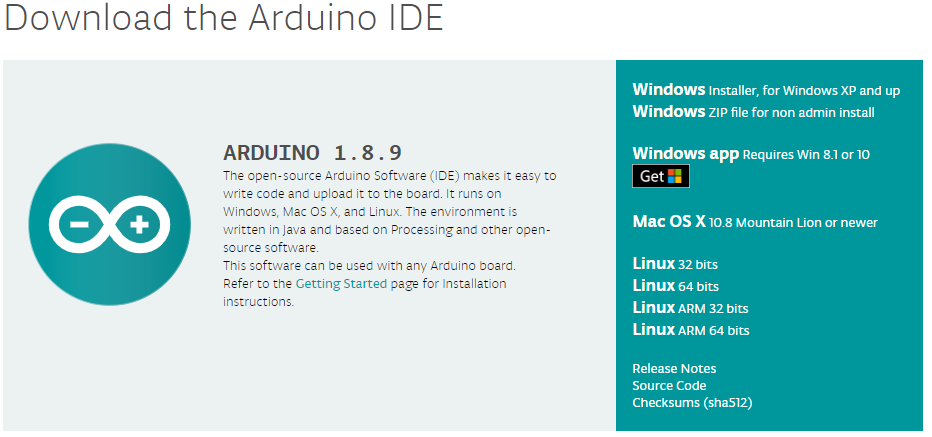
Cicle de vida d'un programa

El cicle de vida d’un programa són els passos necessaris per poder executar un programa a la placa Arduino:

* Editar
* Compilar (Verificar)
* Carregar el programa i Executar

Presentar el resultat amb imatges i/o vídeos de comprovació.

1. Descarregar i instal·lar l’Arduino IDE (<https://www.arduino.cc/en/Main/Software>).

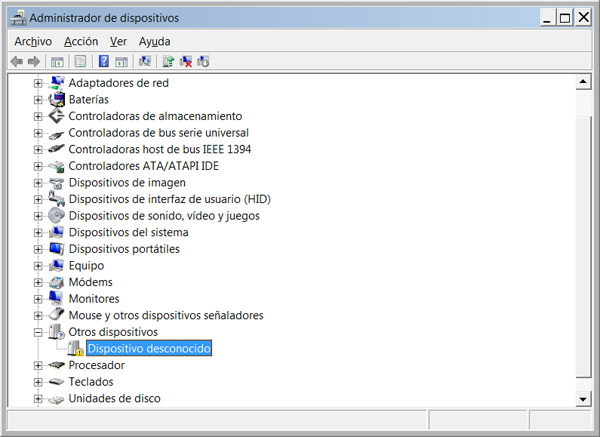


1. Connectar la placa Arduino a una entrada USB del PC. Si tot va bé sense fer res el PC detectarà l’Arduino i li assignarà un COM. Indicar en quin COM s’ha connectat l’Arduino al PC, es pot veure a l’Administrador de Dispositius, com es mostra en la següent imatge:

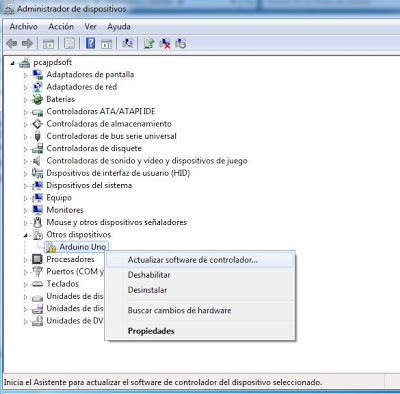
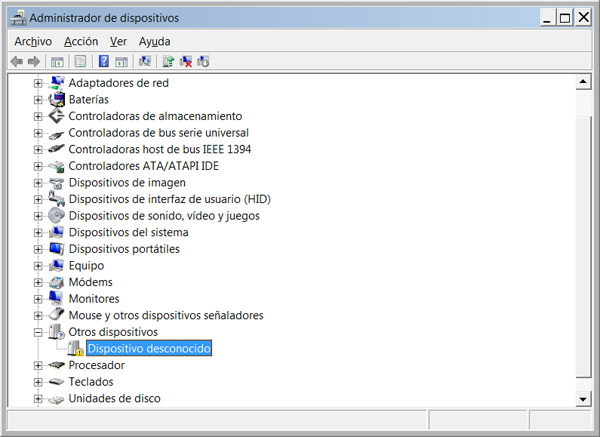


En aquest cas l’Arduino està al COM3

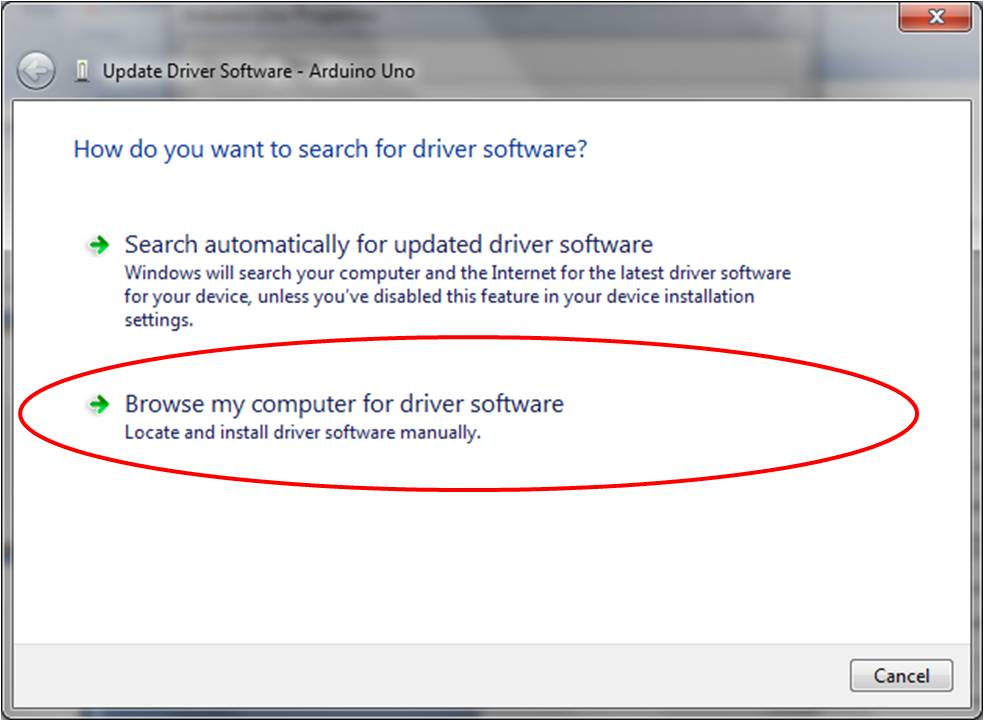
1. Si en l’apartat anterior, el PC no ha detectat l’Arduino (és molt habitual amb plaques Arduino no originals), a l’Administrador de Dispositius veurem lo següent:



Cal instal·lar els drivers de forma manual, els passos a seguir són:

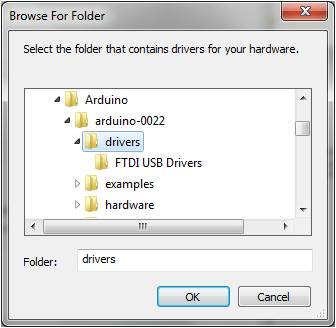


Prémer boto dret del ratolí sobre el dispositiu desconegut



Triar l’opció de buscar al nostre PC

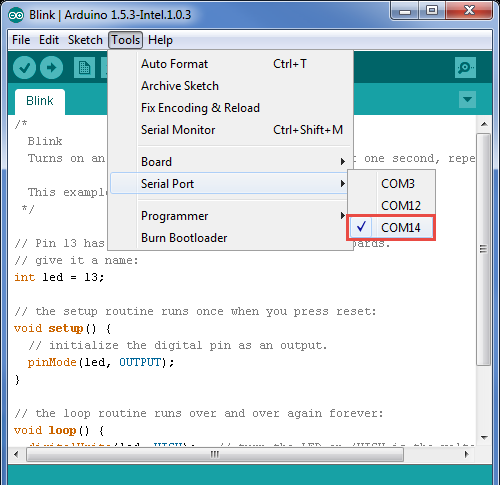
Anar a la carpeta on hi ha l’Arduino instal·lat (normalment a la carpeta Arxius de programa) i triar una de les següents carpetes Drivers o FTDI USB Drivers o amd64 o i386, depèn del PC, s’ha de provar amb totes. A vegades cal repetir el procés dos cops.



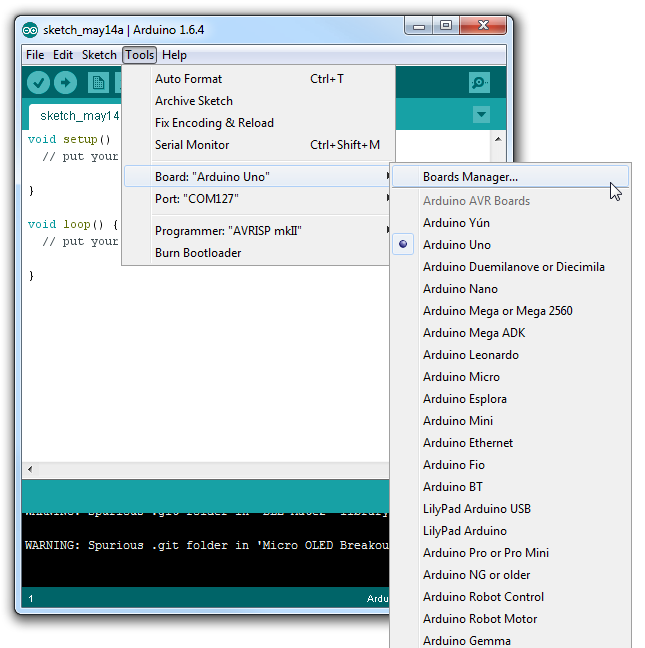
Si no s’instal·len els drivers fent lo explicat anteriorment, cal descarregar-los d’Internet, pots provar a la web del venedor del teu Arduino o buscar a Google FTDI USB Drivers. Un cop els tenim al PC, cal realitzar els passos indicats anteriorment però en aquest cas enlloc d’anar a la carpeta d’Arduino cal anar a la carpeta dels Drivers descarregats. Un cop finalitzat tornar a repetir l’exercici 2.

1. Carregar el primer programa a l’Arduino. Per carregar un programa a l’Arduino cal obrir i configurar l’Arduino IDE.

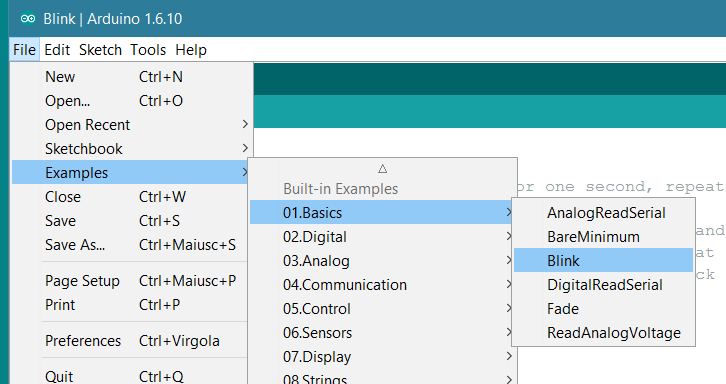
**Primer pas.** Configurar el COM, en aquest cas l’Arduino s’ha connectat al COM14



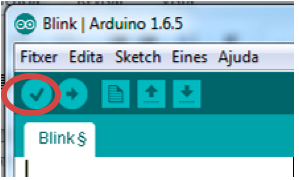
**Segon pas.** Configurar la placa Arduino que és connecta, en aquest cas la UNO



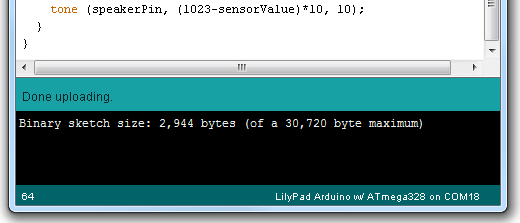
**Tercer pas.** Obrir o editar el program que es vol carregar, en aquest cas el programa d’exemple **Blink**. Aquest programa farà un intermitent d’1 segon sobre un led que porta la pròpia placa i està connectat al pin 13. De moment, no és important entendre el programa. Ens hem de fixar en tots els passos que cal seguir.



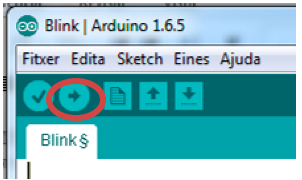
**Quart pas.** Compilar el programa. El programa està escrit en un llenguatge basat en C++ i és un llenguatge d’alt nivell que el micro no l’entén. És necessari fer una traducció a codi màquina. D’això s’encarrega el programa compilador.



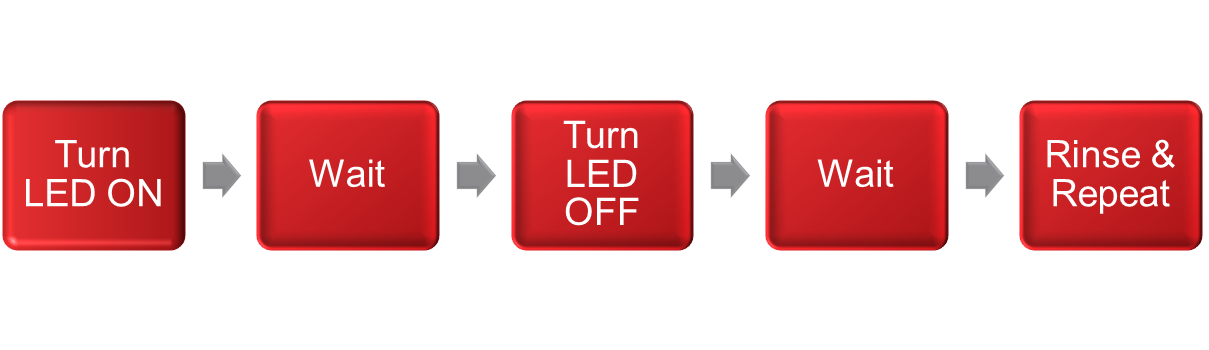
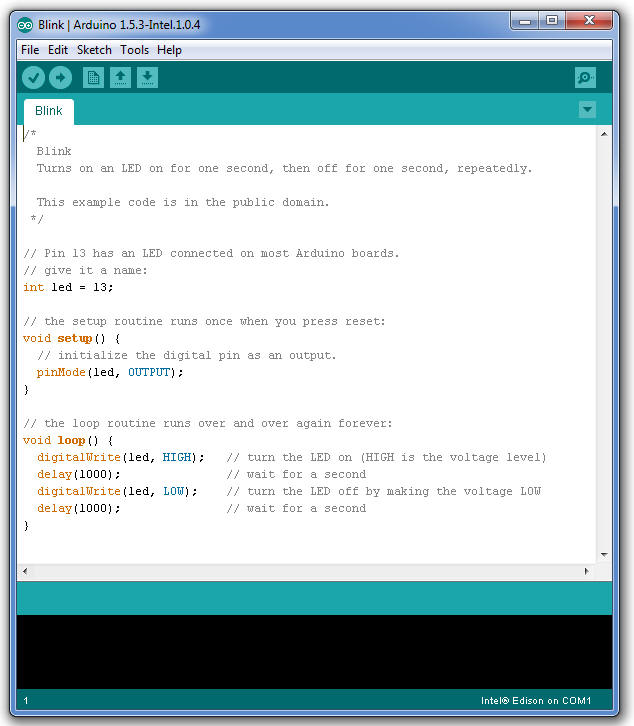
Si hi ha errors, a la finestra de sortida, tindrem el llistat dels errors que s’han produït i que haurem de corregir fins que tinguem una compilació sense cap error. Si no hi ha errors, obtenim:

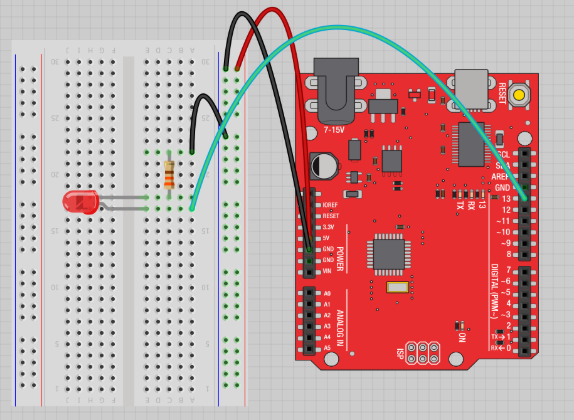


**Cinquè pas.** Carregar el programa a l’Arduino. Una vegada hem compilat podem transferir el programa al uControlador a través del cable que tenim connectat a l’USB. Veurem que els leds TX i RX de la placa Arduino fan pampallugues indicant que s’ha establert la comunicació. Si tot ha anat bé veurem el led de la sortida 13 fent intermitent.



1. De l’exercici anterior canviar els delay(1000) per delay(200), carregar a l’Arduino i explicar que ha passat. Per què és coloca el darrer delay?
2. void setup() {
3. // initialize digital pin LED\_BUILTIN as an output.
4. pinMode(LED\_BUILTIN, OUTPUT);
5. }
6. // the loop function runs over and over again forever
7. void loop() {
8. digitalWrite(LED\_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
9. delay(1000);                       // wait for a second
10. digitalWrite(LED\_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
11. delay(200);                       // wait for a second
12. }





Afegir un led i una resistència al pin 13 tal i com s’indica a la següent imatge, per tal de veure millor els resultats.

1. Modifica l’exercici 5, per tal que el led imiti el batec d’un cor.

void setup() {

  // initialize digital pin LED\_BUILTIN as an output.

  pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

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

  delay(200);                    // wait for a second

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

  delay(200);

  digitalWrite(LED\_BUILTIN, HIGH);

  delay(200);

  digitalWrite(LED\_BUILTIN, LOW);

  delay(1000);                    // wait for a second

}

1. Modifica l’exercici 5, trobar la màxima velocitat que el led pot fer pampallugues i que l’ull humà pugui detectar. 1ms delay? 2ms delay? 3ms delay?

10ms delay.

void setup() {

  // initialize digital pin LED\_BUILTIN as an output.

  pinMode(LED\_BUILTIN, OUTPUT);

}

// the loop function runs over and over again forever

void loop() {

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

  delay(10);                    // wait for a second

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

  delay(10);

  digitalWrite(LED\_BUILTIN, HIGH);

  delay(10);

  digitalWrite(LED\_BUILTIN, LOW);

  delay(10);                    // wait for a second

}

1. Indicar on es troben tots aquests elements a l’Arduino IDE:
2. Verify: Compiles and approves your code. It will catch errors in syntax (like missing semicolons or parentheses).
3. Upload: Verify and Sends your code to the board.
4. New: This buttons opens up a new code window tab.
5. Open: This button will let you open up an existing sketch.
6. Save: This saves the currently active sketch.
7. Serial Monitor: This will open a window that displays any serial information your 101 board is transmitting. It is very useful for debugging.
8. Sketch Name: This shows the name of the sketch you are currently working on.
9. Code Area: This is the area where you compose the code for your sketch.
10. Message Area: This is where the IDE tells you if there were any errors in your code.
11. Text Console: The text console shows complete error messages. When debugging, the text console is very useful.
12. Board and Serial Port: Shows you what board and the serial port selections.
13. Add file project.
14. Serial Plotter.
15. Library Manager.

L

M

J

K

H

G

F

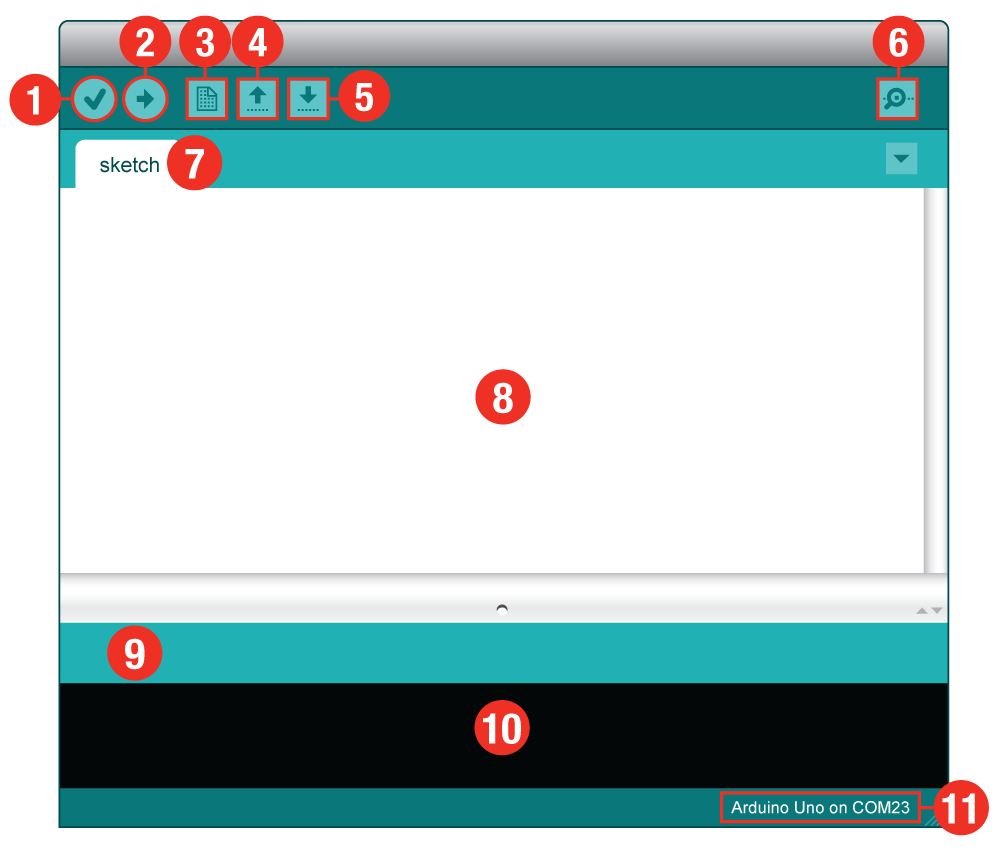
E

D

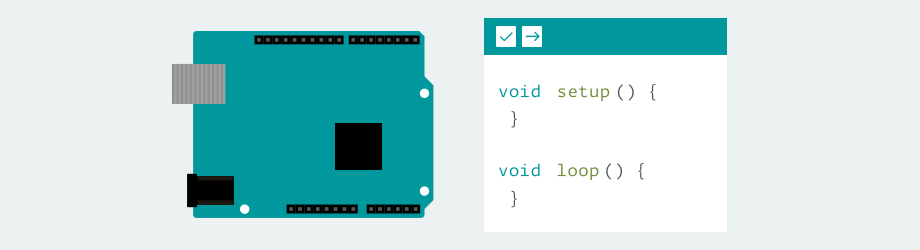
C

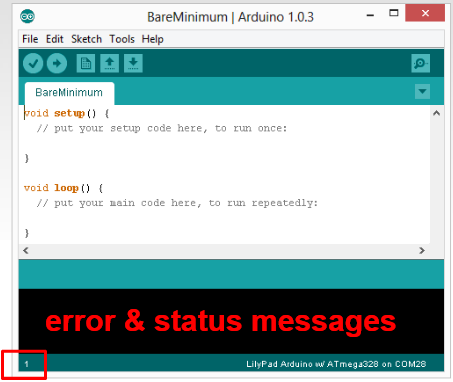
B

A



1. El codi mínim que necessita l’Arduino per funcionar és:





1. Per a tots els exercicis proposats utilitzarem la següent **plantilla**. Crear un sketch anomenat plantilla, per ser utilitzat posteriori durant tot el repte per iniciar qualsevol sketch.





**Requadres rosa, els comentaris, que NO s’envien a l’Arduino**

**Requadres verds, el codi, que SI s’envien a l’Arduino**

## 

## Watch out Curly braces!



## 

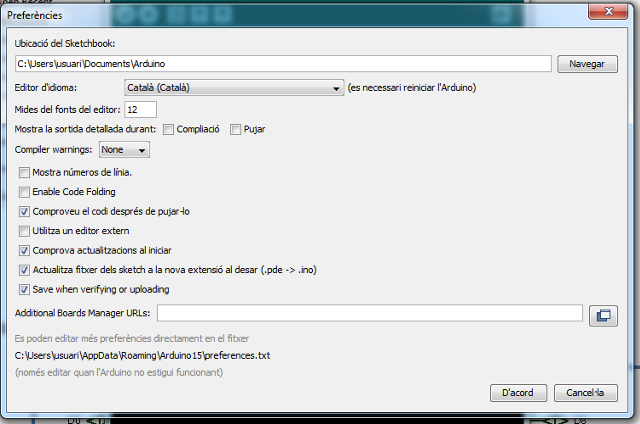
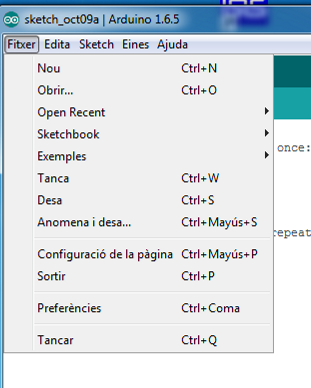
## Per saber-ne més

En el cas de no disposar d’una placa Arduino, es poden provar els programes amb simulació, utilitzant les eines de Proteus:

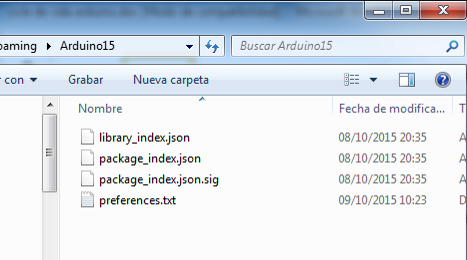
* ISIS: Disseny d'esquemes.
* VSM: Simulació de circuits.

L’Arduino quan compila genera una sèrie de fitxers i els posa en una carpeta que està molt amagada. En aquesta carpeta ens interessa el fitxer amb extensió .hex que és el que porta el codi binari del programa i per tant és el que hem de carregar al uControlador. El que farem és crear una carpeta amb el nom HEX a la nostra carpeta de treball i modificarem les preferències de l’Arduino perquè guardi els fitxers a la nostra carpeta HEX.

Cal seguir aquests passos:



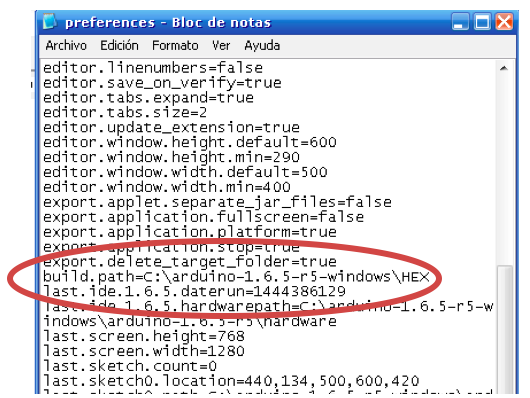
Anar a la carpeta on hi ha el fitxer preferences.txt, que obrirem per modificar. Abans és aconsellable fer una còpia de seguretat d’aquest fitxer pel que pugui passar.



Abans d’obrir el fitxer, tanqueu l’arduino. Obriu el fitxer. Heu d’afegir una línia després de export.delete\_target\_folder=true

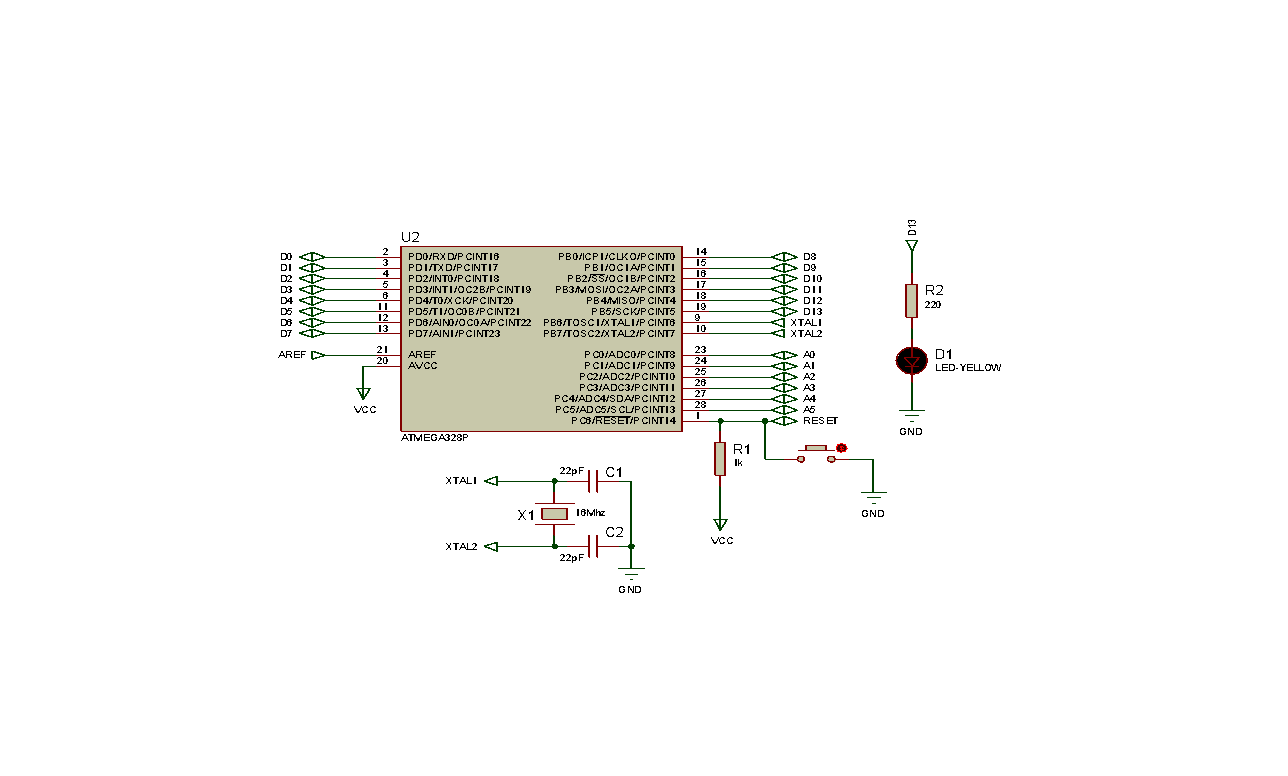
build.path=C:\arduino-1.6.5-r5-windows\HEX

És la carpeta on volem que ens generi els arxius HEX.

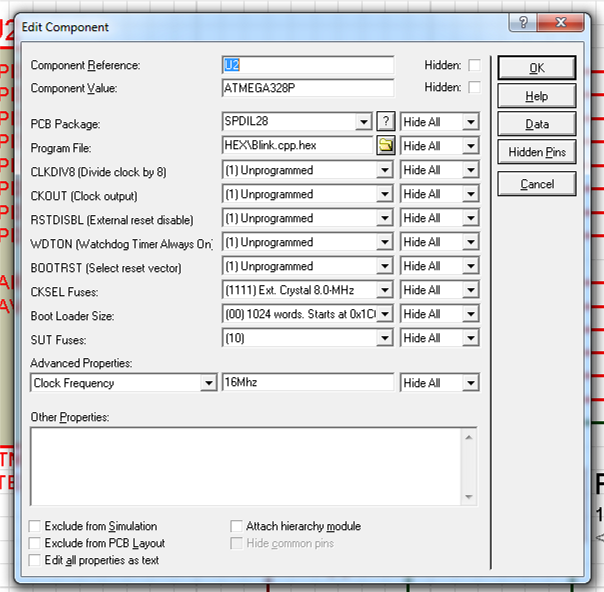


Deseu el fitxer. Per comprovar que tot ha anat bé, obriu l’Arduino, obriu el programa Blink i compileu-lo. Ara aneu a la carpeta HEX i comproveu que hi teniu els fitxers de la compilació.

Anem a fer la simulació amb ISIS. Primer s’ha de dibuixar l’esquema.



Hem d’editar les propietats del uControlador ATMEGA i a la pestanya PROGRAM FILE hi posem el programa blink.cpp.hex que trobarem a la carpeta HEX que abans hem creat. La resta de pestanyes les deixeu com les de la imatge de sota:



Ara ja podrem fer la simulació. Heu d’observar que el LED D1 connectat al pin D13 del PORT PB5 fa intermitent. [Per més informació sobre simulació d’Arduino amb Proteus visitar aquesta web.](http://www.huborarduino.com/programacion/curso-programacion)