



Universidade do Estado de Minas Gerais
Fundação Educacional de Ituiutaba
Engenharia de Computação



CURSO BÁSICO ARDUINO

João Ludovico Maximiano Barbosa
Rafael Caetano da Silva
2016

INTRODUÇÃO

O arduino é uma plataforma de prototipagem eletrônica *open-source*, projetada com um microcontrolador Atmel AVR de placa única, com suporte de entrada/saída embutido, uma linguagem de programação padrão, a qual tem origem em *Wiring*, e é essencialmente C/C++ (Arduino Build Process, s.d.).

O objetivo do arduino é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de se usar por pessoas que não tem um conhecimento avançado nesta área. Ele é baseado nos microcontroladores da Atmel ATMEGA8 e ATMEGA168, com componentes complementares para facilitar a programação e incorporação para outros circuitos (HomePage, s.d.) (Introduction, s.d.).

Um fator importante sobre ele é a maneira padrão que os conectores são expostos, permitindo a unidade central de processamento (CPU - *Central Processing Unit*) ser interligado a outros módulos expansivos, conhecidos como *shields*. Estes *Shields* são placas de circuito impresso normalmente fixadas no topo do aparelho através de uma conexão alimentada por pinos-conectores, que disponibilizam várias funções específicas, desde a manipulação de motores até sistemas de rede sem fio, facilitando o desenvolvimento de várias aplicações (Shields, s.d.).

O arduino ainda é pré-programado com um *bootloader* que simplifica o carregamento de programas para o *chip* de memória *flash* embutido, comparado com outros microcontroladores que usualmente necessitam de um *chip* programador externo (O que é Arduino?, 2012).

Seu grande objetivo é facilitar a prototipagem, implementação ou emulação do controle de sistemas interativos, a nível doméstico, comercial ou móvel, a um custo reduzido. Sendo possível enviar ou receber informações de basicamente qualquer sistema eletrônico (Fonseca, Beppu, & Vega, 2010).

CONHECENDO A IDE DE DESENVOLVIMENTO DO ARDUINO

A IDE que será apresentada é a Arduino1.0.5. Vamos conhecer alguns ícones desta IDE que são mostrados na Figura 1.

- Editor de texto (Retângulo Vermelho): Esta é a área na qual você deve digitar o seu código.
- Botão “**Verify**” (Retângulo Preto): Compila o código;
- Botão “**Upload**” (Retângulo Laranja): Compila o código e envia para o arduino;
- Botão “**New**” (Retângulo Roxo): Abre um novo editor de texto do arduino em branco;
- Botão “**Open**” (Retângulo Cinza): Abre um arquivo previamente salvo;
- Botão “**Save**” (Retângulo Azul): Salva o arquivo;
- Botão “**Serial Monitor**” (Retângulo Amarelo): Abre a janela de comunicação serial com o arduino.

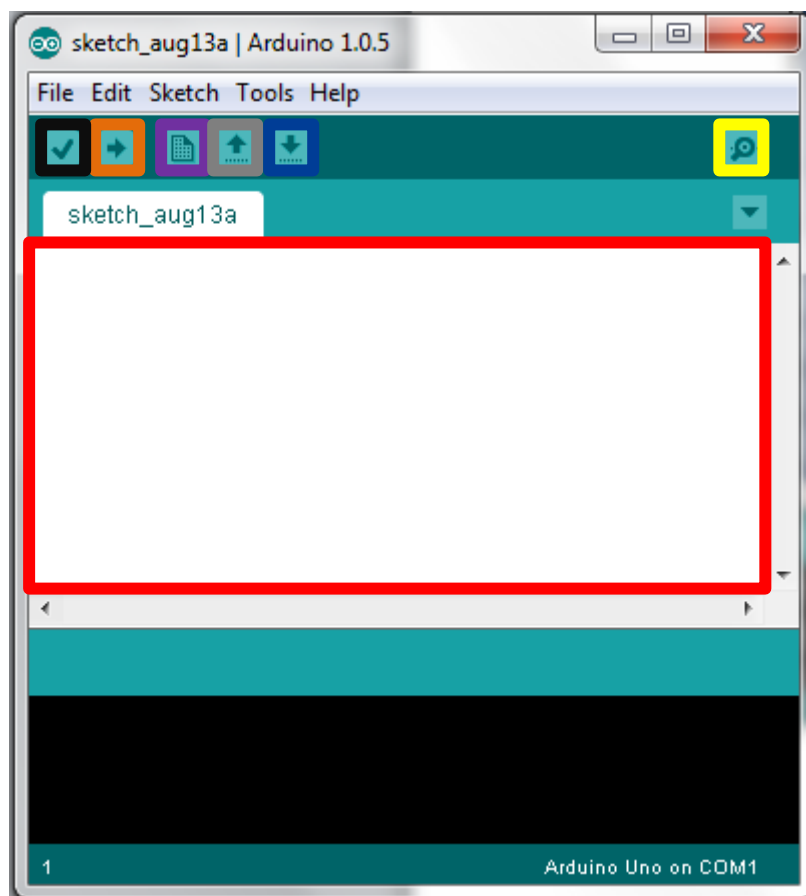


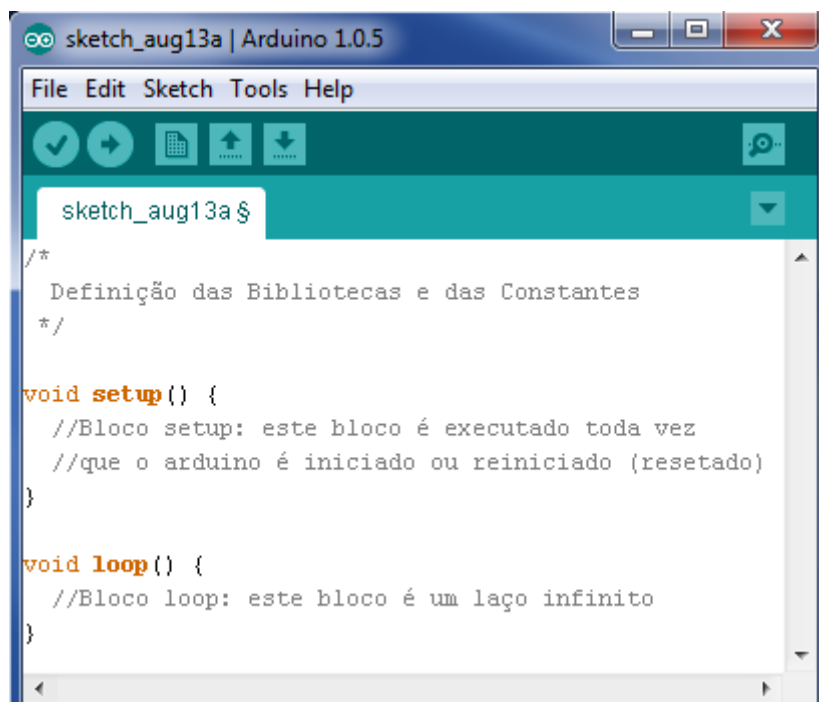
Figura 1 – IDE Arduino 1.0.5

ESTRUTA BÁSICA DE UM CÓDIGO ARDUINO

A estrutura básica de um código no arduino é dividido em três partes:

- Definição de Bibliotecas e Constantes;
- Bloco SETUP; e
- Bloco LOOP;

Na primeira parte do código importamos as bibliotecas e as constantes que serão utilizadas no código. Na segunda parte no bloco SETUP colocamos os códigos que queremos que seja carregado toda vez que o arduino é ligado ou reiniciado. E na ultima parte no bloco LOOP colocamos os códigos que o arduino irá rodar a todo instante, ou seja, em um laço infinito. Na Figura 2 podemos visualizar esta estrutura básica.

The image is a screenshot of the Arduino IDE interface. The title bar at the top reads 'sketch_aug13a | Arduino 1.0.5'. Below the title bar is a menu bar with 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Underneath the menu bar is a toolbar with icons for opening a file, saving, uploading, and downloading. The main text area contains the following code:

```
/*  
  Definição das Bibliotecas e das Constantes  
  */  
  
void setup() {  
  //Bloco setup: este bloco é executado toda vez  
  //que o arduino é iniciado ou reiniciado (resetado)  
}  
  
void loop() {  
  //Bloco loop: este bloco é um laço infinito  
}
```

Figura 2 – Estrutura Básica de um código no Arduino

IMPRIMINDO “HELLO WORLD” ATRAVÉS DO SERIAL MONITOR

Para imprimir ou receber dados na janela serial monitor, é necessário iniciar a comunicação serial no bloco SETUP através do comando “Serial.begin(9600);”.

Para imprimir os dados no serial monitor podemos utilizar estes dois comandos:

- **Serial.print(variável);**
- **Serial.println(variável);**

A diferença entre estes os dois comandos é que o Serial.println quebra a linha (da um ENTER ou volta a linha) após imprimir o texto ou o valor da variável, e o Serial.print **não** quebra a linha após imprimir o texto.

Para imprimir um texto com os comandos anteriores é necessário colocar o texto entre aspas duplas.

Na Figura 3 temos o exemplo de um código que imprimirá “HELLO WORLD” uma vez no Serial Monitor.

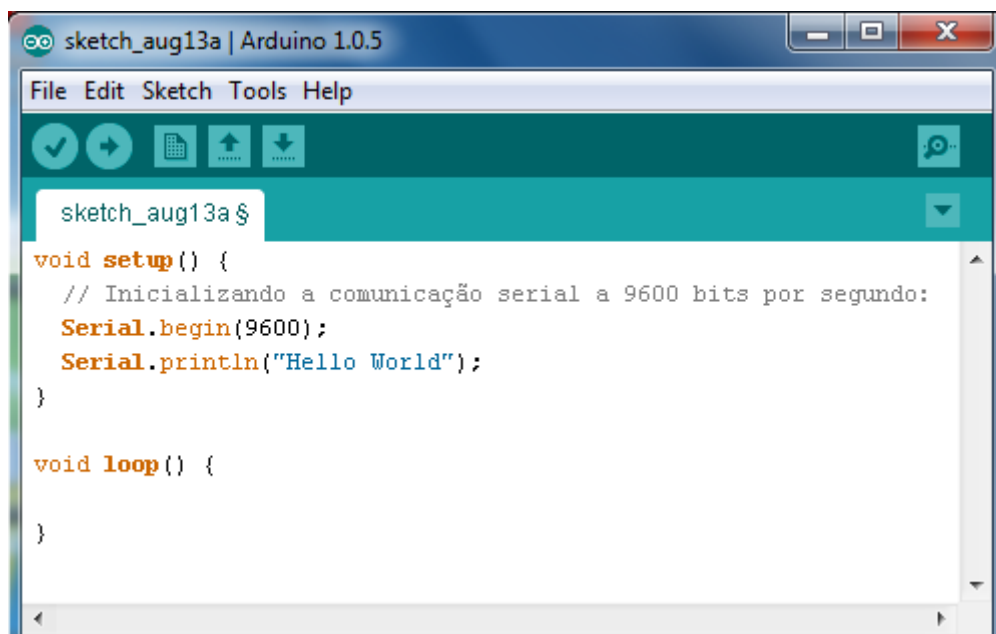


Figura 3 – Código para imprimir “HELLO WORLD” no Serial Monitor

Digite o código da Figura 3 no editor de texto da IDE do arduino, salve o código através do Botão “Save”, e envie o código para o arduino com o Botão “UPLOAD”. Para abrir a janela Serial Monitor clique no Botão “Serial Monitor”.

Para compreender melhor o bloco SETUP e LOOP realize os seguintes testes separadamente e observe o que acontece:

- Reinicie o arduino, apertando o botão “reset” da placa arduino;
- Tire a instrução “Serial.println” do bloco SETUP e coloque-o no bloco LOOP;

LIGANDO E DESLIGANDO UM LED

Para realizarmos este exemplo, será necessário montar um simples circuito com um led conforme descrito a seguir.

Conecte um resistor de 220 ohm no pino 13, em seguida ligue a perna mais longa do diodo emissor de luz LED (a perna positivo, chamado de ânodo) à resistência. Prenda a perna curta (a perna negativo, chamado de cátodo) no GND (ground ou terra) conforme mostra a Figura 4. Feito isso conecte a placa arduino ao computador através do cabo USB, inicie a IDE arduino, e insira o código que será descrito posteriormente.

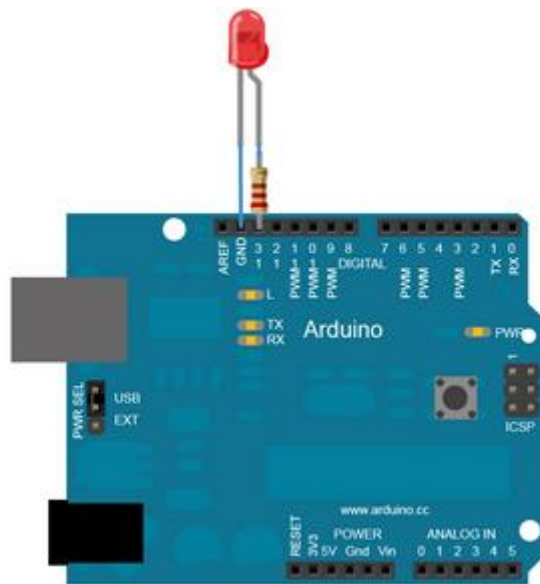


Figura 4 – Circuito de ligação do led com o arduino

Como iremos acionar o LED através da energia do arduino, teremos que habilitar a porta que o LED está conectado ao arduino como saída. Para isso usamos a seguinte instrução que geralmente é colocada no bloco SETUP:

- **pinMode(numPin, OUTPUT);**

Onde: numPin é o número do pino que será definido como saída;

Para definirmos um pino como entrada trocamos a palavra OUTPUT para INPUT, conforme mostra o exemplo abaixo:

- **pinMode(numPin, INPUT);**

Após definir o pino que o LED está conectado como saída, vamos aprender os comandos para ligar ou desligar o LED.

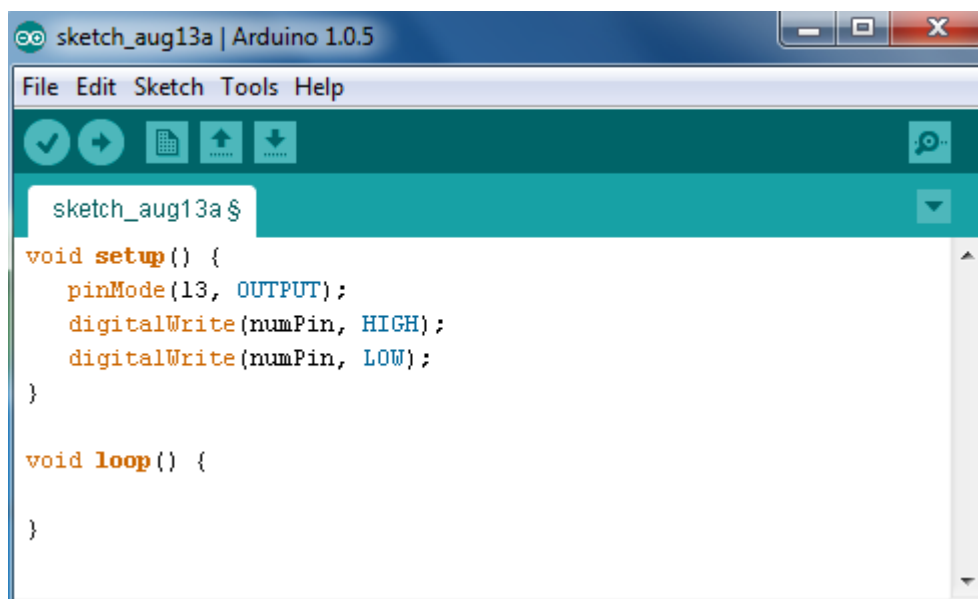
- **Ligar:** digitalWrite(numPin, HIGH);
- **Desligar:** digitalWrite(numPin, LOW);

Para esta atividade vamos conhecer mais um comando que irá nos auxiliar a ver estes comandos a funcionar. O comando é o delay que para a execução normal do arduino por um tempo determinado, sua sintaxe é mostrada abaixo:

- **delay(tempoEmMilissegundos);**

Onde tempoEmMilissegundo, é o tempo em milissegundos que o arduino ficará pausado (parado).

Na Figura 5 é mostrado o exemplo de um código para acender e apagar um LED logo em seguida.



```
sketch_aug13a | Arduino 1.0.5
File Edit Sketch Tools Help
sketch_aug13a $
void setup() {
  pinMode(13, OUTPUT);
  digitalWrite(numPin, HIGH);
  digitalWrite(numPin, LOW);
}

void loop() {
}
```

Figura 5 – Código para acender e apagar um LED

Atividades:

- Acrescente um delay de 1 segundo após as instruções de ligar e desligar o LED no código da Figura 5;
- Mude o trecho de código de ligar e desligar o LED do bloco SETUP para o bloco LOOP, juntamente com os DELAYS;
- Remova os Delays;

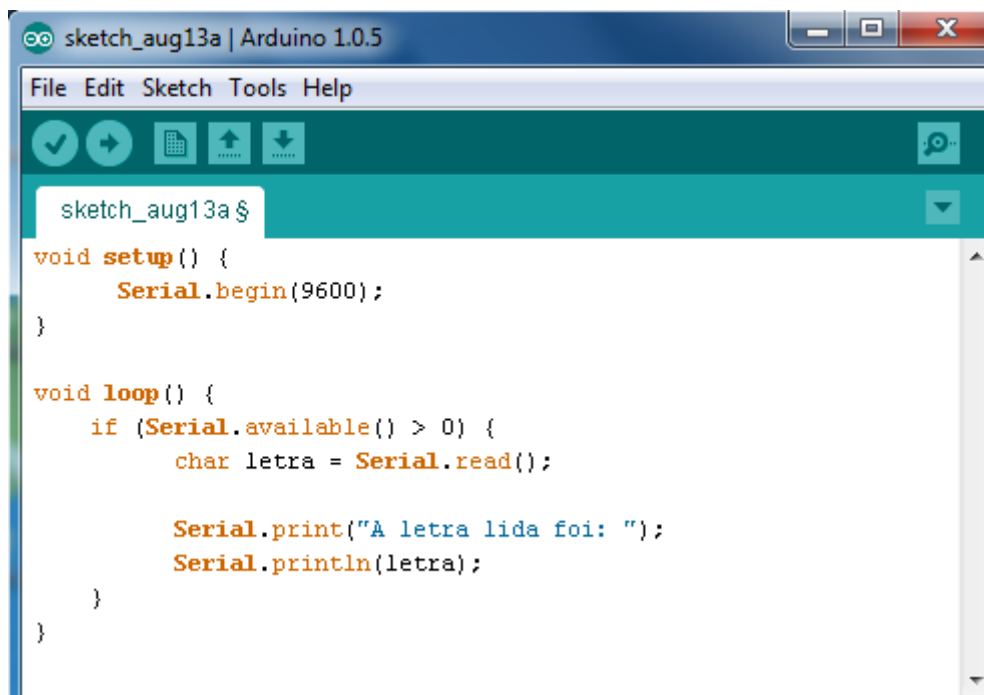
LENDO CARACTERES ATRAVÉS DO SERIAL MONITOR

Para este exemplo necessitamos conhecer mais dois comandos para trabalhar com a porta serial para comunicar o arduino com o computador.

O primeiro comando é o **Serial.available()**, que retorna o número de bytes (caracteres) disponível para leitura da porta serial.

O segundo comando é o **Serial.read()**, que retorna o primeiro byte (caracter) de entrada de dados seriais disponíveis (ou -1 se não há dados disponíveis).

Na Figura 6 podemos ver um exemplo de como ler os dados da porta serial e imprimi-los no SERIAL MONITOR.

The image shows the Arduino IDE interface. The title bar reads 'sketch_aug13a | Arduino 1.0.5'. The menu bar includes 'File', 'Edit', 'Sketch', 'Tools', and 'Help'. Below the menu is a toolbar with icons for checking, running, uploading, and downloading. The main text area shows the following code:

```
sketch_aug13a $  
void setup() {  
    Serial.begin(9600);  
}  
  
void loop() {  
    if (Serial.available() > 0) {  
        char letra = Serial.read();  
  
        Serial.print("A letra lida foi: ");  
        Serial.println(letra);  
    }  
}
```

Figura 6 – Código para ler caracteres da entrada serial do arduino

Para enviar os dados para o arduino abra o Serial Monitor clicando no Botão “Serial Monitor” mostrado no início do curso, feito isso irá abrir uma janela semelhante ao da Figura 7 onde os dados devem ser inseridos no retângulo vermelho, e para envia-los para o arduino clique no botão “SEND” retângulo verde ou tecle “ENTER”.

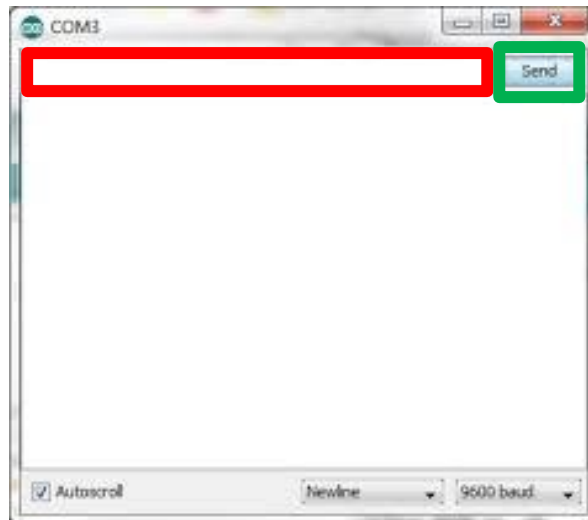


Figura 7 – Janela do Serial Monitor

Atividade:

- Acenda e apague um LED através do serial monitor, onde:
 - Se o caracter lido for ‘L’ (L minúsculo) o LED deverá acender;
 - Se o caracter lido for ‘D’ (D minúsculo) o LED deverá apagar;

Referências

Arduino Build Process. (s.d.). Acesso em 21 de Jan. de 2016, disponível em Arduino: <http://arduino.cc/en/Hacking/BuildProcess>

Fonseca, E. G., Beppu, M. M., & Vega, A. S. (2010). *Apostila Arduino*. Acesso em 21 de Jan. de 2013, disponível em Departamento de Engenharia de Telecomunicações UFF: http://www.telecom.uff.br/pet/petws/downloads/tutoriais/arduino/Tut_Arduino.pdf

HomePage. (s.d.). Acesso em 21 de Jan. de 2016, disponível em Arduino: <http://www.arduino.cc/>

Introduction. (s.d.). Acesso em 19 de Jan. de 2016, disponível em Arduino: <http://arduino.cc/en/Guide/Introduction>

O que é Arduino? (2012). Acesso em 22 de Jan. de 2016, disponível em Tecnologia Embarcada: <http://www.jlp.net.au.net/platarduino.html>

Shields. (s.d.). Acesso em 21 de Jan. de 2016, disponível em Arduino: <http://www.arduino.cc/en/Main/ArduinoShields>