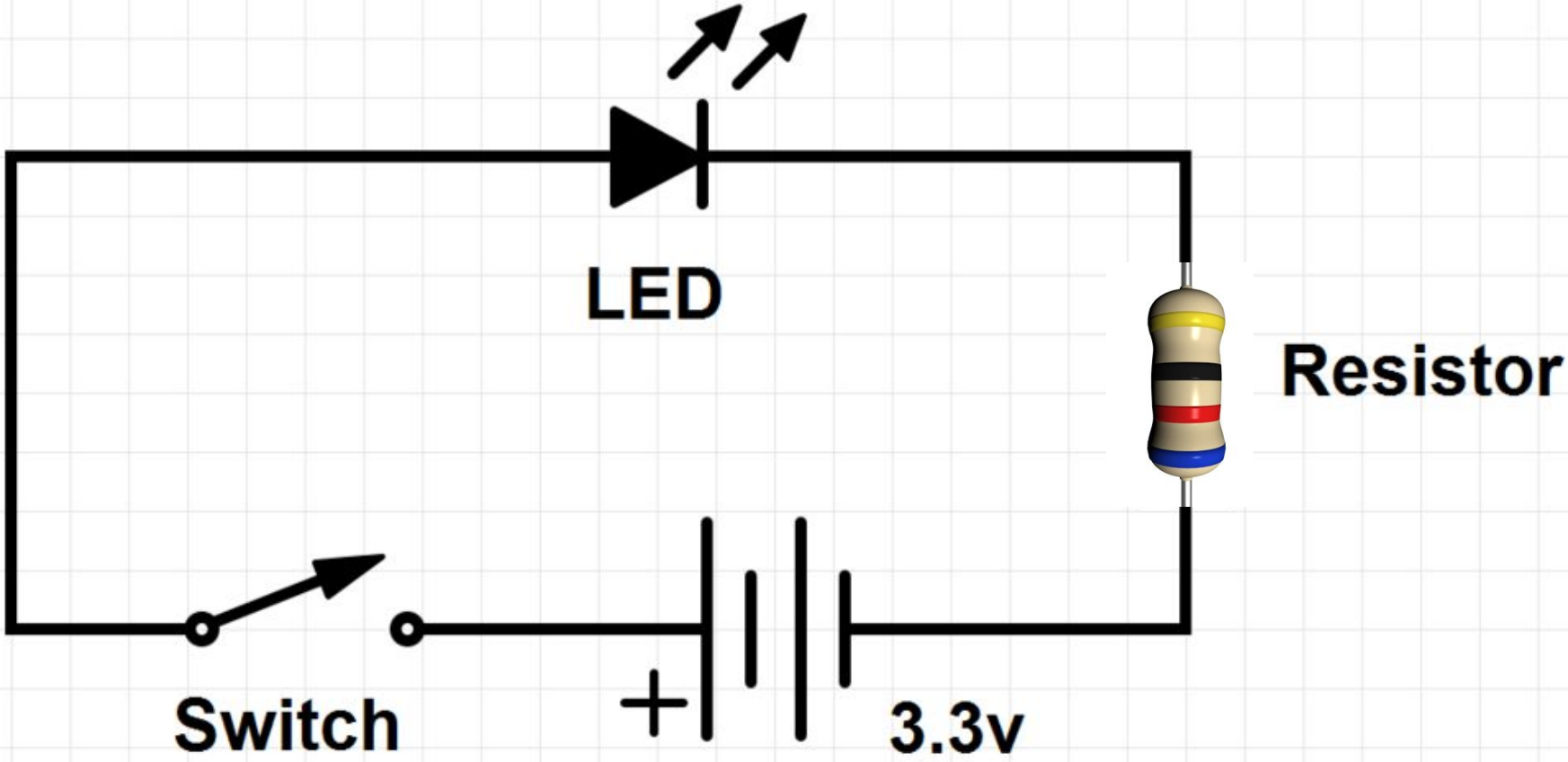
What is a circuit?

- An **electric circuit** is a path in which electrons from a voltage or current source flow. **Electric** current flows in a closed path called an **electric circuit**.
- We build electronic devices by adding different components to a circuit. Everything from the simplest light-up toy to the most complex supercomputer is an electronic circuit.

What are soft circuits?



Basic Circuit Example



Circuit Basics - Vocabulary

- Current - the rate of flow of electric charge, opposite of electron flow - measured in amperes or amps, denoted A
- Voltage - potential difference in electromotive force - measured in volts, denoted V
- Resistance - the amount by which a conductor opposes the passage of current - measured in Ohms, denoted Ω
- Ground (verb) - to connect an element to a negative node

Circuit Basics - Vocabulary

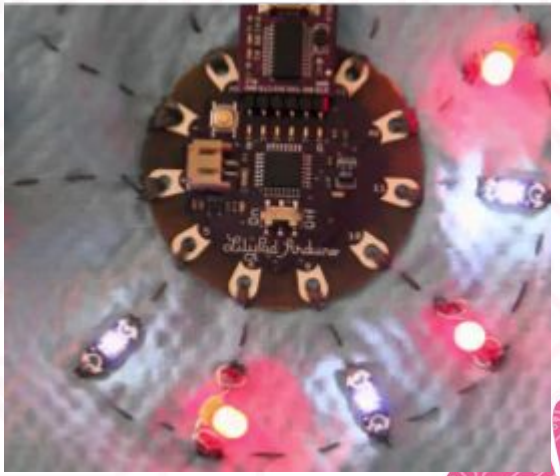
- Series - Hardware elements connected in series have no branches in the wire connecting them
- Parallel - Hardware elements connected in parallel are on different branches of wire

	Series	**Parallel**
Resistors	Simple addition	Reciprocal addition
Capacitors	Reciprocal addition	Simple addition
Inductors	Simple addition	Reciprocal addition
Voltage Sources	Simple addition	N/A
Current Sources	N/A	Simple addition

Projects

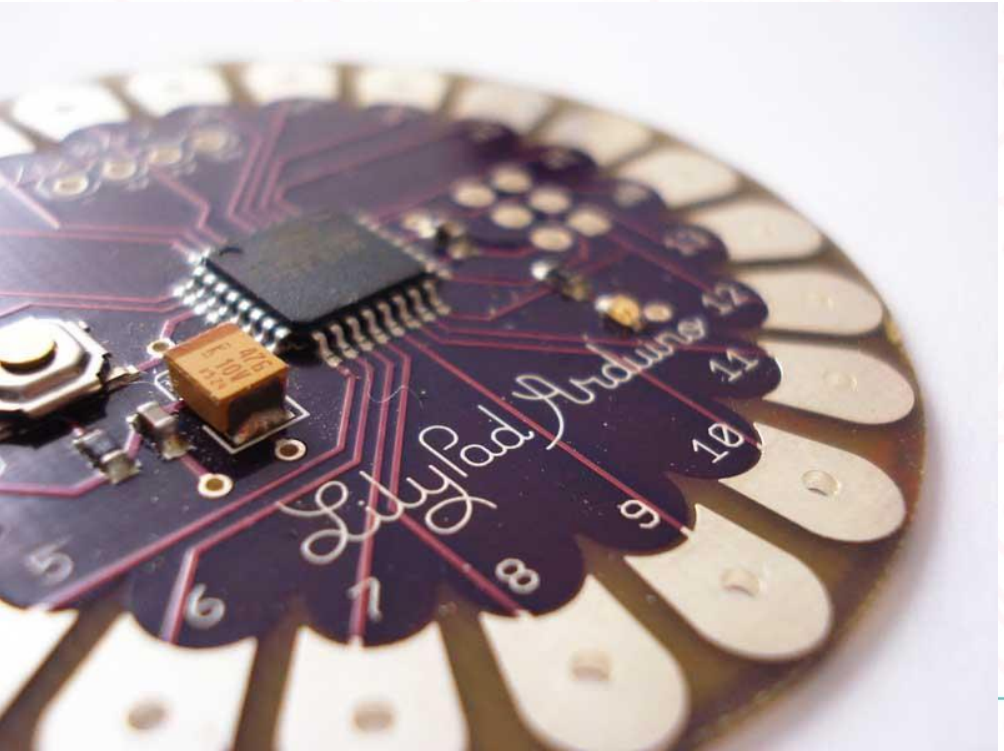
The main project

- What if you wanted to make your circuits more complex?
  - LED lights up in response to touch
  - LED blinks in a certain pattern
  - Play music
- We'll need to use code!



What is a LilyPad Arduino?

- Arduino is a microcontroller (mini computer) that you can program and add inputs (like a touch sensor) and outputs (like LEDs)

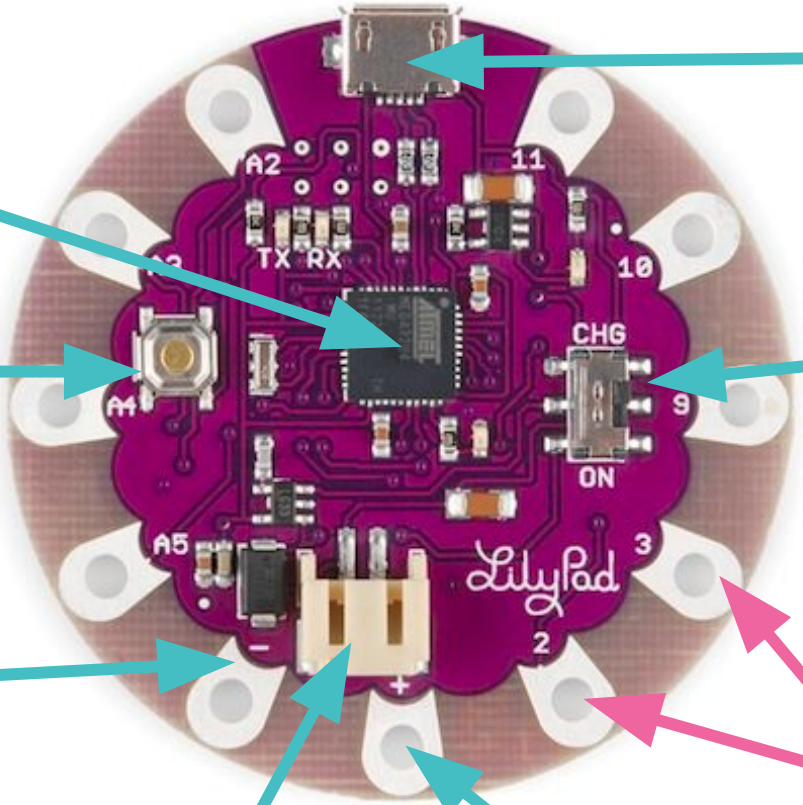


- LilyPad Arduino is specifically for soft circuits
- We'll write code and upload it through a USB cable

Parts of a Lilypad

Microcontroller
Code gets uploaded on here

USB port
Connect to computer to charge battery or upload code



Reset button

On/off switch
"CHG" is like OFF, but when plugged into computer, battery will charge

Always ground (0 volts)

Input/output pins

Battery port

Always "on" (3.3 volts)

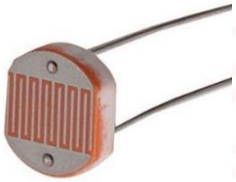
What are we going to do?

- We'll connect different inputs and outputs to the LilyPad's pins, then use code to control the inputs and outputs

Inputs

Snaps

Touch



Light sensor

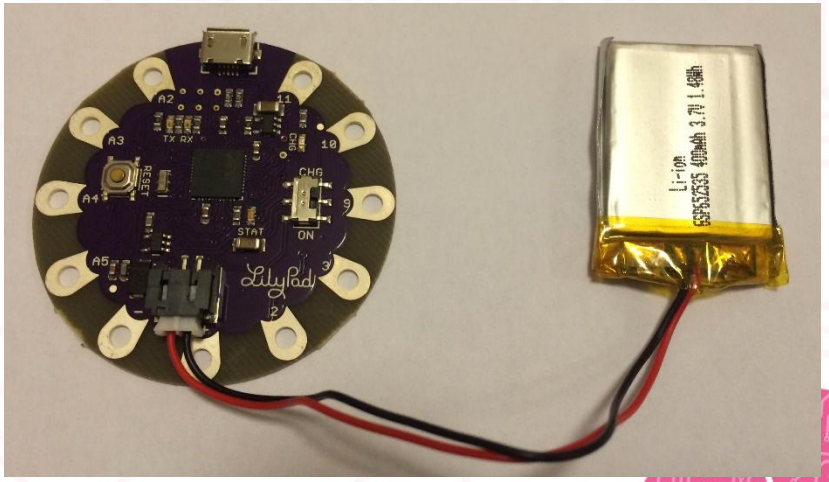
Outputs

LED

Speaker

About the battery

- Lithium
- Rechargeable – charges when LilyPad is connected to computer
- Outputs max 3.7 volts



Project Idea: LED Bracelet

Project Idea: LED Tote Bag

Project Idea: LED Stuffed Animal

Project Idea: Wearables

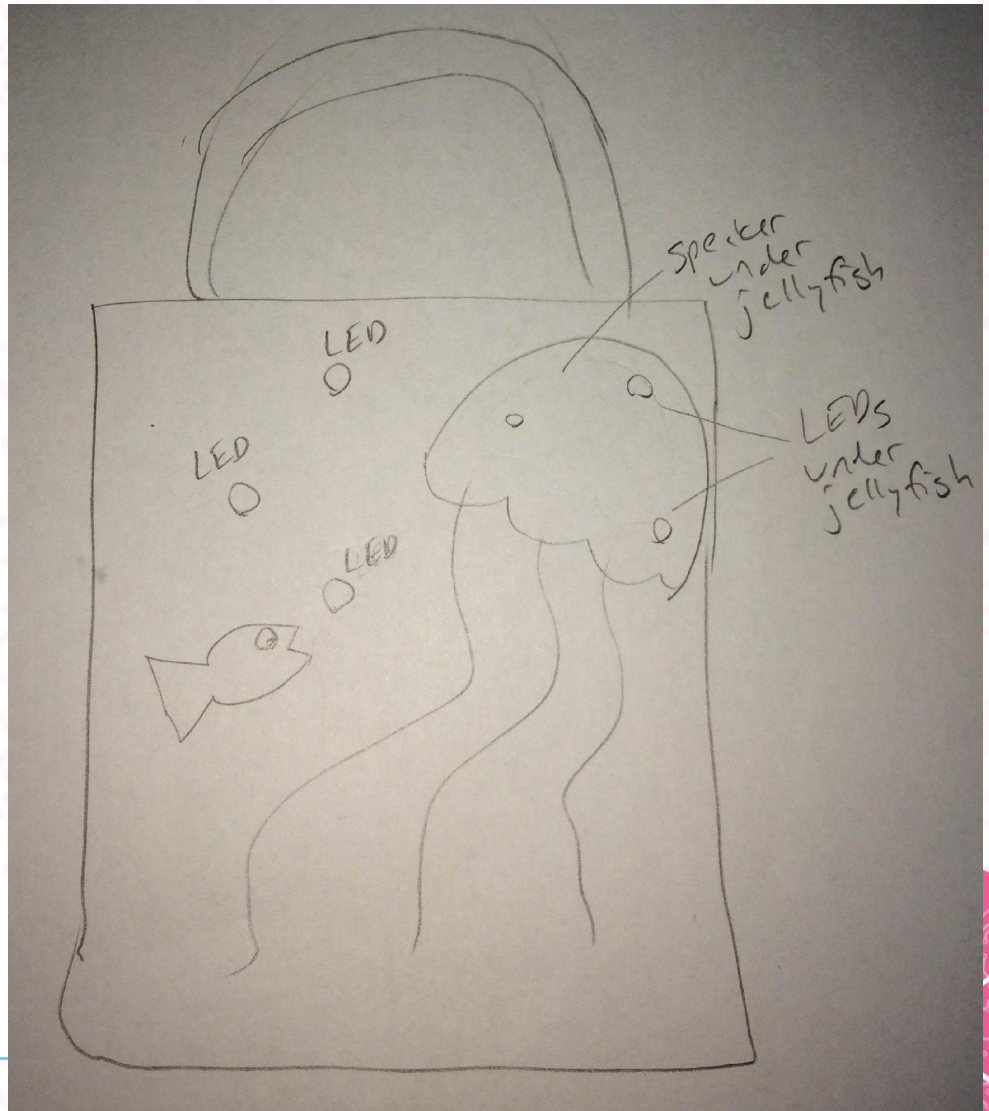
Plan Your Project

Planning your project

List the things you want your creation to do. Example:

- When I touch the fish, blue LEDs will light up
- A different note will play when each LED lights up
- The jellyfish will light up

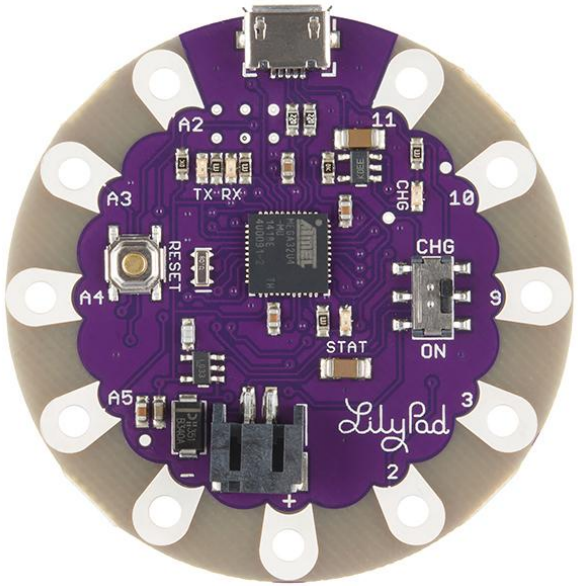
Planning: diagram



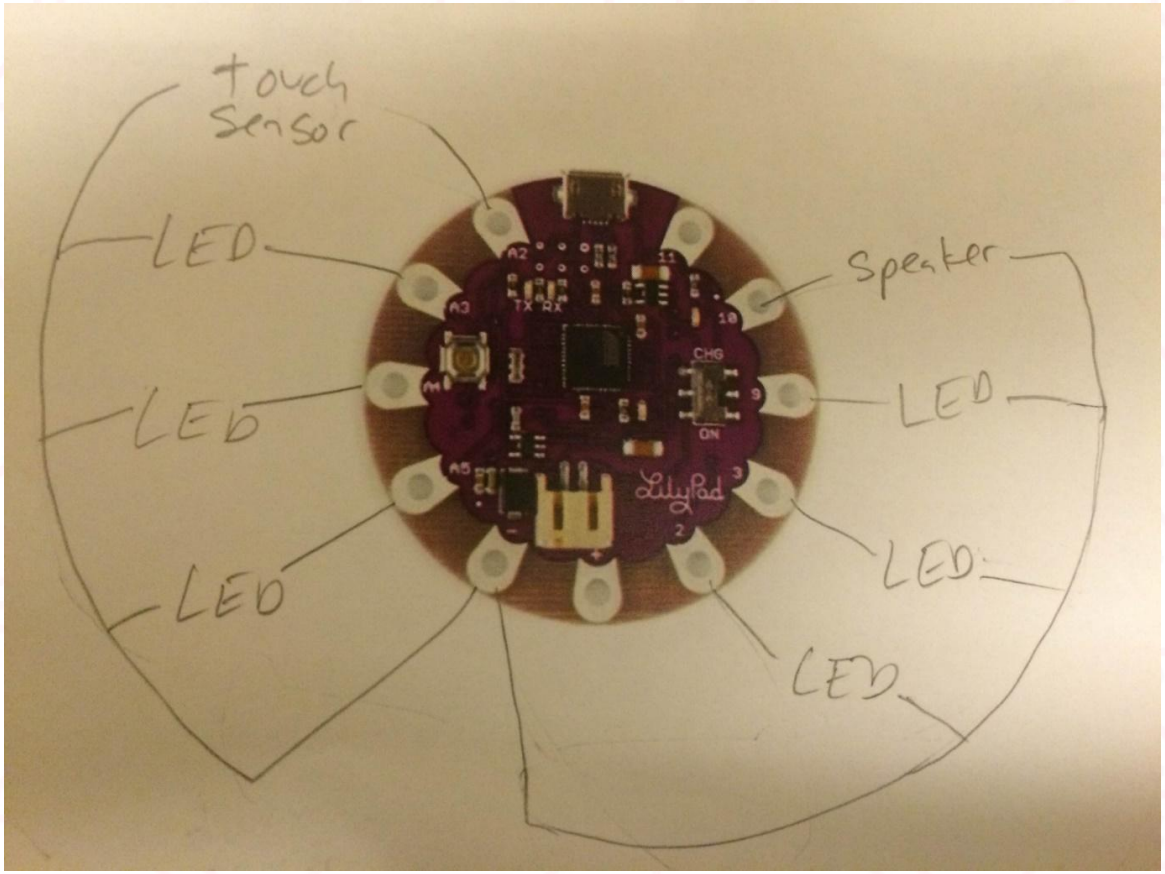
Planning: Circuit Diagram

WIRE	LAMP INCANDESCENT
CONDUCTORS CONNECTED CONNECTED NOT CONNECTED	FUSE RESISTORS FIXED VARIABLE (POTENTIOMETER) RHEOSTAT
GROUND	
CELL	SWITCH
BATTERY	VOLTMETER
OR	AMMETER

Include ALL hardware elements in your drawing, including the LilyPad, LEDs, the battery, switches, and resistors.

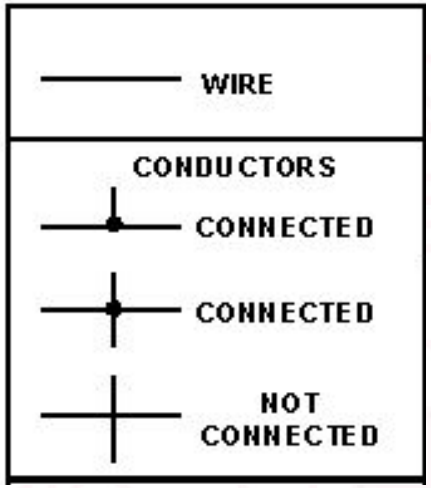


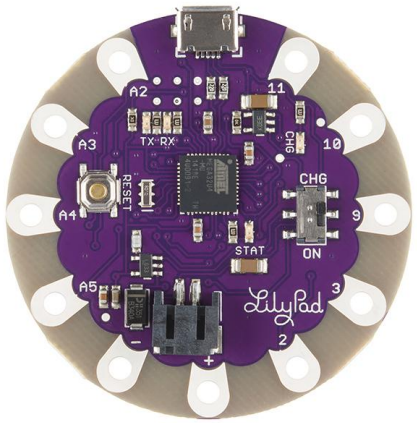
Planning: Circuit Diagram Example



Note: Touch sensor/light sensor can only be used on “A” pins! Remember everything has to complete the circuit to work...so make sure to tie it back to ground.

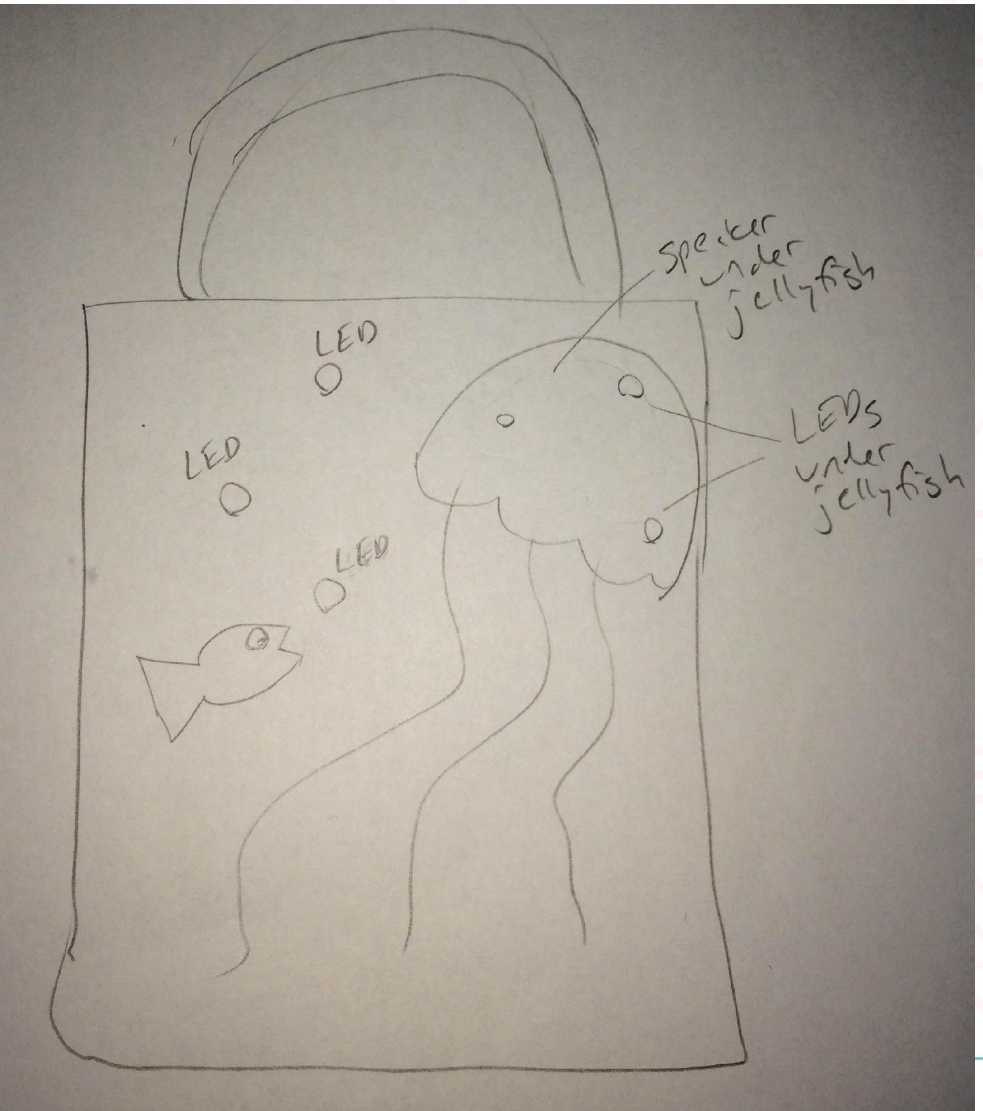
Planning: Wire (Thread) Path

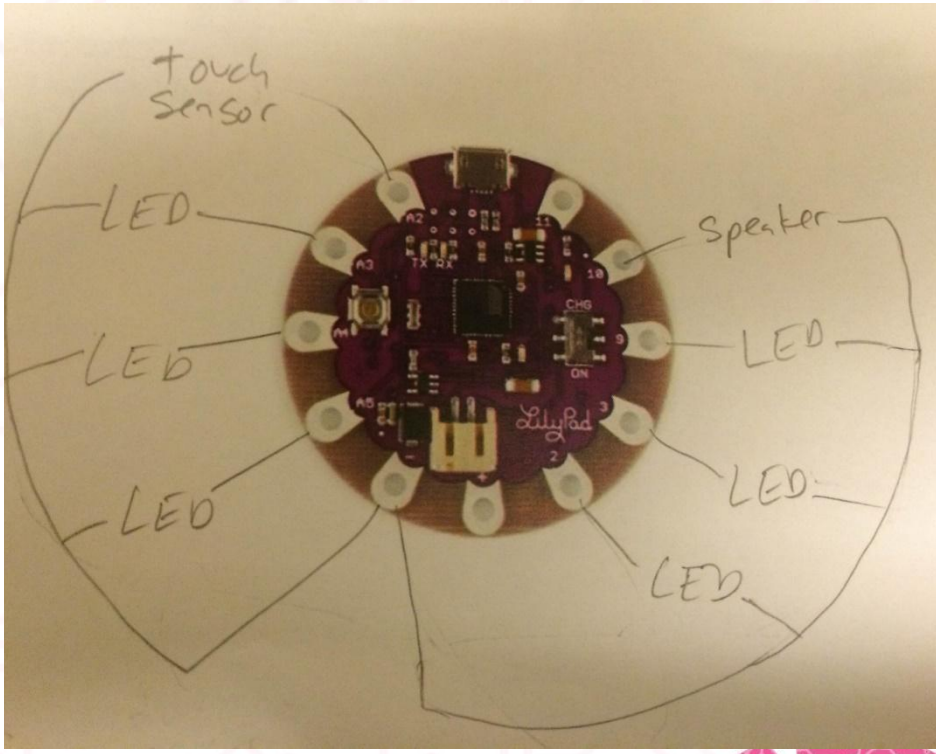




- Plan the path that you will sew to connect your hardware elements
- Do NOT cross two threads - this will cause a **short**
- Sew several loops of thread around each pin - a single loop will cause a **weak connection**

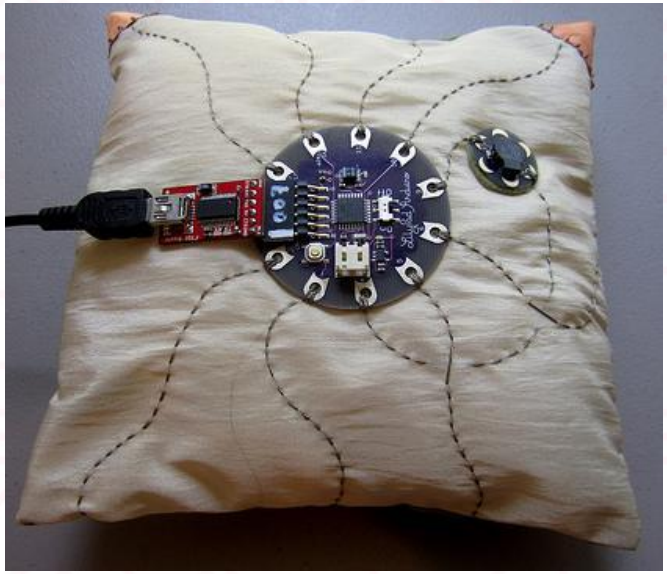
Planning: Complete Design





Tips and tricks

- Conductive thread will show unless you hide it
- Don’t cross wires!
- Use short stitches to prevent wires from crossing
- Lay out everything and use alligator clips to test before sewing
- Use appropriate lengths of thread to reduce breaks and tangles



Gather Your Materials

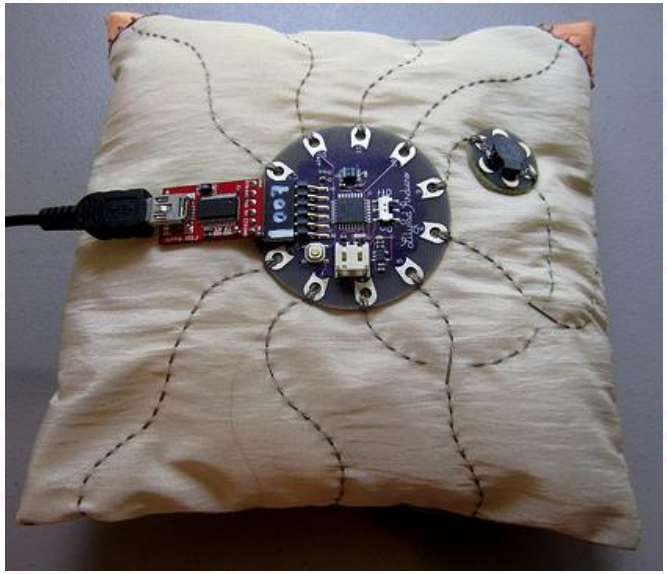
Materials

- 1 Arduino LilyPad microcontroller
- Felt/material to sew onto
- Decorations
- LEDs **and** resistors
- Needle
- Conductive thread
- Battery
- Switches, speakers, other optional hardware

Sew Your Circuits

Tips and tricks

- Conductive thread will show unless you hide it
- Don’t cross wires!
- Use short stitches to prevent wires from crossing
- Lay out everything and use alligator clips to test before sewing
- Use appropriate lengths of thread to reduce breaks and tangles



LUNCH BREAK

Let's start looking at code!

Open up the Lilypad_blink.ino file

The blink code

- **What is code?**
  - Lines of text that the computer understands and uses to execute actions.
- **What are comments?**
  - Lines of text that the computer will ignore, but you can use to write notes to yourself.
  - Either start with // on each line or multiple lines enclosed by /* and */
- **Arduino code has two main parts – setup() and loop()**
  - The lines of code in setup() run immediately after you turn on the LilyPad. The setup() code is enclosed by curly brackets { and }.
  - After setup() runs, the lines of code in loop() run over and over again until LilyPad is turned off. Loop() code is also enclosed by curly brackets.
- **Every line of code ends with a semicolon ;**

Connect LilyPad to computer

- Connect your LilyPad to your computer with the USB cable
  - When the switch is in CHG position, the CHG light will turn on → this means the battery is charging
  - Turn the switch to ON position. Switch must be in ON position for code to be uploaded.
- In Arduino program, go to Tools→Board
  - Select “LilyPad Arduino USB”
- Go to Tools->Port
  - Select port that the LilyPad is on

Upload the code

- This button  compiles your code
  - Compiling checks that there are no errors in the code
- This button  uploads your code to the LilyPad
- Try uploading the code! LED on LilyPad board should blink.

How does the blink code work?

- The LED on the LilyPad board is connected to pin #13. We'll call this `ledPin`. `int ledPin = 13` tells the code that `ledPin` is 13. `int` means that 13 is an integer.
- `pinMode(ledPin, OUTPUT)` sets `ledPin` as an output pin. Later, we will try input pins.
- We put `pinMode(ledPin, OUTPUT)` in the setup() section because we only need to do it once, when the LilyPad is turned on.

How does the blink code work? Continued.

- After the code in the `setup()` section runs, the code in `loop()` will run forever.
- `digitalWrite(ledPin, HIGH)` sends a HIGH voltage to `ledPin`
  - HIGH on the LilyPad corresponds to 3.7 Volts
- `digitalWrite(ledPin, LOW)` sends a LOW voltage to `ledPin`
  - LOW on the LilyPad corresponds to 0 Volts
- `delay(1000)` means to wait for 1000 milliseconds (1 second)

Serial prints

- Go to Tools→Serial Monitor
  - You will see it printing “LED ON” and “LED OFF”
- These are printed by the lines of code `Serial.println(“LED ON”)`
- You can put a Serial line anywhere you want to print a message
- This can help you find problems (bugs) in your code

Playing around with the blink code

- File→Save As… a new file (myblink.ino)
- Try changing the `1000` in `delay(1000)` to a different number to make it blink slower or faster
- At the top of the code before setup(), add the statement `int mydelay = 1000;`
  - Now, change the `delay(1000)` lines to `delay(mydelay)`
  - `Mydelay` is called a variable
  - Changing the value of a variable is easier than changing multiple lines of code
- Instead of `delay(1000)`, try `delay(random(1000,5000))`. This will pick a random number between 1 second and 5 seconds.

Adding external LED

- Change

```
int ledPin = 13;
```

  to `int ledPin = 9`

- Take an alligator clip. Attach one end to pin 9 of the LilyPad, one end to the long pin of the LED.
- Take another alligator clip. Attach one end to the ground (-) pin of the LilyPad, and one end to the short end of the LED.
- Upload the code to see the external LED blink!

Reading inputs

- Now we're going to try reading an input
- File->Save As… a new file (myInput.ino)
- At the top of the file before setup(), add this line:

```
int readPin = A4;
```

- Inside setup(), add these lines:

```
pinMode(readPin, INPUT);
digitalWrite(readPin, HIGH);
```

Reading inputs continued

- Delete everything inside of loop() but leave the curly brackets
- Inside of loop(), add the line:

```
int readValue = digitalRead(readPin);
```

- After that, add the line :

```
Serial.println(readValue);
```

- After that, add the line :

```
delay(1000);
```

Reading inputs Continued

- Upload the code
- Connect one end of the alligator clip to pin A4
- Open the Serial window (Tools→Serial Monitor)
- Try touching the other end of the alligator clip to the ground (-) pin of the LilyPad – watch what happens in the Serial window. `digitalRead` is reading the value on the pin.
- We made the pin high in setup() with `digitalWrite()`. But when you touch the alligator clip to the (–) pin, the voltage on the pin goes to 0.

Using if() statements

- Comment out these lines of code by adding // to the start of the line:

```
// Serial.println(readValue);
// delay(1000);
```

- Add these lines of code inside loop():

```
if(readValue == 0)
{
    digitalWrite(ledPin, HIGH);
}
else
{
    digitalWrite(ledPin, LOW);
}
```

Using if() statement and reading an input

- Upload the code
- Connect an LED between ledPin and ground (-) pin.
- LED should light up when you touch pin A4 to ground
- For an example of this code, look at the file Lilypad_input_LED.ino

Analog inputs

- File→Save As… a new file (touch.ino)
- Remove these lines of code inside loop()

```
if(readValue == 0)
{
    digitalWrite(ledPin, HIGH);
}
else
{
    digitalWrite(ledPin, LOW);
}
```

Analog inputs continued

- Change this line of code:

```
int readValue = digitalRead(readPin);
```

- To this:

```
int readValue = analogRead(readPin);
```

- After that, add these lines:

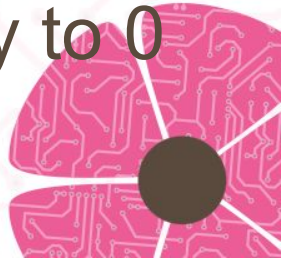
```
Serial.println(readValue);
delay(1000);
```

Testing the analog input

- Upload the code
- Open the Serial window (Tools→Serial Monitor)
- Try connecting an alligator clip to pin A4 and touch it to ground (-) pin, see what happens in the Serial window
- Instead of `digitalRead()`, which reads a 0 (low voltage) or 1 (high voltage) from a pin, we are using `analogRead()`, which reads a value from 0 to 1023. 1023 corresponds to high voltage.

Playing with the touch sensor

- Connect another alligator clip to the ground (-) pin
- Touch the two alligator clips together so that pin A4 is connected to ground. What happens to the value?
- Touch both alligator clips to your finger but not touching each other. What happens?
- Pin A4 is being connected to the ground pin through your finger. This lowers the voltage on pin A4, but it doesn't get lowered completely to 0 because your finger has resistance.



Using if() statements with the analog input

- Comment out this line of code by adding // to the start of the line:

```
// delay(1000);
```

- Add this code after the statements you already have in loop()

```
if(readValue < 900)
{
    Serial.println("I've been touched!");
    delay(1000);
}
```

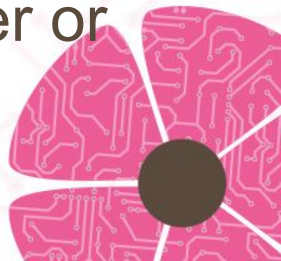
- Upload the code and open the Serial window. See if touching the alligator clips prints out the message, you may need to adjust the value of 900.
- For an example of using this code to light an LED, see LilyPad_touch_LED.ino

Analog input with light sensor

- Connect one pin of the light sensor (photoresistor) to pin A4 and the other pin to ground (-) pin
- Open the Serial window and notice how the value changes when you cover and uncover the light sensor

Playing sounds

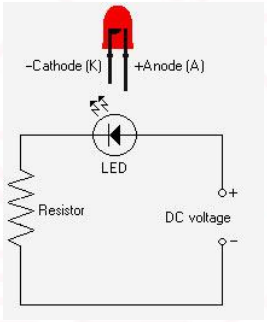
- Open the file LilyPad_speaker.ino
- Connect one pin of the speaker to speakerPin and the other pin of the speaker to ground (-)
- `tone(speakerPin, NOTE_E4)` plays a note of frequency `NOTE_E4` to a speaker connected to `speakerPin`
  - The different `NOTE`s are defined in the pitches.h file
- The tone will continue to play until you play a different tone or you use `noTone(speakerPin)` to stop the tone
- Use `delay()` after `tone()` to make notes shorter or longer



Debugging

Debugging

Polarity = having a property determining direction or orientation, such as the poles of a magnet (+/-), or the direction requirement of current through an LED



Common Hardware Issues:

- Weak pin connections - add more thread around pins to stabilize connections
- Shorts - find crossed wires and re-sew the path
- Unlit LEDs - check LED **polarity** and reverse if needed

What we've learned so far

- What is code
- How the LilyPad code works
- How to read inputs from pins
- How to control external components (outputs)

Inputs

Snaps

Touch

Photoresistor

Outputs

LED

Speaker

Thank you!