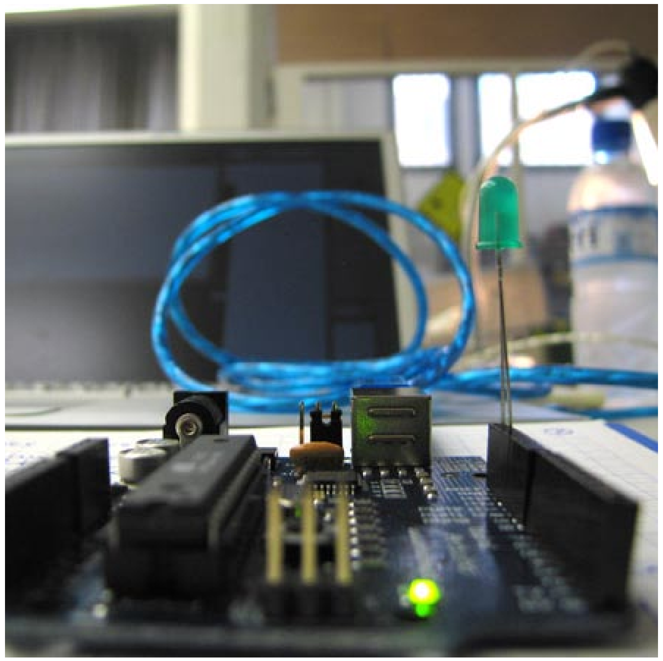
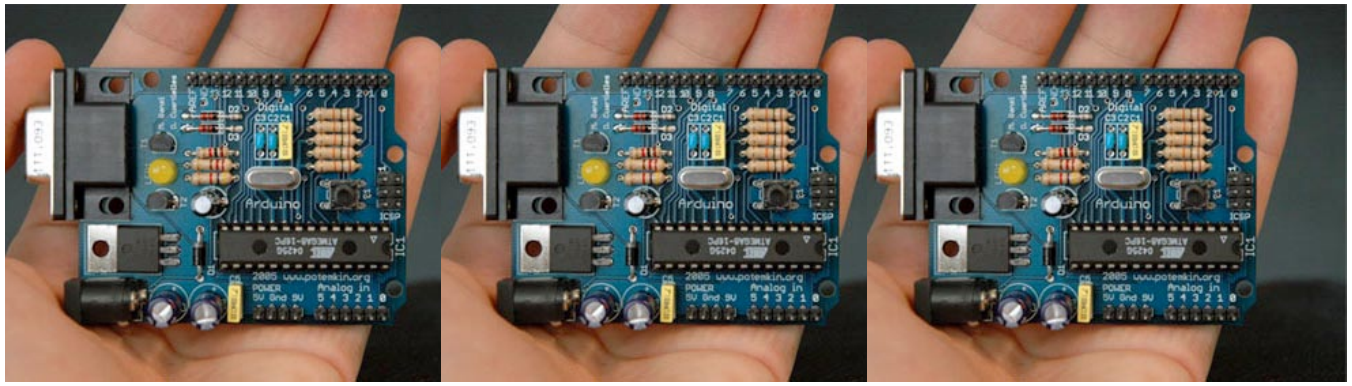
interaction design lab

Arduino

• Workshop

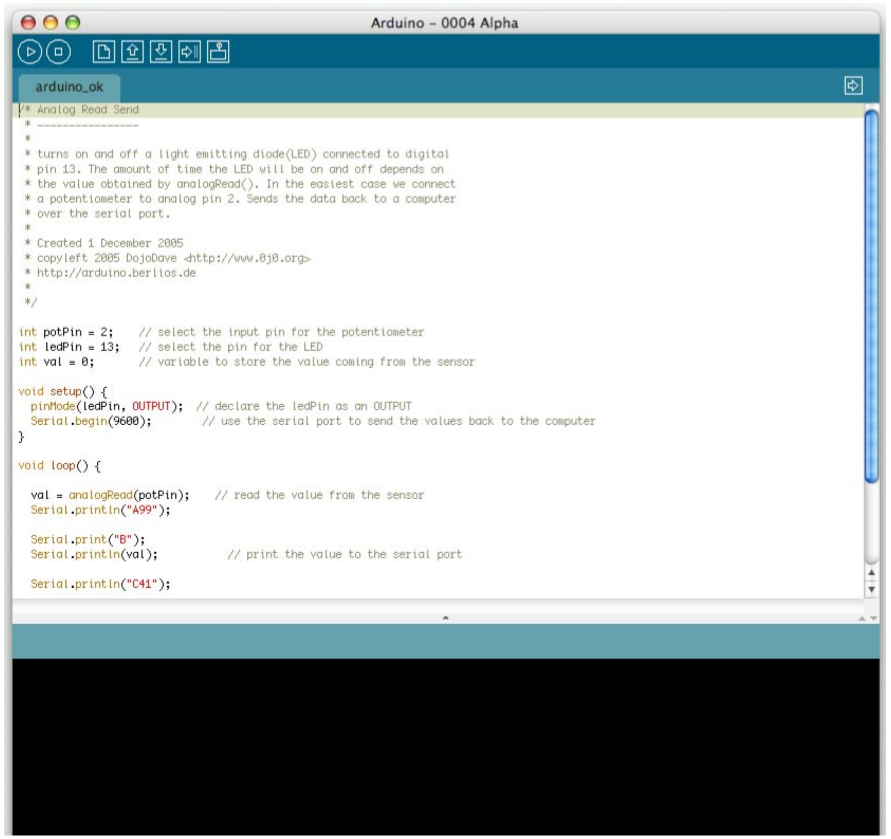


Index



01.0 introduction

- about Arduino
- when/how arduino started
- Arduino target user



```
/* Analog Read Send
 * ----------------
 *
 * turns on and off a light emitting diode(LED) connected to digital
 * pin 13. The amount of time the LED will be on and off depends on
 * the value obtained by analogRead(). In the easiest case we connect
 * a potentiometer to analog pin 2. Sends the data back to a computer
 * over the serial port.
 *
 * Created 1 December 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 */

int potPin = 2;    // select the input pin for the potentiometer
int ledPin = 13;   // select the pin for the LED
int val = 0;       // variable to store the value coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin as an OUTPUT
  Serial.begin(9600);        // use the serial port to send the values back to the computer
}

void loop() {

  val = analogRead(potPin);    // read the value from the sensor
  Serial.println("A99");

  Serial.print("B");
  Serial.println(val);           // print the value to the serial port

  Serial.println("C41");
```

about Arduino

Arduino is an open-source physical computing platform based on a simple i/o board and a IDE integrated development environment that implements the Processing/Wiring language.

Arduino can be used to develop stand-alone interactive objects and installation or can be connected to software on your computer.

when/how Arduino started

The first Arduino board was produced on January 2005.

The project is the result of the collaboration of many people but in the beginning the need of such a board was thought by David Cuartielles and Massimo Banzi. They are both teacher of physical computing and were used to built different boards for each course to teach to their students how to create interactive prototype.

They met at Interaction Design Institute Ivrea and agreed on the spot on many ideas and on the need to built a standar cheep board as a teaching tool.

The rest of the team is composed by David Mellis, who is responsible for building the software engine, Tom Igoe and Gianluca Martino.

Arduino target user
artists and designer

One relevant feature of Arduino is the limited range of programming skills required to work with it.

Inside the Arduino software there is an engine that translates the high level, more "user friendly", programme into something understandable by the micro-processor.

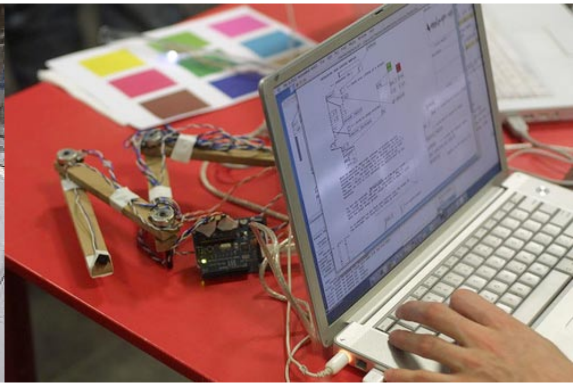
As a matter of fact the Integrated Development Environment uses a language similar to Java while the board is programmed in C, a language that requires more sophisticated programming skills.

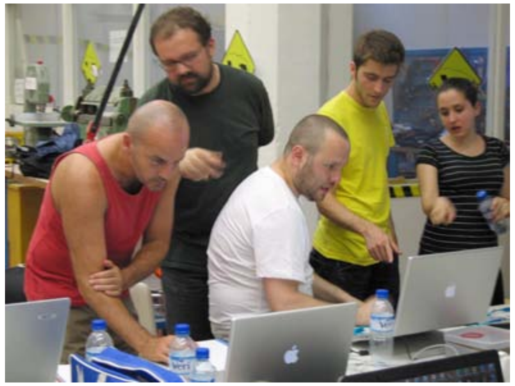
This enables people like artists and designers to quickly create prototypes to communicate their ideas.

Building complicated prototypes requires more time and more skills but this can all be learned as long as the initial phase is facilitated through a workshop.

The low cost of the Arduino board is an important quality for the Team.

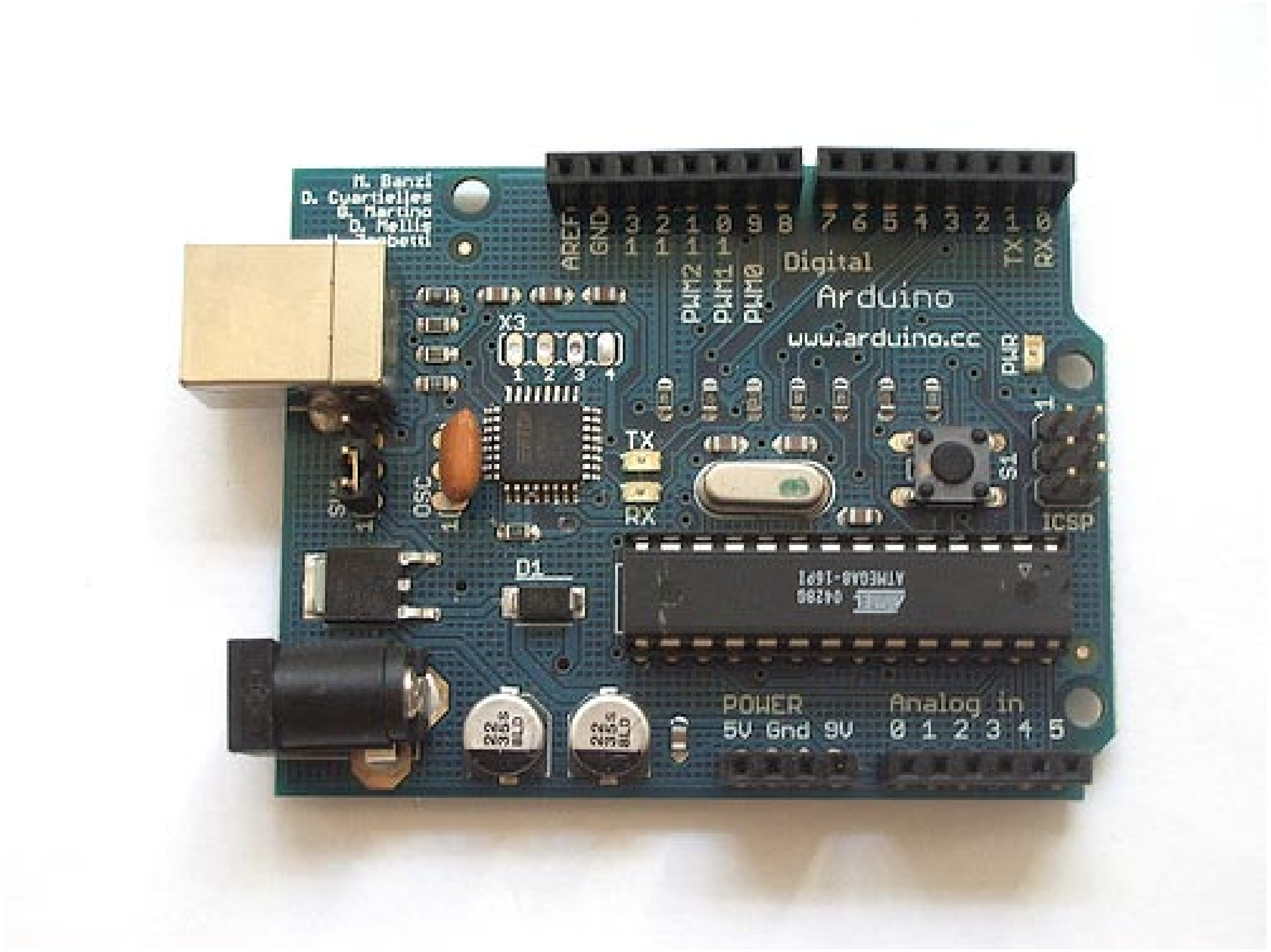
The cost of an Arduino board is around 20 EUR + VAT and with other 30/50 EUR it's possible to buy a good selection of sensors, some LEDs, a breadboard, wires and a multimetre. With less than 75 EUR is therefore possible to have a complete kit to experiment and create prototypes with.





02.0 the Arduino way

- introduction
- tinkering
- collaboration/community
- software tools



introduction
the Arduino way

The Arduino philosophy is based on making design rather then talking about it. It is a constant search for faster and more accurate ways to build better prototypes. We have explored many prototyping techniques and developed ways of thinking with our hands.
The classic engineering relies on a strict process for getting from A to B while the arduino way is based on maybe getting lost in the way and finding C instead. This is the process of tinkering that we are so fond about; playing with the medium in an open-ended way, finding the unexpected. In this search we also selected a number of software packages that enable that process, this constant manipulation of the software and hardware medium.

Another concept we developed is the "opportunistic prototyping": why spend time and energy building from scratch, a process that requires time and profound technical knowledge, while we can take already made devices and hack them in order
to exploit the hard work done by large companies and good engineers?
This become evident in Ivrea where the heritage of olivetti is represented by a few junkyards where computer parts, electronic components and devices of any sort have been dumped after the demise of the italian company. We could buy those devices for a few euros and hack them into out prototypes dramatically shortening the loop.

The last element is the community. Engaging people and push them to share by being the first to share.
We're standing
on the shoulders of the giants of opensource here.

Tinkering

the “MacGyver” style

From the “Tinkering” exhibition at the Exploratorium, San Francisco:

“Tinkering is what happens when you try something you don’t quite know how to do, guided by whim, imagination, and curiosity.

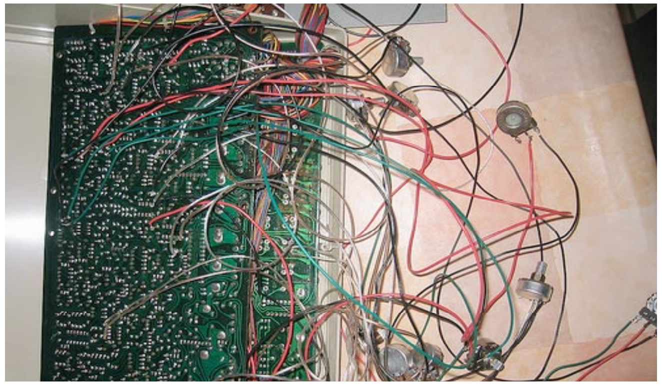
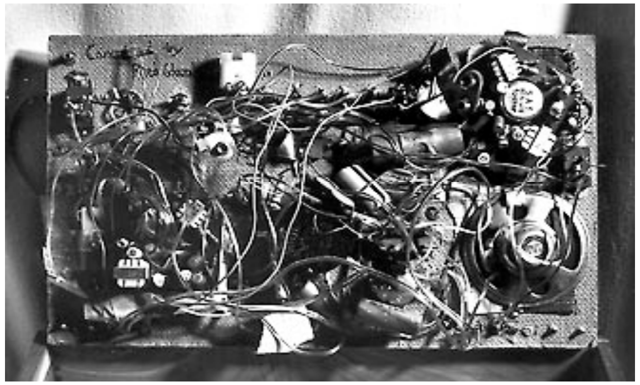
When you tinker, there are no instructions - but there are also no failures, no right or wrong ways of doing things. It’s about figuring out how things work and reworking them.

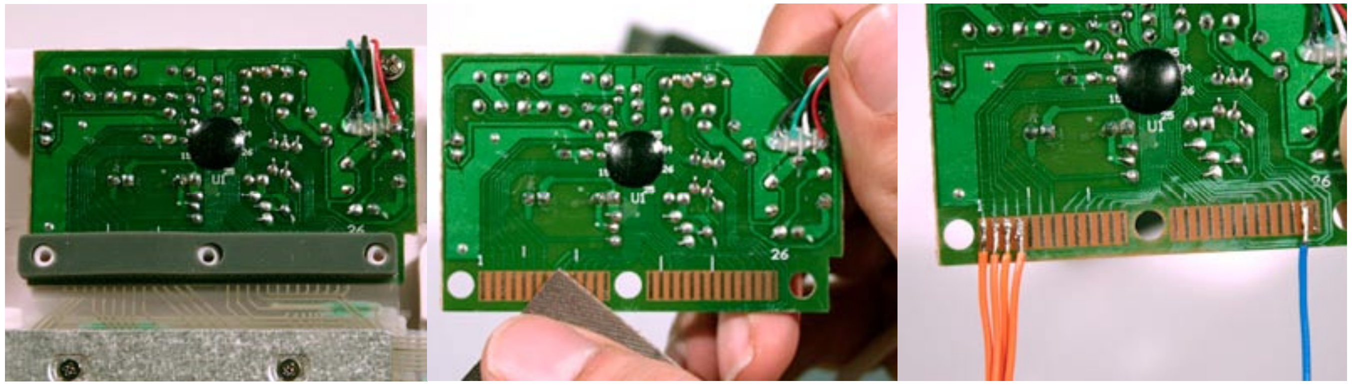
Contraptions, machines, wildly mismatched objects working in harmony - This the stuff of tinkering.”

This definition says a lot about the Arduino way of prototyping. Actually there is not a manual on how to do things but a reference collection of samples that people can modify and combine with other examples to learn about the logic of the programme and the board.

It’s a “hands on” way of working in which even junk becomes a source for learning and building prototypes. The reuse of material from other fields is an other big knowledge and material source. Learning how to do things by looking how other things work and can be hacked.

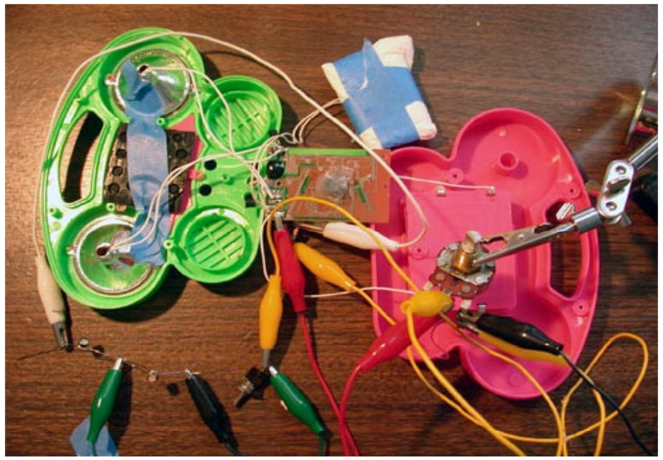
Circuit bending and patching are two key words in this learning school.



Keyboard hacks are an other example of the Arduino way. Its circuit is perfect for creating an input device to communicate with the computer.

Hacking it is extremely easy, once the keyboatd is opened and the protection layer of the contacts re-moved with a piece of sandpaper it's ready to become an input device with as many inputs as the number of keys on the keyboard.



LOW TECH SENSORS AND ACTUATORS

FOR ARTISTS AND ARCHITECTS

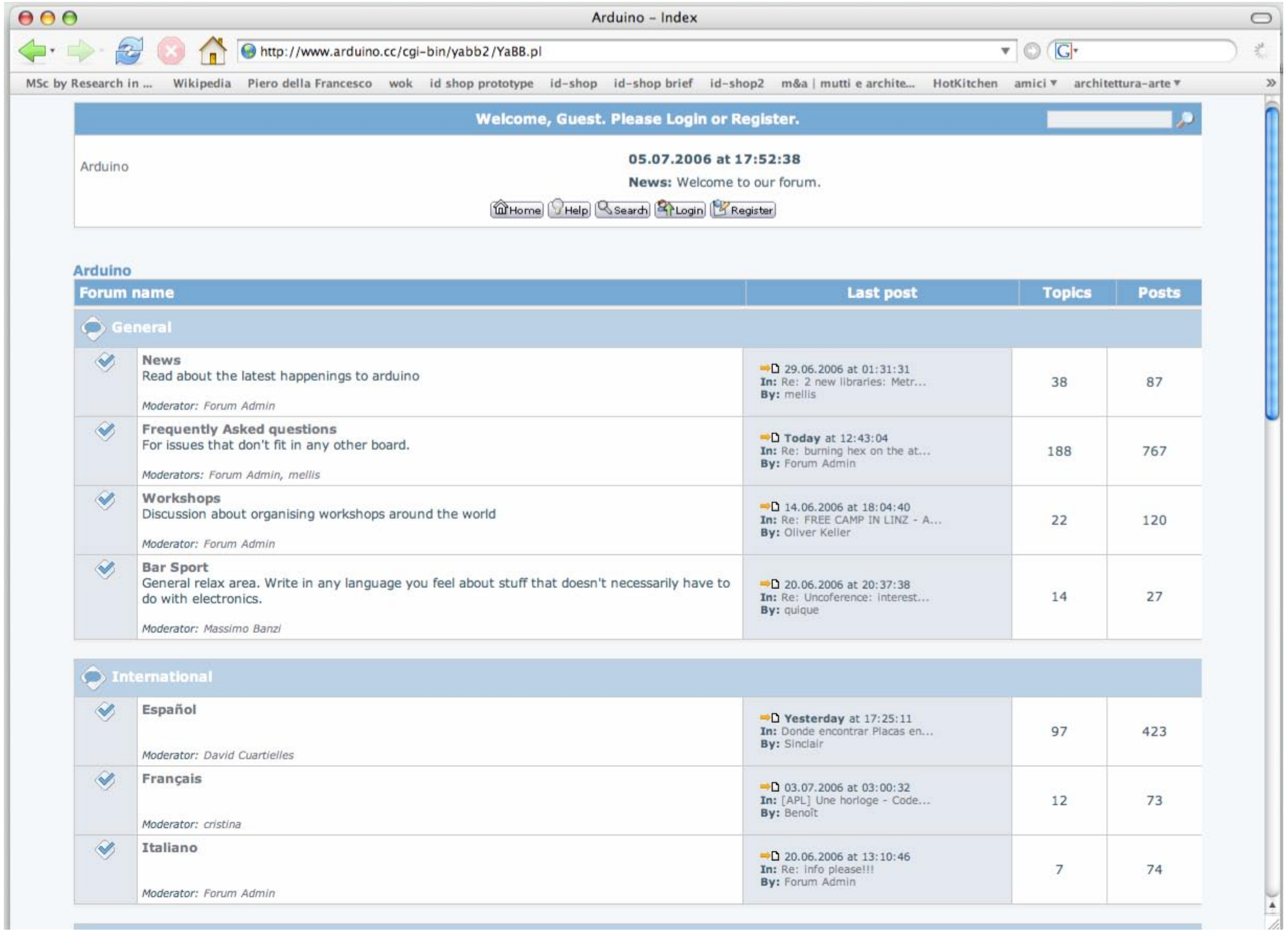
USMAN HAQUE & ADAM SOMLAI-FISCHER

a research project sponsored and commissioned by FACT Foundation for Art and Creative Technology, Liverpool U.K.

Toys are a great source of cheap components to play with. Many toys have simple sensor that react to light, pressure and proximity. Often toys also have little motors, LEDs, little speakers, ... all small electronic and mechanical elements that can be removed and used in an other contests.

A reference with regard to this is the "LOW TECH SENSORS AND ACTUATORS" initiative for artists and architects

by Usman Haque [www.haque.co.uk]
and Adam Somlai-Fischer [www.aether.hu]
http://lowtech.propositions.org.uk/



Arduino - Index
http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl
MSc by Research in ... Wikipedia Piero della Francesco wok id shop prototype id-shop id-shop brief id-shop2 m&a | mutti e archite... HotKitchen amici architettura-arte
Home Help Search Login Register

Welcome, Guest. Please Login or Register.

Arduino

05.07.2006 at 17:52:38

News: Welcome to our forum.

Arduino

Forum name	Last post	Topics	Posts
General			
News Read about the latest happenings to arduino *Moderator: Forum Admin*	29.06.2006 at 01:31:31 **In:** Re: 2 new libraries: Metr... **By:** mellis	38	87
Frequently Asked questions For issues that don't fit in any other board. *Moderators: Forum Admin, mellis*	**Today** at 12:43:04 **In:** Re: burning hex on the at... **By:** Forum Admin	188	767
Workshops Discussion about organising workshops around the world *Moderator: Forum Admin*	14.06.2006 at 18:04:40 **In:** Re: FREE CAMP IN LINZ - A... **By:** Oliver Keller	22	120
Bar Sport General relax area. Write in any language you feel about stuff that doesn't necessarily have to do with electronics. *Moderator: Massimo Banzi*	20.06.2006 at 20:37:38 **In:** Re: Uncoference: interest... **By:** quique	14	27
International			
Español *Moderator: David Cuartielles*	**Yesterday** at 17:25:11 **In:** Donde encontrar Placas en... **By:** Sinclair	97	423
Français *Moderator: cristina*	03.07.2006 at 03:00:32 **In:** [APL] Une horloge - Code... **By:** Benoît	12	73
Italiano *Moderator: Forum Admin*	20.06.2006 at 13:10:46 **In:** Re: info please!!! **By:** Forum Admin	7	74

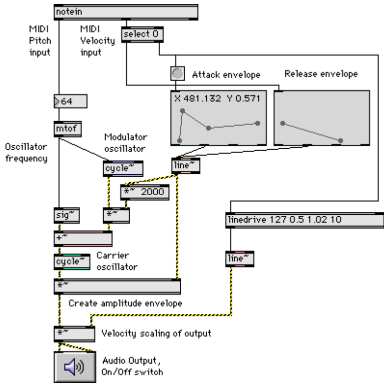
collaboration/community

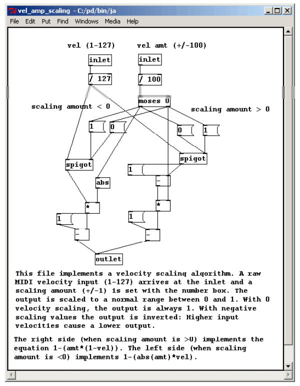
An outstanding quality of the Arduino project is the community that was created as was proved by the Honorary Mentions in the Prix Ars Electronica-International Competition for CyberArts 2006 in the Digital Communities section (http://www.aec.at/en/prix/communities/winners2006.asp)

From the early beginning Arduino based itself on the collaboration of people as an open-source project, a value they are very proud of.

In simple words, open source means that the code for the programme is public and poeple can contribute to it downloading the software, modifying it, improving it and then sending it back.
Forum for questions and opinions are opened in many languages even if the english and spanish community are, so far, the most active ones.
(http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl)

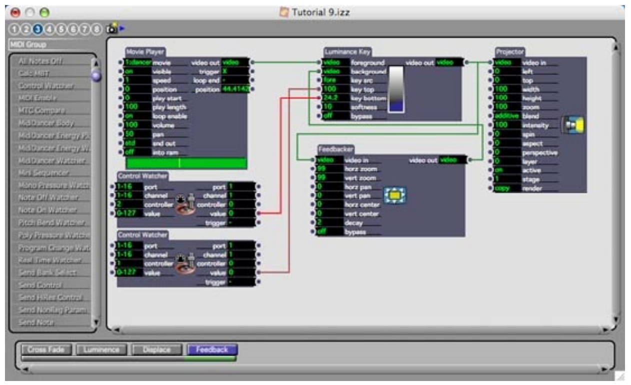
Arduino wants to develop internationally but at a local level. That's one of the reason why workshops are being organized around Europe, in order to create local groups to spread the knowledge from.





This file implements a velocity scaling algorithm. A raw MIDI velocity input (1-127) arrives at the inlet and a scaling amount (+/-1) is set with the number box. The output is scaled to a normal range between 0 and 1. With 0 velocity scaling, the output is always 1. With negative scaling values the output is inverted: Higher input velocities cause a lower output.

The right side (when scaling amount is >0) implements the equation 1-(amt*(1-vel)). The left side (when scaling amount is <0) implements 1-(abs(amt)*vel).



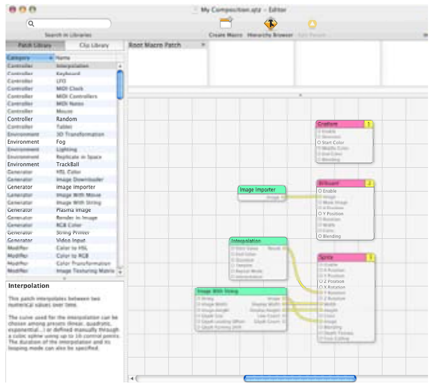
software tools

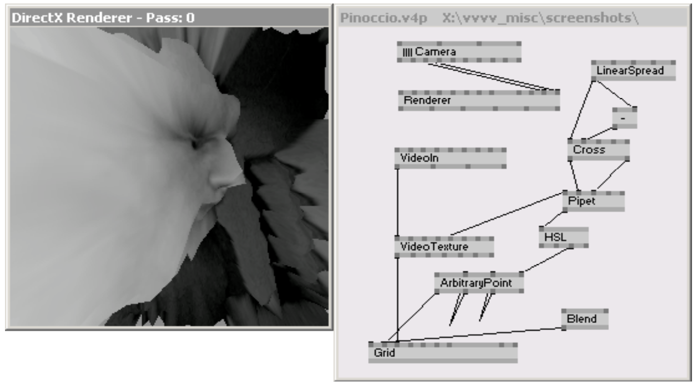
There are different software programmes which Arduino boar can interface with.

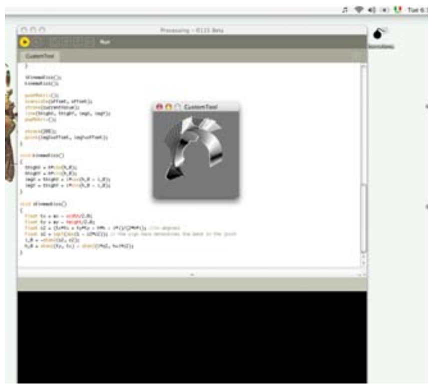
MAX
PURE DATA
QUARTZ COMPOSER
EYES WEB
VVVV
ISADORA
PROCESSING
FLASH
DIRECTOR

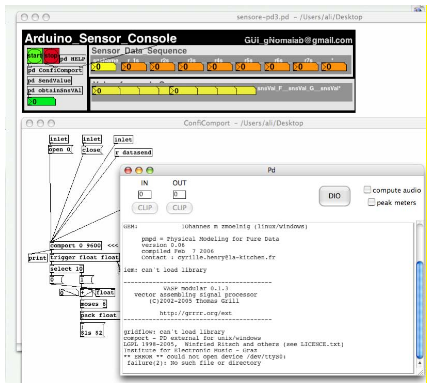
This wide flexibility of collaboration with different software packages provide the possibility for the user to work with their favorite software and the most suited one depending on the prototype or performance being created.

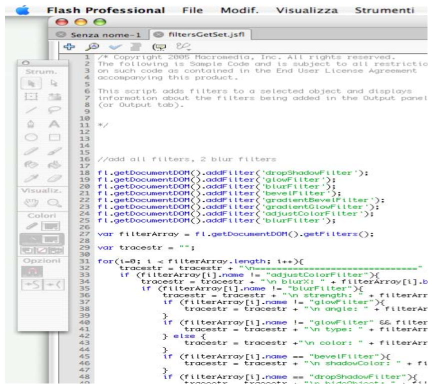
In the official website in the playground section (http://www.arduino.cc/playground/) there is a number of pages dedicated to the interfacing of Arduino to various softwares.





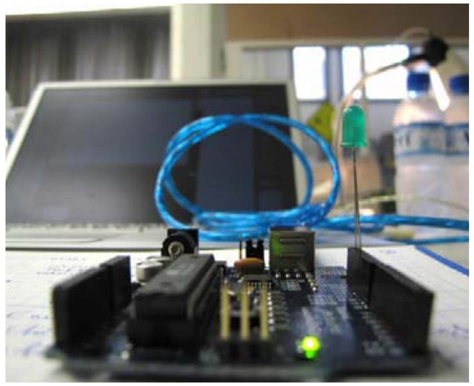






```
/* Copyright 2005 Macromedia, Inc. All rights reserved.
The following is Sample Code and is subject to all restrictio
on such code as contained in the End User License Agreement
accompanying this product.

This script adds filters to a selected object and displays
information about the filters being added in the Output panel
(or Output tab).


*/




//add all filters, 2 blur filters

fl.getDocumentDOM().addFilter('dropShadowFilter');
fl.getDocumentDOM().addFilter('glowFilter');
fl.getDocumentDOM().addFilter('blurFilter');
fl.getDocumentDOM().addFilter('bevelFilter');
fl.getDocumentDOM().addFilter('gradientBevelFilter');
fl.getDocumentDOM().addFilter('gradientGlowFilter');
fl.getDocumentDOM().addFilter('adjustColorFilter');
fl.getDocumentDOM().addFilter('blurFilter');

var filterArray = fl.getDocumentDOM().getFilters();

var tracestr = "";

for(i=0; i < filterArray.length; i++){
    tracestr = tracestr + "\n================================"
    if (filterArray[i].name != "adjustColorFilter"){
        tracestr = tracestr + "\n blurX: " + filterArray[i].b
        if (filterArray[i].name != "blurFilter"){
            tracestr = tracestr + "\n strength: " + filterArr
            if (filterArray[i].name != "glowFilter"){
                tracestr = tracestr + "\n angle: " + filterAr
            }
            if (filterArray[i].name != "glowFilter" && filter
                tracestr = tracestr + "\n type: " + filterArr
            } else {
                tracestr = tracestr +"\n color: " + filterArr
            }
            if (filterArray[i].name == "bevelFilter"){
                tracestr = tracestr + "\n shadowColor: " + fi
            }
            if (filterArray[i].name == "dropShadowFilter"){
```

interaction design lab

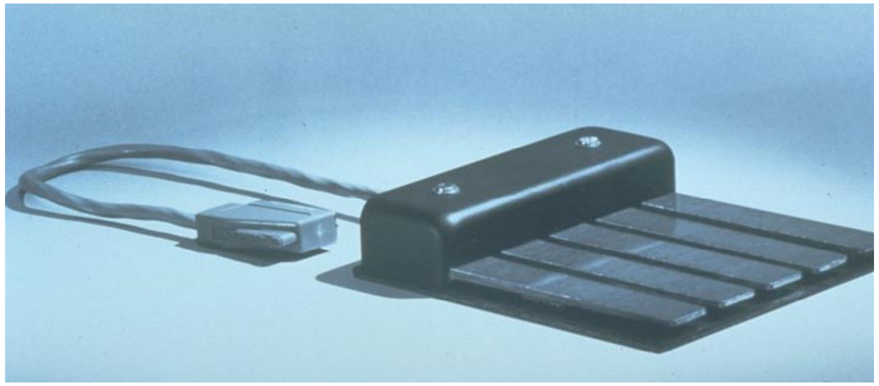


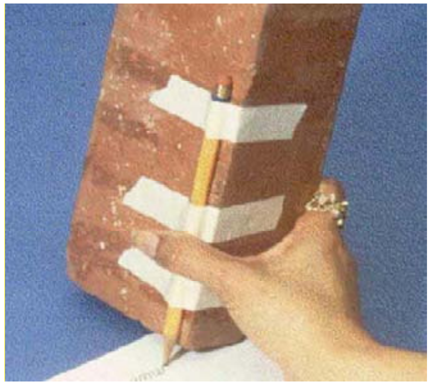
03.0 the workshop

- introduction
- Arduino today
- reference projects
- “hans on”

"*Many years ago, I dreamed that digital technology could greatly augment our collective human capabilities for dealing with complex, urgent problems.*"

-- Douglas Engelbart





introduction

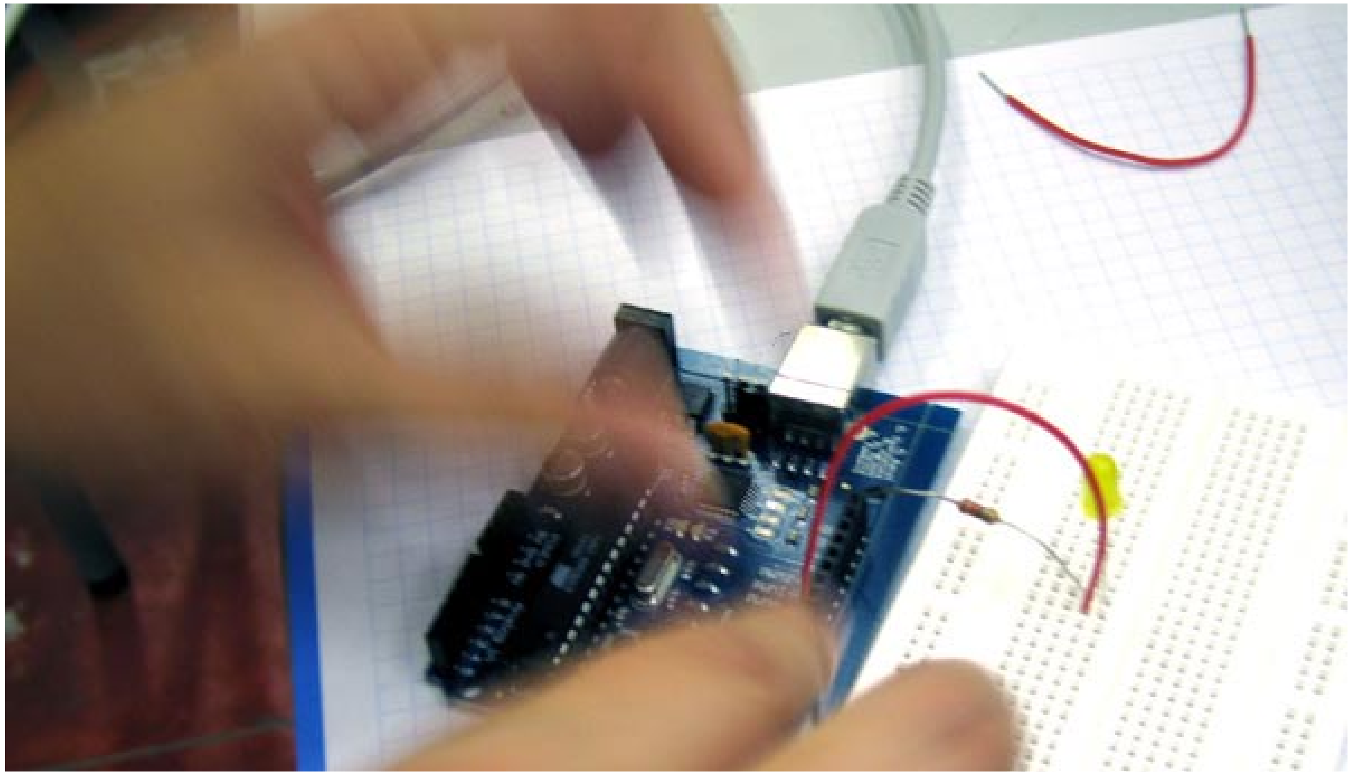
The first slides shown at the workshop introduce the concept of interaction design giving possible definitions of it.

Interaction Design is :
- the creation of meaningful relationships between us (humans) and artifacts
- the creation of beautiful relationships between us (humans) and artifacts
- the creation of controversial relationships between us (humans) and artifacts

Then it's briefly summarised how it developed in time:
70's User Centered Design
80's Human Computer Interface
90's Ubiquitous Computing
00's Palpable Computing

Interface design and physical computing are the two sides of interaction design.

Designing through prototypes is a central concept in the Arduino way: once you have an idea and start creating the prototype, it develops, changes, transforms into something else you would have not thought about before starting. Sometimes even mistakes can be usefull and lead to
interesting discoveries and solutions.
Prototypes don't need to be perfect, they can be rough and physical (paper, cardboard, wood, plastics functional or semi-functional) or imaginary (through experiential prototypes on video). The important is that the concept goes through and is communicated to the audience.

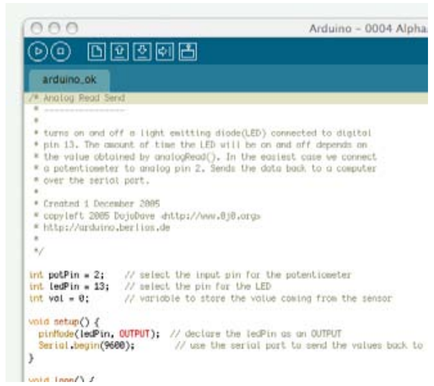


Arduino today

The Arduino system can be reasumed as follows:

the 3 basic parts:
• the circuit board
• the IDE Integrated Development Environment (Software)
• the reference collection of examples

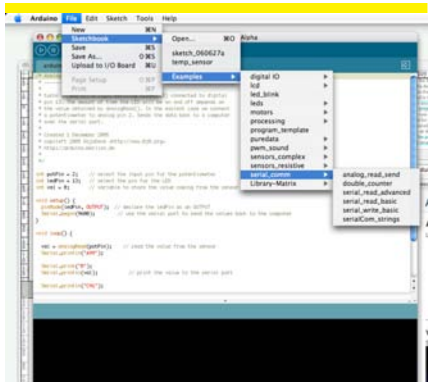
Arduino practice can be divided into 4 field of action:
• Education: in which a series of workshop are organised to spread the knowledge of Arduino.
• Research: The development and improvement is going on thanks to the research of the team themselves and to the many external people collaborating at the open-source project.
• Commercial projects: the board reached a good level of reliability and is now starting to be implemented into objects or used for installation at a commercial level.
• Competition: at design festivals and fairs, competitions are organised for people to create the better prototype in the shortest time or with the "Junk" they manage to collect in a day.

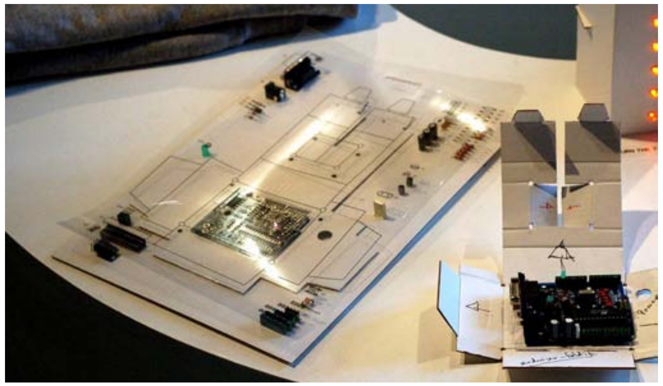
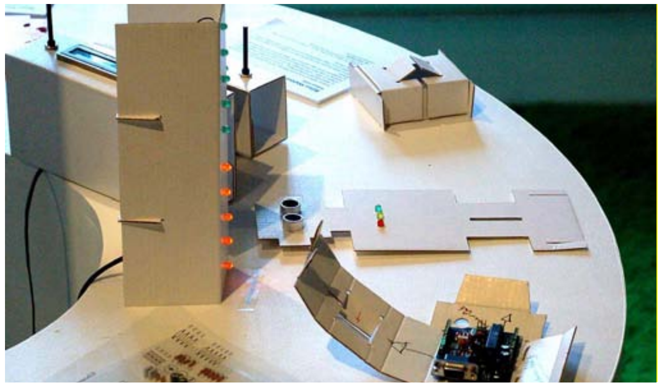
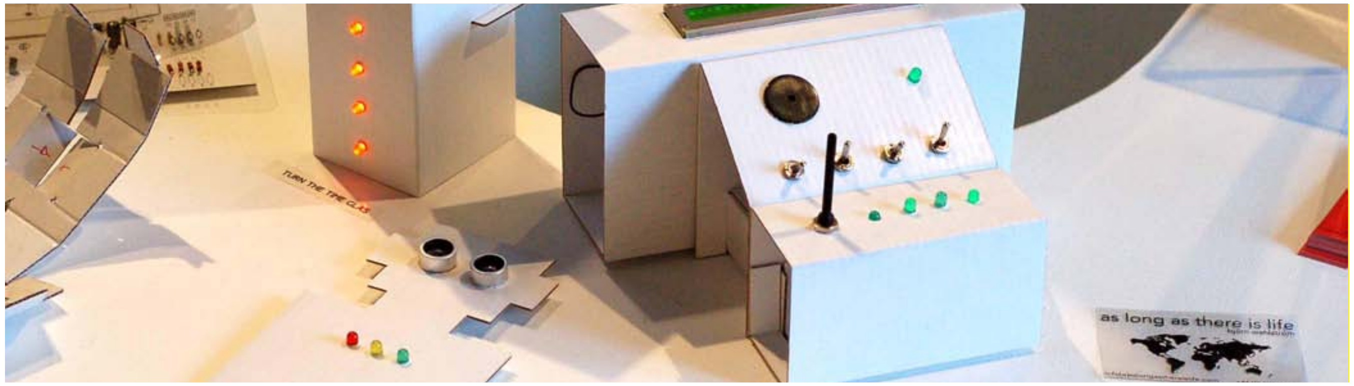


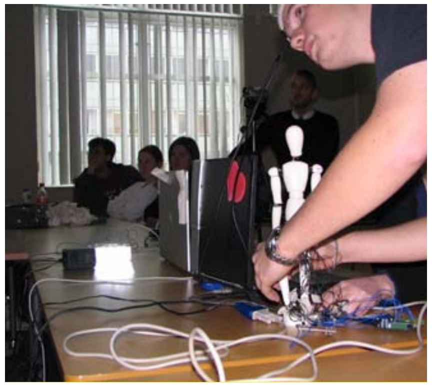
Arduino - 0004 Alpha

arduino_ok

```
/* Analog Read Send
 * ------------------
 *
 * turns on and off a light emitting diode(LED) connected to digital
 * pin 13. The amount of time the LED will be on and off depends on
 * the value obtained by analogRead(). In the easiest case we connect
 * a potentiometer to analog pin 2. Sends the data back to a computer
 * over the serial port.
 *
 * Created 1 December 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 */

int potPin = 2;    // select the input pin for the potentiometer
int ledPin = 13;   // select the pin for the LED
int val = 0;       // variable to store the value coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin as an OUTPUT
  Serial.begin(9600);        // use the serial port to send the values back to
}

void loop() {
```





reference projects

some student and commercial projects

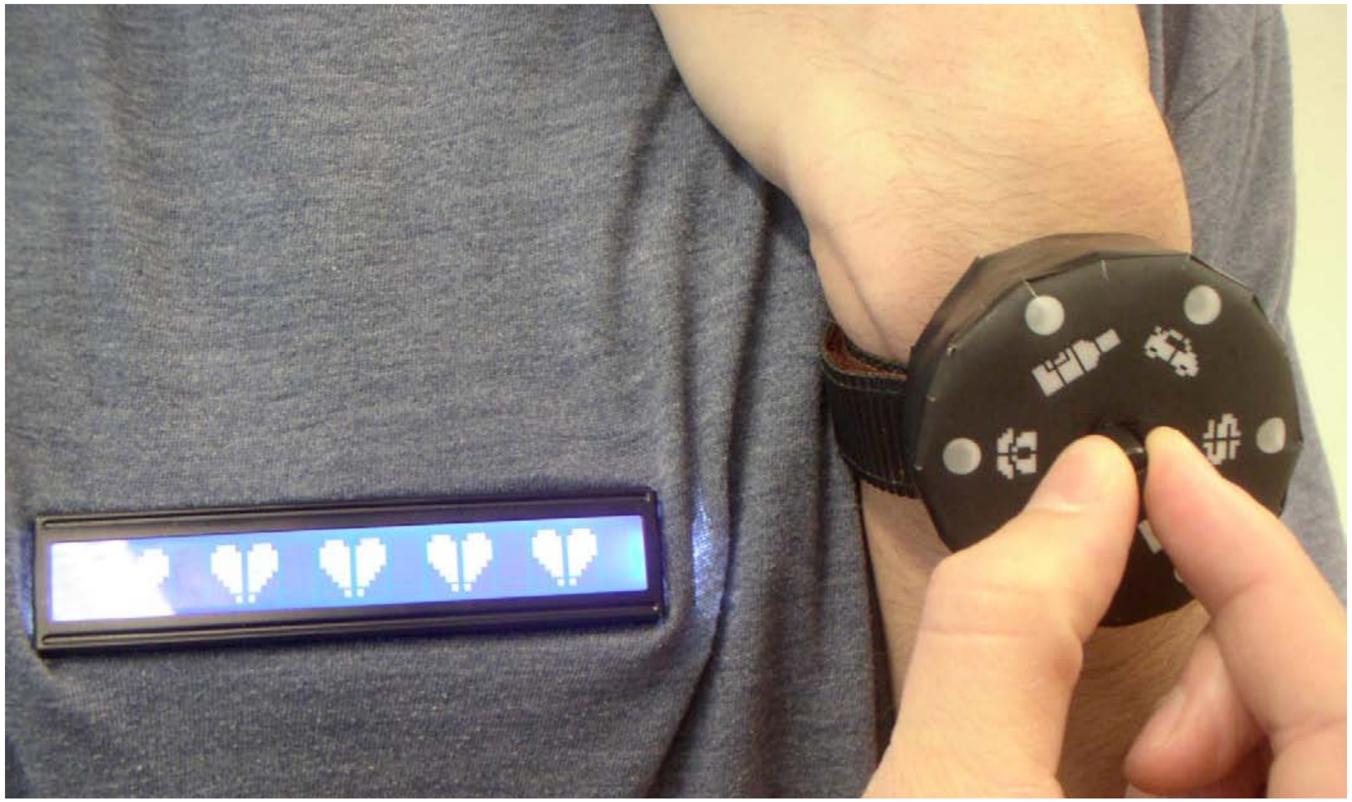
** **2006 Arduino.youFoldIt by Bjorn Wahlstrom shown at FDC**

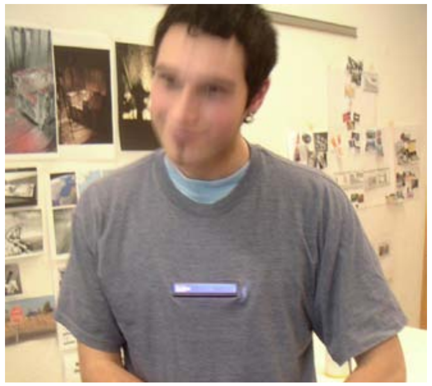
this is the exhibition material from Bjorn Wahlstrom, a designer from Malmo, Sweden. As part of his final thesis in Material and Virtual Design he created a collection of foldable shapes aiming to offer an alternative to wrap electronic prototypes in cardboard.

In the pictures we see:
- the Arduino serial KIT
- the box where to place the board in
- an example of a digital sandclock made by Bjorn Wahlstrom in collaboration with Thomas Ness and Per Nilsson
- an example of a musical instrument composed by three different foldable structures

** **Interaction Design Course K3**

A group of students make a game interface based on handling a puppet. The toy is connected to a flash game where the character will have to fight the dark side of the force.



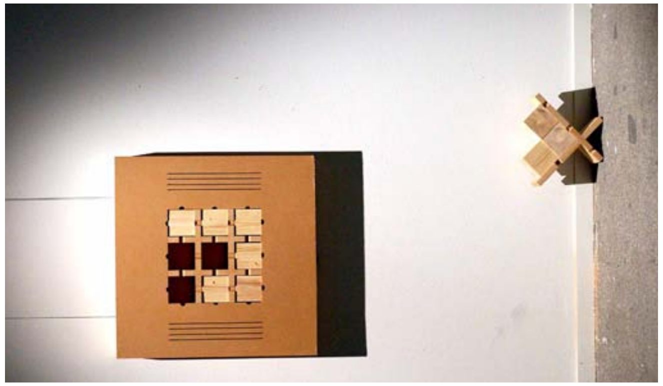
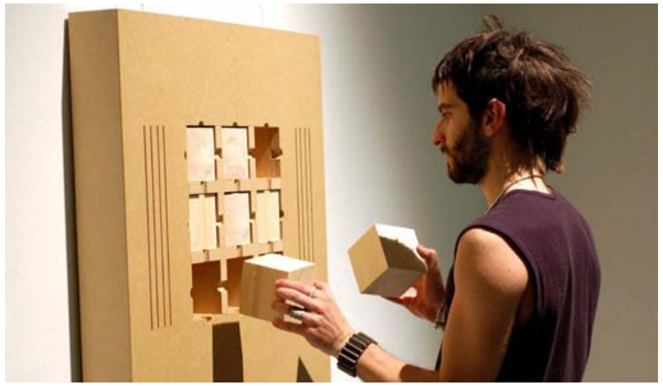


** 2006 FlirstShirt Basel january 2006

by Matthias Rohrbach, Simon Stotz, Simon Palmieri, and Valerie Lau

Text by Valeri Lau
FLIRT SHIRT
"In the heat of the night, when everybody makes his way to the bars and discotheques,it's the perfect time to wear our interactive flirt shirt. As you arrive at the club, you see all the sexy people ready to make this night a night to remember. The air is filled with lovely flavoured fragrances of lovely looking girls, you can't wait to try your luck. But as the night gets older your voice gets weaker. The music is loud, you can't understand a word and when you finally gained enough courage by drinking enough alcohol it seems like you won't have a chance to flirt with that beautiful girl at the bar without screaming at her. But don't worry, this is exactly the moment, our flirt shirt goes into action: Just turn on the LCD on the shirt, choose the symbol you want to display by spinning the clock like input device and without a single word you will have caught her eyes. Heart, dollar signs, swiss crosses, your phone number, the knight rider light animation and a popping bottle of champaign are the symbols, with which you can try to capture her heart.

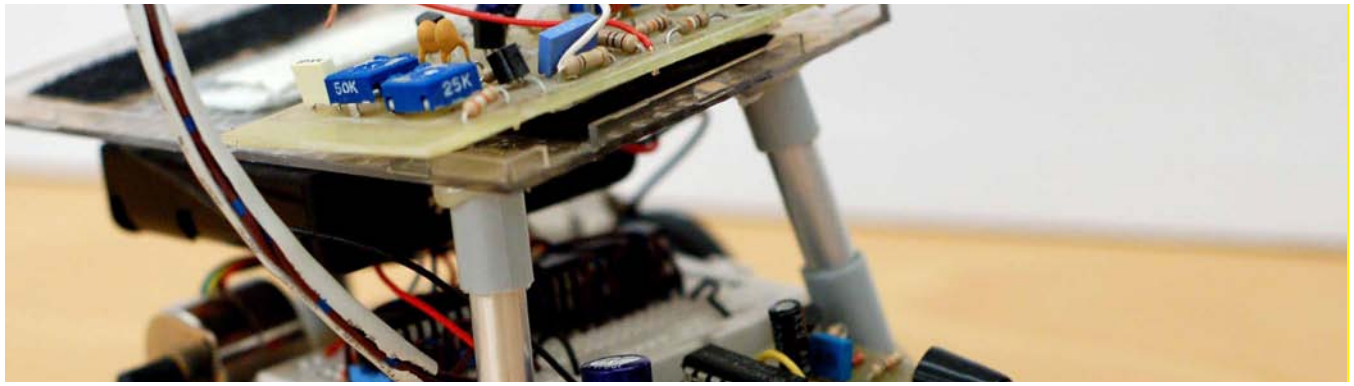
Technical stuff:
LCD display: we redefined some of the preset characters on the display to create the symbols and animations.
Clock like input device: We used a digital potentiometer to be able to go back and forth in the list of symbols. By connecting everything to two arduino boards correctly with a hell of a bunch of wires and having written an elegant code, we finally made it work. Tadaa. "

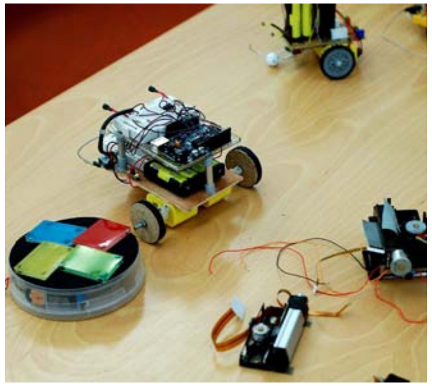
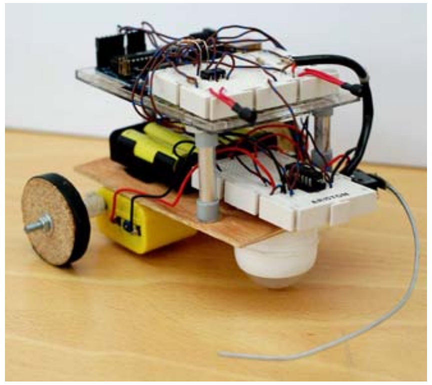
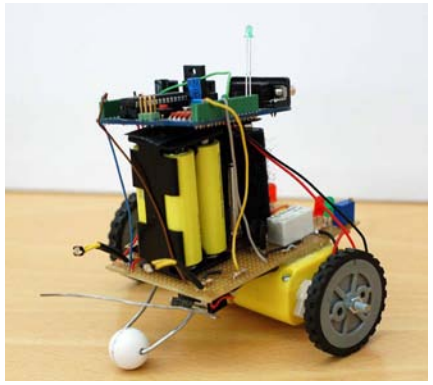



** 2005 hip to be square

by Andreas Karlsson, Mikel Jelm, Sebastian Ibarra

This is an MP3 player activated through the combination of boxes placed inside the shelf. There are 512 possible combinations, thus there are 512 mp3 songs that could be chosen. The interaction modes could be shifted from pure on-off to several flow modes.

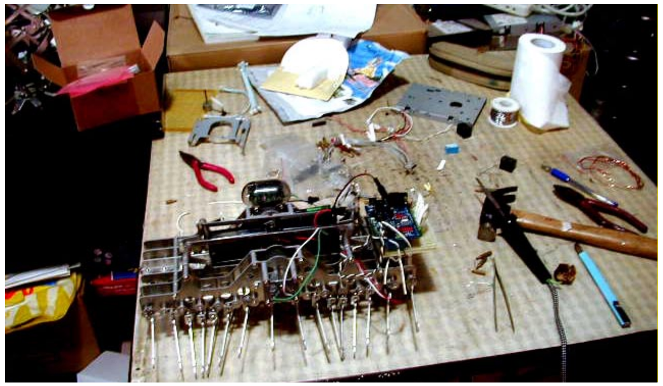
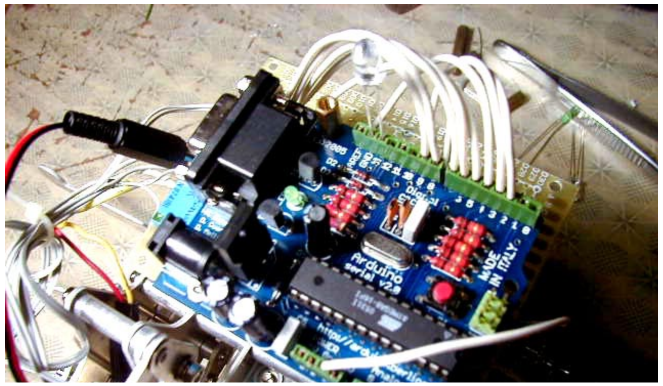


** 2006 robots for education in Spain

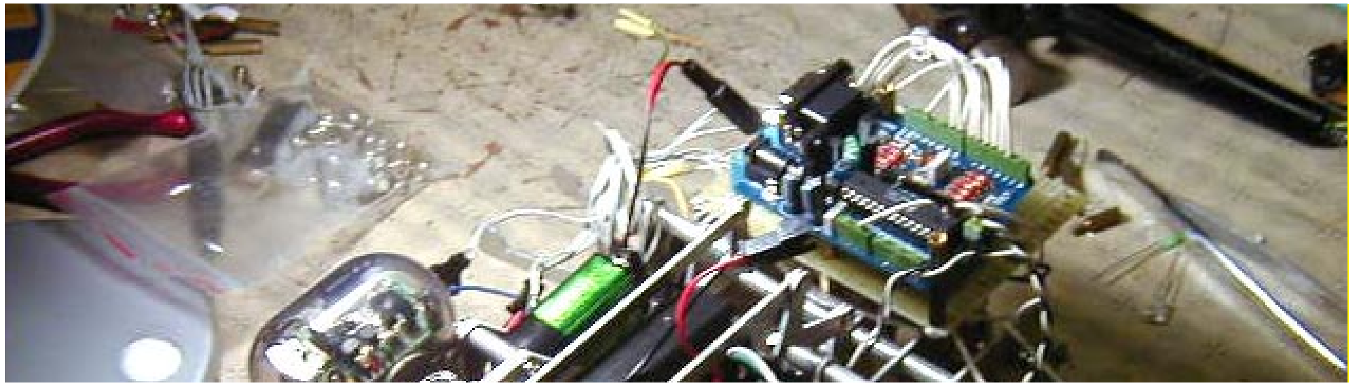
by Juan Carlos Alonso

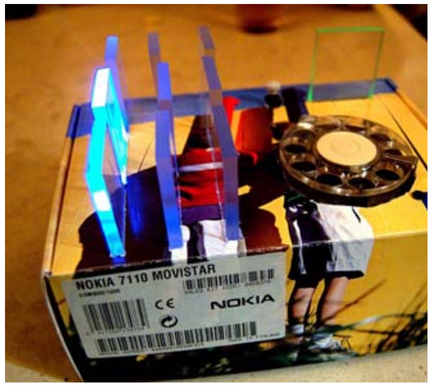
JC Alonso is a teacher at a secondary school in Mostoles, near Madrid. He is also one of the initiators of Dorkbot Madrid. He is interested in teaching his technology classes using state of the art of contemporary technical tools. He is building robots out of scrap, old electronic components, roll-on deodorant balls, and arduino boards.
He presented this robots as part of the Interactivos workshop in Spain (www.interactivos.org) during a special session oriented to secondary school teachers. One of his constructions was featured at Barcelona's Electronic Music Festival Sonar 2006, as part of the catalogue announcing a talk about Arduino at Sonarama, the space for experimental music.

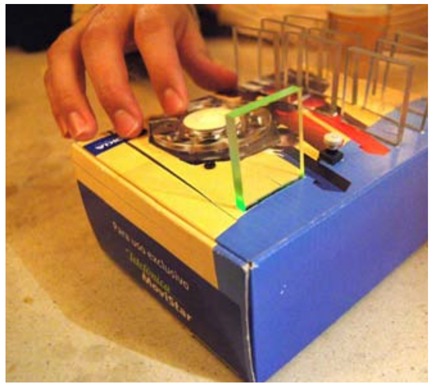
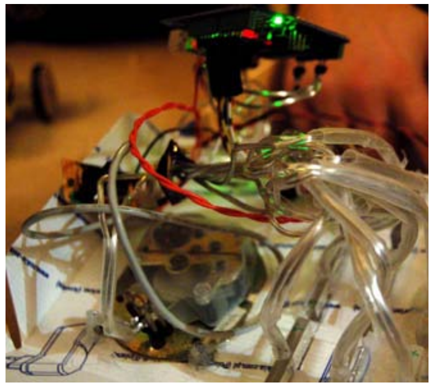



** 2005 Fallen Robotpic China

Fallen is a chinese engineering student that approached us asking for arduino boards to make a technology migration at this university. After we sent him a couple of boards for him to try out as a present, he came back with this collection of pictures of a 100-feet robot moving on his living room's floor. He controls his robot via Bluetooth using a PDA or a flash interface on his computer. All the intelligence is made with Arduino.

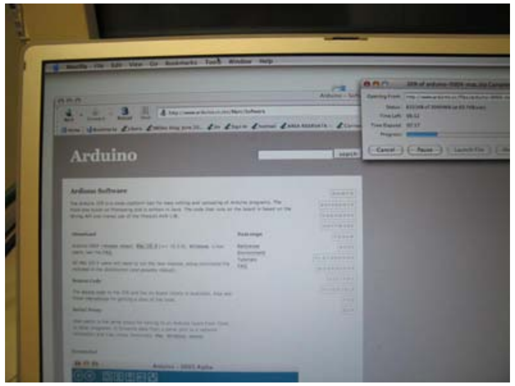




** 2006 Follow the number game by Igor Gonzalez

Igor is an engineer working at Telefonica, Spain. In his spare time he likes to design prototypes for kids to play with. In this case he made a small machine to show kids how to count. After counting the amount of lamps lighted up on the box, kid will have to dial up the number.



“hands on”

Now we are ready to start the practical part.
It’s structured with a serie of exercises in increasing difficulty by which the logic of the software and the board circuit is tought.

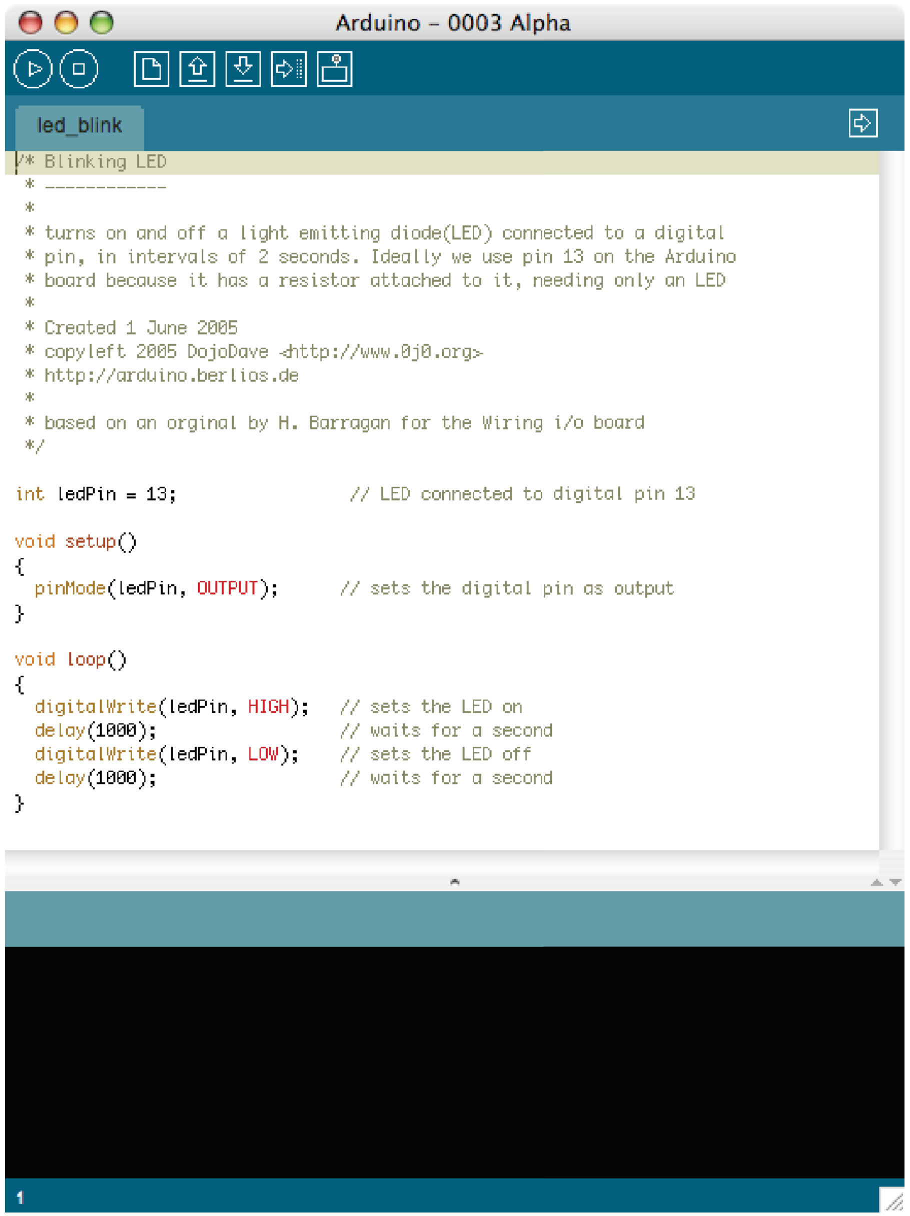
The exercises are the following. Before going to the next exercise, Massimo and David check that everybody has understood and managed to get the application working.

The exercises can be listed as follow:

Installing the application
Blinking an LED
What is electricity
The Breadboard
Reading a Pushbutton
Trying out different On/Off Sensors
Use the light sensor instead of the pushbutton
Analogue inputs
Try different Resistive sensors
Serial Communication
One day exercise
Analogue Output (PWM)
Driving bigger loads (motors etc)
Complex sensors
Talking to software

04.0 the exercises

- installing the software
- Arduino board
- exercises
- final prototype



```
/* Blinking LED
 * ------------
 *
 * turns on and off a light emitting diode(LED) connected to a digital
 * pin, in intervals of 2 seconds. Ideally we use pin 13 on the Arduino
 * board because it has a resistor attached to it, needing only an LED
 *
 * Created 1 June 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 * based on an orginal by H. Barragan for the Wiring i/o board
 */

int ledPin = 13;                 // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(1000);                  // waits for a second
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(1000);                  // waits for a second
}
```

installing the software

The first step is about getting the software downloaded on each student computer.

On the website www.arduino.cc at the section "software" there are the three version for MAC, Windows and Linux users. In each downloadpackage is a read me file that explains the different step to do to get it working properly.

As the programme is installed a first exercise need to be opened:
file/sketchbook/open/examples/digital IO/led_blink

Then the software is organized as follows:
In the beginning there is a comment part (*) where it's explained what is the example about, what hardware component is required to get it working and the credits of who created it.
Then follow the three main parts:

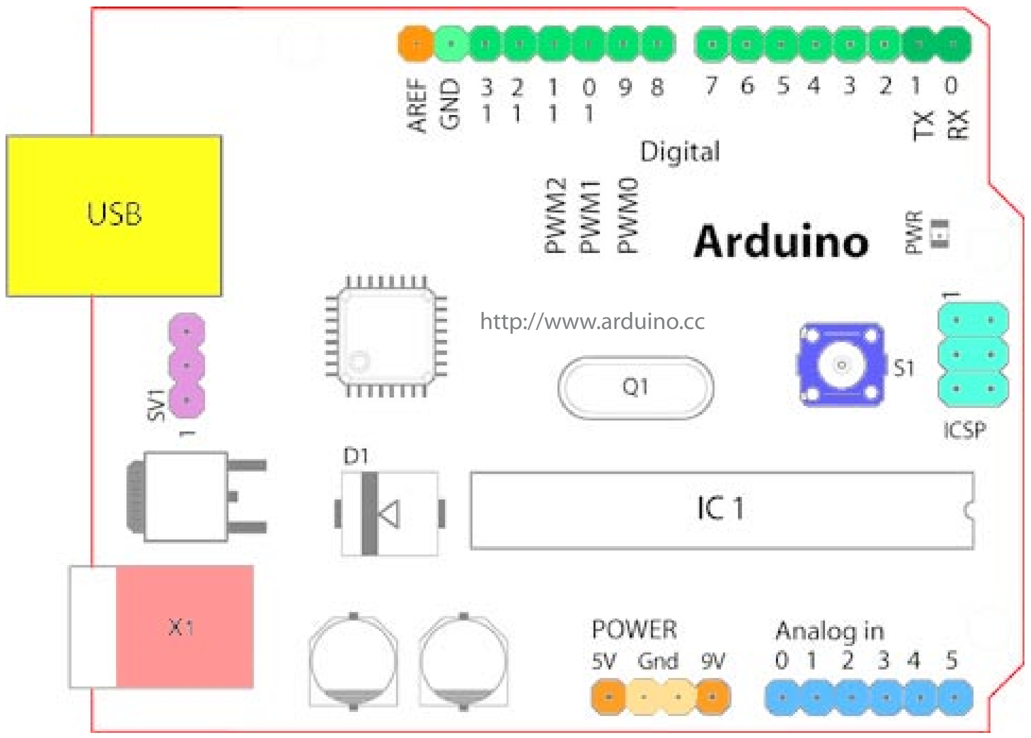
• all the variables involved are declare as "int"

• void setup()
setup() is the preparation: the set of the different variable features. In the setup section, always at the top of your programme, you would set pinMode, initialize serial communication, etc.

• void loop()
loop() is execution: the commands for the software to execute in loop like reading inputs, triggering outputs, etc.

* Variable Declaration
* Function Declaration

Arduino board
subtitle

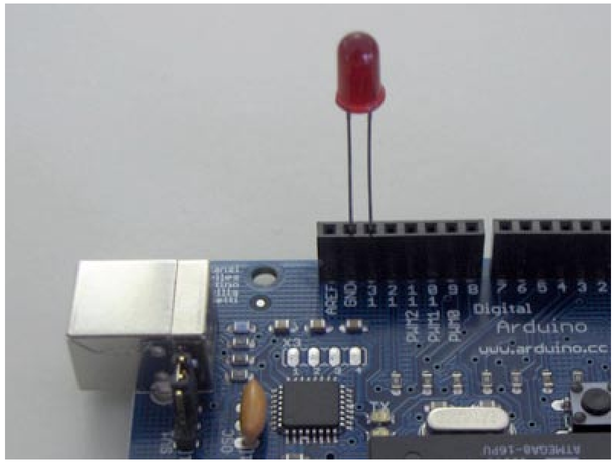


Starting clockwise from the top center:

- Analog Reference pin (orange)
- Digital Ground (light green)
- Digital Pins 2-13 (green)
- Digital Pins 0-1/Serial In/Out - TX/RX (dark green)
  - These pins cannot be used for digital i/o (digitalRead and digitalWrite) if you are also using serial communi cation (e.g. Serial.begin).
- Reset Button - S1 (dark blue)
- In-circuit Serial Programmer (blue-green)
- Analog In Pins 0-5 (light blue)
- Power and Ground Pins (power: orange, grounds: light orange)
- External Power Supply In (9-12VDC) - X1 (pink)
- Toggles External Power and USB Power (place jumper on two pins closest to desired supply) - SV1 (purple)
- USB (used for uploading sketches to the board and for serial communication between the board and the computer; can be used to power the board) (yellow)

THE CODE:

```
/* Blinking LED
 * turns on and off a light emitting diode(LED)
 * connected to a digital
 * pin, in intervals of 2 seconds.
 * Ideally we use pin 13 on the Arduino
 * board because it has a resistor attached to it,
 * needing only an LED
 * Created 1 June 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 * based on an orginal by H.
 * Barragan for the Wiring i/o board
 */

int ledPin = 13;                 // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT);      // sets the digital pin as output
}
void loop()
{
  digitalWrite(ledPin, HIGH);   // sets the LED on
  delay(1000);                  // waits for a second
  digitalWrite(ledPin, LOW);    // sets the LED off
  delay(1000);                  // waits for a second
}
```



exercises
(some examples)

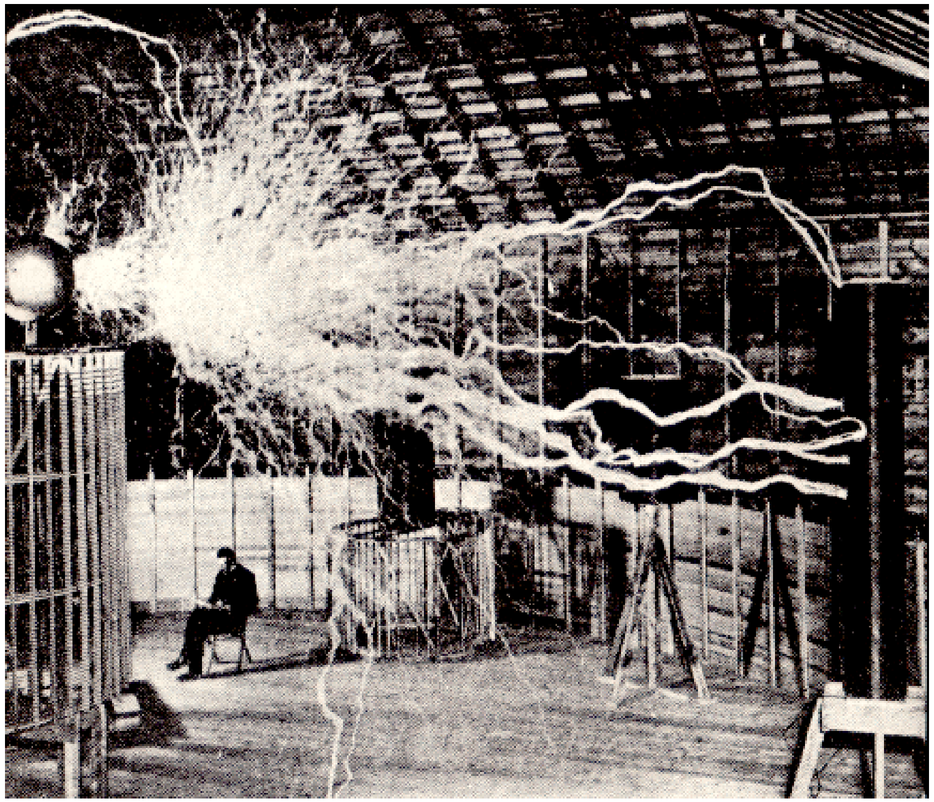
● the blinking LED

The first exercises is about making an LED blink. It's a very basic exercise but necessary to understand the order in which to proceed and get to know different parts of the board.

A first process to get confident with is the following:
-compile (to check that the programme is written right)
-reset the board(with the button on the board)
-export the programme to the board

As a microcontroller, Arduino doesn't have any pre-established output devices. Willing to provide newcomers with some help while debugging programs, it is proposed the use of one of the board's pins plugging a LED that we will make blink indicating the right functionality of the program.

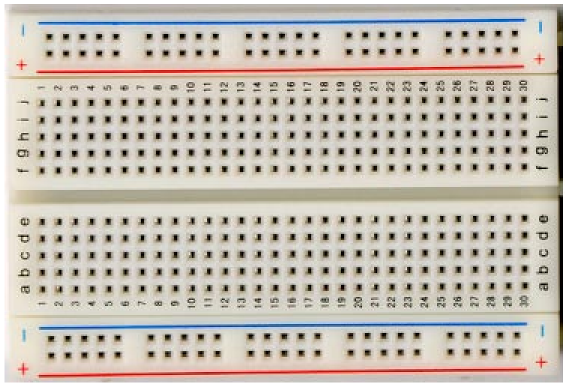
LEDs have polarity, which means they will only light up if you orient the legs properly. The long leg is typically positive, and should connect to pin 13. The short leg connects to GND; the bulb of the LED will also typically have a flat edge on this side. If the LED doesn't light up, trying reversing the legs (you won't hurt the LED if you plug it in backwards for a short period of time).



● What is electricity

The three most basic units in electricity are voltage (V), current (I) and resistance (r). Voltage is measured in volts, current is measured in amps and resistance is measured in ohms.
A neat analogy to help understand these terms is a system of plumbing pipes. The voltage is equivalent to the water pressure, the current is equivalent to the flow rate, and the resistance is like the pipe size.
There is a basic equation in electrical engineering that states how the three terms relate. It says that the current is equal to the voltage divided by the resistance.
I = V/r

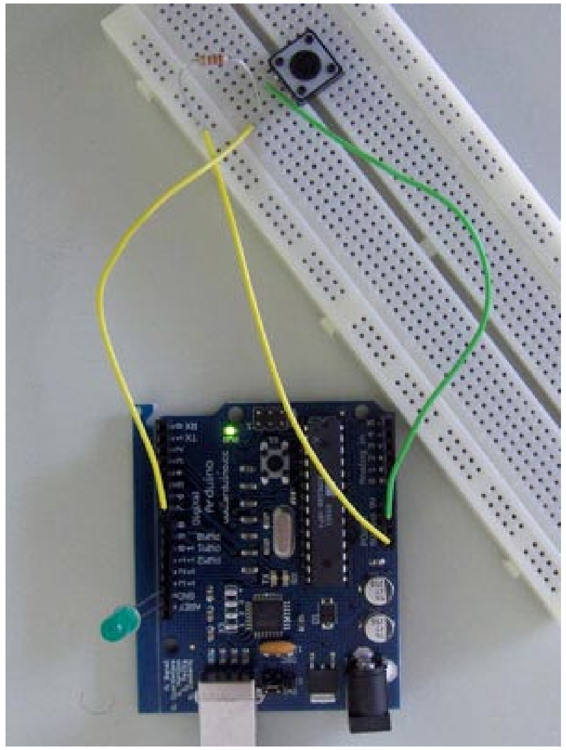
Electricity is a form of energy. Electricity is the flow of electrons. All matter is made up of atoms, and an atom has a center, called a nucleus. The nucleus contains positively charged particles called protons and uncharged particles called neutrons. The nucleus of an atom is surrounded by negatively charged particles called electrons. The negative charge of an electron is equal to the positive charge of a proton, and the number of electrons in an atom is usually equal to the number of protons. When the balancing force between protons and electrons is upset by an outside force, an atom may gain or lose an electron. When electrons are "lost" from an atom, the free movement of these electrons constitutes an electric current. For historical reasons, electric current is said to flow from the most positive part of a circuit to the most negative part. The electric current thus defined is called conventional current. It is now known that, depending on the conditions, an electric current can consist of a flow of charged particles in either direction, or even in both directions at once. The positive-to-negative convention is widely used to simplify this situation.



• The Breadboard

The breadboar is a device used to build circuits without the need of soldering. In the central part all the pins lined under a number are all connected while the ones next to the coloured lines are connected horizontally. By plugging components in the right holes, comples circuits can be built and modified in the space of a few minutes.

```
/* Basic Digital Read
 * ------------------
 *
 * turns on and off a light emitting diode(LED) connected to digital
 * pin 13, when pressing a pushbutton attached to pin 7. It illustrates the
 * concept of Active-Low, which consists in connecting buttons using a
 * 1K to 10K pull-up resistor.
 *
 * Created 1 December 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 */
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 released)
    digitalWrite(ledPin, LOW);  // turn LED OFF
  } else {
    digitalWrite(ledPin, HIGH);  // turn LED ON
  }
}
```



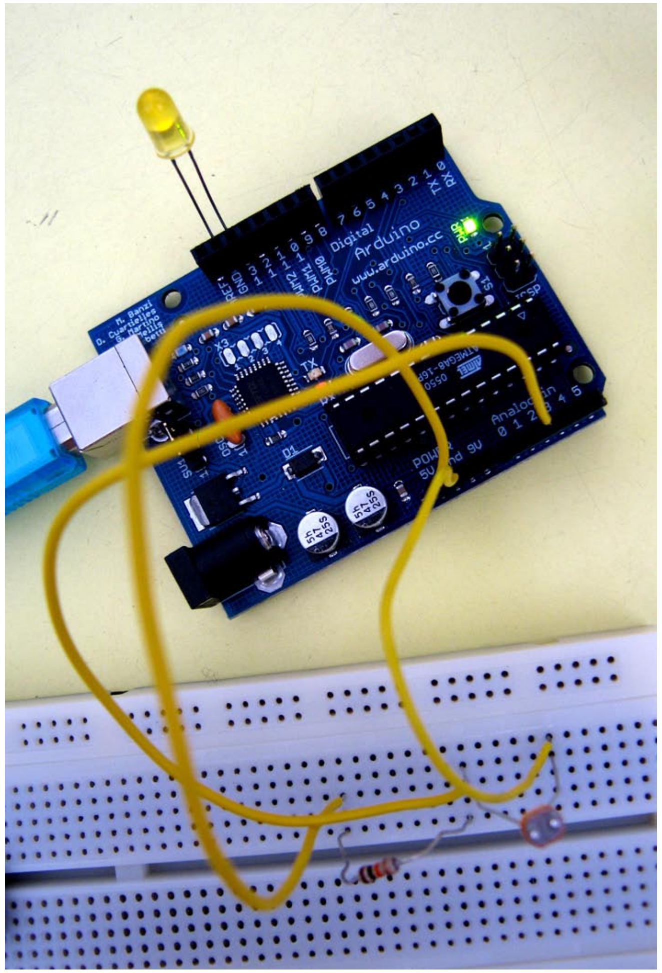
• Reading a Pushbutton

The pushbutton is a component that connects two points in a circuit when you press it. The example turns on an LED when you press the button.

We connect three wires to the Arduino board. The first goes from one leg of the pushbutton through a pull-up resistor (here 10 KOhms) to the 5 volt supply. The second goes from the corresponding leg of the pushbutton to ground. The third connects the pushbutton to a digital i/o pin (here pin 7) that reads the button's state.

When the pushbutton is open (unpressed) there is no connection between the two legs of the pushbutton, so the pin is connected to 5 volts (through the pull-up resistor) and we read a HIGH. When the button is closed (pressed), it makes a connection between its two legs, connecting the pin to ground, so that we read a LOW. (The pin is still connected to 5 volts, but the resistor in-between them means that the pin is "closer" to ground.) (LOW=zero=FALSE, HIGH=one=TRUE)

```
/* Analog Read Send
 * ----------------
 * turns on and off a light emitting diode(LED)
connected to digital
 * pin 13. The amount of time the LED will be on
and off depends on
 * the value obtained by analogRead(). In the easi-
est case we connect
 * a potentiometer to analog pin 2. Sends the data
back to a computer
 * over the serial port.
 * Created 1 December 2005
 * copyleft 2005 DojoDave <http://www.0j0.org>
 * http://arduino.berlios.de
 *
 */

int potPin = 2;    // select the input pin for the

potentiometer
int ledPin = 13;   // select the pin for the LED
int val = 0;       // variable to store the value
coming from the sensor

void setup() {
  pinMode(ledPin, OUTPUT);  // declare the ledPin
as an OUTPUT
  Serial.begin(9600);        // use the serial port
to send the values back to the computer
}

void loop() {
  val = analogRead(potPin);    // read the value
from the sensor
  Serial.println(val);           // print the value
to the serial port
  digitalWrite(ledPin, HIGH);  // turn the ledPin
on
  delay(val);                  // stop the program
for some time
  digitalWrite(ledPin, LOW);   // turn the ledPin
off
  delay(val);                  // stop the program
for some time
}
```



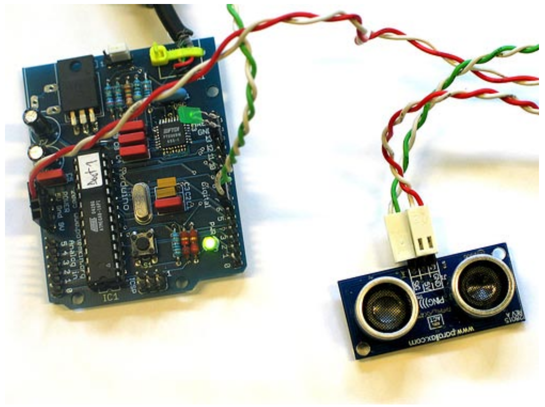
• Use the light sensor instead of the pushbutton

A LDR light sensor is a variable resistor. Notice, that in this case we use a pull-up resistor (which means that it is located before the sensor). Still, we measure the voltage 'between' this resistor and the light sensor. The circuit on your breadboard should look something like in the picture besides.

If everything is set up right, you can connect the board to the serial (or USB) port of your computer and, if necessary, to the power supply.

The more light the sensor receive the quicker is the blinking of the LED.

With the function:
"`Serial.println(val);`" it's possible to send numeric values to the computer where your favourite software can read them and use the sensor.



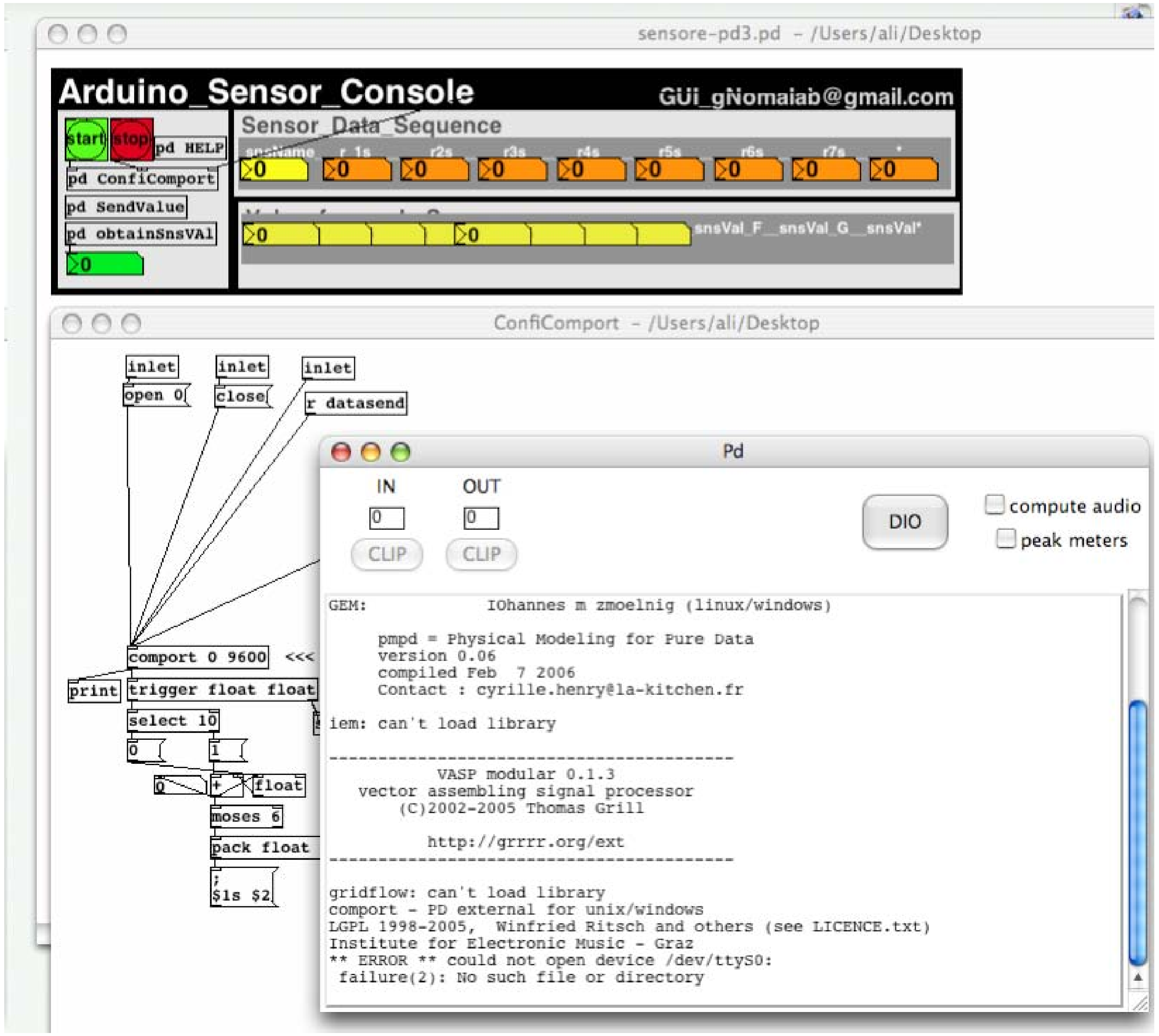
```
/* Ultrasound Sensor
 * Reads values (00014-01199) from an ultrasound sensor (3m sensor)
 * and writes the values to the serialport.
 *
 * http://www.xlab.se | http://www.0j0.org
 * copyleft 2005 Mackie for XLAB | DojoDave for DojoCorp
 */

int ultraSoundSignal = 7; // Ultrasound signal pin
int val = 0;
int ultrasoundValue = 0;
int timecount = 0; // Echo counter
int ledPin = 13; // LED connected to digital pin 13

void setup() {
  beginSerial(9600);                  // Sets the baud rate to 9600
  pinMode(ledPin, OUTPUT);            // Sets the digital pin as
output
}

void loop() {
 timecount = 0;
 val = 0;
 pinMode(ultraSoundSignal, OUTPUT); // Switch signalpin to output

/* Send low-high-low pulse to activate the trigger pulse of the
sensor
 */
digitalWrite(ultraSoundSignal, LOW); // Send low pulse
delayMicroseconds(2); // Wait for 2 microseconds
digitalWrite(ultraSoundSignal, HIGH); // Send high pulse
delayMicroseconds(5); // Wait for 5 microseconds
digitalWrite(ultraSoundSignal, LOW); // Holdoff

/* Listening for echo pulse
 */
pinMode(ultraSoundSignal, INPUT); // Switch signalpin to input
val = digitalRead(ultraSoundSignal); // Append signal value to val
while(val == LOW) { // Loop until pin reads a high value
  val = digitalRead(ultraSoundSignal);
}

while(val == HIGH) { // Loop until pin reads a high value
  val = digitalRead(ultraSoundSignal);
  timecount = timecount +1;            // Count echo pulse time
}

/* Writing out values to the serial port
 */

ultrasoundValue = timecount; // Append echo pulse time to ultra-
soundValue

serialWrite('A'); // Example identifier for the sensor
printInteger(ultrasoundValue);
serialWrite(10);
serialWrite(13);

/* Lite up LED if any value is passed by the echo pulse
 */

if(timecount > 0){
  digitalWrite(ledPin, HIGH);
}

/* Delay of program
 */
delay(100);
}
```

• Complex sensors

The PING range finder is an ultrasound sensor from Parallax able of detecting objects up to a 3 mts distance. The sensor has 3 pins, two are dedicated to power and ground while the third one is used both as input and output.

The pin dedicated to reading has to be shifting configuration from input to output according to the PING specification sheet. First we have to send a pulse that will make the sensor send an ultrasound tone and wait for an echo. Once the tone is received back, the sensor will send a pulse over the same pin as earlier. The width of that pulse will determine the distance to the object.

The example shown here was assembled by Marcus Hannerstig, while the software was created by David Cuartielles.



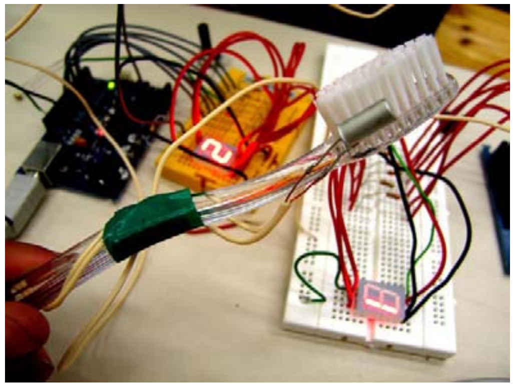
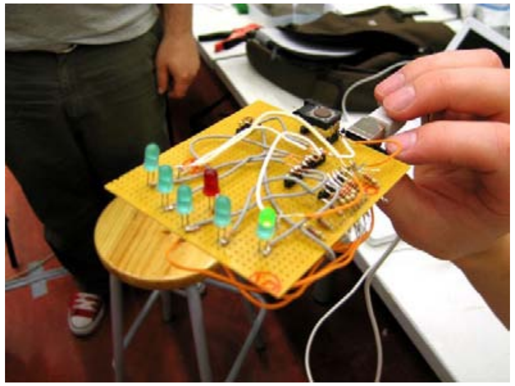
```
GEM:            IOhannes m zmoelnig (linux/windows)

     pmpd = Physical Modeling for Pure Data
     version 0.06
     compiled Feb  7 2006
     Contact : cyrille.henry@la-kitchen.fr

iem: can't load library

-----------------------------------------
           VASP modular 0.1.3
   vector assembling signal processor
       (C)2002-2005 Thomas Grill

          http://grrrr.org/ext
-----------------------------------------

gridflow: can't load library
comport - PD external for unix/windows
LGPL 1998-2005,  Winfried Ritsch and others (see LICENCE.txt)
Institute for Electronic Music - Graz
** ERROR ** could not open device /dev/ttyS0:
 failure(2): No such file or directory
```

• Talking to software

Arduino "talks", transmits data in Serial format, so any other program/programming language that has serial capabilities can easily comunicate with Arduino. Some programs (like Flash) don't natively speak Serial but they can still communicate with Arduino through an intermidiary, that like a translator enables them to talk to each other. Read through the specific sections and you will find references on how to interface Arduino with every possible software that has been connected so far.

With the function (as in the light sensor exercise):
`Serial.println(val);`

the software is able to receive a digital value from the sensor. This value is then easily usable in other programming software like PD, VVVV,Max,..
The different patches to communicate between Arduino and the other programmes are available to download in the playground section at www.arduino.cc

final prototype

After everybody managed to get all the different exercises working people divide into group of 2-3 people and develop together an interactive prototype.

David and Massimo made a list of possible prototype that they tested people is able to realize with the knowledge acquired during the workshop.

Everybody need to use a particular sensor then it's up to them if they want it to combine it with other technologies and software.

Here some example of project realized:

• A cocktail-shaker that lights up some green LEDs when the drink is ready.

• A toothbruch that through a dysplys says to you if you are brushing your teeth at the right speed.

• A catching-timer that works by setting the time bending it on one side to set the time and on the other to let the time count down start.

• A bath that change the lights, the music and lights up a fan according to the water temperature.

• A musical instrument, a flute, that creates sintetised sound and instead of reacting to the air been blowed, reacts to the light and different pressure of the buttons.

• A LED lamps that lights up when you go nearer to it.

Credits

idlab
Massimo Banzi
Ailadi Cortelletti

partner
David Cuartielles