

Arduino 101

Programming

Week 3

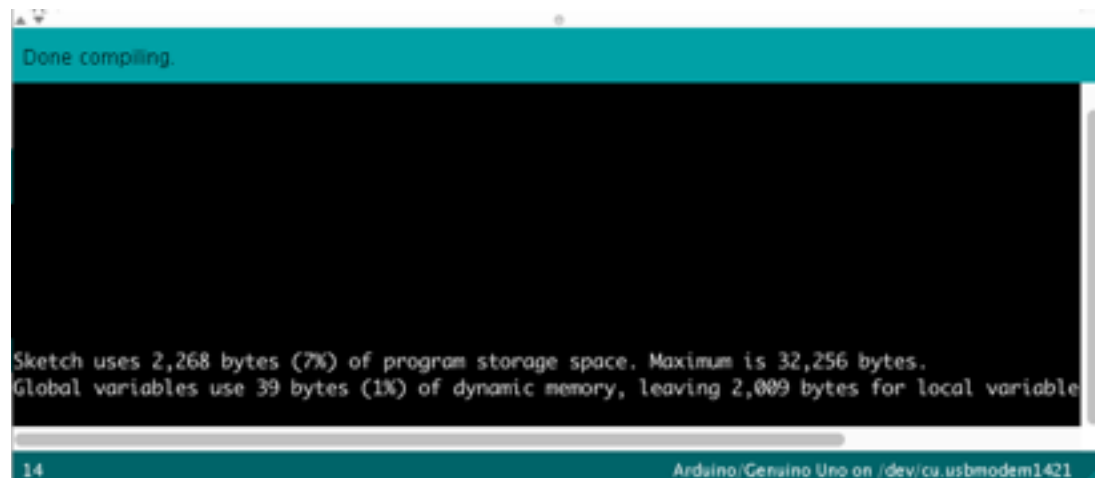
Arduino IDE

Line Numbers



Go to **File -> Preferences** and turn on "Display Line Items".

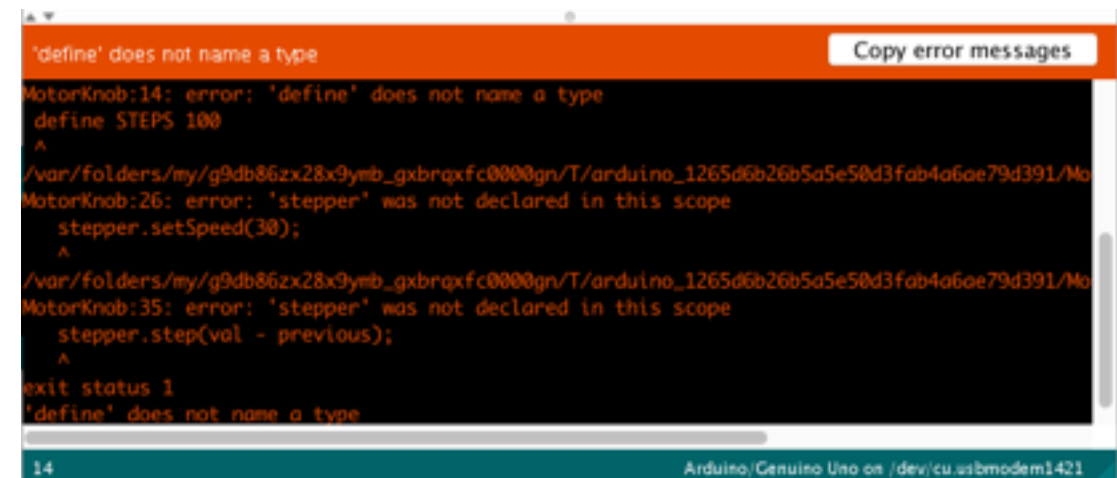
The Message Panel



Done compiling.

Sketch uses 2,268 bytes (7%) of program storage space. Maximum is 32,256 bytes.
Global variables use 39 bytes (1%) of dynamic memory, leaving 2,009 bytes for local variables.

14 Arduino/Genuino Uno on /dev/cu.usbmodem1421



'define' does not name a type Copy error messages

MotorKnob:14: error: 'define' does not name a type
define STEPS 100
^
/var/folders/my/g9db86zx28x9ymb_gxbrqxfc0000gn/T/arduino_1265d6b26b5a5e50d3fab4a6ae79d391/Mo
MotorKnob:26: error: 'stepper' was not declared in this scope
stepper.setSpeed(30);
^
/var/folders/my/g9db86zx28x9ymb_gxbrqxfc0000gn/T/arduino_1265d6b26b5a5e50d3fab4a6ae79d391/Mo
MotorKnob:35: error: 'stepper' was not declared in this scope
stepper.step(val - previous);
^
exit status 1
'define' does not name a type

14 Arduino/Genuino Uno on /dev/cu.usbmodem1421

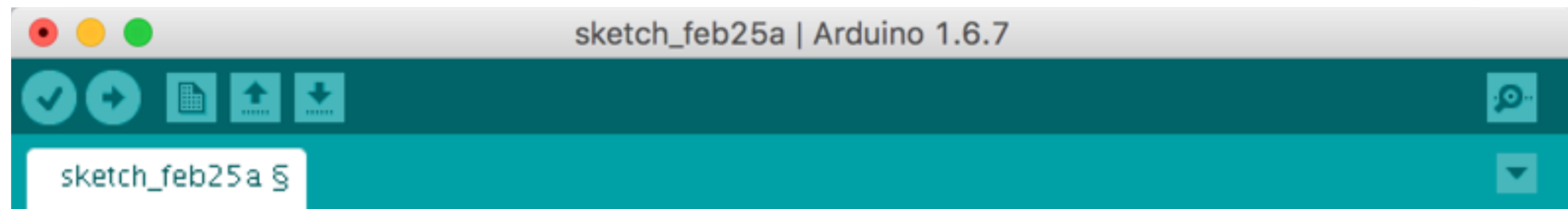
The “Message Pane” could display confirmation, error, or other messages that you programmed into your sketch.

Block Completion Indicators

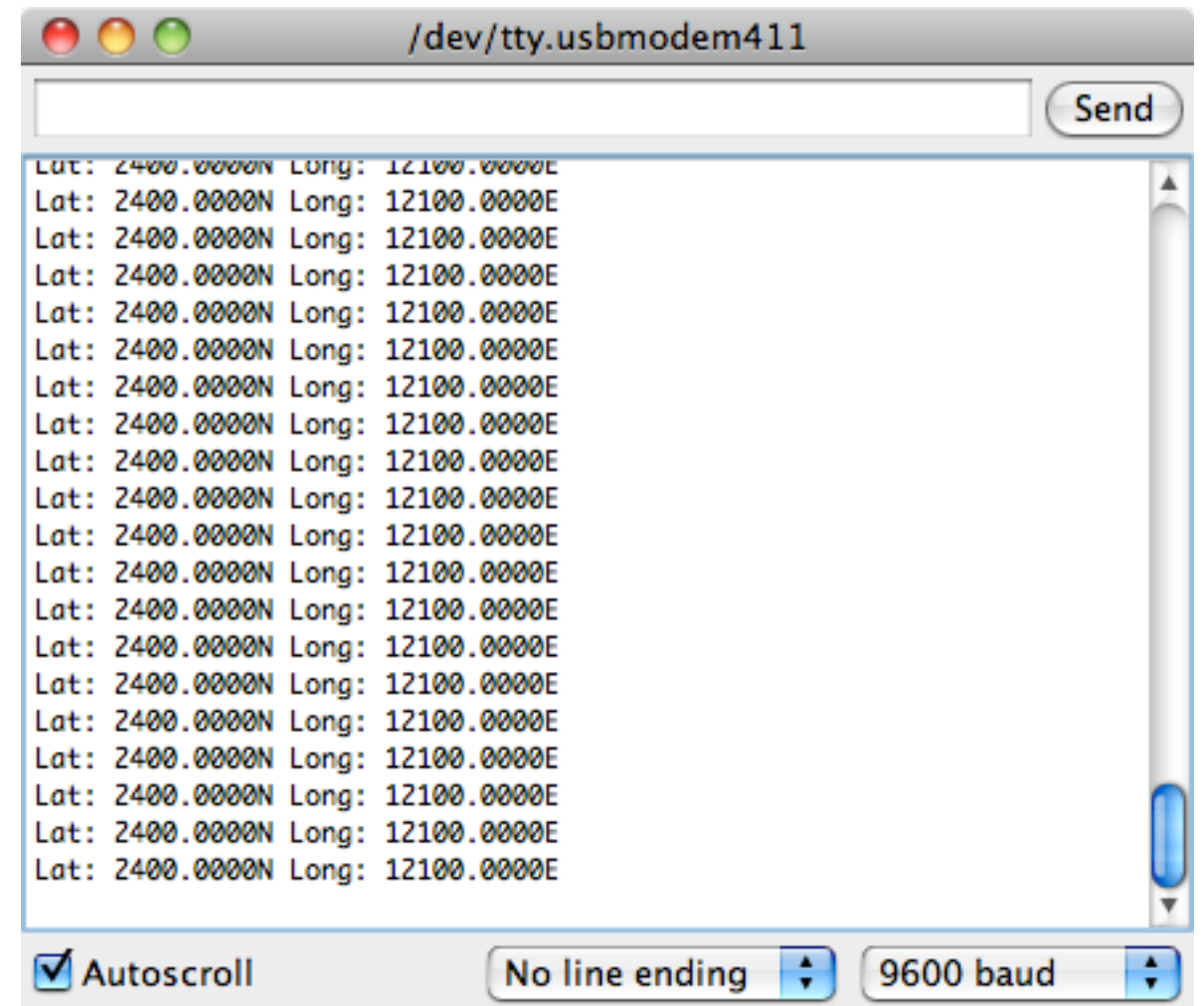
```
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

```
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

The Toolbar



Serial Monitor



The Compiler



Code

Variables

```
int pushButton = 0 ;
```

Variables are the constructs that programming languages use to store changing information in a program

Data Types

Integer (int): 2^{16} different values. It could range from -32,768 to 32,767

Bytes: 2^8 or 256 different values. It ranges from 0 to 255

Long: 2^{32} different values. They are for storing very large values. It can range from -2,147,483,648 to 2,147,483,647.

Boolean (bool): It can only be true or false, and ideally take up just one bit in memory

Functions

```
int counter(int number) {  
    int count = number + 1;  
    return count;  
}
```

Functions are blocks of programming code that perform a specific function.

Functions

```
void loop() {  
    // put your main code here, to run repeatedly:  
    int count = counter (1);  
}
```

Functions

```
boolean isNegative(int number) {  
    boolean result = false;  
    if (number < 0) {  
        result = true;  
    }  
    return result;  
}
```

Keywords

i.e. for, if, while, digitalWrite(), Serial, int, byte, and String.

Math Operators

Operation	Symbol
Addition	+
Subtraction	-
Multiplication	*
Division	/
Modulus	%

Math Operators

Operation	Symbol
Increment value	++
decrement value	--
add right value to left	+=
subtract right value from left	-=
multiply right value by left	*=
divide right value by left	/=

Comparison Operators

Operation	Symbol
Greater than	>
Less than	<
Greater than or equal to	>=
Less than or equal to	<=
Equal	==
Not equal	!=

Program Flow

```
int counter = 0;

void setup() {
  Serial.begin(9600);
  Serial.println("Program is starting");
}
void loop() {
  counter++;
  Serial.print("Loop Function Count ");
  Serial.println(counter);
  delay(1000);
}
```

Conditional Statements

```
int threshold = 45;
int sensorlevel = analogRead(A0); // read an analog input

if (sensorLevel > threshold) {
    digitalWrite(13, HIGH); // turn LED on
} else {
    digitalWrite(13, LOW); // turn LED off
}
// code flow continues here after the conditional
```

They help testing if a statement is true or false.

While Loop

```
// read the button on digital input 3:  
while (digitalRead(3) == HIGH) {  
    blink();  
}  
// code flow continues here after the conditional
```

They direct the program to continue in a loop until that condition is no longer true.

For Loop

```
for (int counter = startingValue; counter < endingValue; counter++) {  
    // do stuff  
}
```

Use it when you want to repeat an action a particular number of times.

Constants

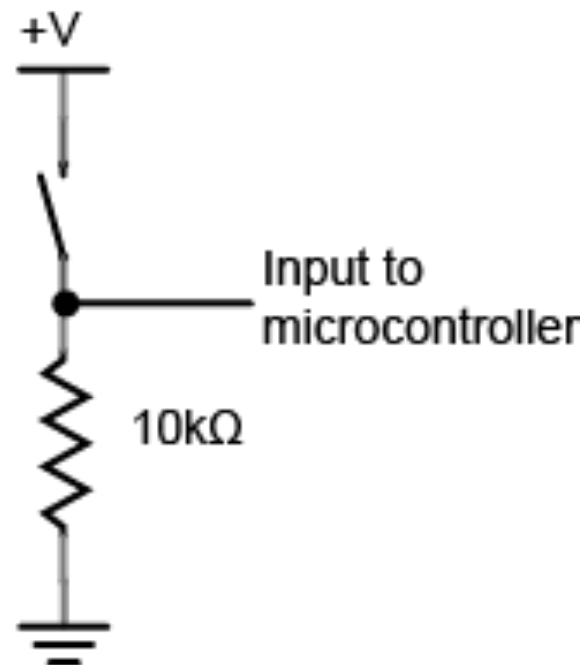
```
const int LEDpin = 3;  
const int sensorMax = 253;
```

```
//Or you can use define:  
#define LEDPin 3  
#define sensorMax 253
```

They're a useful way to label numbers that get used repeatedly and doesn't change its value within your program.

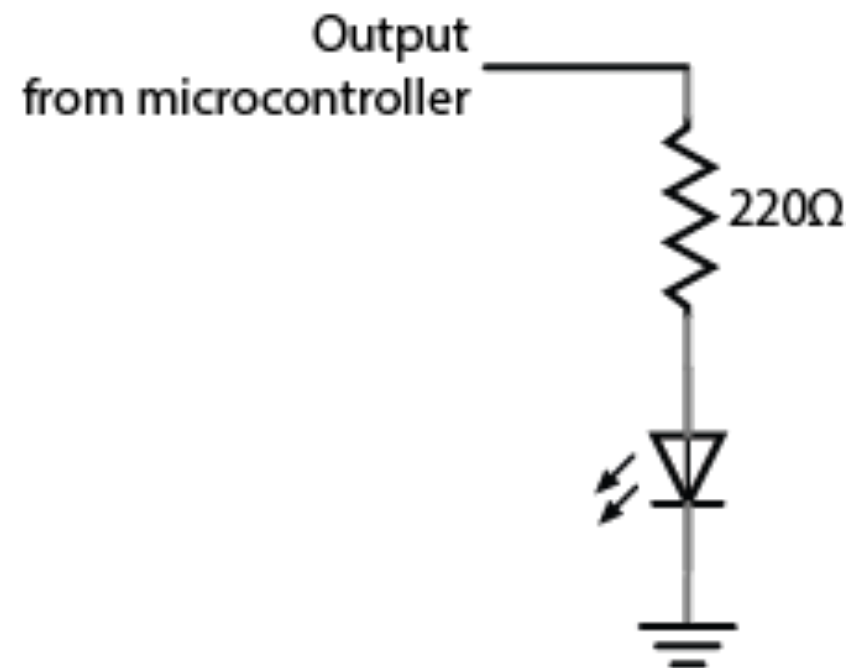
Digital Input & Output

Digital Input



If voltage is flowing, the circuit is on. If it's not flowing, the circuit is off.

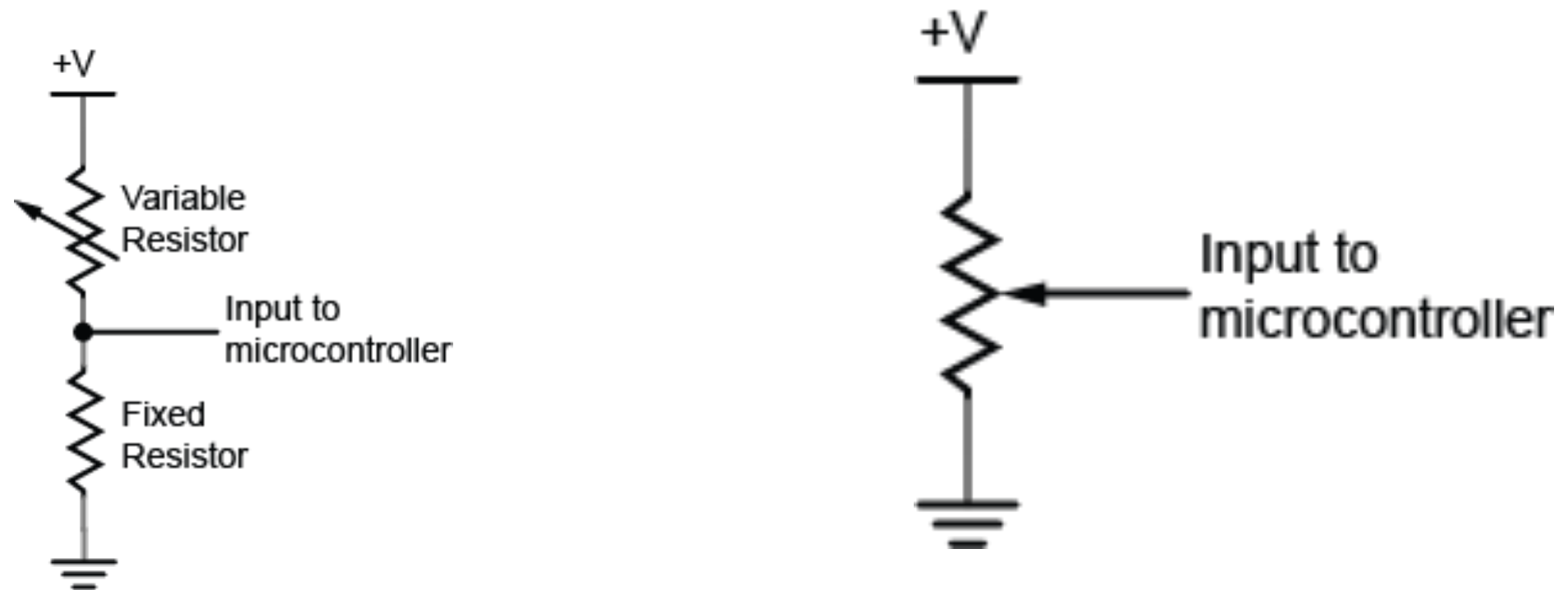
Digital Output



With a digital output you can either turn something off or on.

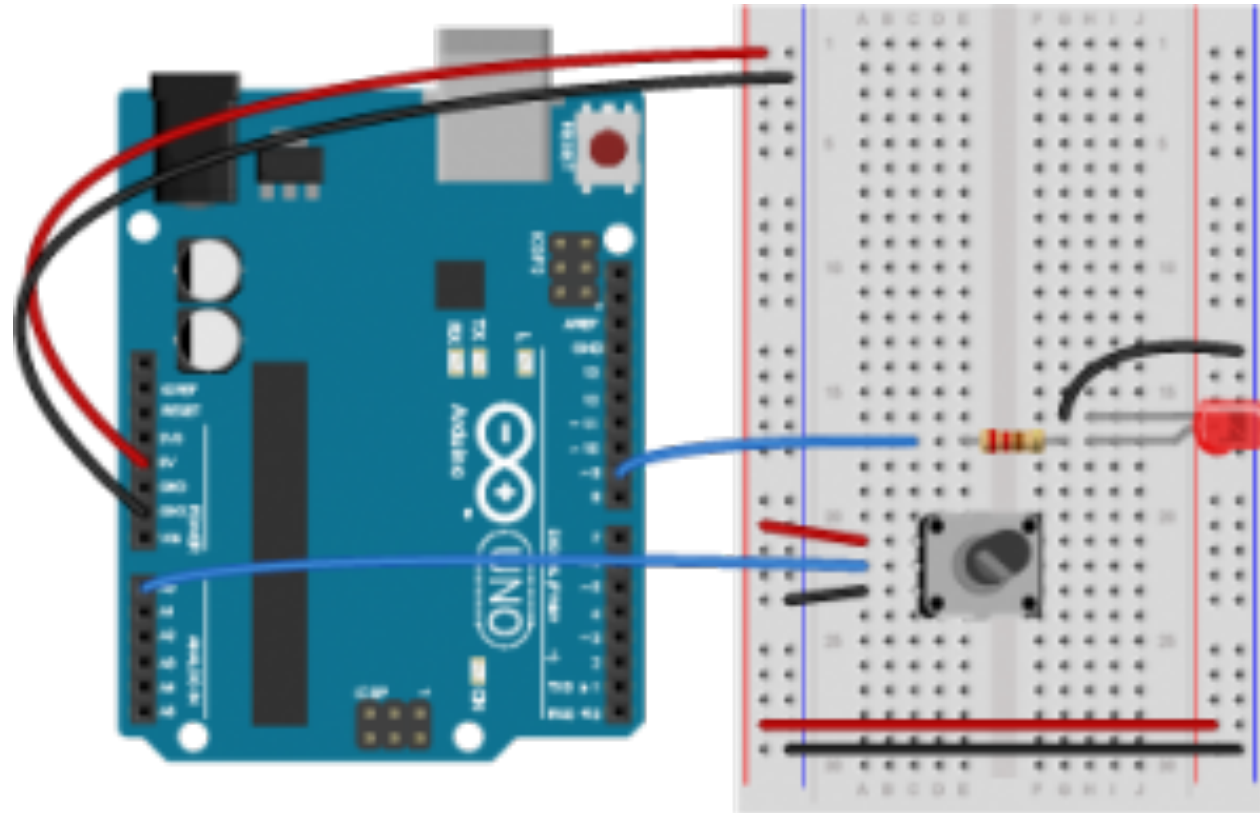
Analog Input

Analog Input



An input that can read a variable voltage, typically from 0 volts to the maximum voltage that powers the microcontroller itself.

Analog Input



Luv-o-meter

A luv-o-meter is a device that measures a person's potential to be a lover, and displays it on a graph of lights. In gaming arcades, the luv-o-meter is usually a handle that a person grips, and his or her grip is measured either for its strength or its sweatiness. Your luv-o-meter can measure any analog physical quantity that you want, providing you have a sensor for it. Make sure the display is clear, so the participant knows what it means, and make sure it is responsive.