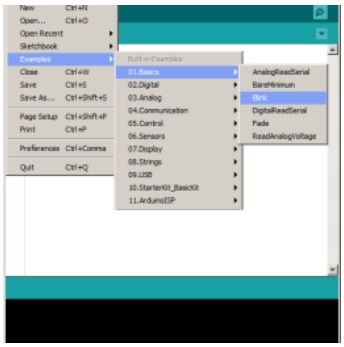
Arduino: primeiros passos para aprender e configurar.



Tutorial: Arduino - Primeiros Passos

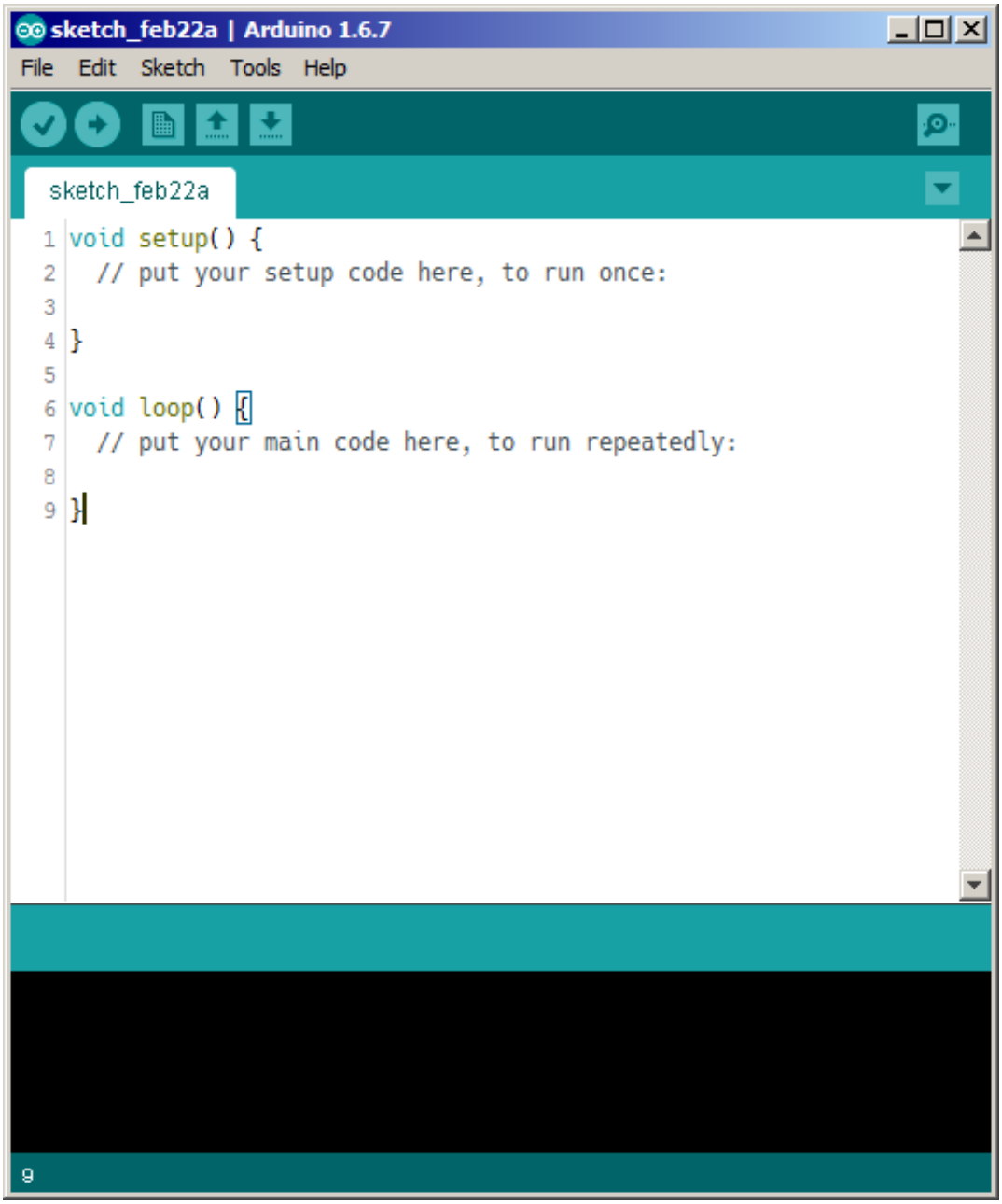
> *Neste tutorial, iremos aprender a configurar o Arduino IDE, carregar um programa de exemplo e modificá-lo.*

1. Introdução

No post "Aprenda Arduino com nosso Mega Kit", você conheceu cada item do Mega Kit e aprendeu a instalar o ambiente de programação, o Arduino IDE. Neste tutorial, iremos aprender a configurar a IDE, carregar um programa de exemplo e modificá-lo.

2. Estrutura básica

Após instalar o Arduino IDE, vá até o ícone no desktop ou busque onde você salvou o aplicativo e abra. Ao abrir, você verá algo como abaixo:



Arduino IDE

Na linha superior, podemos ver os menus e abaixo dela, alguns ícones. A área branca é onde se digita o programa e abaixo dela, temos uma região para diversos tipos de mensagens. Na parte braca, podemos ler:

```
void setup() {
  // put your setup code here,
```

```
1
2
3
4
5
6
7
8
```

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

Essa é a estrutura básica de um programa de Arduino. Esse programa está vazio, nele existem duas partes importantes: o que vem depois de *void setup()* entre **{}** (chaves) e também o que vem depois de *void loop()* entre **{}**.

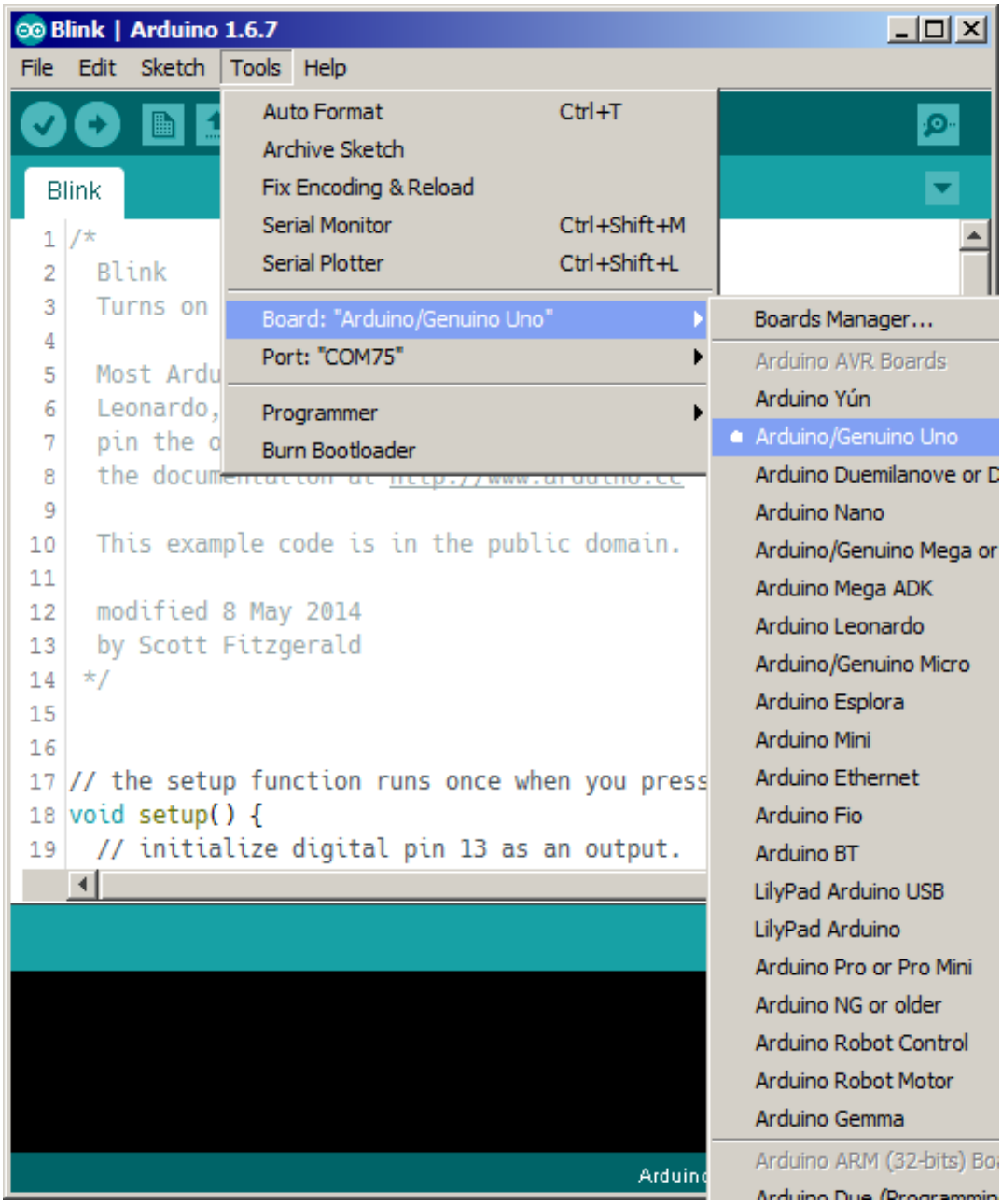
Um programa é um conjunto de comandos que são executados em sequência, como em uma receita de bolo.

Tudo que vem dentro da seção *setup* é executado somente uma vez quando ligamos o Arduino. *Setup* quer dizer “ajuste” e é nessa seção que colocamos toda a configuração para que as coisas funcionem depois. Todos comandos dentro de *loop* são executadas ordem sequencial até o final, e depois o primeiro comando volta a ser executado até desligarmos a placa.

3. Placa

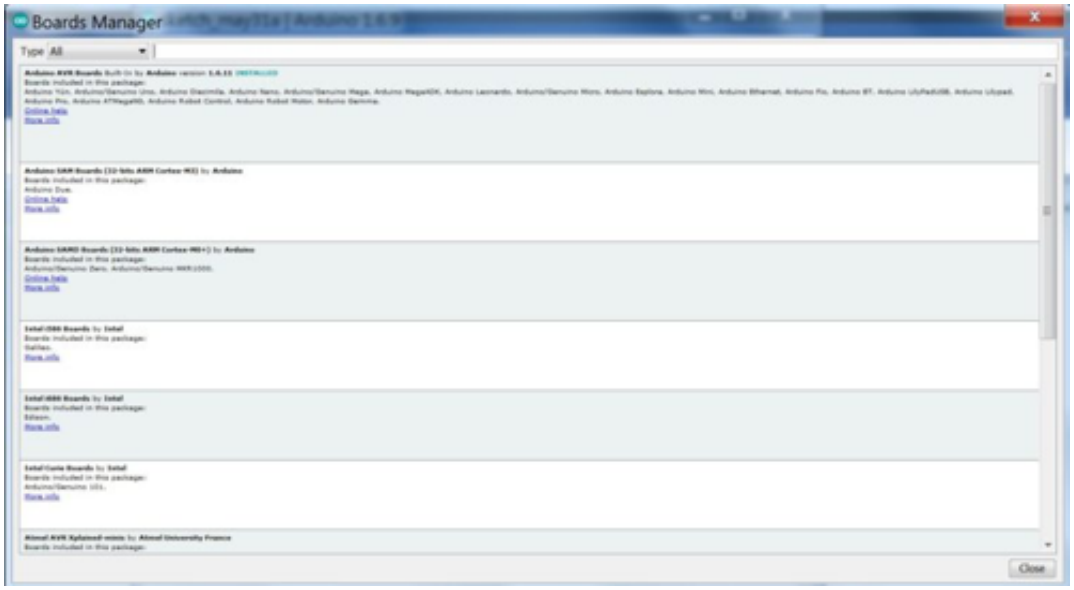
Agora, vamos conectar o Arduino UNO ao USB do computador e enviar o programa para a placa. É muito fácil!

Depois de conectado, precisamos dizer a IDE qual modelo de placa estamos usando. Vá na aba “Tools”/”Ferramentas”, “Board”/”Placa” e selecione “Arduino/Genuino Uno”.



Selecionando o modelo

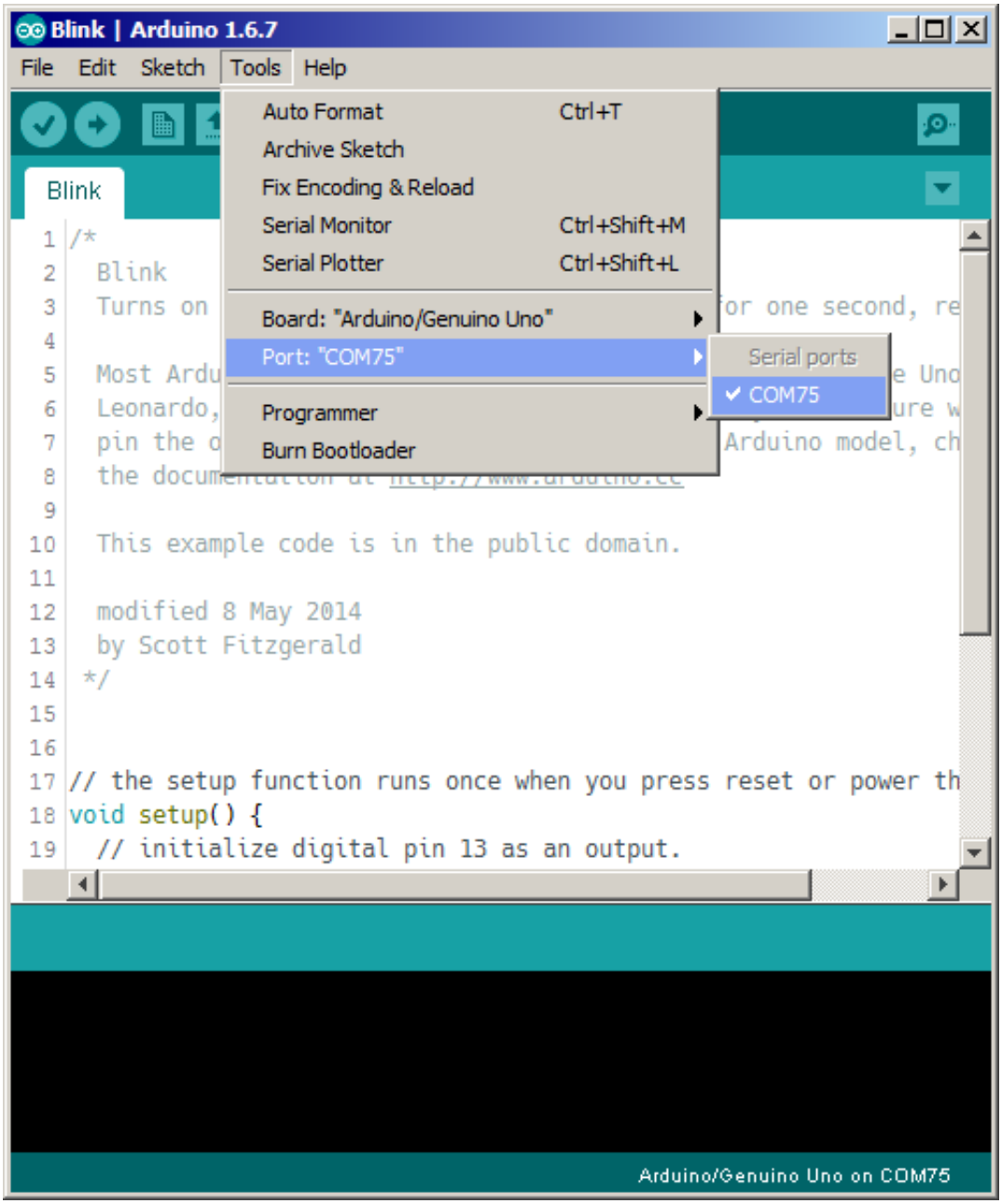
Caso ela não esteja nas opções, instale-a seguindo os comandos: “Tools”/”Ferramentas”, “Board”/”Placa”, “Boards Manager” e digite o nome da placa. Em seguida, clique em “Install”/”Instalar”.



“Board Manager”

4. Porta

Só precisamos fazer mais um ajuste e estaremos prontos para enviar o programa! Na mesma aba “Tools”/”Ferramentas”, temos a opção “Port”/”Porta”. No Windows, a porta aparecerá da seguinte forma: “COM + número”, como por exemplo, “COM75”. Selecione a porta onde sua placa está conectada e, caso o LED do Arduino não acenda, troque a porta no programa ou no computador (entrada USB).



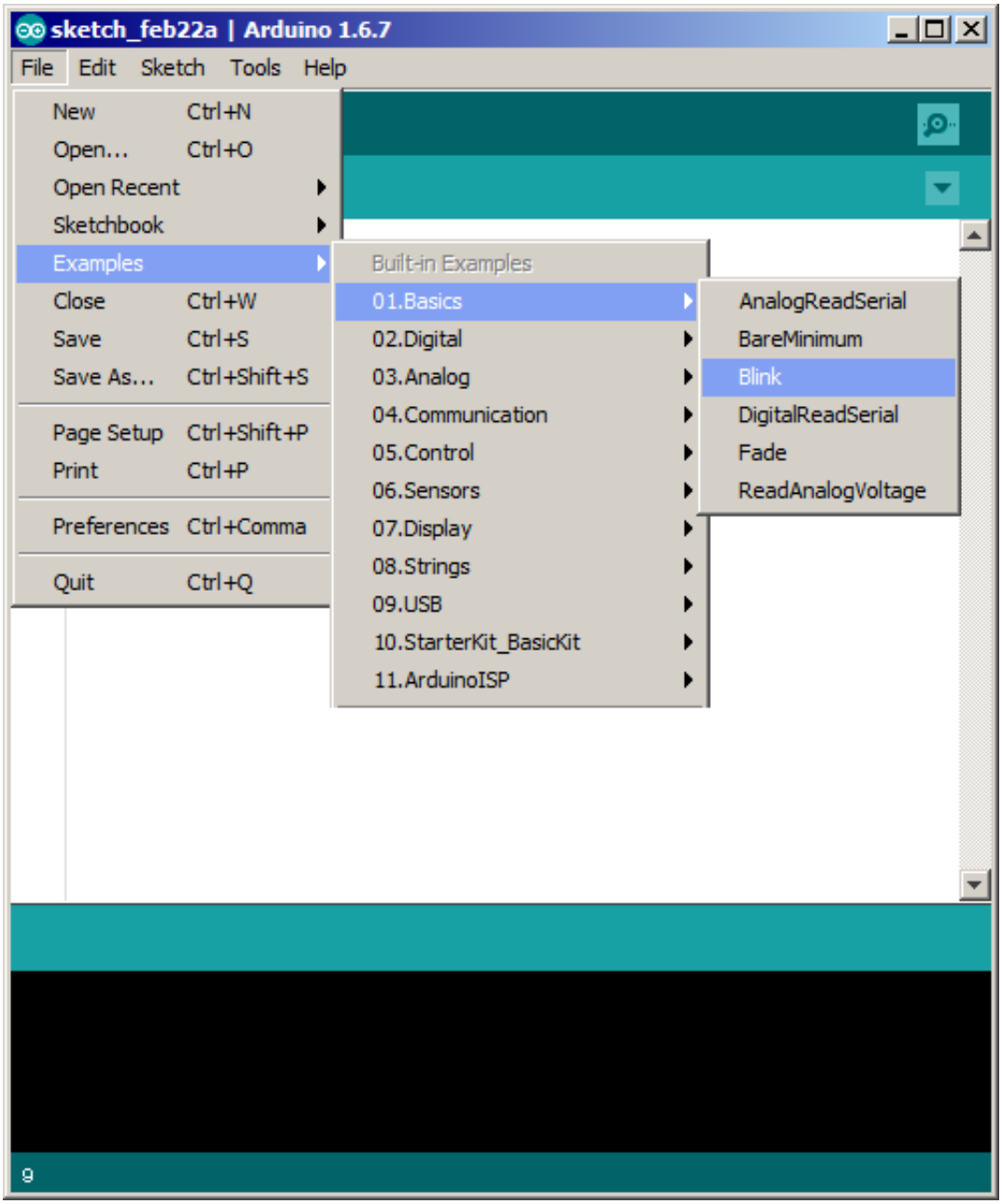
Selecionando a porta

No Mac e no Linux, teremos mais opções, mas geralmente a porta correta é a que se encontra no topo, elas aparecem dessa forma: “/dev/tty.usbmodemXXXX” ou “/dev/ttyUSBn”.

Lembrando que a opção “Porta”/”Port” só estará ativada se o Arduino estiver conectado na entrada USB.

5. Exemplo “Blink”

Abra “File”/Arquivo”, vá em “Examples”/”Exemplos”, depois em “Basics” e selecione “Blink”.



Abrindo o exemplo “Blink”

Aparecerá o seguinte programa:

```
/*
  Blink
```

```
1  /*
2    Blink
3    Turns on an LED on for one second, then off for one second, repeatedly.
4    Most Arduinos have an on-board LED you can control. On the UNO,
5  MEGA and ZERO
6    it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set
7  to
8    the correct LED pin independent of which board is used.
9    If you want to know what pin the on-board LED is connected to on your
10 Arduino model, check
11   the Technical Specs of your board  at
```

```
12  https://www.arduino.cc/en/Main/Products
13
14    This example code is in the public domain.
15    modified 8 May 2014
16    by Scott Fitzgerald
17
18    modified 2 Sep 2016
19    by Arturo Guadalupi
20
21    modified 8 Sep 2016
22    by Colby Newman
23  */
24  // the setup function runs once when you press reset or power the board
25  void setup() {
26    // initialize digital pin LED_BUILTIN as an output.
27    pinMode(LED_BUILTIN, OUTPUT);
28  }
29  // the loop function runs over and over again forever
30  void loop() {
31    digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the
32  voltage level)
33    delay(1000);                       // wait for a second
34    digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the
35  voltage LOW
36    delay(1000);                       // wait for a second
      }
```

Na primeira parte, temos uma anotação que explica o que o programa faz, quem fez e quando foi feito. Tudo que está entre os símbolos /* e */ é considerado uma anotação/comentário de mais de uma linha e não vale como programa. Ainda nesse bloco, os símbolos // também se referem a uma anotação/comentário, mas apenas de uma linha.

No *setup*, temos o comando *pinMode*. Esse é o primeiro comando (ou instrução) que o programa vai executar. Traduzindo para o português, ele significa: O *LED_BUILTIN* da placa é uma saída.

No Arduino UNO, *LED_BUILTIN* se refere ao pino 13 que está conectado ao LED da placa.

Cada linha de comando termina com o símbolo **;** (ponto e vírgula). Essa é a forma de dizer ao programa que chegamos ao fim do comando. Entre os símbolos **()** (parênteses) estão os parâmetros, ou seja, valores, do

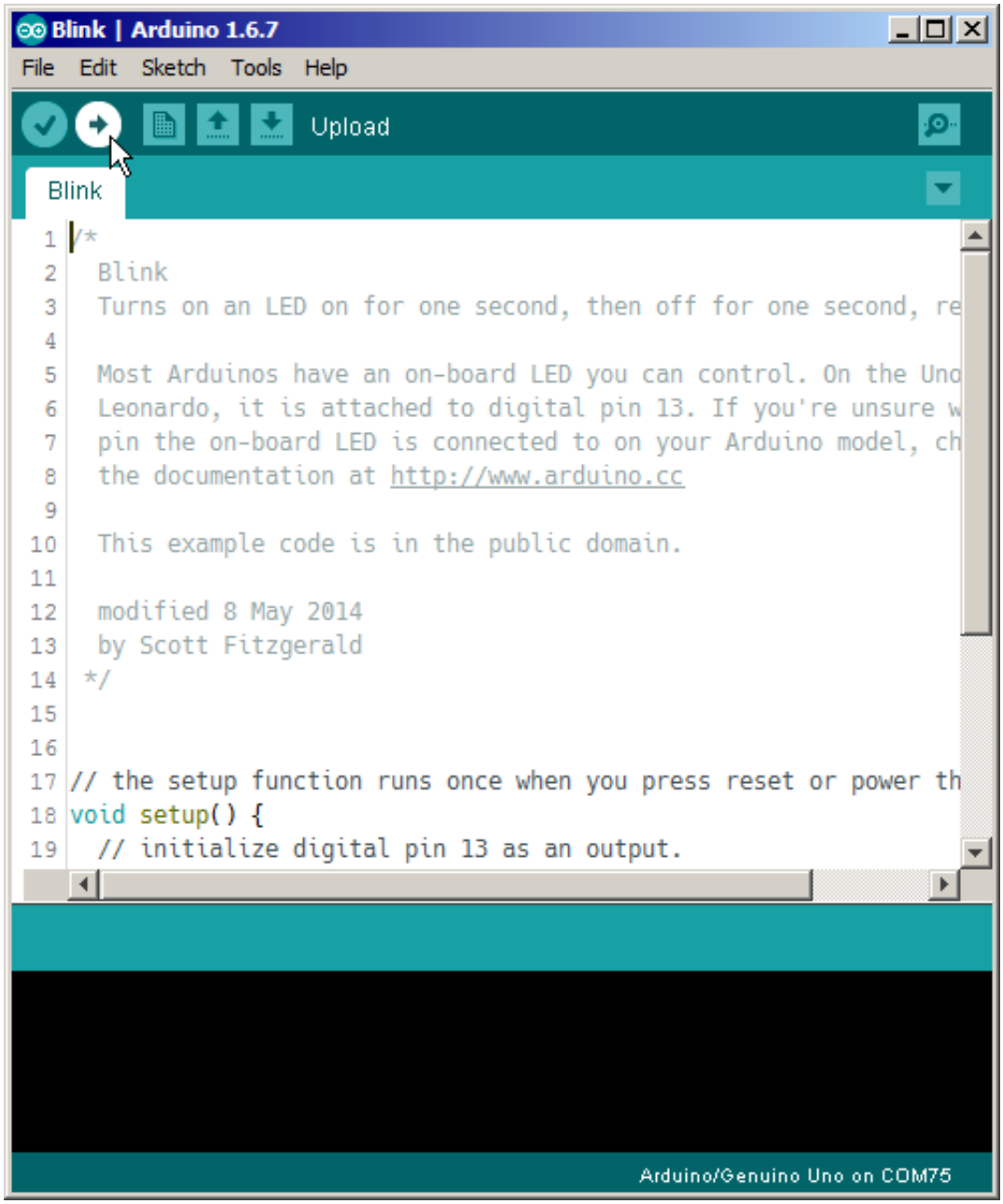
comando: *LED_BUILTIN* e *OUTPUT* (saída).

No *loop*, o comando *digitalWrite* é usado para ligar (*HIGH*) ou para desligar (*LOW*) a eletricidade, neste caso na saída 13, onde está ligado o LED. O comando *delay* é usado para fazer o Arduino aguardar por determinado tempo antes de executar o próximo comando. O valor 1000 é equivalente a um segundo, assim como 2000 faria esperar por dois segundos. Quando o último comando é executado, o programa volta para a primeira do *loop* até que a placa seja desligada.

Vamos experimentar mudar as durações dentro de delay e enviar novamente para o Arduino. Na primeira linha, aonde tem *delay(1000)*, escreva *delay(2000)* e depois envie com o botão “Upload”. Você vai notar que o LED fica aceso por mais tempo e apagado o mesmo tempo que ficava antes.

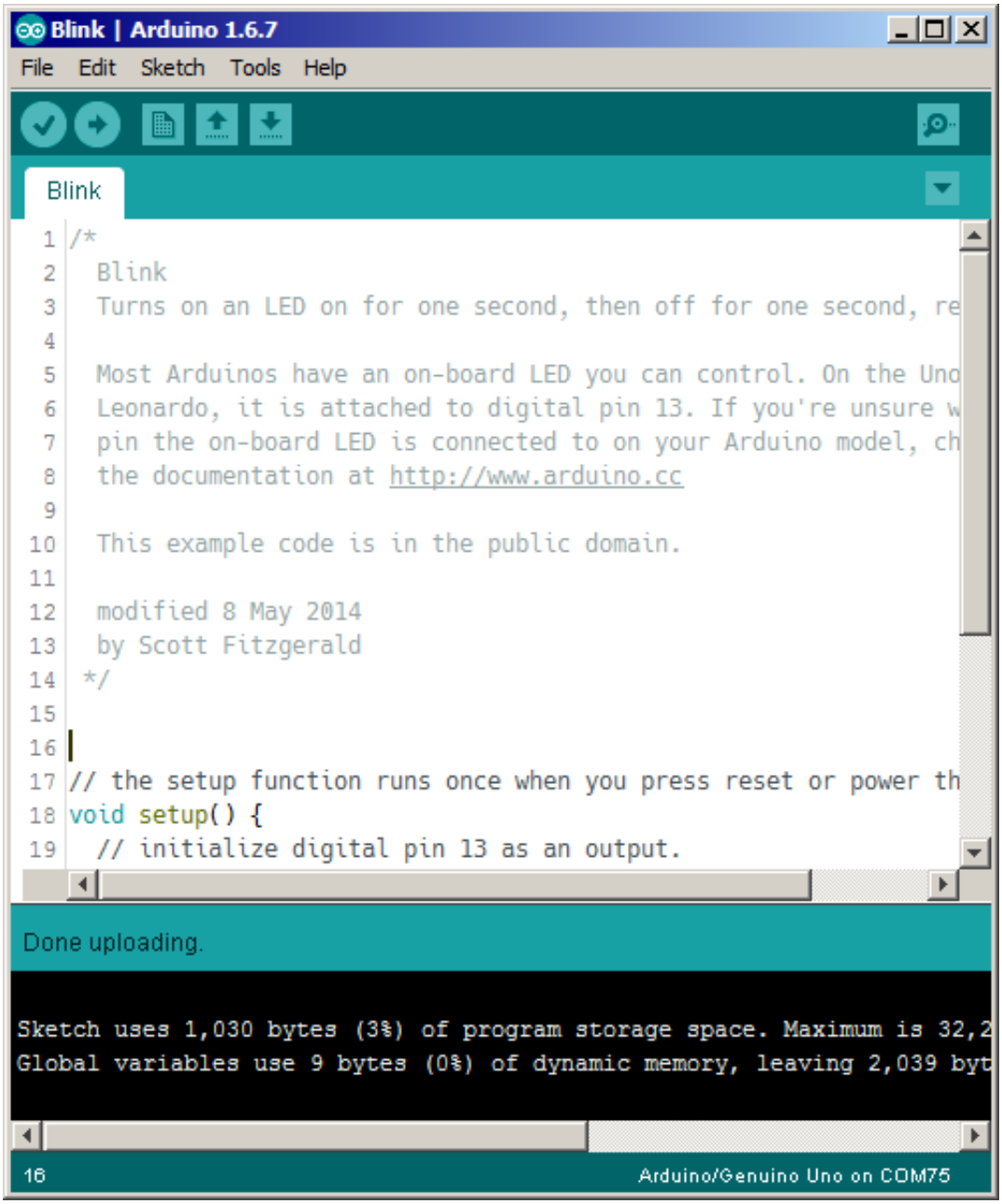
6. Upload

Agora, aperte o botão “Upload”, conforme a imagem a seguir:



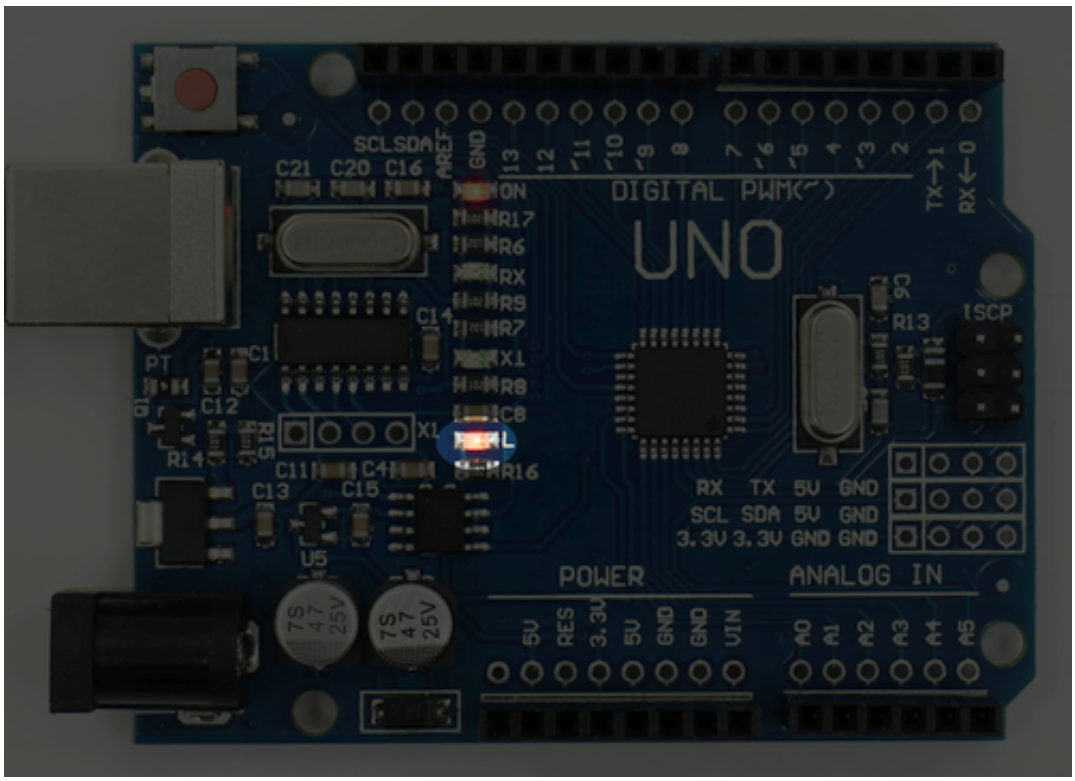
Botão Upload

A IDE vai automaticamente fazer o necessário para enviar o programa para a placa e, no final, aparecerá algo como na imagem abaixo. Note que agora existem mensagens na parte de baixo da IDE:



Programa enviado

O “Done uploading” significa que está tudo correto! Agora sua placa Arduino deve estar piscando um LED:



LED piscando na placa

7. Dicas

Caso você tenha algum problema, o programa não esteja funcionando ou a placa não esteja respodendo, tente essas dicas:

- Desconecte o cabo da fonte de energia e o USB e tente reconectá-los.
- Troque a porta (entrada USB).

No próximo tutorial, iremos aprender a fazer as “luzes” de uma ambulância usando o Arduino.