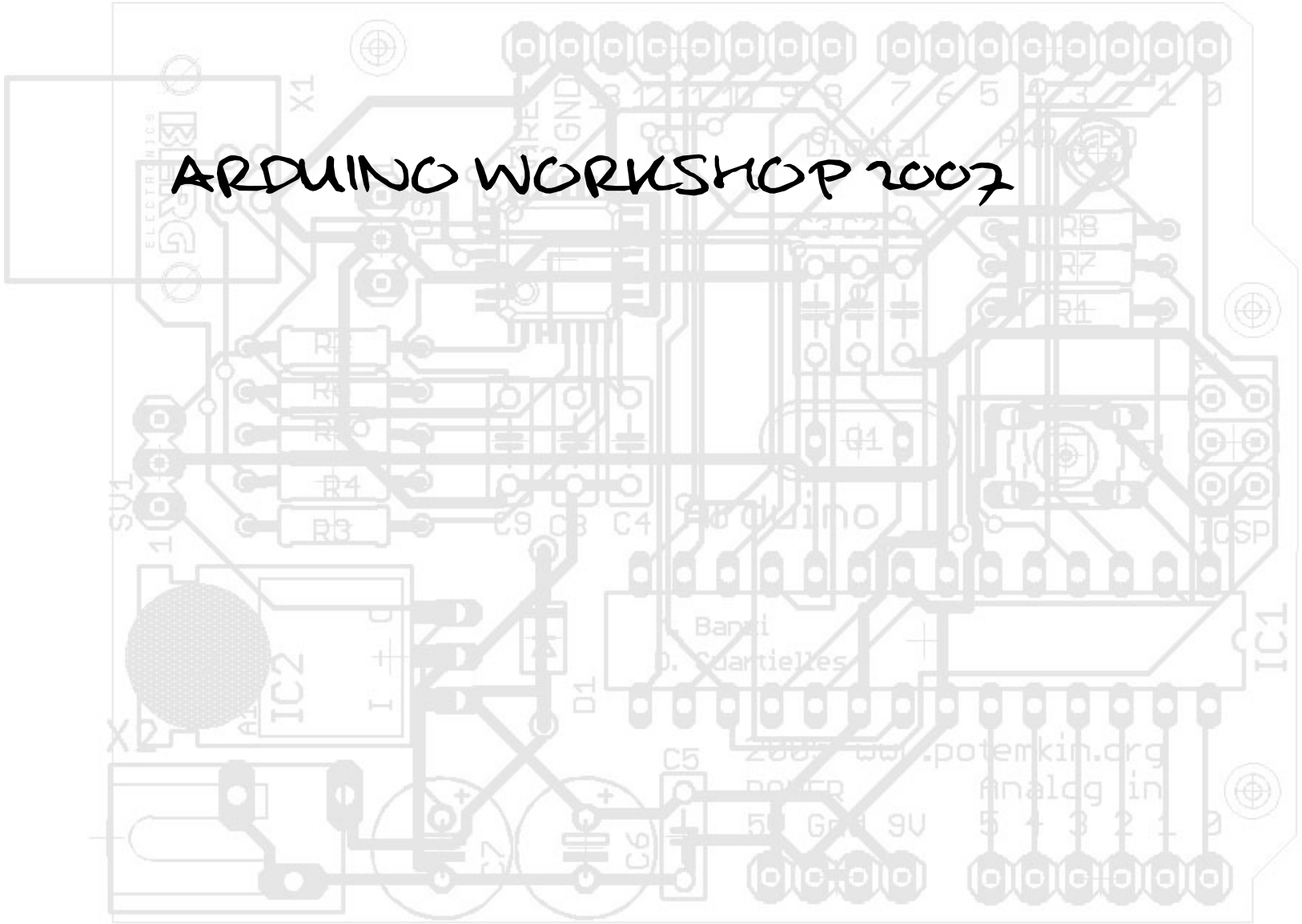


ARDUINO WORKSHOP 2007



PRESENTATION

WHO ARE WE?

- **Markus Appelbäck**
 - Interaction Design program at Malmö University
 - Mobile networks and services
 - Mecatronics lab at K3, Malmö University
 - Developer, Arduino community
 - Music, programming, sound and image translations, interactive installations, haptics and tactile feedback...
- **Marcus Hannerstig**
 - Interaction Design program at Malmö University
 - Interactive installations
 - Mecatronics lab at K3, Malmö University
- **Marcus Ericsson**
 - Interaction Design program at Malmö University
 - Light installations
 - Snowboard
 - Mecatronics lab at K3, Malmö University
- **David Sjunnesson**
 - Interaction Design program at Malmö University
 - Light installations
 - Mecatronics lab at K3, Malmö University

AGENDA

DAY 1

- INTRODUCTION
- ARDUINO IDE
- ARDUINO PROGRAM STRUCTURE
 - PRACTICAL EXAMPLE (LED BLINK)
- PROGRAMMING BASICS
- DIGITAL OUTPUT
- BASIC DIGITAL SENSORS
 - PRACTICAL EXAMPLE (READ DIGITAL INPUTS)
- DIGITAL INPUT
- PROGRAMMING CONTROL STATEMENTS
- MULTIPLE OUTPUTS
 - PRACTICAL EXAMPLE (SEQUENCING LEDS)
- PROGRAMMING | LOOPS
 - PRACTICAL EXAMPLE (SEQUENCING LEDS)

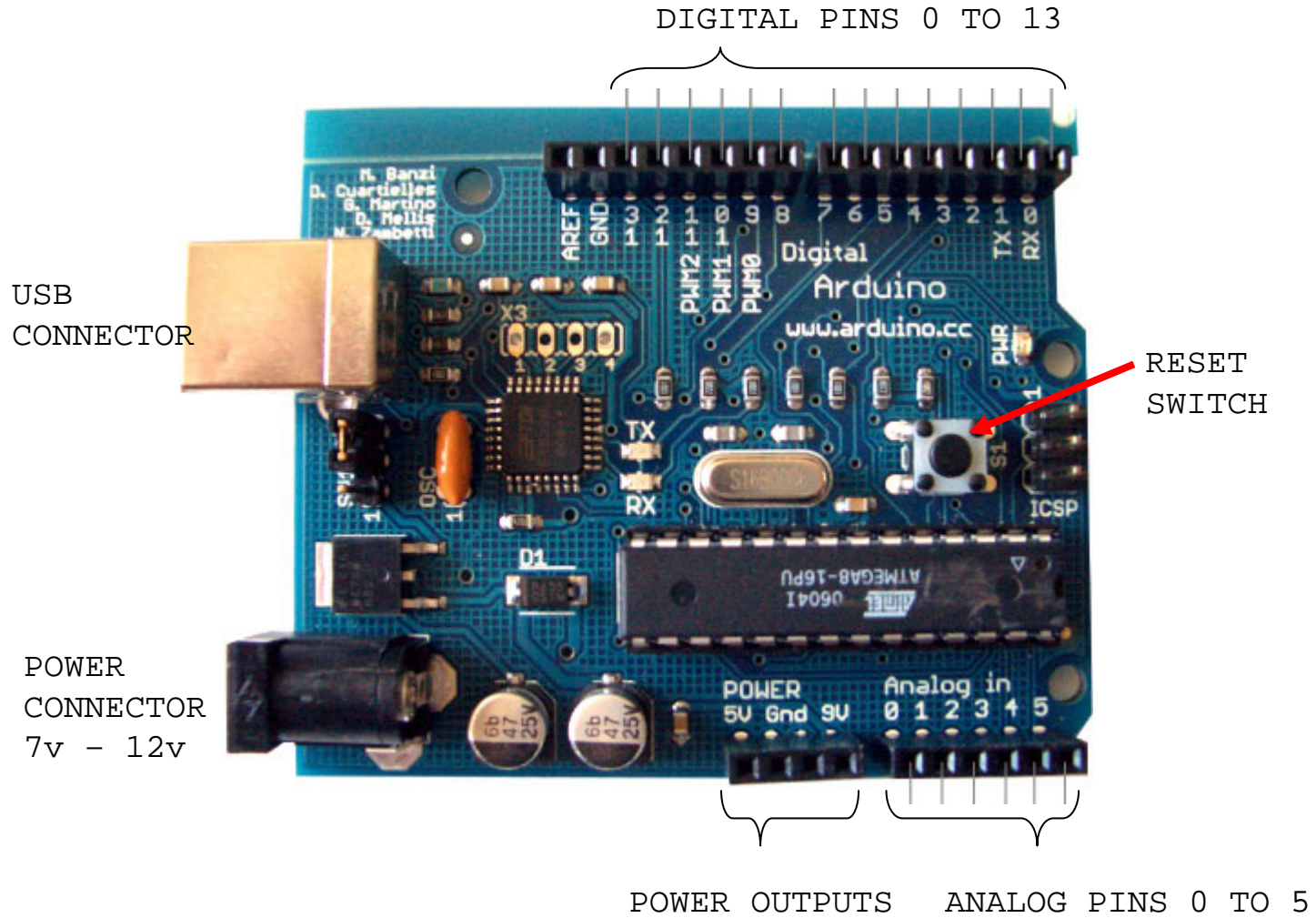
INTRODUCTION

THE PLAN!

- **SHORT INTRODUCTIONS**
 - ELECTRONICS
 - PROGRAMMING
- **HANDS-ON EXERCISES**
 - TUTORING
- **DESCRIBING THE EXEMPLES**
 - HOW IT WORKS
 - HOW WE CAN ALTER IT
- **WORK OUR SELVES THROUGH A NUMBER OF SENSORS**

INTRODUCTION

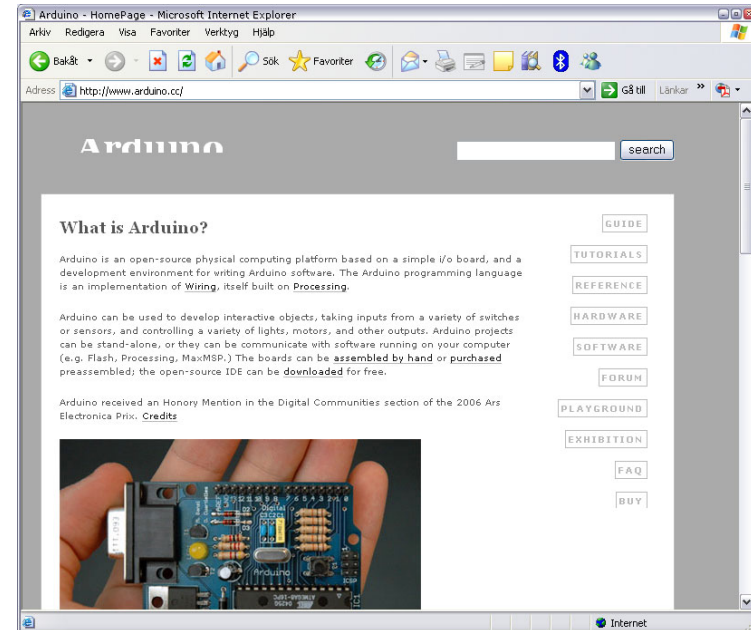
BASIC ARDUINO KNOWLEDGE



ARDUINO IDE

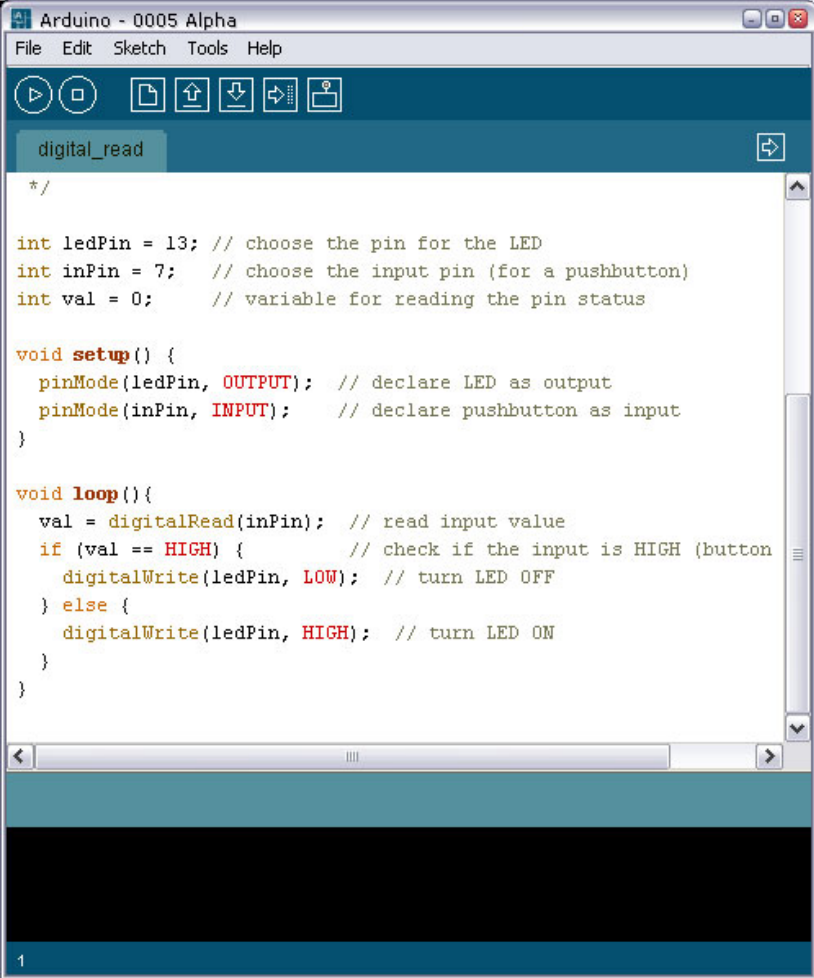
DOWNLOAD AND INSTALL

- **DOWNLOAD FROM:**
<http://www.arduino.cc>
 - **THE FULL VERSION NOT EXPERT**
- **UNZIP**
- **CREATE SHORTCUTS**
- **INSTALL DRIVER**



ARDUINO IDE

- "INTEGRATED DEVELOPMENT ENVIRONMENT"
- USED FOR PROGRAMMING THE ARDUINO
- 3-STEP
 - WRITE CODE
 - COMPILE
 - UPLOAD
- <http://www.arduino.cc>
- PROCESSING



```
Arduino - 0005 Alpha
File Edit Sketch Tools Help

digital_read

*/

int ledPin = 13; // choose the pin for the LED
int inPin = 7;   // choose the input pin (for a pushbutton)
int val = 0;    // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(inPin, INPUT);  // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin); // read input value
  if (val == HIGH) {       // check if the input is HIGH (button
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON
  }
}
```

ARDUINO IDE

FILE MENU
NEW, SAVE,
OPEN,
EXAMPLES...

EDIT MENU
CUT, PASTE,
FIND...

TOOLS MENU
COM-PORT
SETTINGS,
TOOLS,
SETTINGS...

```

Arduino - 0005 Alpha
File Edit Sketch Tools Help

digital_read

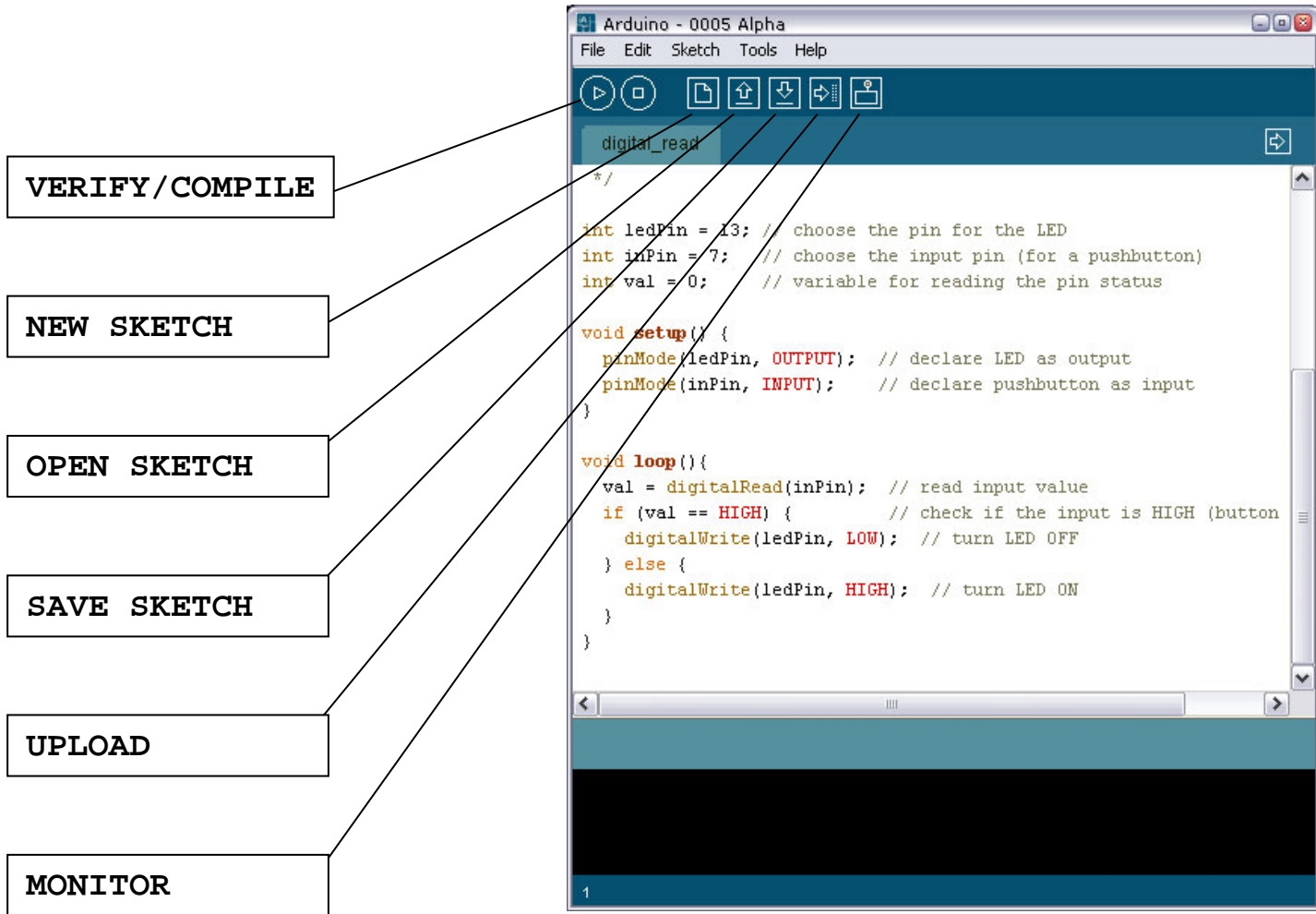
*/

int ledPin = 13; // choose the pin for the LED
int inPin = 7;  // choose the input pin (for a pushbutton)
int val = 0;    // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(inPin, INPUT);  // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin); // read input value
  if (val == HIGH) {       // check if the input is HIGH (button
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON
  }
}
  
```


ARDUINO IDE



ARDUINO IDE

PROGRAM STRUCTURE, VARIABLES

```

Arduino - 0005 Alpha
File Edit Sketch Tools Help

digital_read

*/
                                variabeldeklarationer

int ledPin = 13; // choose the pin for the LED
int inPin = 7;  // choose the input pin (for a pushbutton)
int val = 0;    // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(inPin, INPUT);  // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin); // read input value
  if (val == HIGH) {        // check if the input is HIGH (button
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON
  }
}
  
```

VARIABLES ARE NAMED CONTAINERS WHERE WE CAN STORE VALUES.

ANY NAME COULD BE GIVEN A VARIABLE (ALPHA NUMERIC CHARACTERS)

THE VALUE IN A VARIABLE CAN BE USED AND CHANGED DYNAMICALLY AS MANY TIMES AS WE LIKE

VARIABLES ARE GOOD FOR STORING VALUES THAT WE KNOW WILL BE USED MORE THAN ONCE IN OUR PROGRAM

ARDUINO IDE

PROGRAM STRUCTURE, SETUP

```

Arduino - 0005 Alpha
File Edit Sketch Tools Help

digital_read

*/

int ledPin = 13; // choose the pin for the LED
int inPin = 7;  // choose the input pin (for a pushbutton)
int val = 0;    // variable for reading the pin status

void setup() {                                     setup-metoden
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(inPin, INPUT);  // declare pushbutton as input
}

void loop(){
  val = digitalRead(inPin); // read input value
  if (val == HIGH) {        // check if the input is HIGH (button
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON
  }
}
  
```

THE SETUP METHOD IS USED FOR INITIALIZING THE ARDUINO BEFORE EXECUTING THE PROGRAM

CONTAINS CALLS TO UNDERLAYING METHODS TO SET APPROPRIATE START UP OPTIONS

DEFINING IF PORTS ARE USED AS INPUTS OR OUTPUTS, IF WE ARE GOING TO USE SERIAL COMMUNICATION AND SO ON

ARDUINO IDE

PROGRAM STRUCTURE, LOOP

```

Arduino - 0005 Alpha
File Edit Sketch Tools Help

digital_read

*/

int ledPin = 13; // choose the pin for the LED
int inPin = 7; // choose the input pin (for a pushbutton)
int val = 0; // variable for reading the pin status

void setup() {
  pinMode(ledPin, OUTPUT); // declare LED as output
  pinMode(inPin, INPUT); // declare pushbutton as input
}

void loop() {
  val = digitalRead(inPin); // read input value
  if (val == HIGH) { // check if the input is HIGH (button
    digitalWrite(ledPin, LOW); // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH); // turn LED ON
  }
}
  
```

THE LOOP IS WHERE ALL THE ACTION IS!!

THIS IS THE PROGRAM RUNNING ON THE ARDUINO AFTER IT HAS BEEN UPLOADED.

EXECUTES FROM TOP TO BOTTOM AND THEN STARTS OVER.

PRACTICAL EXAMPLE

LETS GET TO WORK!!

- CONNECT THE ARDUINO TO YOUR COMPUTER
- CHECK THE COM-PORT
 - INSTALLED CORRECTLY?
 - COM-PORT NUMBER?
- SET THE CORRECT COMPORT IN THE IDE
- OPEN THE "led_blink" EXAMPLE UNDER SKETCHBOOK -
EXAMPLES
- COMPILE & UPLOAD
- A BUILT IN LED AT PORT 13 IS NOW SUPPOSED TO BLINK
IN ONE SECOND INTERVALS
- EXAMAIN THE CODE AND SEE IF YOU CAN ALTER THE
INTERVAL FOR THE LED.

PRACTICAL EXAMPLE

INSTALLING THE DRIVER

- LOCATE THE FOLDER YOU UNZIPPED THE IDE TO..
- IN THAT FOLDER YOU WILL FIND A FOLDER CALLED DRIVERS
- IN THAT FOLDER YOU WILL FIND A ZIP-FILE CALLED "FTDI USB Drivers.zip"
- UNZIP IT!
- CONNECT THE ARDUINO..
- A MESSAGE SEARCHING FOR A DRIVER APPEARS..
- CHOOSE "SPECIFY LOCATION"
- LOCATE THE UNZIPPED FOLDER AND PRESS OK
- IF EVERYTHING WENT OK A NEW DEVICE IS FOUND.
- SAME PROCEDURE... "SPECIFY LOCATION"
- THE COM-PORT SHOULD NOW APPEAR IN THE ARDUINO IDE'S TOOLS MENU UNDER COM-PORT

PROGRAMMING

BASIC ARDUINO METHODS

- `pinMode(pin, MODE);`
 - SETS THE PORT ("*pin*") IN A SPECIFIC MODE, INPUT OR OUTPUT
 - USUALLY MADE IN THE SETUP METHOD, BUT ALLOWED IN LOOP METHOD
- `digitalWrite(pin, STATE);`
 - DIGITAL PINS HAVE TWO STATES, ON OR OFF!
WRITES A VALUE TO A DIGITAL PORT ("*pin*"). HIGH EQUALS ON, LOW EQUALS OFF.
- `delay(ms);`
 - DELAYS ANY FURTHER EXECUTION FOR A SPECIFIED NUMBER OF MILLISECONDS ("*ms*").
 - PROGRAM HALTS UNTIL SPECIFIED TIME HAVE ELAPSED.

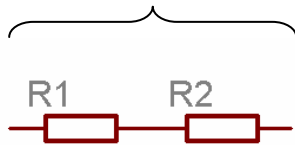
ELECTRONICS

CURRENT, VOLTAGE AND RESISTORS

- **CURRENT: THE NUMBER OF ELECTRONS**
- **VOLTAGE: THE PREASURE**
- **RESISTOR: RESTRICTING THE NUMBER OF ELECTRONS**

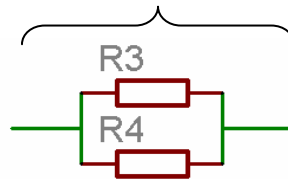
SERIES CIRCUIT

$$R_{tot} = R1 + R2$$

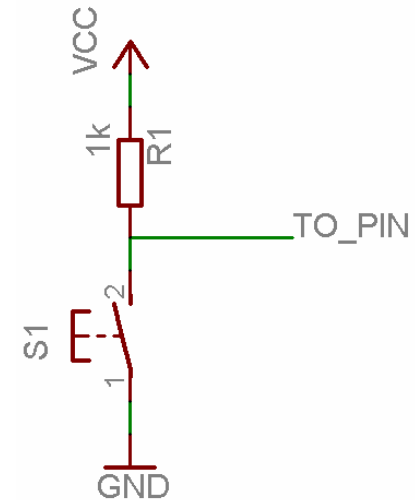


PARALLEL CIRCUIT

$$R_{tot} = (R3 * R4) / (R3 + R4)$$



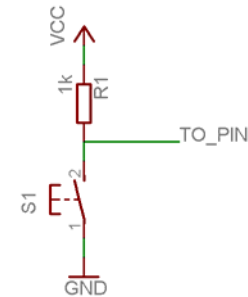
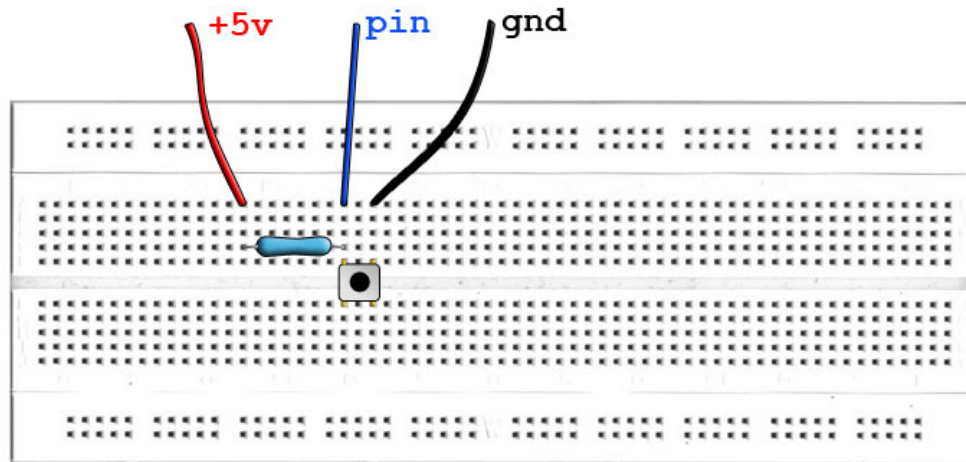
- **ELECTRONS TAKE THE EASIEST WAY!**
 - HAVE TO CHOOSE BETWEEN RESISTOR AND WIRE, IT CHOOSES WIRE...



PRACTICAL EXAMPLE

BUTTONS

- CONNECT THE PUSHBUTTON USING A BREADBOARD:



- OPEN THE "digital_read" EXAMPLE IN SKETCHBOOK - EXAMPLES
- COMPILE & UPLOAD

PROGRAMMING

BASIC ARDUINO METHODS

- `pinMode(pin, MODE);`
 - SETS THE PORT ("*pin*") IN A SPECIFIC MODE, INPUT OR OUTPUT
 - USUALLY MADE IN THE SETUP METHOD, BUT ALLOWED IN LOOP METHOD
- `digitalRead(pin);`
 - RETURNS THE STATE OF THE PORT, ON OR OFF!
RETURNS A VALUE REPRESENTING THE STATE (HIGH OR LOW).
- `if(condition) { }`
 - THIS STATEMENT EVALUATES THE "*condition*". THE CONDITION IS EITHER TRUE OR FALSE. IF ITS TRUE THE CODE WITHIN THE CURLY BRACES IS EXECUTED.
- `else { }`
 - CORRESPONDS TO AN IF-STATEMENT. IF PRECEDING IF-STATEMENT IS FALSE THE ELSE-STATEMENT WILL BE EXECUTED.

PROGRAMMING

VARIABLES

- **DEFINED TYPES**
- **THE TYPE DETERMINS THE SIZE OF THE VALUE THIS VARIABLE IS ABLE TO STORE**
- **THREE BASIC VARIABLE TYPES**
 - **INT - 16 BIT, (-32768 TO 32767)**
PORT NUMBERS, VALUES READ FROM PORTS, RESULTS FROM CALCULATIONS...
 - **CHAR - 8 BIT (1 BYTE), (0 TO 256)**
CHARACTERS (A-Z), SMALL VALUES, NOT GOOD FOR CALCULATIONS, SENDING DATA OVER SERIAL...
 - **LONG - 32 BIT, (-2147483648 TO 2147483647)**
RESULTS FROM CALCULATIONS, COUNTING MILLISECONDS...
- **USE APPROPRIATE TYPE, PROGRAM SIZE LIMITED!**
- **USE NAMES THAT ARE EASY TO REMEMBER**
 - NOT myVariable, myValue, port...
 - INSTEAD firstInput, nbrOfOutputs...

ELECTRONICS

THE LED

- "LIGHT EMITTING DIODE"
- EMITTING LIGHT WHEN CURRENT GOING THROUGH
- HAVE TO BE PLACED THE RIGHT WAY (+ AND -)



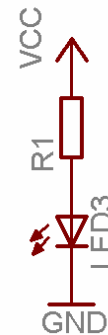
SHORT LEG TO GROUND



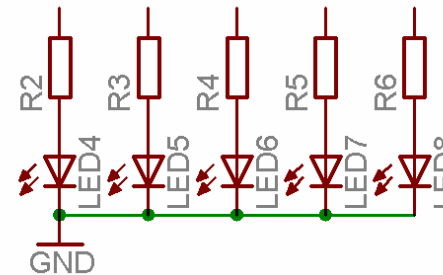
POLARIZED
SHORT LEG
TO GROUND



NOT
WORKING!



LIMIT
CURRENT!

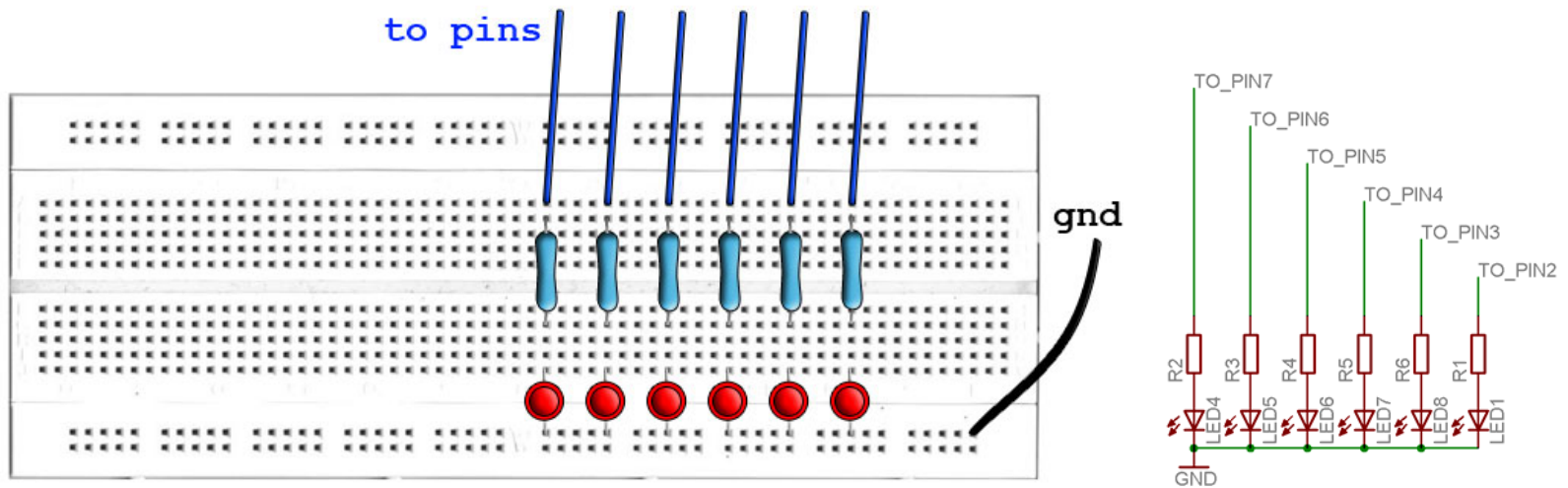


MULTIPLE LEDs,
COMMON GROUND

PRACTICAL EXAMPLE

KNIGHT RIDER

- CONNECT LEDS USING BREADBOARD



- "knight_rider1" FROM EXAMPLES
- "knight_rider2" FROM EXAMPLES

PROGRAMMING

ARRAYS

- COLLECTION OF VARIABLES
- ONE NAME
 - `int outPin1 = 2;`
 - `int outPin2 = 3;`
 - ...
 - `int outPins[6] = {2,3,4,5,6,7};`
- INDEX TO RETREIVE ONE SPECIFIC
 - `outPins[index];`
- ANY TYPE
 - `int, byte, long..`

PROGRAMMING

LOOPS

- `for(counter=0; counter<end; counter++) { }`

Counter
initialization

condition

Counter
increment
decrement

- IF CONDITION IS TRUE EXECUTE CODE ELSE END LOOP
- AT THE END CHECKS CONDITION AGAIN
- IF CONDITION IS TRUE EXECUTE CODE ELSE END LOOP
- AT THE END...

FIN