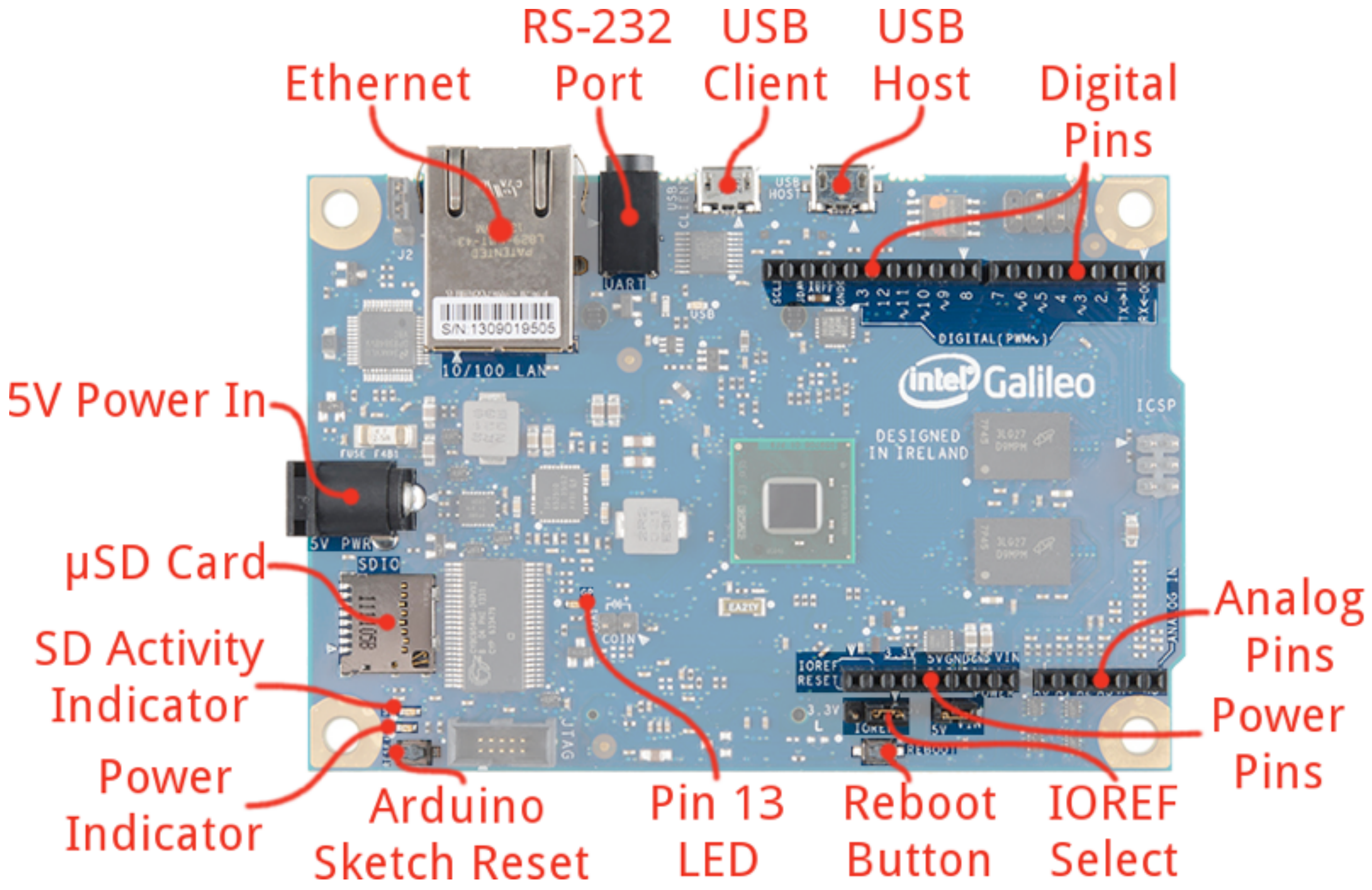
First Steps

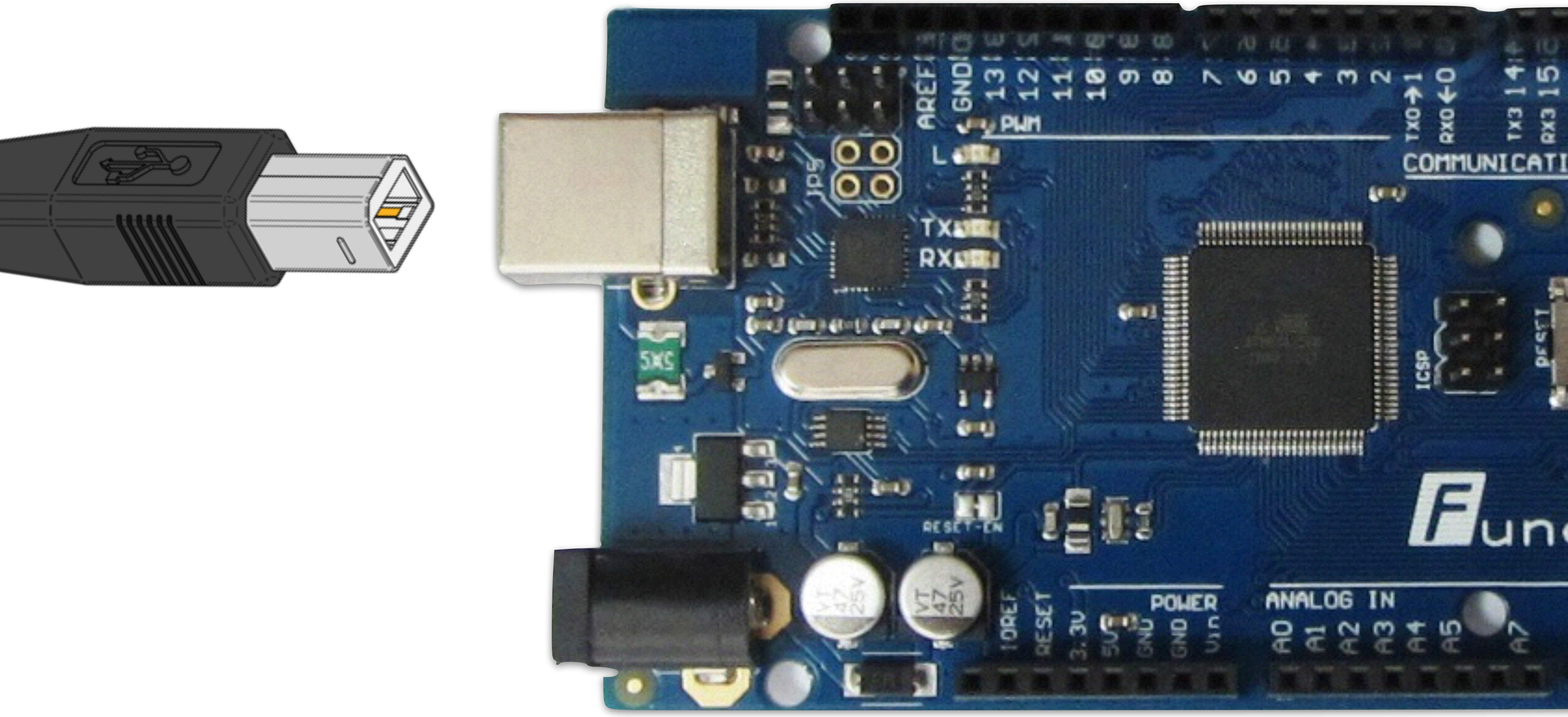
Programming for Engineers
Winter 2015

Andreas Zeller, Saarland University

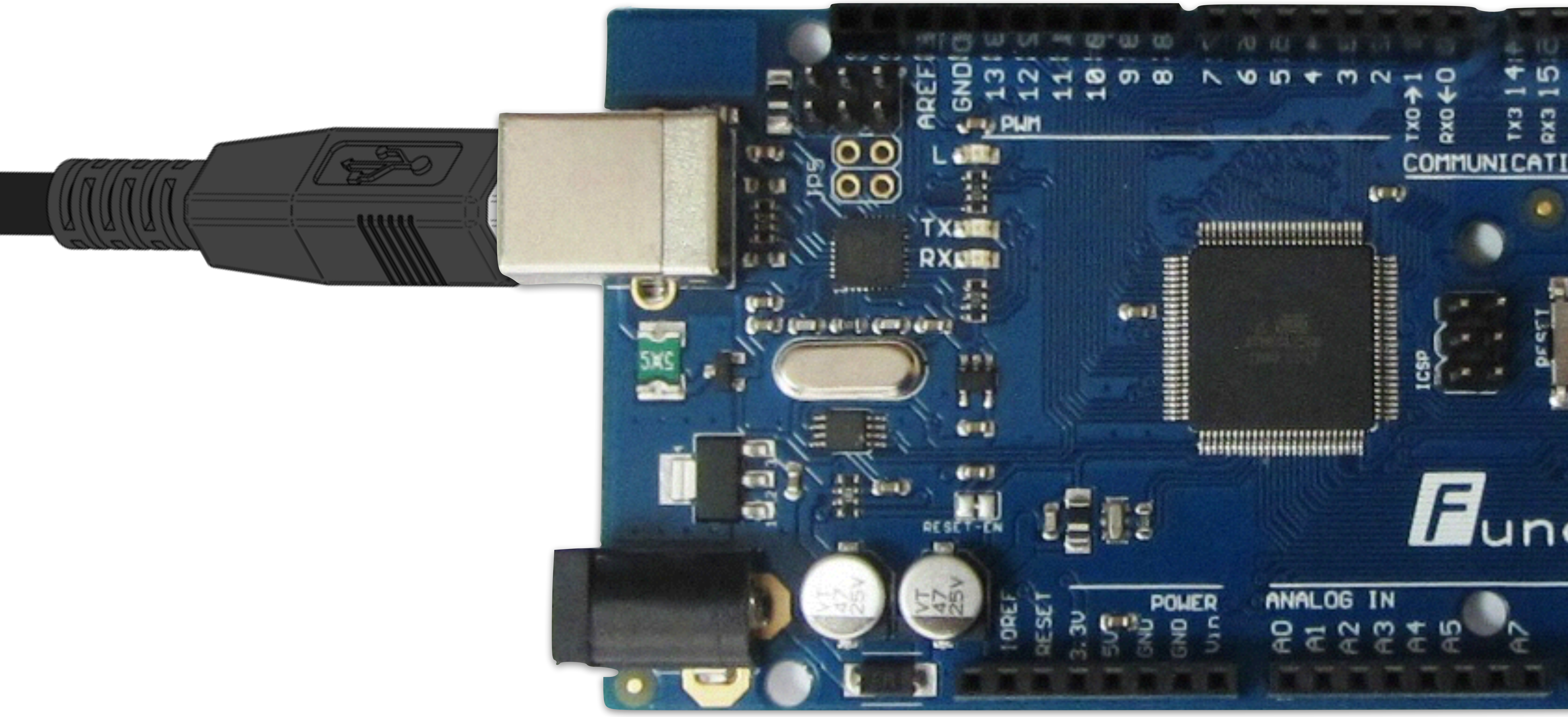
The Arduino Board



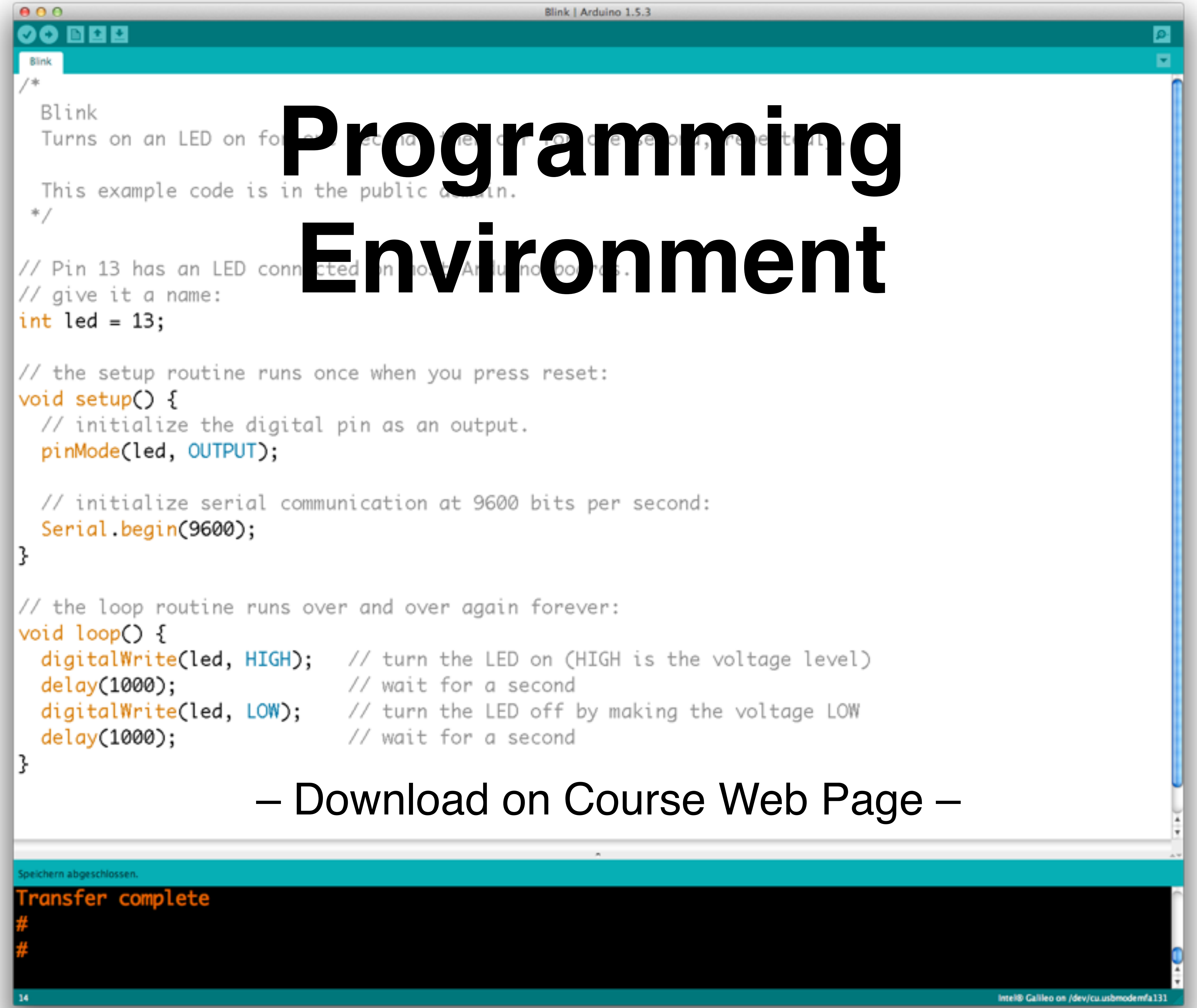
USB Connection



USB Connection



Programming Environment



```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);

  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

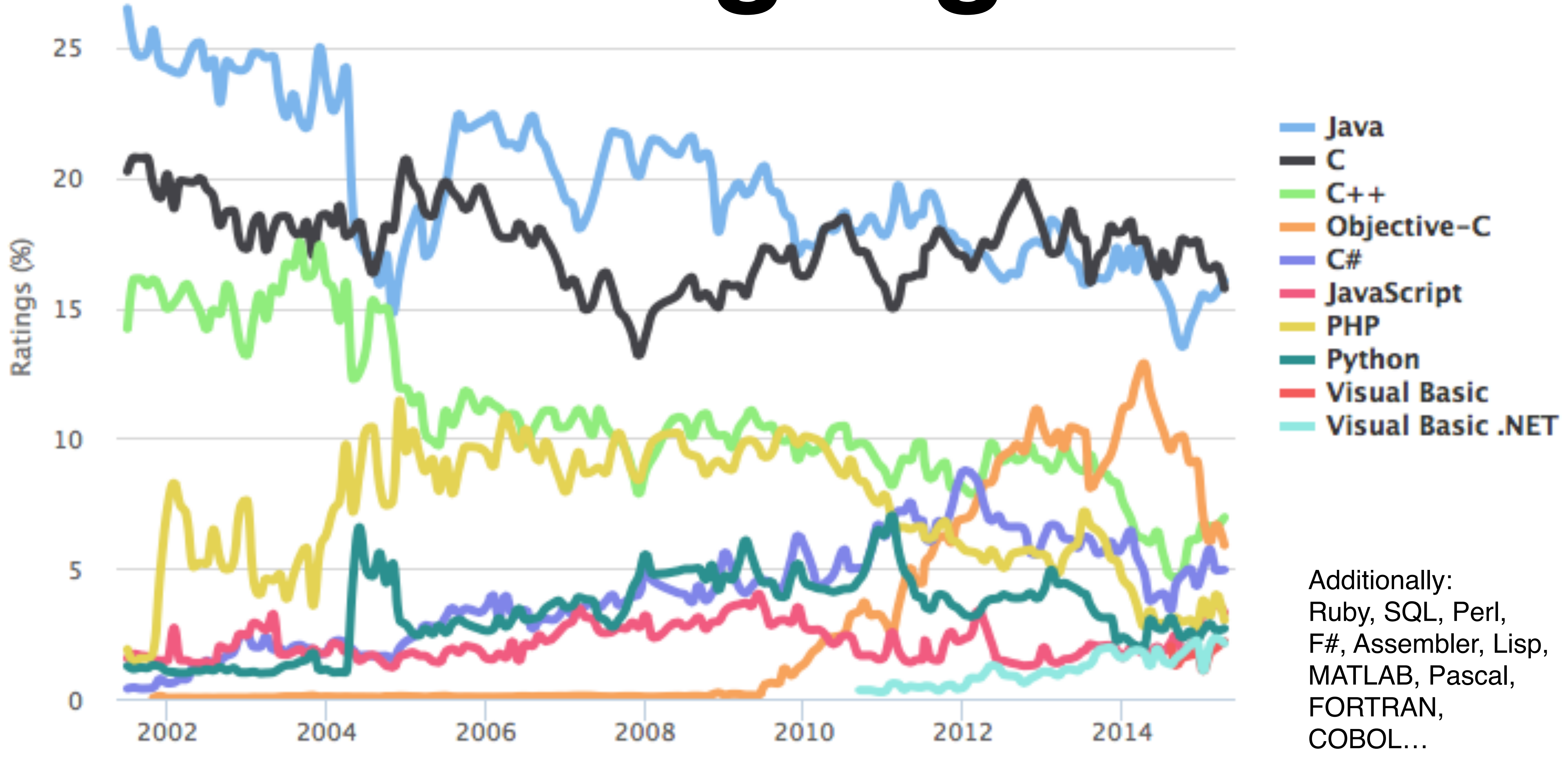
```
Speichern abgeschlossen.
Transfer complete
#
#
```

– Download on Course Web Page –

A Program

- Determines what the computer should do
- Written in a programming language
- Consists of *instructions*

Programming Languages

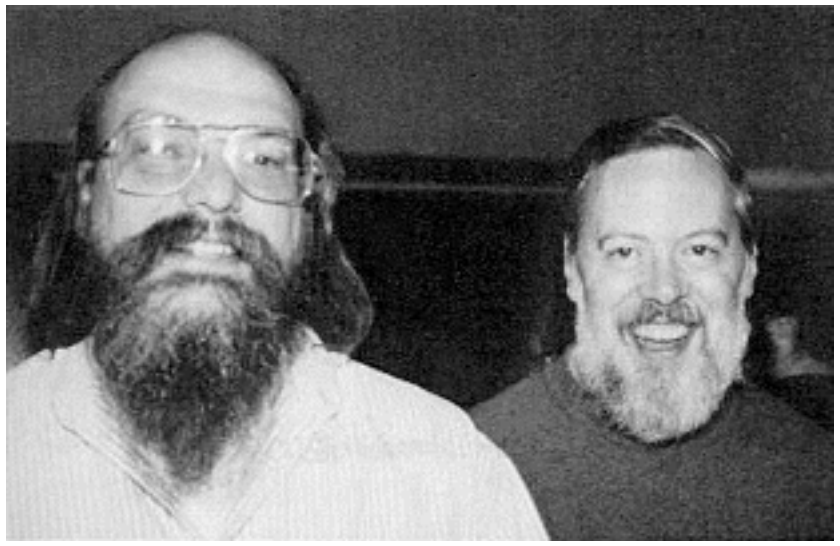


Additionally:
Ruby, SQL, Perl, F#, Assembler, Lisp, MATLAB, Pascal, FORTRAN, COBOL…

TIOBE Programming Community Index – April 2015

C

- Our programming language
- Developed in 1969–1973 in the UNIX Bell Labs (as a successor of B)
- One of the most influential programming languages



Ken Thompson and Dennis Ritchie,
Inventors of the C language

A Program in C

- consists of instructions*:*

```
digitalWrite(led, HIGH);
```

- *which can be assembled into functions*:

```
void setup() {
   pinMode(led, OUTPUT);
}
```

- *Comments explain the purpose*:

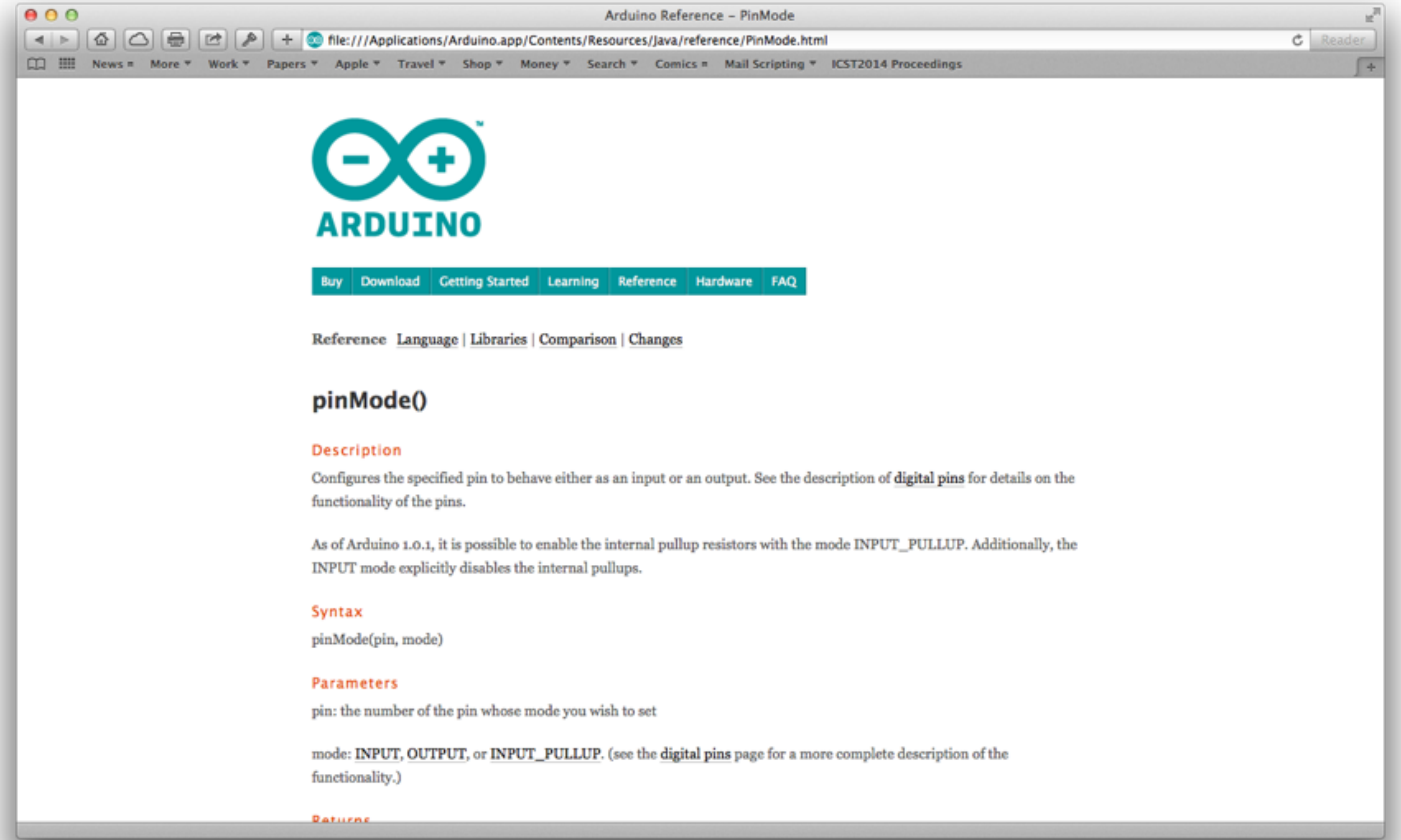
```
delay(1000);   // Wait one second
```

Instructions

- First we consider *function calls.*
- The Arduino Platform provides thousands of predefined *functions*.
- Each function provides a *service*.

`pinMode()`	Configure pin as input/output
`digitalWrite()`	Write out data digitally
`delay()`	Wait

All Functions


Arduino Reference – PinMode
file:///Applications/Arduino.app/Contents/Resources/Java/reference/PinMode.html
News More Work Papers Apple Travel Shop Money Search Comics Mail Scripting ICST2014 Proceedings
ARDUINO
Buy Download Getting Started Learning Reference Hardware FAQ

Reference Language | Libraries | Comparison | Changes

pinMode()

Description

Configures the specified pin to behave either as an input or an output. See the description of digital pins for details on the functionality of the pins.

As of Arduino 1.0.1, it is possible to enable the internal pullup resistors with the mode INPUT_PULLUP. Additionally, the INPUT mode explicitly disables the internal pullups.

Syntax

pinMode(pin, mode)

Parameters

pin: the number of the pin whose mode you wish to set

mode: INPUT, OUTPUT, or INPUT_PULLUP. (see the digital pins page for a more complete description of the functionality.)

Returns

In Arduino Menu: Help → Reference

Function Calls

- Most functions have parameters that determine their mode of operation

```
digitalWrite(pin_number, value)
```

- A value (argument) must be provided for each parameter

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

function name

argument for *value*

argument for *pin_number*

Predefined Functions

- Every Arduino program *(Sketch)* starts with two functions:

`setup()`	Called once at the beginning
`loop()`	Called repeatedly

- The content of these two functions determines what happens in the

Defining Functions

- A function like setup() and loop() is defined as a sequence of instructions surrounded by {…}

```
void setup() {
    Instruction 1;
    Instruction 2;
    …
}
```

- Every instruction ends with a “;”

Comments

- Comments serve to *make programs easier for humans to understand*
- Either // … until end of line or /* … */

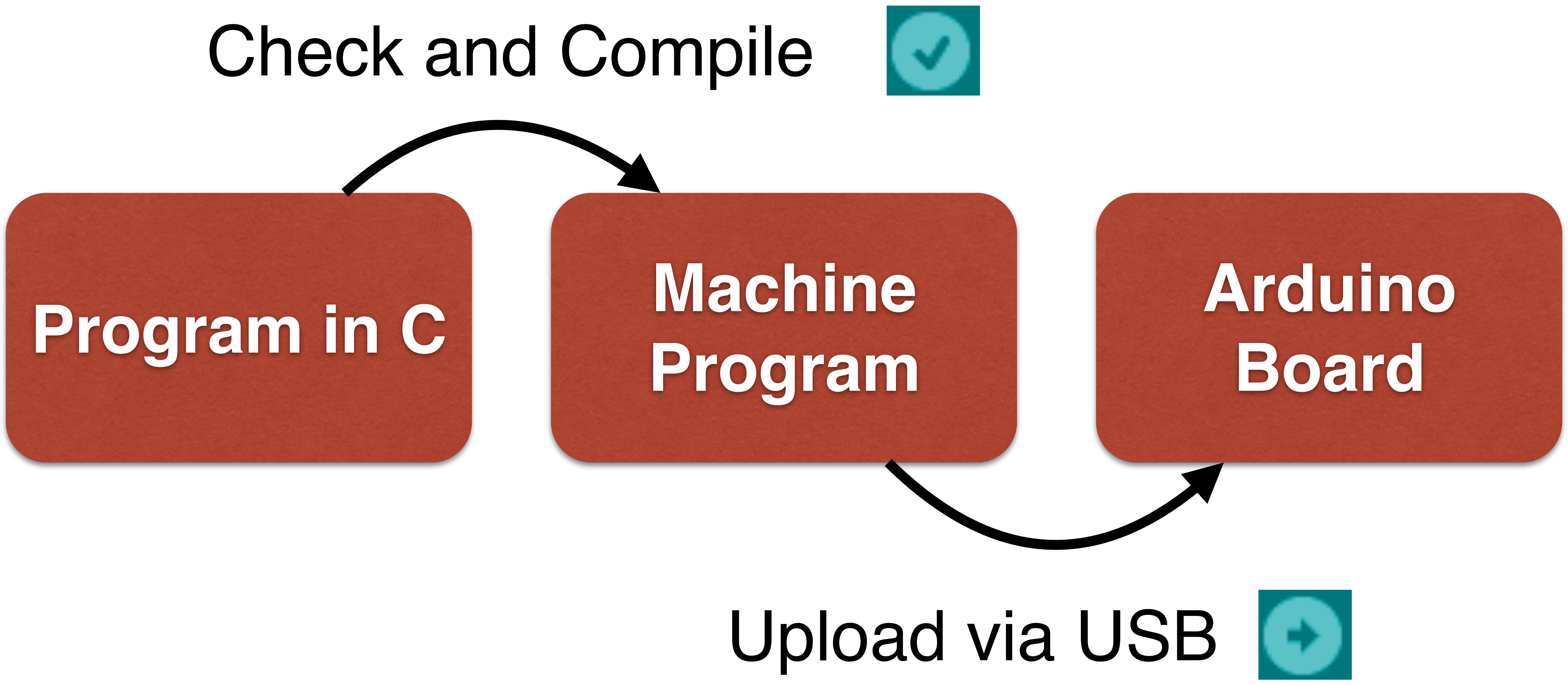
```
/* Pin 13 has an LED connected
on most Arduino boards. */

// setup() runs once when you press reset
```

- The computer *ignores all comments*

Example: Blink 3x

```
void setup() {
    // configure PIN 13 (built-in LED) as output
    pinMode(13, OUTPUT);

    // turn the LED on (HIGH is the voltage level)
    digitalWrite(13, HIGH);

    // wait for a second
    delay(1000);

    // turn the LED off by making the voltage LOW
    digitalWrite(13, LOW);

    // wait for a second
    delay(1000);

    // turn the LED on
    …
}
```

From Program to Processor



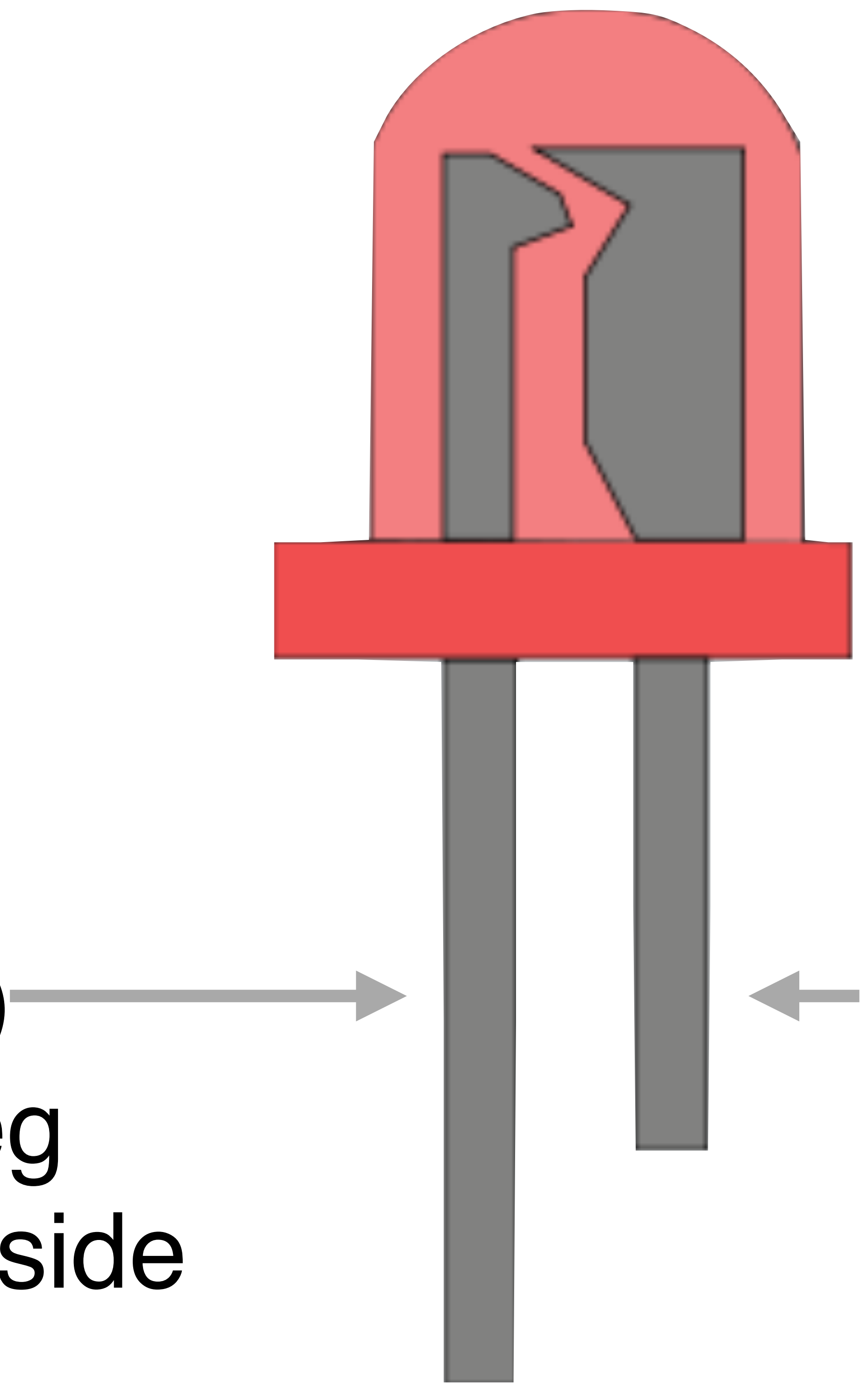
Repetition

- After the setup() function has been called, the loop() function gets called repeatedly.

Example: Blink forever

```
void setup() {
    // configure PIN 13 (built-in LED) as output
    pinMode(13, OUTPUT);
}

void loop() {
    // turn the LED on (HIGH is the voltage level)
    digitalWrite(13, HIGH);

    // wait for a second
    delay(1000);

    // turn the LED off by making the voltage LOW
    digitalWrite(13, LOW);

    // wait for a second
    delay(1000);
}
```

A LED



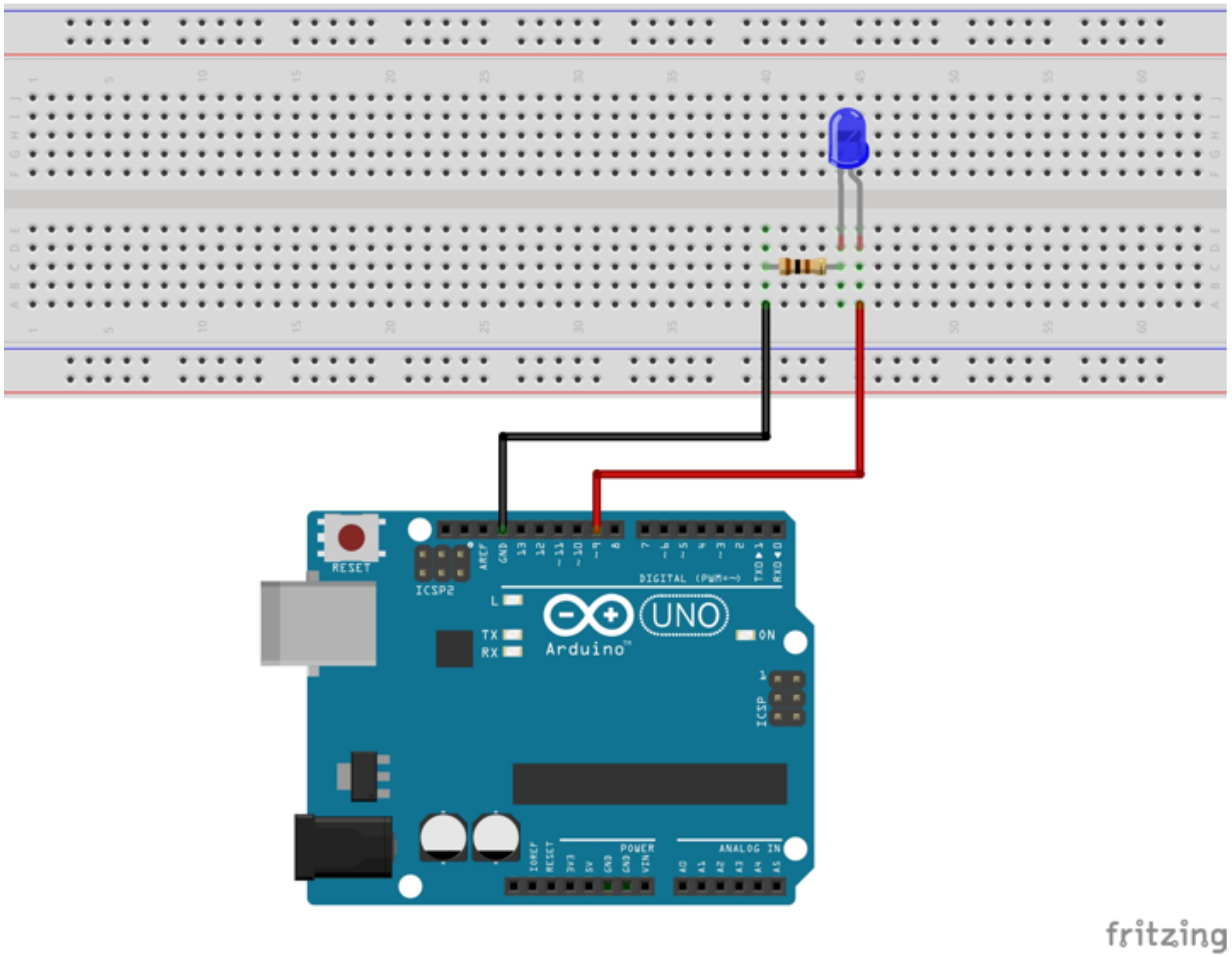
Anode (+)
- long leg
- round side

Cathode (–)
- short leg
- flat side

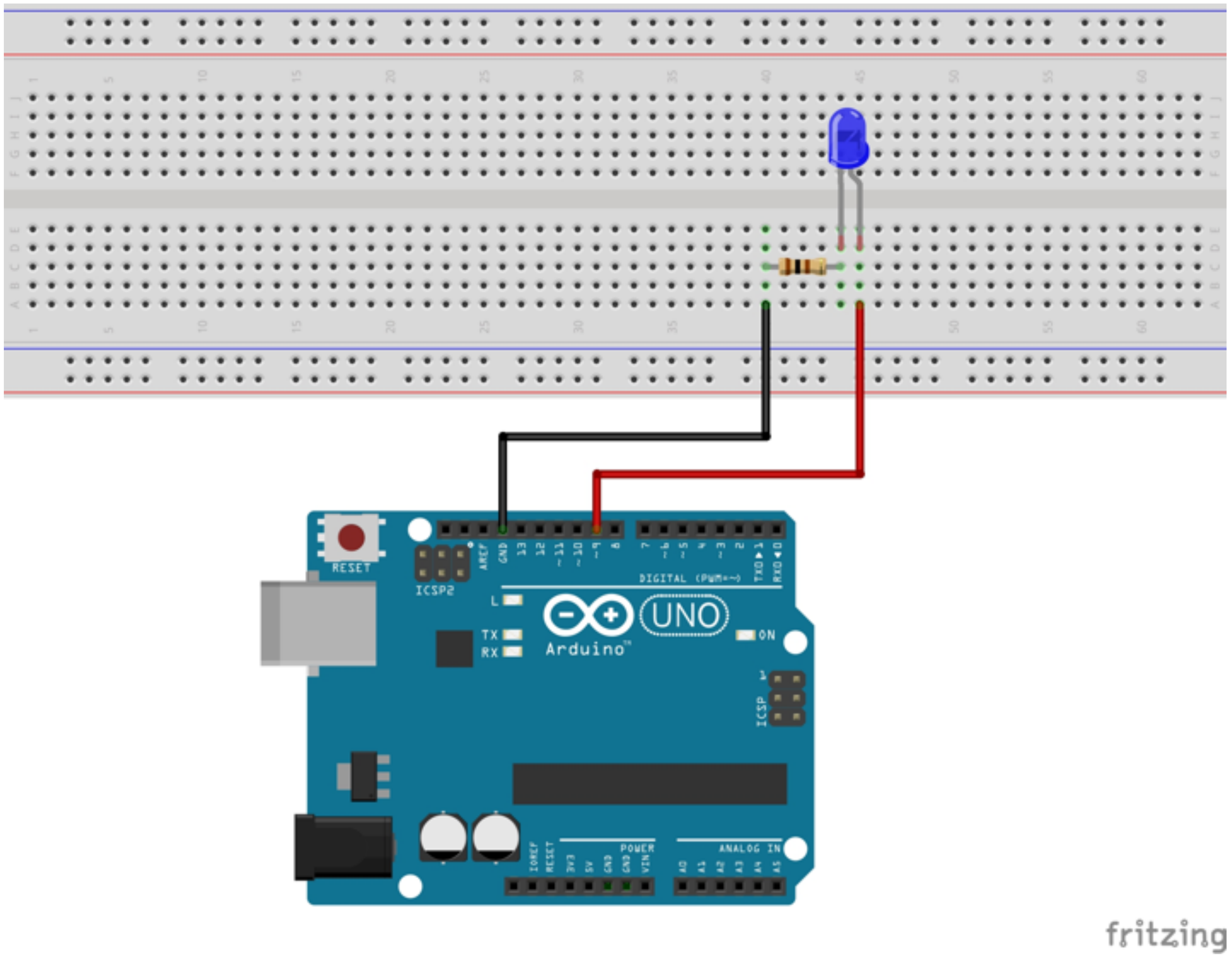
Connecting a LED

- To connect an LED to 5V, a resistor is needed*:*
    - *200Ω* for red, yellow
    - *100Ω* for white, green, blue, IR
- Cathode (–, short leg) to GND, Anode (+, long leg) to port

Connecting a LED



The Correct Port



The Correct Port

- To connect the LED to a different port (e.g. port 9), the port number must be changed in the entire program
- In a large program this would become problematic very quickly
- Solution: *Variables*

Variables

- *Variables are used to store values*.
- The instruction

  ```
  int led = 13;
  ```

  introduces `led` as a variable holding the value 13.
- After this instruction, the value can be accessed via the name `led`.

Types

- The type of a variable determines which values it can hold
- `int` – integer numbers
- Further types: `float`, `char`, `void`

Symbolic Blinking

```
// Pin 13 has an LED connected on most
// Arduino boards.  Give it a name:
int led = 13;

void setup() {
   pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```

Blinking Faster

```
// Pin 13 has an LED connected on most
// Arduino boards.  Give it a name:
int led = 13;

// Blinking delay (in ms)
int blink_delay = 250;

void setup() {
   pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(blink_delay);
    digitalWrite(led, LOW);
    delay(blink_delay);
}
```

Alternating Blinking

```
int led_red   = 12;
int led_green = 13;

void setup() {
   pinMode(led_red, OUTPUT);
   pinMode(led_green, OUTPUT);
}

void loop() {
    digitalWrite(led_red, HIGH);
    digitalWrite(led_green, LOW);
    …
}
```

Identifiers

- All names for variables and functions *(identifiers) consist of* a–z, A-Z, 0–9 and _ (underscore)
- Identifiers must not begin with 0–9
- An identifier can only be assigned once in a sketch.

Identifiers

- `delay`, `Delay` and `DELAY` are different identifiers
- Convention:
  - `Delay` – a Class
  - `DELAY` – a *Macro*
  - `_delay` – *intern*

} we don’t do this!

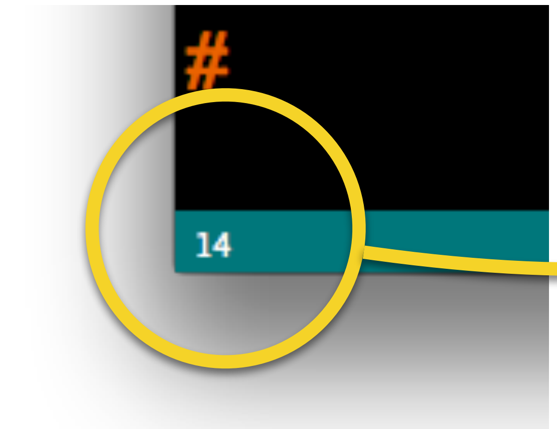
In Case of Errors

- On errors: *error message*

Line Column

```
Blink.ino:7:5: error:
    redefinition of 'int on_delay'
```

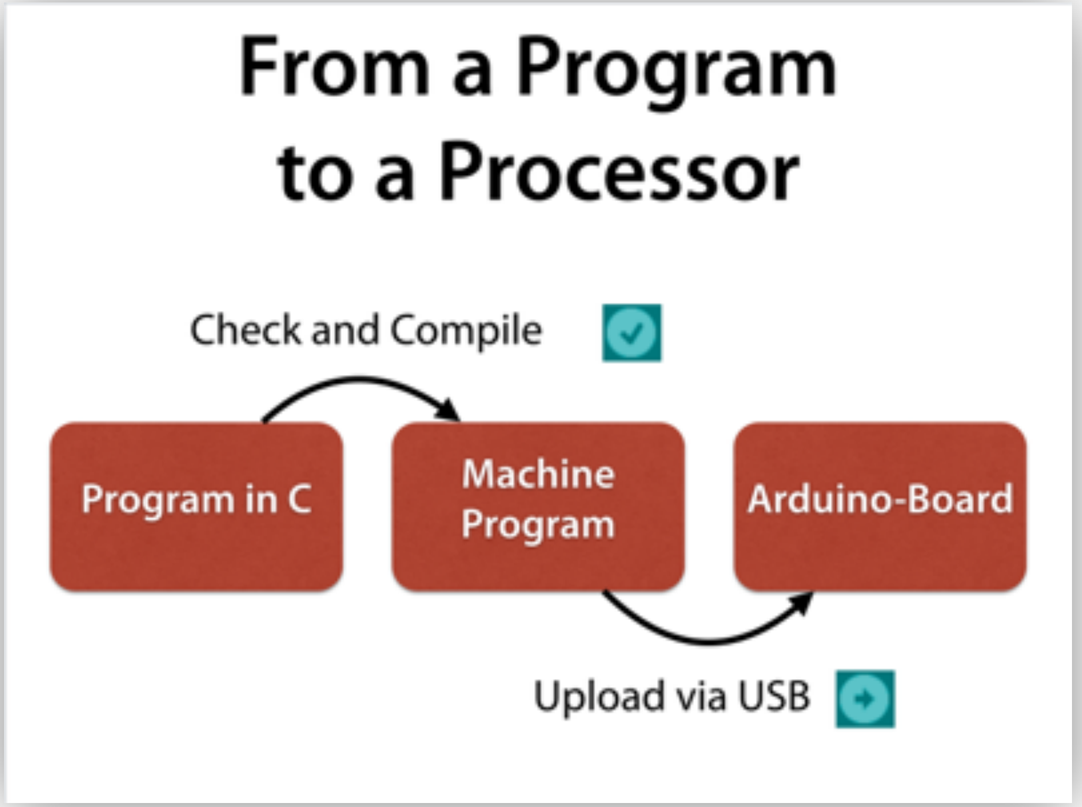
error message



Current line

Preview

- Morse-Code
- Functions with parameters
- Control structures

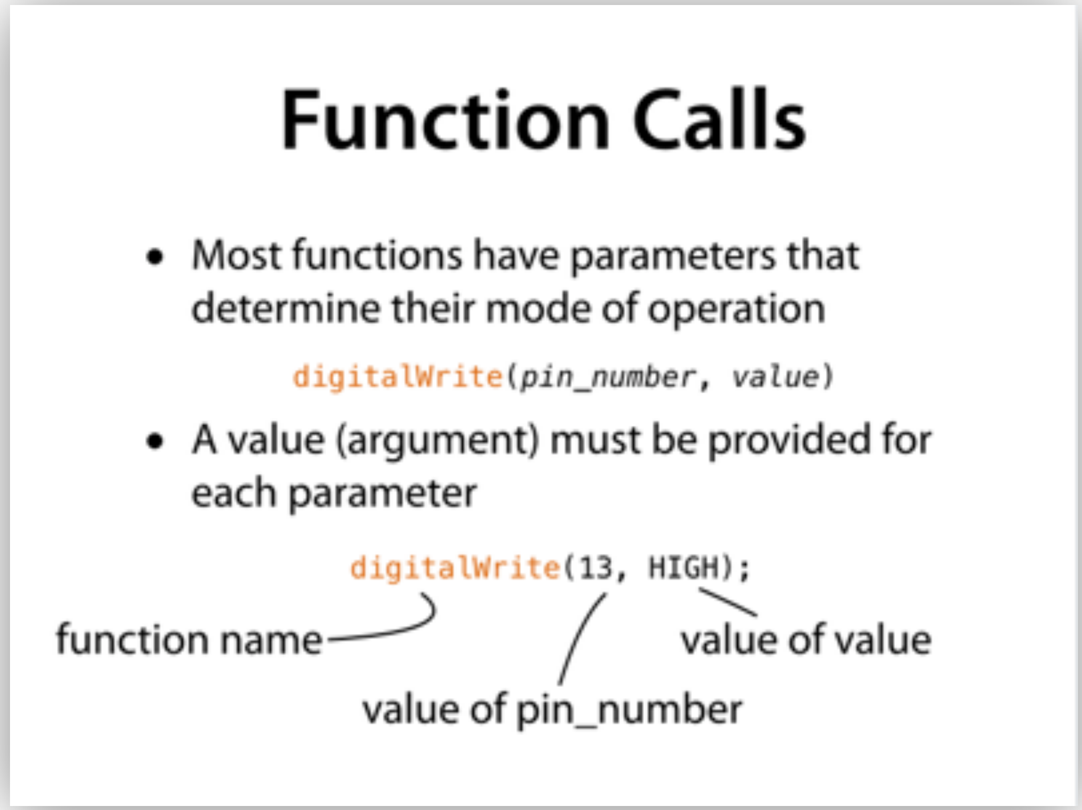


From a Program to a Processor

Check and Compile

Program in C → Machine Program → Arduino-Board

Upload via USB



Function Calls

- Most functions have parameters that determine their mode of operation

```
digitalWrite(pin_number, value)
```

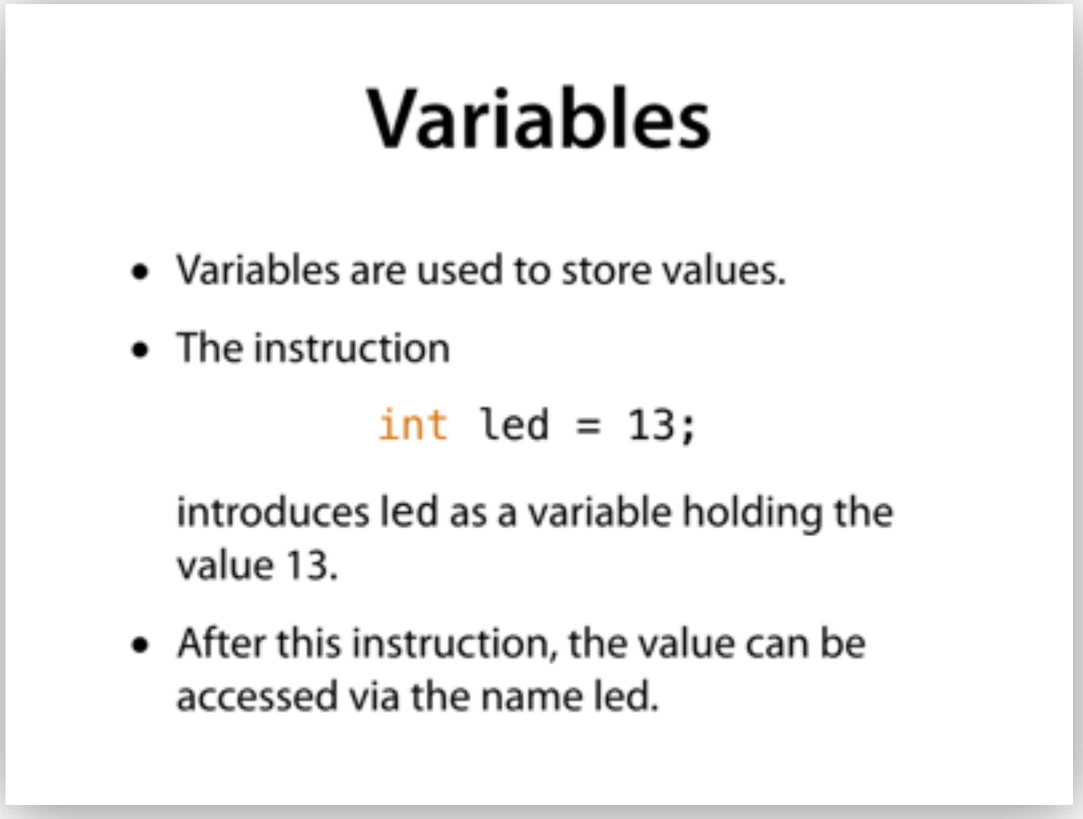
- A value (argument) must be provided for each parameter

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

function name

value of value

value of pin_number



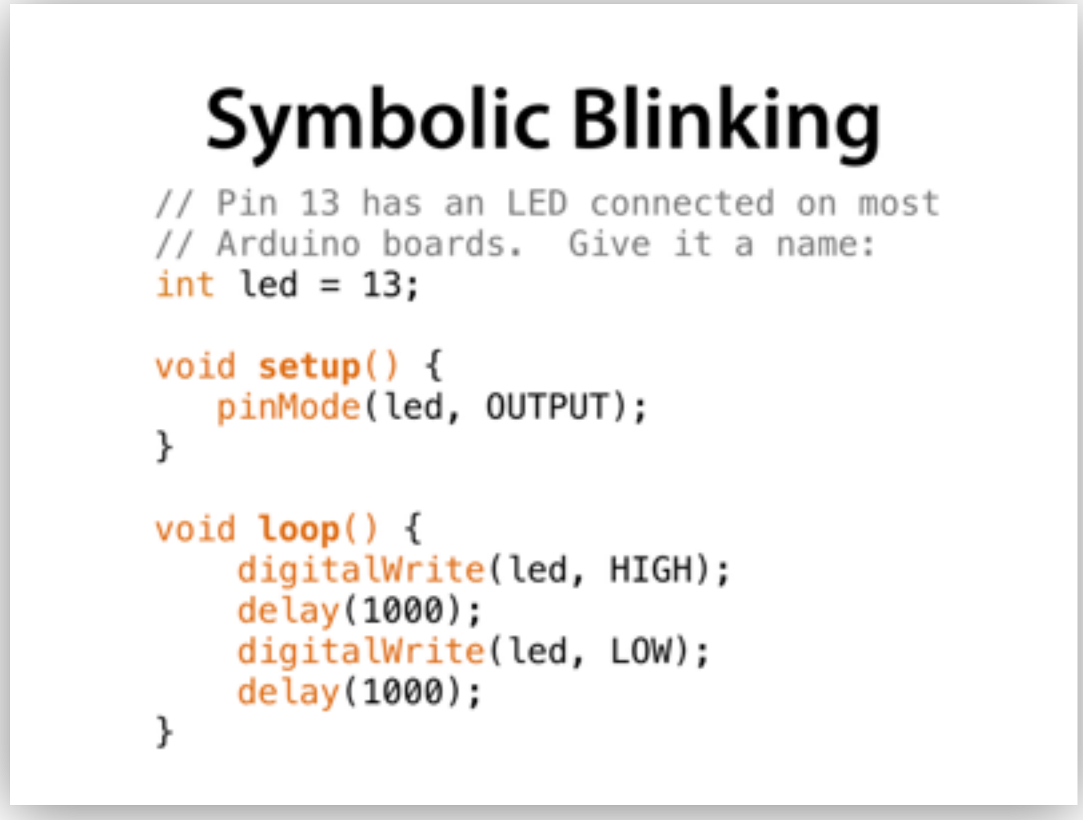
Variables

- Variables are used to store values.
- The instruction

```
int led = 13;
```

  introduces led as a variable holding the value 13.

- After this instruction, the value can be accessed via the name led.



Symbolic Blinking

```
// Pin 13 has an LED connected on most
// Arduino boards.  Give it a name:
int led = 13;

void setup() {
   pinMode(led, OUTPUT);
}

void loop() {
    digitalWrite(led, HIGH);
    delay(1000);
    digitalWrite(led, LOW);
    delay(1000);
}
```