

Arduino Básico

Michael McRoberts

Novatec

Original English language edition published by Apress Inc., Copyright © 2010 by Apress, Inc.. Portuguese-language edition for Brazil copyright © 2011 by Novatec Editora. All rights reserved.

Edição original em Inglês publicada pela Apress Inc., Copyright © 2010 pela Apress, Inc.. Edição em Português para o Brasil copyright © 2011 pela Novatec Editora. Todos os direitos reservados.

© Novatec Editora Ltda. 2011.

Todos os direitos reservados e protegidos pela Lei 9610 de 19/02/1998.

É proibida a reprodução desta obra, mesmo parcial, por qualquer processo, sem prévia autorização, por escrito, do autor e da Editora.

Editor: Rubens Prates

Tradução: Rafael Zanolli

Revisão gramatical: Marta Almeida Sá

Revisão técnica: Edgard Damiani

Editoração eletrônica: Camila Kuwabata / Carolina Kuwabata

ISBN: 978-85-7522-274-4

Histórico de impressões:

Setembro/2011 Primeira edição

Novatec Editora Ltda.

Rua Luís Antônio dos Santos 110

02460-000 – São Paulo, SP – Brasil

Tel.: +55 11 2959-6529

Fax: +55 11 2950-8869

E-mail: novatec@novatec.com.br

Site: www.novatec.com.br

Twitter: twitter.com/novateceditora

Facebook: facebook.com/novatec

LinkedIn: linkedin.com/in/novatec

Dados Internacionais de Catalogação na Publicação (CIP)
(Câmara Brasileira do Livro, SP, Brasil)

McRoberts, Michael
 Arduino básico / Michael McRoberts ; [tradução
 Rafael Zanolli]. -- São Paulo : Novatec
 Editora, 2011.

 Título original: Beginning arduino
 ISBN 978-85-7522-274-4

 1. Arduino (Linguagem de programação para
 computadores) 2. Eletrônicos - Processamento de
 dados I. Título.

11-05551

CDD-005.133

Índices para catálogo sistemático:

1. Arduino : Linguagem de programação :
 Computadores : Processamento de dados
 005.133
 PRL20110824

Introdução

Desde que o Arduino Project teve início em 2005, mais de 150.000 placas Arduino foram vendidas em todo o mundo. O número de placas-clone não oficiais sem dúvida supera o de placas oficiais, assim, é provável que mais de 500 mil placas Arduino e suas variantes tenham sido vendidas. Sua popularidade não para de crescer, e cada vez mais pessoas percebem o incrível potencial desse maravilhoso projeto de fonte aberta para criar projetos interessantes rápida e facilmente, com uma curva de aprendizagem relativamente pequena.

A maior vantagem do Arduino sobre outras plataformas de desenvolvimento de microcontroladores é a facilidade de sua utilização; pessoas que não são da área técnica podem, rapidamente, aprender o básico e criar seus próprios projetos em um intervalo de tempo relativamente curto. Artistas, mais especificamente, parecem considerá-lo a forma perfeita de criar obras de arte interativas rapidamente, e sem conhecimento especializado em eletrônica. Há uma grande comunidade de pessoas utilizando Arduinos, compartilhando seus códigos e diagramas de circuito para que outros os copiem e modifiquem. A maioria dessa comunidade também está muito disposta a auxiliar outros desenvolvedores. Você descobrirá que o Fórum do Arduino é o melhor local para buscar por respostas rápidas.

Entretanto, apesar da enorme quantidade de informação disponível aos iniciantes na Internet, a maioria desses dados está espalhada em várias fontes, fazendo com que seja complicado rastrear as informações necessárias. É aqui que este livro entra. Em suas páginas, temos 50 projetos criados para conduzi-lo passo a passo pela programação de seu Arduino. Quando você utiliza pela primeira vez seu Arduino (ou qualquer outro gadget, nesse sentido), deseja conectá-lo, ligar um LED e ver as luzes piscando imediatamente. Não quer ter de, primeiro, ler páginas e páginas de teoria. Como autor, compreendo a empolgação dos iniciantes, e justamente por isso você, desde o início, conectará dispositivos ao seu Arduino, fará o upload do código e verá tudo em funcionamento. Esta é, acredito, a melhor forma de aprender sobre um tema, especialmente um como a computação física, que é o tópico do Arduino.

Como utilizar este livro

O livro inicia com uma introdução sobre o Arduino, explicando como montar o hardware, instalar o software, fazer o upload de seu primeiro sketch e garantir que seu Arduino e o software estejam funcionando corretamente. Depois, explico o IDE do Arduino e como utilizá-lo, antes que você mergulhe em projetos que irão de criações básicas a tópicos avançados. Cada projeto inicia com uma descrição de como preparar o hardware, com o código necessário para que ele funcione. Depois, explico mais detalhadamente como o hardware e o código funcionam. Tudo será explicado em passos claros e simples, com muitos diagramas e fotos que tornarão muito fácil garantir que você acompanhe o projeto corretamente.

Você encontrará alguns termos e conceitos no livro que talvez não entenda a princípio. Não se preocupe, eles se tornarão claros à medida que você trabalhar nos projetos.

De que você necessita

Para ser capaz de acompanhar os projetos deste livro, você necessitará de diversos componentes. Como eles podem ser caros, sugiro que inicie comprando apenas os componentes para os projetos dos primeiros capítulos (os componentes listados no início das páginas dos projetos). Conforme avança pelo livro, você poderá adquirir os componentes necessários para os projetos subsequentes.

Há alguns outros itens de que você necessitará ou que pode considerar úteis. Evidentemente, você deverá obter uma placa Arduino ou uma das muitas placas-clone disponíveis no mercado, como Freeduino, Seeduino (sim, é assim que se escreve), Boarduino, Sanguino, Roboduino ou qualquer uma das outras variantes “duino”. Todas são compatíveis com o IDE do Arduino, com seus Shields e com tudo o mais que você pode utilizar com uma placa Arduino oficial. Lembre-se de que o Arduino é um projeto de código aberto e, portanto, livre para criação de clones ou de outras variantes. Entretanto, se você deseja apoiar a equipe de desenvolvimento da placa Arduino original, adquira uma placa oficial de um dos distribuidores reconhecidos. Em setembro de 2010, a mais recente variação da placa Arduino era a Arduino Uno.

Você deverá ter acesso à Internet para fazer o download do IDE (Integrated Development Environment, ou Ambiente de Desenvolvimento Integrado) do Arduino — o software utilizado para escrever seu código —; e também precisará da Internet para fazer o download dos exemplos de código deste livro (caso você não queira digitá-los manualmente) e de todas as bibliotecas de código que possam ser necessárias para que seu projeto funcione.

Você também necessitará de uma mesa bem iluminada, ou de outra superfície plana para dispor seus componentes; essa mesa deve estar próxima de seu PC ou laptop para permitir o upload do código para o Arduino. Lembre-se de que você está trabalhando com eletricidade (ainda que uma corrente contínua de baixa voltagem); portanto, uma superfície de metal deverá ser coberta por um material não condutivo, como uma toalha ou papel, antes que você posicione seus materiais. Da mesma forma, será interessante ter, ainda que não seja essencial, um par de alicates para cortar fios, alicates de ponta fina, e um desencapador de fios. Um caderno e uma caneta também serão úteis para esboçar diagramas aproximados, detalhar conceitos e projetos etc.

Por fim, os elementos necessários mais importantes são o entusiasmo e a disposição em aprender. O Arduino foi projetado como uma forma simples e barata de você se envolver com eletrônica de microcontroladores, e nada é difícil demais, desde que você esteja disposto a, ao menos, tentar. Este livro irá ajudá-lo nessa jornada, e apresentará a você esse empolgante e criativo hobby.

O que exatamente é um Arduino?

A Wikipédia afirma que *“Um Arduino é um microcontrolador de placa única e um conjunto de software para programá-lo. O hardware consiste em um projeto simples de hardware livre para o controlador, com um processador Atmel AVR e suporte embutido de entrada/saída. O software consiste de uma linguagem de programação padrão e do bootloader que roda na placa.”*

Em termos práticos, um Arduino é um pequeno computador que você pode programar para processar entradas e saídas entre o dispositivo e os componentes externos conectados a ele (Figura 1.1). O Arduino é o que chamamos de plataforma de computação física ou embarcada, ou seja, um sistema que pode interagir com seu ambiente por meio de hardware e software. Por exemplo, um uso simples de um Arduino seria para acender uma luz por certo intervalo de tempo, digamos, 30 segundos, depois que um botão fosse pressionado. Nesse exemplo, o Arduino teria uma lâmpada e um botão conectados a ele. O Arduino aguardaria pacientemente até que o botão fosse pressionado; uma vez pressionado o botão, ele acenderia a lâmpada e iniciaria a contagem. Depois de contados 30 segundos, apagaria a lâmpada e aguardaria um novo apertar do botão. Você poderia utilizar essa configuração para controlar uma lâmpada em um closet, por exemplo.

Esse conceito poderia ser estendido pela conexão de um sensor, como um sensor de movimento PIR, para acender a lâmpada quando ele fosse disparado. Esses são alguns exemplos simples de como você poderia utilizar um Arduino.

O Arduino pode ser utilizado para desenvolver objetos interativos independentes, ou pode ser conectado a um computador, a uma rede, ou até mesmo à Internet para recuperar e enviar dados do Arduino e atuar sobre eles. Em outras palavras, ele pode enviar um conjunto de dados recebidos de alguns sensores para um site, dados estes que poderão, assim, ser exibidos na forma de um gráfico.

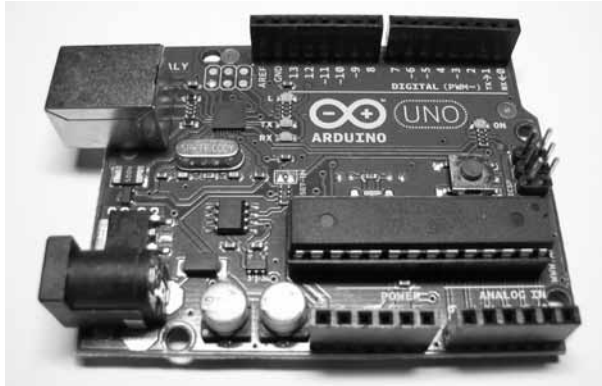


Figura 1.1 – Arduino Uno.

O Arduino pode ser conectado a LEDs, displays (mostradores) de matriz de pontos (Figura 1.2), botões, interruptores, motores, sensores de temperatura, sensores de pressão, sensores de distância, receptores GPS, módulos Ethernet ou qualquer outro dispositivo que emita dados ou possa ser controlado. Uma pesquisa na Internet revelará muitos outros projetos em que um Arduino foi utilizado para ler dados e controlar uma enorme quantidade de dispositivos.

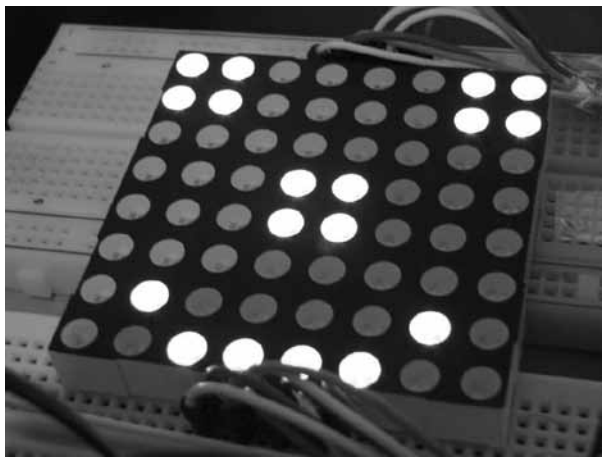


Figura 1.2 – Display de matriz de pontos controlado por um Arduino (imagem cortesia de Bruno Soares).

A placa do Arduino é composta de um microprocessador Atmel AVR, um cristal ou oscilador (relógio simples que envia pulsos de tempo em uma frequência especificada,

para permitir sua operação na velocidade correta) e um regulador linear de 5 volts. Dependendo do tipo de Arduino que você utiliza, ele também pode ter uma saída USB, que permite conectá-lo a um PC ou Mac para upload ou recuperação dos dados. A placa expõe os pinos de entrada/saída do microcontrolador, para que você possa conectá-los a outros circuitos ou sensores.

A mais recente placa do Arduino, a Uno, difere das versões prévias por não utilizar o chip FTDI, que conduz a USB para a serial. Em vez disso, ela utiliza um Atmega8U2, programado como um conversor USB para serial. Isso confere à placa muitas vantagens quando comparada à sua predecessora, a Duemilanove. Primeiro, o chip Atmega é muito mais barato que o chip FTDI, diminuindo o preço das placas. Segundo, e mais importante, ele permite que o chip USB tenha seu firmware atualizado, para que o Arduino seja exibido em seu PC como outro dispositivo, tal como um mouse ou joystick de jogos. Isso abre uma série de novas possibilidades para o Arduino. Infelizmente, a mudança para esse tipo de novo chip USB tornou muito mais difícil para fabricantes de clones criarem clones do Arduino Uno.

Para programar o Arduino (fazer com que ele faça o que você deseja) você utiliza o IDE do Arduino, um software livre no qual você escreve o código na linguagem que o Arduino compreende (baseada na linguagem C). O IDE permite que você escreva um *programa de computador*, que é um conjunto de instruções passo a passo, das quais você faz o upload para o Arduino. Seu Arduino, então, executará essas instruções, interagindo com o que estiver conectado a ele. No mundo do Arduino, programas são conhecidos como *sketches* (rascunho, ou esboço).

O hardware e o software do Arduino são ambos de fonte aberta, o que significa que o código, os esquemas, o projeto etc. podem ser utilizados livremente por qualquer pessoa e com qualquer propósito. Dessa forma, há muitas placas-clone e outras placas com base no Arduino disponíveis para compra, ou que podem ser criadas a partir de um diagrama. De fato, nada impede que você compre os componentes apropriados e crie seu próprio Arduino em uma matriz de pontos ou em sua PCB (Printed Circuit Board, placa de circuito impresso) feita em casa. A única ressalva que a equipe do Arduino impõe é que você não utilize a palavra “Arduino”. Esse nome é reservado à placa oficial. Daí a existência de nomes para as placas-clone como Freeduino, Roboduino etc.

Como os projetos são de fonte aberta, qualquer placa-clone é 100% compatível com o Arduino e, dessa forma, qualquer software, hardware, shield etc. também será 100% compatível com o Arduino genuíno.

O Arduino também pode ser estendido utilizando os *shields* (escudos), que são placas de circuito contendo outros dispositivos (por exemplo, receptores GPS, displays de LCD, módulos de Ethernet etc.), que você pode simplesmente conectar ao seu Arduino para

obter funcionalidades adicionais. Os shields também estendem os pinos até o topo de suas próprias placas de circuito, para que você continue a ter acesso a todos eles. Você não tem de utilizar um shield se não quiser; pode fazer exatamente o mesmo circuito utilizando uma protoboard, Stripboard, Veroboard, ou criando sua própria PCB. A maioria dos projetos deste livro foi feita utilizando circuitos em uma protoboard.

Há muitas variantes diferentes do Arduino. A versão mais recente é o Arduino Uno. A versão anterior, a popular Duemilanove (2009 em italiano), é a placa que você provavelmente encontrará na vasta maioria dos projetos para Arduino na Internet. Você também pode obter versões Mini, Nano e Bluetooth do Arduino. Outra nova adição aos produtos disponíveis é o Arduino Mega 2560, que oferece mais memória e um número maior de pinos de entrada/saída. As novas placas utilizam um novo bootloader, o Optiboot, que libera mais 1,5 kB de memória flash e permite uma inicialização mais rápida.

Talvez o Arduino mais versátil, e daí o mais popular, seja o Arduino Uno (ou seu predecessor, o Duemilanove). Isso ocorre porque ele utiliza um chip padrão de 28 pinos, ligado a um soquete de circuito integrado (CI). A beleza desse sistema é que, se você criar alguma coisa com um Arduino e depois quiser transformá-la em algo permanente, em vez de utilizar uma placa Arduino relativamente cara, poderá simplesmente retirar o chip da placa e colocá-lo em sua própria placa de circuito, em seu dispositivo personalizado. Dessa forma, você tem um dispositivo embarcado personalizado, o que é muito bacana.

Então, com um pouco mais de investimento, você pode substituir o chip AVR em seu Arduino por um novo. Note que o chip deve ser pré-programado com o Arduino Bootloader (software programado no chip para habilitá-lo a utilizar o IDE do Arduino), mas você pode comprar um AVR Programmer para gravar o bootloader você mesmo, ou comprar um chip já programado; a maioria dos fornecedores de componentes para o Arduino oferece essas opções. Também é possível programar um chip utilizando um segundo Arduino; não é difícil encontrar instruções online nesse sentido.

Se você fizer uma pesquisa online por “Arduino”, ficará surpreso com o grande número de sites dedicados ao Arduino, ou que apresentam projetos interessantes criados com ele. O Arduino é um dispositivo incrível, e possibilitará que você crie de tudo, desde obras de arte interativas (Figura 1.3) até robôs. Com um pouco de entusiasmo para aprender como programar um Arduino e fazê-lo interagir com outros componentes, assim como tendo um pouco de imaginação, você poderá construir tudo que quiser.

Este livro fornecerá as habilidades necessárias para que você dê seus primeiros passos nesse empolgante e criativo hobby. Agora que você já sabe o que é o Arduino, vamos ligá-lo ao seu computador e colocá-lo para funcionar.



Figura 1.3 – Instalação de arte Anthros, por Richard V. Gilbank, controlada utilizando um Arduino.

Primeiros passos

Esta seção explicará como configurar seu Arduino e o IDE pela primeira vez. São dadas instruções para Windows e Mac (executando o OSX 10.3.9 ou mais recente). Caso você esteja utilizando Linux, consulte as instruções Getting Started no site do Arduino, em www.arduino.cc/playground/Learning/Linux. Também suponho que você esteja utilizando um Arduino Uno. Se tiver uma placa diferente, como a Duemilanove (Figura 1.4), consulte a página correspondente no guia Getting Started no site do Arduino.

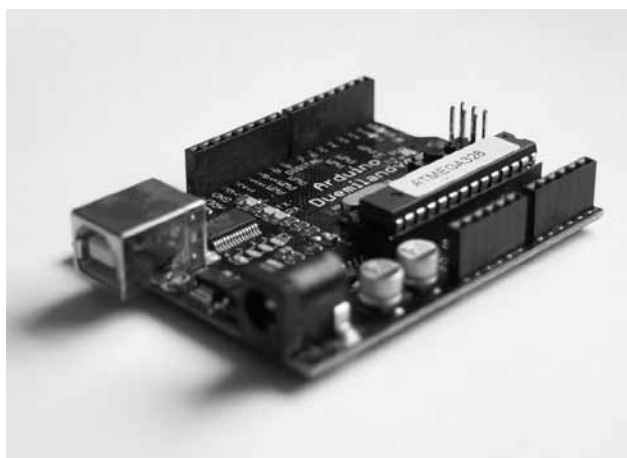


Figura 1.4 – Arduino Duemilanove (imagem cortesia de Snorpey).

Você também necessitará de um cabo USB (com plug do tipo A para B), do mesmo tipo utilizado na maioria das impressoras USB modernas. Se você tiver um Arduino Nano, necessitará de um cabo USB A para Mini-B. Não conecte ainda o Arduino, espere até que eu lhe diga para fazê-lo.

A seguir, faça o download do IDE do Arduino, o software que você utilizará para escrever seus programas (ou sketches) e, depois, fazer o upload deles para a placa. Para o IDE mais recente, acesse a página de download do Arduino, em <http://arduino.cc/en/Main/Software>, e obtenha a versão apropriada para seu sistema operacional.

Instalação no Windows XP

Assim que você tiver feito o download do IDE mais recente, descompacte o arquivo e clique duas vezes na pasta descompactada para abri-la. Você verá os arquivos do Arduino e suas subpastas. A seguir, conecte seu Arduino utilizando o cabo USB e verifique se o LED verde de energia (PWR) acende. O Windows lhe avisará “Novo hardware encontrado: Arduino” e o Assistente de instalação de novo hardware será exibido. Clique Avançar e o Windows tentará carregar os drivers. Esse processo falhará, mas não se preocupe, isso é normal.

Em seguida, clique com o botão direito no ícone Meu Computador em seu Desktop e escolha Gerenciar. A janela de Gerenciamento do Computador abrirá. Agora, desça até o Gerenciador de Dispositivos, na lista Ferramentas do Sistema, e clique nele. Na janela, à direita, você verá uma lista de seus dispositivos. O Arduino Uno fará parte dessa lista com um ponto de exclamação amarelo sobre ele, para indicar que o dispositivo não foi instalado corretamente. Clique com o botão direito sobre esse ícone e escolha “Atualizar Driver”. Escolha “Não, não agora” na primeira página e clique em Avançar. Então, escolha “Instalar de uma lista ou local específico (avançado)” e clique novamente em Avançar. Agora clique em “Incluir este local na pesquisa” e clique em Procurar. Navegue até a pasta Drivers do IDE descompactado do Arduino e clique em Avançar. O Windows instalará o driver e você poderá, então, clicar no botão Concluir.

O Arduino Uno agora será exibido na opção Portas (COM & LPT), na lista de dispositivos, e mostrará o número de porta atribuído a ele (por exemplo, COM6). Para abrir o IDE, clique duas vezes no ícone do Arduino em sua pasta.

Instalação no Windows 7 e Vista

Assim que você tiver feito o download do IDE mais recente, descompacte o arquivo e clique duas vezes na pasta descompactada para abri-la. Você verá os arquivos do Arduino e suas subpastas. A seguir, conecte seu Arduino utilizando o cabo USB e verifique se o LED verde de energia (PWR) acende. O Windows tentará instalar automaticamente os drivers para o Arduino Uno e falhará. Isso é normal, por isso, não se preocupe.

Clique no botão **Iniciar** do Windows e depois em **Painel de Controle**. Agora, clique em **Sistema e Segurança**, depois em **Sistema** e, então, em **Gerenciador de Dispositivos**, na lista, no lado esquerdo. O Arduino fará parte da lista como um dispositivo com um ponto de exclamação amarelo sobre ele, para mostrar que não foi instalado corretamente. Clique com o botão direito no Arduino Uno e escolha **Atualizar driver**.

Depois, escolha **Procurar software de driver no computador** e, na janela seguinte, clique no botão **Procurar**. Navegue até a pasta **Drivers** dentro da pasta **Arduino** que você descompactou antes e clique em **OK**, e depois em **Avançar**. O Windows tentará instalar o driver. Uma caixa de segurança do Windows será exibida afirmando que o “Windows não pode verificar o editor deste software de driver”. Clique em **Instalar driver mesmo assim**. A janela de instalação fará o necessário. Se tudo der certo, você terá outra janela dizendo que o software de driver foi atualizado com êxito. Finalmente, clique em **Fechar**. Para abrir o IDE, clique duas vezes no ícone do Arduino em sua pasta.

Instalação no Mac OSX

Faça o download do arquivo de imagem de disco (.dmg) mais recente para o IDE. Abra o arquivo .dmg (Figura 1.5).

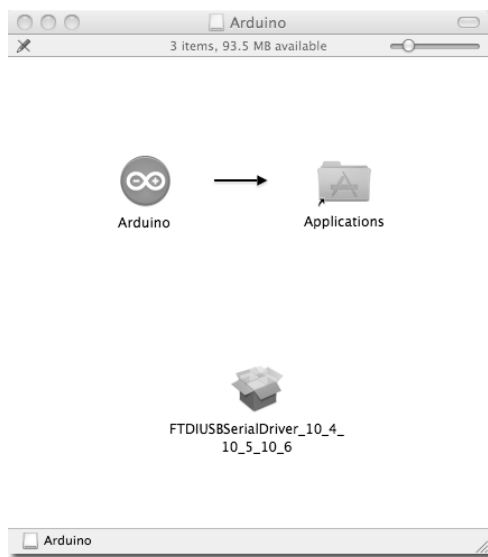


Figura 1.5 – Arquivo .dmg do Arduino aberto no OSX.

Arraste o ícone do Arduino sobre a pasta **Aplicações**, soltando-o nela. Se você estiver utilizando um Arduino mais antigo, como o Duemilanove, terá de instalar o driver FTDI USB Serial. Clique duas vezes no ícone do pacote e siga as instruções para fazê-lo. Para o Uno e o Mega 2560, não será necessário instalar nenhum driver.

Para abrir o IDE, vá até a pasta **Aplicações** e clique no ícone do Arduino.

Seleção de placa e porta

Assim que você abrir o IDE, ele terá visual semelhante ao da figura 1.6.

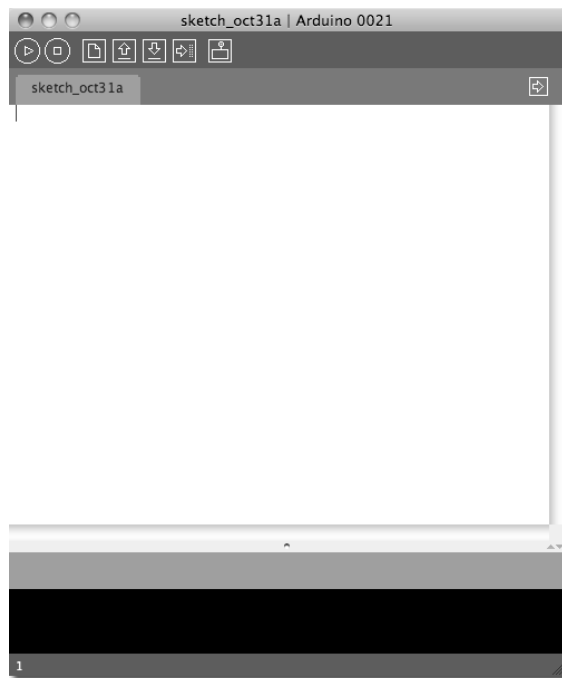


Figura 1.6 – IDE do Arduino quando aberto pela primeira vez.

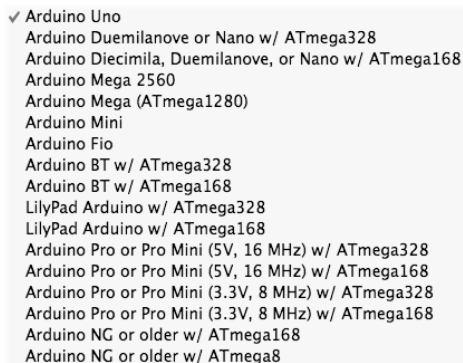
Agora, vá ao menu e clique em Tools. Então, clique em Board (Figura 1.7).



Figura 1.7 – Menu Tools do Arduino.

Você será apresentado a uma lista de placas (Figura 1.8). Se você tiver uma Uno, escolha de acordo. Se tiver uma Duemilanove, ou outra variante do Arduino, escolha a opção apropriada a partir dessa lista.

A seguir, clique no menu Tools novamente. Em seguida, clique em Serial Port e escolha na lista a porta apropriada para seu Arduino (Figura 1.9). Você agora está pronto para fazer o upload de um sketch de exemplo e testar se a instalação deu certo.

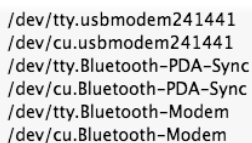


```

✓ Arduino Uno
  Arduino Duemilanove or Nano w/ ATmega328
  Arduino Diecimila, Duemilanove, or Nano w/ ATmega168
  Arduino Mega 2560
  Arduino Mega (ATmega1280)
  Arduino Mini
  Arduino Fio
  Arduino BT w/ ATmega328
  Arduino BT w/ ATmega168
  LilyPad Arduino w/ ATmega328
  LilyPad Arduino w/ ATmega168
  Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
  Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
  Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
  Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
  Arduino NG or older w/ ATmega168
  Arduino NG or older w/ ATmega8

```

Figura 1.8 – Menu Boards do Arduino.



```

/dev/tty.usbmodem241441
/dev/cu.usbmodem241441
/dev/tty.Bluetooth-PDA-Sync
/dev/cu.Bluetooth-PDA-Sync
/dev/tty.Bluetooth-Modem
/dev/cu.Bluetooth-Modem

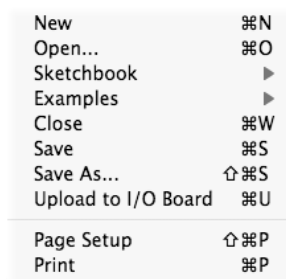
```

Figura 1.9 –Lista Serial Port.

Upload de seu primeiro sketch

Agora que você instalou os drivers e o IDE, e tem a placa e a porta correta selecionadas, é o momento de fazer o upload de um sketch de exemplo para o Arduino e testar se tudo está funcionando corretamente, antes de avançar ao seu primeiro projeto.

Primeiro, clique no menu File (Figura 1.10) e depois em Examples.



```

New           ⌘N
Open...       ⌘O
Sketchbook    ►
Examples      ►
Close         ⌘W
Save          ⌘S
Save As...    ⇧⌘S
Upload to I/O Board ⌘U

Page Setup    ⇧⌘P
Print         ⌘P

```

Figura 1.10 – Menu File.

Você será apresentado a uma enorme lista de exemplos. Vamos experimentar um bem simples. Clique em Basics e depois em Blink (Figura 1.11). O sketch Blink será carregado no IDE.

Depois, clique no botão Upload (o sexto botão a partir da esquerda) e olhe para seu Arduino. (Se você tem um Arduino Mini, NG ou outra placa, pode ser necessário apertar o botão reset na placa antes de apertar o botão Upload). As luzes RX e TX devem começar a piscar para mostrar que os dados estão sendo transmitidos de seu

computador para a placa. Assim que o upload do sketch tiver sido feito, as palavras “Done uploading” (upload completo) serão exibidas na barra de status do IDE e as luzes RX e TX pararão de piscar.

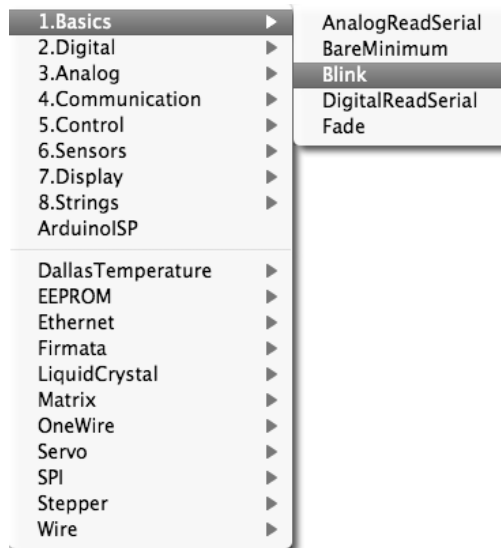


Figura 1.11 – Menu Examples.

Depois de alguns segundos, você deverá ver que o LED do pino 13 (o pequenino LED ao lado dos LEDs RX e TX) começa a piscar, acendendo e apagando em intervalos de um segundo. Se isso estiver acontecendo, você conectou seu Arduino, instalou os drivers e o software, e fez o upload de um sketch de exemplo com êxito. O sketch Blink é um sketch muito simples que pisca o LED 13 (Figura 1.12), o pequenino LED verde (ou laranja) soldado na placa (e que também está conectado ao pino digital 13 do microcontrolador).

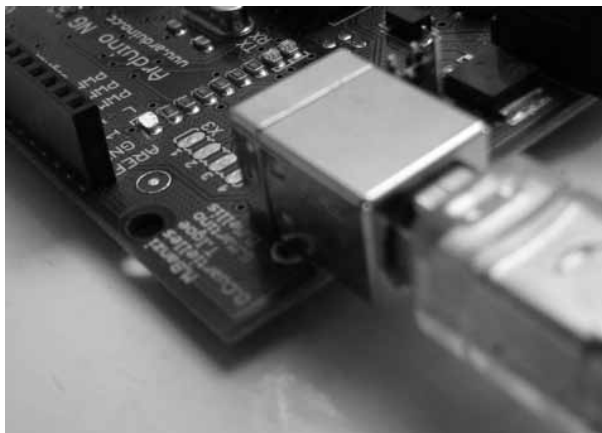


Figura 1.12 – LED 13 piscando.

Antes de avançar ao Projeto 1, vamos analisar o IDE do Arduino. Explicarei cada parte do programa.

IDE do Arduino

Quando você abrir o IDE do Arduino, verá algo semelhante à figura 1.13. Caso esteja utilizando Windows ou Linux, pode haver algumas pequenas diferenças, mas o IDE é basicamente o mesmo, independentemente do sistema operacional utilizado.

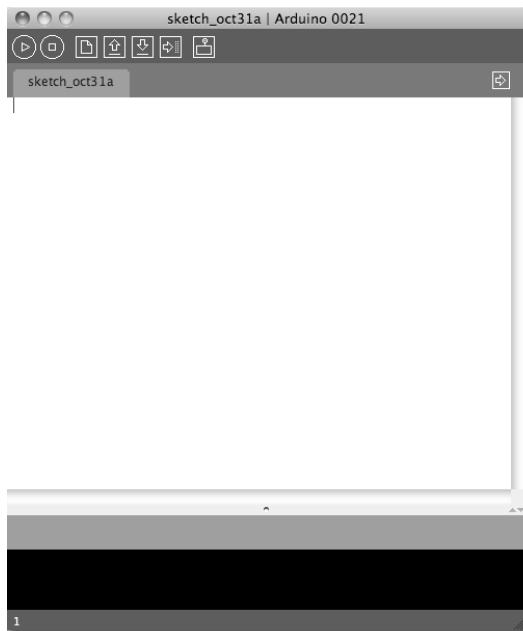


Figura 1.13 – Visual do IDE quando a aplicação abre.

O IDE é dividido em três partes: a Toolbar no topo, o código ou a Sketch Window no centro, e a janela de mensagens na base. A Toolbar consiste de sete botões. Sob a Toolbar há uma guia, ou um conjunto de guias, com o nome do arquivo do sketch. Também há um botão posicionado no lado direito.

Ao longo do topo há a barra de menus, com os itens File, Edit, Sketch, Tools e Help. Os botões na Toolbar (Figura 1.14) fornecem acesso conveniente às funções mais utilizadas dentro desses menus.



Figura 1.14 – Toolbar.

Os botões da Toolbar e suas funções estão listados na tabela 1.1.

Tabela 1.1 – Funções dos botões da Toolbar

Verify/Compile	Verifica se há erros no código
Stop	Interrompe o monitor serial, ou desmarca os outros botões
New	Cria um sketch em branco
Open	Mostra uma lista de sketches, em seu Sketchbook, para abrir
Save	Salva o sketch atual em seu sketchbook
Upload	Faz o upload do sketch atual para o Arduino
Serial Monitor	Exibe os dados seriais enviados do Arduino

O botão **Verify/Compile** é utilizado para verificar se seu código está correto e livre de erros, antes que você faça o upload para a placa do Arduino.

O botão **Stop** interrompe a operação do monitor serial e também desmarca outros botões selecionados. Enquanto o monitor serial está em operação, você pode pressionar o botão **Stop** para obter um instantâneo dos dados seriais, até então enviados, e analisá-los. Isso é especialmente útil se você estiver enviando dados para o monitor serial mais rápido do que puder lê-los.

O botão **New** cria um novo sketch em branco, pronto para receber seu código. O IDE pede que você digite um nome e forneça uma localização para seu sketch (tente utilizar a localização padrão, se possível) e, então, oferece um sketch em branco, pronto para receber seu código. A guia no topo do sketch exibe o nome que você deu a ele.

O botão **Open** apresenta uma lista de sketches armazenados em seu sketchbook, bem como uma lista de sketches de exemplo que você pode experimentar com diversos periféricos. Os sketches de exemplo são muito valiosos para iniciantes, por formarem uma base a partir da qual poderão construir seus próprios sketches. Abra o sketch apropriado para o dispositivo com o qual está conectando, e modifique o código conforme sua necessidade.

O botão **Save** salva o código na janela de sketch para seu arquivo. Uma vez feito isso, você receberá uma mensagem “Done saving” (“Gravação completa”) na base de sua janela de código, indicando que o sketch foi salvo.

O botão **Upload to I/O Board** faz o upload do código contido na janela de sketch atual para seu Arduino. Certifique-se de que você selecionou corretamente a placa e a porta (no menu **Tools**) antes do upload. É essencial que você salve seu sketch antes de fazer o upload para sua placa, caso ocorra algum erro que trave seu sistema ou o IDE. Também é aconselhável pressionar o botão **Verify/Compile** antes do upload para garantir que não existam erros que terão de ser depurados primeiro.

O monitor serial é uma ferramenta muito útil, especialmente para depuração de código. O monitor exibe os dados seriais enviados de seu Arduino (USB ou placa serial). Você também pode enviar dados de volta ao Arduino utilizando o monitor serial. Clique no botão **Serial Monitor** para abrir uma janela como a da figura 1.15.

No canto inferior direito você pode ver a taxa de transmissão (Baud Rate) na qual os dados seriais devem ser enviados de/para o Arduino. A taxa de transmissão é a taxa por segundo em que alterações de estado ou bits (dados) são enviados de/para a placa. A configuração padrão é 9600 baud, o que significa que, se você quisesse enviar um livro pela linha de comunicação serial (nesse caso, seu cabo USB), 1.200 letras ou símbolos de texto seriam enviados por segundo ($9600 \text{ bits} / 8 \text{ bits por caractere} = 1.200 \text{ bytes ou caracteres}$). Bits e bytes serão explicados futuramente.

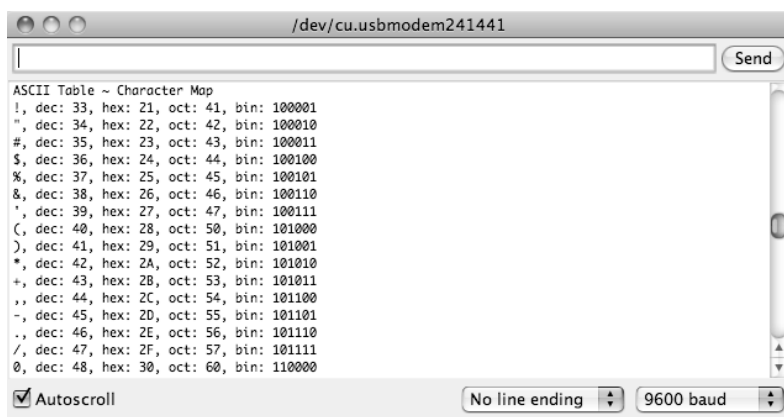


Figura 1.15 – Janela serial em uso.

No topo, há uma caixa de texto em branco, para que você digite o texto a ser enviado de volta para o Arduino, e um botão **Send**, para enviar o texto. Note que o monitor serial não pode receber nenhum dado serial, a menos que você tenha preparado o código em seu sketch para que isso ocorra. Da mesma forma, o Arduino não receberá nenhum dado, a menos que você o tenha codificado para tanto.

Por fim, a área central da janela é o local em que seus dados seriais serão exibidos. Na imagem anterior, o Arduino está executando o sketch `ASCIITable` (do exemplo `Communications`). Esse programa faz a saída de caracteres ASCII do Arduino pela serial (o cabo USB) para o PC, onde o monitor serial os exibe.

Para iniciar o monitor serial, pressione o botão **Serial Monitor**. Para interrompê-lo, pressione o botão **Stop**. Em um Mac ou Linux, a placa do Arduino reiniciará sozinha (reexecutando o código desde o início), quando você clicar no botão **Serial Monitor**.

Assim que você estiver capacitado para a comunicação serial de, e para, o Arduino, poderá utilizar outros programas, como Processing, Flash, MaxMSP etc. para realizar a comunicação entre o Arduino e seu PC. Você utilizará o monitor serial futuramente, ao ler dados de sensores e fazer com que o Arduino envie esses dados para o monitor serial em um formato legível para seres humanos.

Na parte inferior da janela do IDE você verá mensagens de erro (em texto vermelho) que o IDE poderá exibir quando tentar se conectar com sua placa, realizar o upload do código ou verificá-lo. No canto inferior esquerdo do IDE você verá um número. Essa é a localização atual do cursor dentro do programa. Se você tiver código em sua janela, e mover o cursor para baixo (utilizando a tecla ↓ de seu teclado), verá que esse número se eleva. Isso é útil para encontrar bugs destacados por mensagens de erro.

No topo da janela do IDE (ou no topo de sua tela, se você estiver utilizando um Mac), você verá os diversos menus em que pode clicar para acessar mais itens de menu (Figura 1.16).

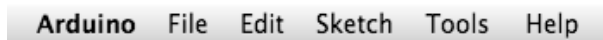


Figura 1.16 – Menus do IDE.

O primeiro menu é o menu **Arduino** (Figura 1.17). A opção **About Arduino** mostra o número da versão atual, uma lista das pessoas envolvidas na criação desse fantástico dispositivo e algumas informações adicionais.



Figura 1.17 – Menu Arduino.

Sob essa primeira opção, há a opção **Preferences**, que exibe a janela de preferências em que você pode alterar várias opções do IDE, como a localização padrão de seu sketchbook etc. A opção **Quit** encerra o programa.

O menu **File** (Figura 1.18) é o local em que você pode acessar opções para criar um novo sketch, analisar sketches armazenados em seu sketchbook (bem como sketches de exemplo), salvar seu sketch ou utilizar a opção **Save As** para dar a ele um novo nome, fazer o upload de seu sketch para a placa de entrada/saída (o Arduino), ou imprimir seu código.

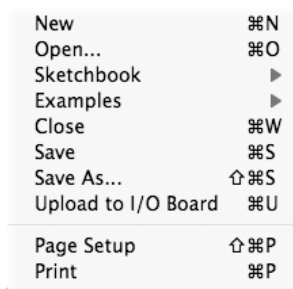


Figura 1.18 – Menu File.

O menu **Edit** (Figura 1.19) oferece opções para que você recorte, copie e cole seções do código. Você também pode selecionar todo o código (**Select All**) ou localizar (**Find**) certas palavras ou frases dentro do código. As opções **Undo** (desfazer) ou **Redo** (refazer) são úteis quando você comete algum erro.



Figura 1.19 – Menu Edit.

O menu **Sketch** (Figura 1.20) contém as funções **Verify/Compile** e outras funções úteis, incluindo a opção **Import Library**, que apresenta uma lista das bibliotecas disponíveis armazenadas em sua pasta **libraries**.



Figura 1.20 – Menu Sketch.

Uma *biblioteca* é um conjunto de código que você pode incluir em seu sketch para aprimorar a funcionalidade de seu projeto. Isso é uma forma de impedir que você tenha de reinventar a roda, recriando algo que já foi feito; em vez disso, você pode reutilizar código escrito por outra pessoa para diversos componentes de hardware comuns. Por exemplo, a biblioteca **Stepper** é um conjunto de funções para controlar

um motor de passo; assim, incluindo essa biblioteca em seu sketch, você pode reutilizar o código diversas vezes em projetos diferentes. Dessa forma, você também pode ocultar do usuário as partes mais complicadas de seu código. Futuramente, falarei com mais detalhes sobre o uso de bibliotecas.

A opção **Show Sketch Folder** abre a pasta em que o sketch está armazenado. A opção **Add File** permite que você adicione outro arquivo de fonte ao seu sketch, permitindo a divisão de sketches maiores em arquivos menores para, depois, adicioná-los ao sketch principal.

O menu **Tools** (Figura 1.21) oferece diversas opções. Você pode selecionar a placa e a porta serial como fez quando configurou o Arduino pela primeira vez. A função **Auto Format** formata seu código para melhorar sua visualização. A opção **Copy for Forum** copia o código da janela de sketch, mas em um formato que, quando colado no fórum do Arduino (e na maioria dos outros fóruns), seja exibido da mesma forma que no IDE, acompanhado da colorização da sintaxe etc. A opção **Archive Sketch** permite que você comprima seu sketch em um arquivo ZIP, perguntando onde você deseja armazená-lo. Por fim, a opção **Burn Bootloader** grava o Arduino Bootloader (o trecho de código do chip para torná-lo compatível com o IDE do Arduino) no chip. Essa opção pode apenas ser utilizada se você tiver um programador AVR e se tiver substituído o chip em seu Arduino, ou caso tenha comprado chips em branco para utilizar em seu próprio projeto embarcado. A menos que pretenda gravar muitos chips, geralmente é mais barato simplesmente comprar um chip ATmega (Figura 1.22) com o Arduino Bootloader pré-programado. Muitas lojas online oferecem chips pré-programados a ótimos preços.



Figura 1.21 – Menu Tools.

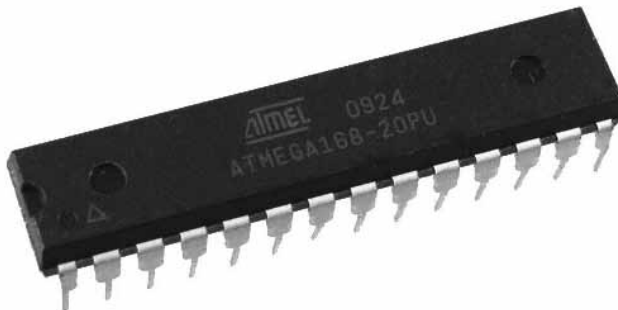


Figura 1.22 – Chip Atmel ATmega, o coração de seu Arduino (imagem cortesia da Earthshine Electronics).

O último menu, **Help**, é o local em que você encontra mais informações sobre o IDE, ou links para as páginas de referência do site do Arduino e outras páginas úteis.

O IDE do Arduino é bem simples, e você aprenderá como utilizá-lo com rapidez e facilidade à medida que avança nos projetos. Conforme você se torna mais proficiente no uso do Arduino e na programação em C (a linguagem de programação utilizada para criar código no Arduino é um dialeto da linguagem C), poderá acabar considerando o IDE do Arduino como sendo básico demais. Caso queira algo com mais funcionalidades, poderá experimentar algum dos programas de IDE profissionais (mesmo os gratuitos), como Eclipse, ArduIDE, GNU/Emacs, AVR-GCC, AVR Studio e até mesmo o XCode da Apple.

Agora que você tem o software de seu Arduino instalado, a placa conectada e funcionando, e um conhecimento básico de como utilizar o IDE, vamos avançar diretamente para o Projeto 1, no qual criaremos um LED piscante.