# Capítulo 6

# Uso do Hardware Livre Arduino em Ambientes de Ensino-aprendizagem

Rafael Machado Alves, Armando Luiz Costa da Silva, Marcos de Castro Pinto, Fábio Ferrentini Sampaio, Marcos da Fonseca Elia

***Abstract***

*This mini-course is intended for teachers and other professionals who wish to work with educational robotics. The main topics discussed are: presentation of theoretical concepts on the use of robotics in education and demonstration the main features of free hardware and software Arduino to create educational projects.*

***Resumo***

*Este mini-curso destina-se aos professores e outros profissionais que desejam trabalhar com robótica educacional. Os principais tópicos abordados são: apresentação de conceitos teóricos sobre o uso da robótica na educação e demonstração dos principais recursos do hardware e do software livre Arduino para a criação de projetos educacionais.*

## 1.1. Introdução

O mini-curso *Uso do Hardware Livre Arduino em Ambientes de Ensino-aprendizagem* tem por objetivo apresentar e explorar as possibilidades educacionais do kit de robótica de baixo custo Arduino. Ao longo do texto é trabalhado o tema robótica educacional, numa visão interdisciplinar, a partir de cinco grandes temas relacionados ao currículo do 2º Ciclo do ensino fundamental e do ensino médio: Resgate histórico da História das Ciências a partir do uso da robótica; Desenvolvimento de aplicações práticas com a robótica (robótica resolvendo problemas no cotidiano); Aprendizado da eletrônica a partir do uso da robótica; Desenvolvimento de projetos interdisciplinares com robótica.

## 1.2. Introdução à Robótica Educacional

### 1.2.1. Qualificação do Principal Problema Abordado

As interações primeiras da criança com o mundo físico são as mais importantes e determinantes para o resto da vida. Piaget [Piaget et al 1970 e 1974] foi o epistemólogo construtivista que melhor sistematizou essas ideias sobre a importância para o desenvolvimento cognitivo das interações dos sujeitos com os objetos da natureza. Suas teorias construtivistas sobre as interações sujeito-objeto (físico), doravante **S-O**, têm servido de referencial teórico para muitos modelos de ensino-aprendizagem.

Em uma linha construtivista análoga, mas com foco na interação social sujeito-sujeito, doravante **S-S**, tem-se os modelos de ensino-aprendizagem centrados nas teorias construtivistas de Vygotsky (1984 e 1993), cujas contribuições conceituais fundamentais para esse tema seriam: (i) a importância que a interação **S-S** traz para construção da linguagem, e esta para a aprendizagem, e esta para o desenvolvimento cognitivo; e (ii) o papel potencial que alguns sujeitos podem ter sobre a aprendizagem de seus pares, estabelecendo um campo de influência que Vygotsky denominava Zona de Desenvolvimento Proximal (ZDP).

Um desdobramento quase que natural das ideias construto-interacionista apontadas acima, seriam aquelas que valorizam a construção do conhecimento por meio da interação com objetos criados pelos próprios sujeitos, ou seja, com os artefatos tecnológicos criados pelos seres humanos que formam o acervo cultural da Humanidade. A interação dos sujeitos com objetos resultantes de sua cultura, aqui representada como **S-C**, tem sido denominada de "construcionismo", sendo Papert [Ackermann, 2001] o seu principal protagonista.

No entendimento do Grupo GINAPE, o uso da Robótica em ambientes de ensino-aprendizagem compõe uma tecnologia educacional potencializora, sob o ponto de vista dos referenciais teóricos construtivistas de Piaget, Vygotsky e Papert. Além da interação sócio-verbal (**S-S**) que o meio escolar proporciona, os alunos têm também a oportunidade de uma interação integrada (**S-O**) e (**S-C**), através da criação de objetos animados, automatizados e comandados pelas suas próprias estratégias cognitivas, sob a supervisão firme de um projeto pedagógico engendrado e executado por seus professores.

### 1.2.2. Robótica Aplicada à Educação

A Robótica Pedagógica ou Robótica Educacional (RE) vem se constituindo numa forma interdisciplinar de promoção do aprendizado de conceitos curriculares. Na aula com RE o aluno pensa, manuseia, constrói, executa, vê o que dá certo, depura o que está errado e reexecuta, ou seja, é o esmiuçar da teoria através da prática.

A atividade com robótica educacional é desafiadora e lúdica, onde o esforço do educando é utilizado na criação de soluções, sejam essas compostas por hardware e/ou software, visando à resolução de um problema proposto – podendo o mesmo ser real. Para Schons *et al*. (2004), a robótica pedagógica "constitui nova ferramenta que se encontra à disposição do professor, por meio da qual é possível demonstrar na prática muitos dos conceitos teóricos, às vezes de difícil compreensão, motivando tanto o professor como principalmente o aluno".

Segundo Zilli (2004), a robótica educacional pode desenvolver as seguintes competências: raciocínio lógico; formulação e teste de hipóteses; habilidades manuais e estéticas; relações interpessoais e intrapessoais; integração de conceitos aprendidos em diversas áreas do conhecimento para o desenvolvimento de projetos; investigação e compreensão; representação e comunicação; trabalho com pesquisa; resolução de problemas por meio de erros e acertos; aplicação das teorias formuladas a atividades concretas; utilização da criatividade em diferentes situações; e capacidade crítica.

A nossa proposta é, portanto, apresentar e explorar a robótica educacional como elemento motivador da aprendizagem e como tema problematizador na resolução de problemas, tirando vantagem dos dispositivos de baixo custo que hoje permitem o interfaceamento homem-máquina (Human Interface Device-HID).

### 1.2.3. Robótica Educacional de Baixo Custo

Atualmente, não há a necessidade de gastar valores elevados na compra de material para trabalhar com a robótica em sala de aula. A utilização de kits de robótica sofisticados pode ser dispensável em se tratando de RE [Filho et al. 2008, Miranda et al. 2007, Id. 2010]. Há também uma tendência promissora que é o uso de Laboratórios Remotos como vem sendo pesquisado em algumas instituições, objetivando estudos a distância e sem custos iniciais na aquisição do kit robótico [D'Abreu *et al*. 2001, Victorino *et al*. 2009, Cruz *et al*. 2009, Souza *et al*. 2011]. Além disso, muitos pesquisadores conseguem bons resultados apostando em sucata [Albuquerque *et al*. 2007, Sasahara *et al*. 2007, Santos *et al*. 2010], democratizando assim o acesso as tecnologias através da Robótica Educacional de Baixo Custo (REBC). A REBC utiliza materiais alternativos, recursos de hardware e de software livre, tais como o projeto Arduino[1], como forma de se viabilizar projetos educacionais na área de RE.

O projeto Arduino (Subseção 1.3.1) prevê uma plataforma de hardware e de software de código aberto de fácil utilização, acessível não somente para especialistas na área de Eletrônica, mas também para hobbystas, artistas e qualquer pessoa interessada na criação de objetos ou ambientes interativos.

### 1.2.4. O Papel do Professor na Robótica Educacional

O papel do professor é muito importante no contexto da robótica educacional, atuando tanto no planejamento de atividades didáticas com os recursos da robótica, quanto na execução da atividade com a robótica agindo como elemento mediador e incentivador para que seus alunos obtenham êxito em suas tarefas. Para tal, torna-se necessário que o professor sinta-se capacitado a trabalhar com tecnologias que envolvam a robótica educacional. Assim, não será possível consolidar a prática da robótica educacional nas escolas brasileiras, sem pensar em uma formação docente adequada para o uso de tecnologias educativas. Segundo Kenski,

> *"(...) é preciso que este profissional tenha tempo e oportunidades de familiarização com as novas tecnologias educativas, suas possibilidades e limites para que, na prática, faça escolhas*

[1] Arduino - Plataforma open-source de prototipagem eletrônica baseada na flexibilidade, com hardware e software fáceis de usar. http://www.Arduino.cc

> *conscientes sobre o uso das formas mais adequadas ao ensino de um determinado tipo de conhecimento, em um determinado nível de complexidade, para um grupo específico de alunos e no tempo disponível."* [Kenski, 1999].

A pouca formação docente existente aliada ao elevado custo de kits comerciais voltados para a robótica educacional ainda contribuem para a pouca atividade desta no Brasil, principalmente no contexto da educação pública.

O presente mini-curso utiliza e avança em algumas idéias e propostas do trabalho de Pinto (2011) para promover o uso da robótica em ambientes de ensino-aprendizagem, procurando envolver professores e outros interessados, mesmo que desprovidos de um background na área.

## 1.3. Arduino em Ambientes de Ensino-aprendizagem

É possível utilizar a Robótica Educacional em sala de aula sem o uso da programação ou computador, somente fazendo o uso de artefatos físicos (como hardwares, elétrico-eletrônicos ou aparatos mecânicos) [Santomauro, 2009], porém os experimentos construídos ficam com um escopo limitado sem a presença de uma lógica dinâmica (software). Desta forma, compreende-se que a utilização da programação, a qual permite criar sistemas inteligentes capazes de reagir a um estímulo, além de expandir os limites de atuação potencializa o seu uso em Robótica Educacional. Isto é possível através do projeto Arduino.

### 1.3.1. Projeto Arduino

O projeto Arduino, nascido na Itália em 2005, constitui uma plataforma de hardware e de software com o objetivo de possibilitar que pessoas não especialistas em programação e/ou em eletrônica possam desenvolver aplicações de objetos e ambientes interativos. Para isso, a proposta do projeto visa tanto a criação de um hardware fácil de manusear e com os recursos necessários para trabalhar com os "mundos" digital e analógico, quanto um software de desenvolvimento acessível para a programação dos projetos interativos.

Uma vez programado, o Arduino controla uma gama de componentes eletrônicos como LEDs, motores, *displays*[2] com base nas instruções recebidas através de sensores como os de luminosidade e temperatura, acoplados a um dos modelos de hardware (Figura 1.1).

[2] Display (ou mostrador, em português) é um dispositivo para a apresentação de informação de modo visual.


**Figura 1.1: Placa eletrônica Arduino**

Hoje, com sete anos de vida, o Arduino virou uma verdadeira "onda" mundial com aplicações em diversos segmentos (música, artes plásticas, educação, meio ambiente, etc.) e com uma infinidade de comunidades espalhadas no planeta trocando experiências sobre seus projetos.

### 1.3.2. Hardware Livre Arduino e Suas Variações

Sob a filosofia "*Open Hardware*", a placa eletrônica Arduino possui toda documentação no site do projeto para aqueles que desejarem montar sua própria placa. Essa documentação está disponível sob a licença *Attribution-ShareAlike 2.5* da empresa *Creative Commons*. Isto proporcionou que, além dos modelos oficiais lançados pela equipe do projeto, diversos outros modelos de hardware surgissem em vários lugares do globo.

A placa eletrônica possui como elemento principal um microcontrolador da família AVR, fabricado pela empresa ATMEL. São provenientes do microcontrolador os recursos do Arduino (Figura 1.2) como: conversores analógicos/digitais (entradas analógicas); as entradas e saídas digitais; as saídas PWM[3] (saídas analógicas). Além disso, a placa eletrônica ainda possui interface serial/USB para comunicação com o computador de desenvolvimento e um regulador de voltagem para adequação da energia necessária (tensão elétrica) ao funcionamento do sistema (5V DC).

[3] Modulação por largura de pulso (MLP) - mais conhecida pela sigla em inglês PWM (*Pulse-Width Modulation*). http://pt.wikipedia.org/wiki/Modula%C3%A7%C3%A3o_por_largura_de_pulso

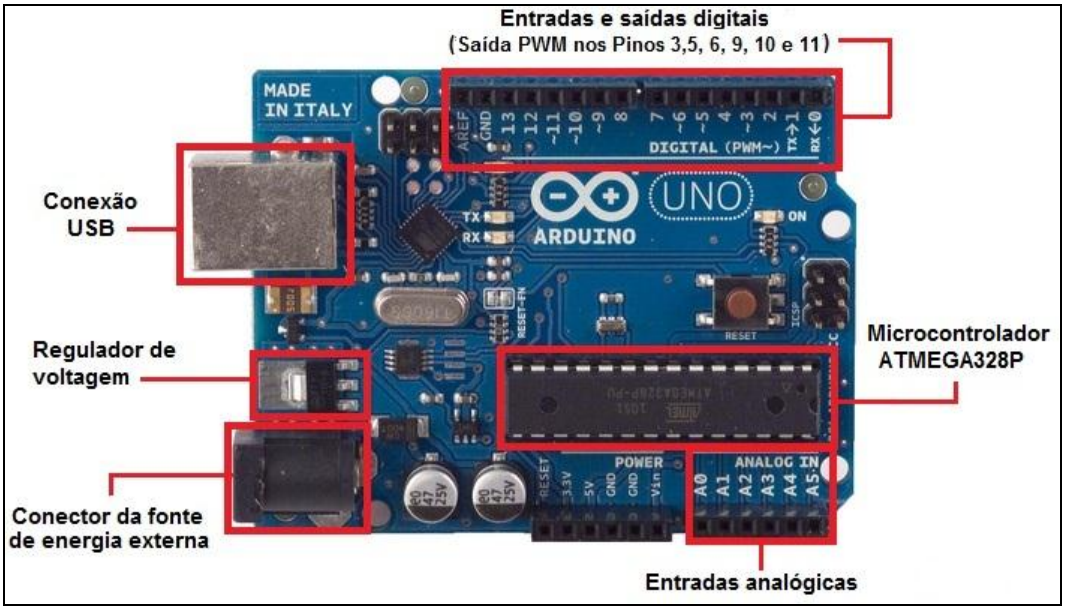



**Figura 1.2: Recursos do hardware livre Arduino**

Para conexão do Arduino com o computador que possui o ambiente de desenvolvimento, é utilizada uma interface serial com conexão USB. Na plataforma Windows a conexão do Arduino é identificada como uma porta serial tipo COM (ex.: COM3) e na plataforma Linux como um dispositivo serial ttyUSB (ex.: ttyUSB1). O cabo utilizado para interconexão é do tipo UBS A/B (o mesmo usado para impressora e *scanners*).

A alimentação elétrica padrão da placa Arduino é 5V (necessária ao funcionamento do microcontrolador) e pode ser obtida de duas formas: do próprio cabo USB ou através de uma fonte externa, tais como baterias ou eliminadores de pilha, que são ligadas ao conector de energia externa. Nos casos de alimentação elétrica com fonte externa, podemos aplicar tensões entre 7 e 20V, pois a placa possui um dispositivo regulador de voltagem que faz a adequação necessária para entregar ao microcontrolador a tensão de trabalho de 5V.

### 1.3.3. Ambiente de programação Arduino

O projeto Arduino também envolve um ambiente de desenvolvimento integrado ao hardware (IDE – *Integrated Developement Environment*) para geração dos programas (denominados de *sketches*) que serão enviados para a placa eletrônica. O IDE do Arduino foi desenvolvido em linguagem JAVA baseado no projeto *Processing*[4], na biblioteca AVR-gcc (para microcontroladores da família AVR) e em outros softwares livres. A linguagem de programação do Arduino é baseada no projeto *Wiring*[5] e pode rodar nas plataformas Windows e Linux (Figura 1.3).

[4] http://processing.org/

[5] http://wiring.org.co/

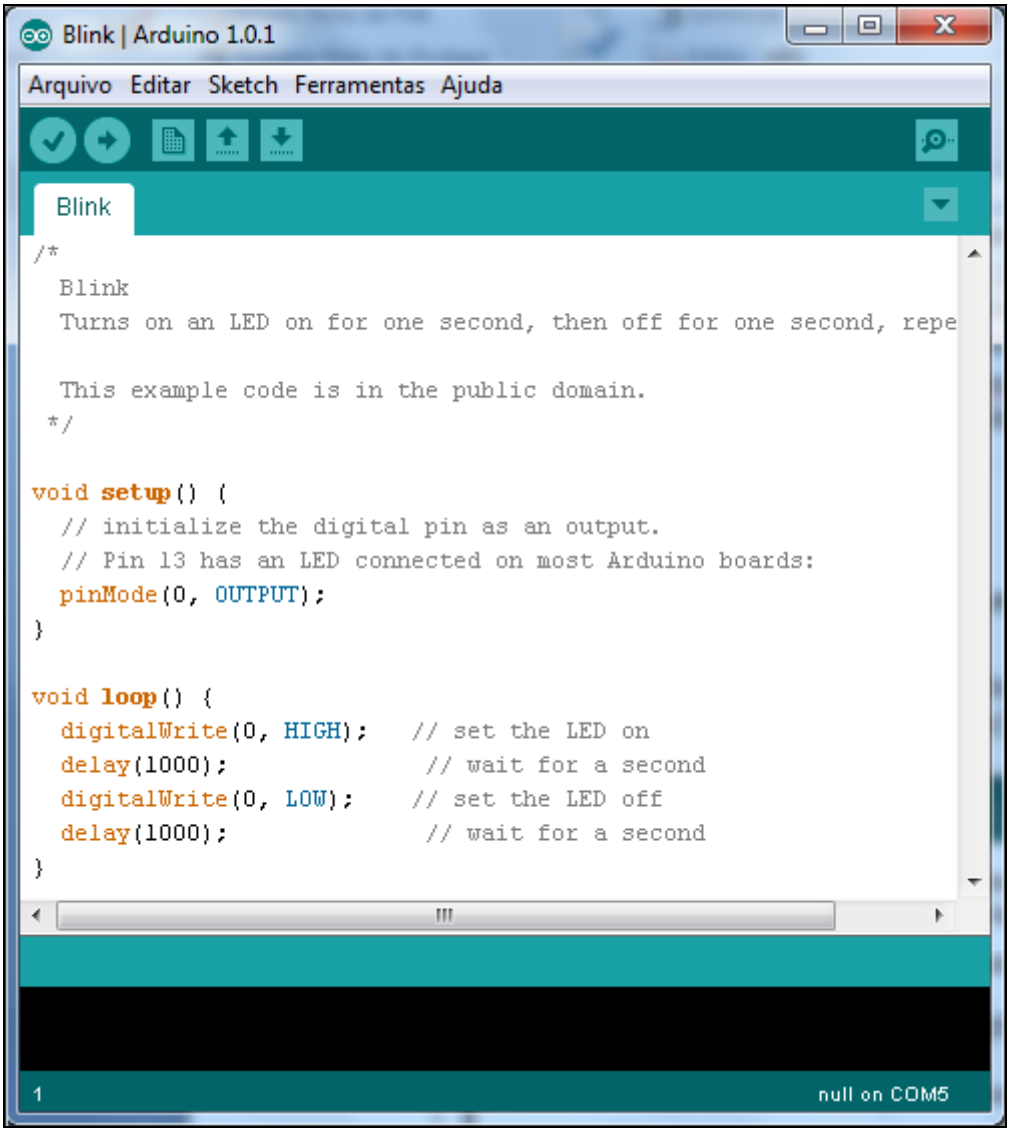



**Figura 1.3: Ambiente de programação Arduino**

Apresentamos a seguir uma breve descrição das funcionalidades dos Menus suspensos e dos Botões de atalho (Figura 1.4) contidos na interface do ambiente de programação Arduino:

- ***Arquivo*** – opções para criação, abertura, fechamento e salvamento de programas. Acesso ao livro de programas do usuário (*sketchbook*), aos programas-exemplo e ao menu de configuração do ambiente;
- ***Editar*** – acesso a funções para copiar e colar códigos de programas. A função *Copy* for Forum copia o código em formato adequado para postagem em fórum, com coloração de sintaxe. A função *Copy* as HTML copia o código para a área de transferência como HTML, apropriado para incorporação em páginas da web;
- ***Sketch*** – possui funções para verificação de erros, importação de bibliotecas para o código em curso (inclui o comando #include para cada biblioteca importada no topo de código) e função para importação de arquivo (aparece em uma nova aba (*tab*) no IDE);

- ***Ferramentas*** – possui funções para a auto-formatação do código criado (realizando a indentação[6] automática ao abrir chaves e colchetes no código), para seleção da placa eletrônica (hardware), para seleção da porta serial para comunicação com a placa eletrônica, além da função para gravação do *bootloader*[7] no microcontrolador presente na placa eletrônica (necessário apenas quando há a substituição do microcontrolador na placa).
-

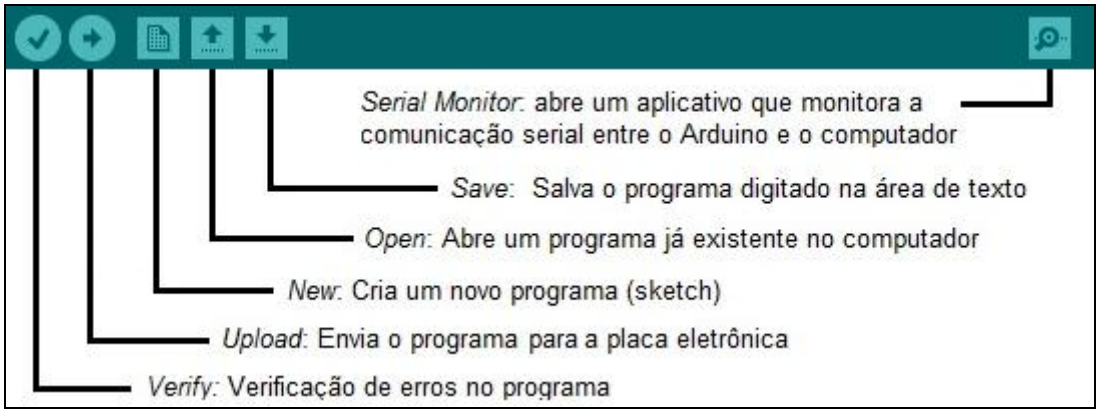



**Figura 1.4: Botões de atalho**

## 1.4. Eletrônica Básica

Antes de iniciarmos o nosso primeiro experimento vamos conhecer três componentes bastante comuns.

### 1.4.1. LEDs

São diodos emissores de luz, mais conhecidos pela sigla em inglês LED (*Light Emitting Diode*). O LED é comumente usado para indicação de eventos ou estados em um circuito eletrônico. A Figura 1.5 mostra o aspecto físico de um LED e seu símbolo eletrônico. A identificação dos terminais do LED é feita através do chanfro que fica do lado do terminal negativo (-) ou através do tamanho dos terminas, onde o positivo (+) é o maior.

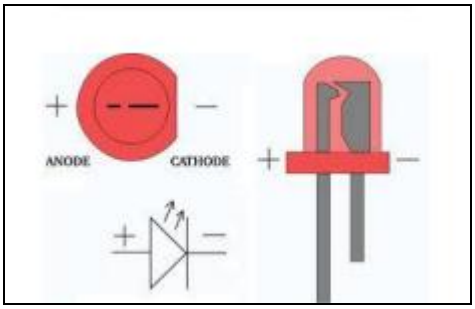



**Figura 1.5: LED**

[6] Site Wikipédia. Identação: http://pt.wikipedia.org/wiki/Indenta%C3%A7%C3%A3o

[7] Site Arduino. Bootloader: http://www.Arduino.cc/en/Hacking/Bootloader?from=Main.Bootloader

Como todo tipo de diodo, o LED somente permite a passagem da corrente quando está polarizado diretamente, ou seja, quando os seus terminais (+) e (-) estão conectados ao lado (+) e (-) da fonte de energia respectivamente (Figura 1.6.a). Caso a ligação esteja invertida (polarizado reversamente), o LED não acenderá (Figura 1.6.b).

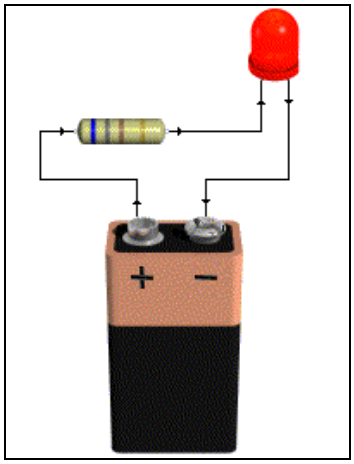



**Figura 1.6.a: Polarização direta**

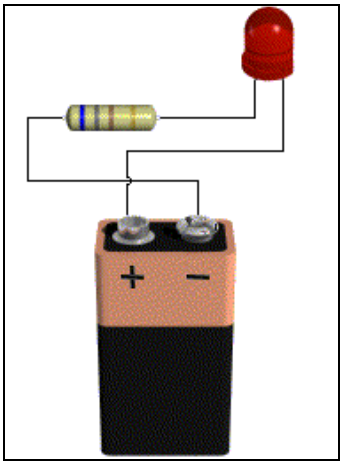



**Figura 1.6.b: Polarização reversa**

O *display* de 7 segmentos (Figura 1.7.a e 1.7.b) frequentemente utilizado como contador de um cronômetro (por ex.: relógio digital) é formado por um conjunto de sete LEDs de formato retangular.

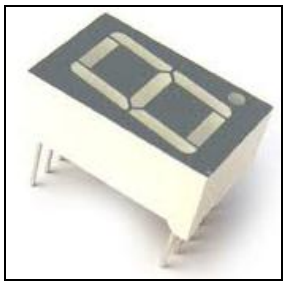


**Figura 1.7.a: Display de 7 segmentos**

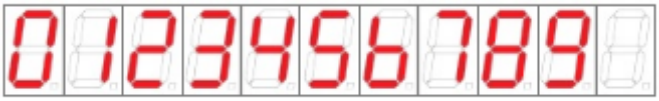


**Figura 1.7.b: Estados do display de 7 segmentos**

### 1.4.2. Resistores

Resistores são componentes elétrico-eletrônicos construídos com a finalidade básica de apresentar o fenômeno físico de "resistência elétrica". Assim, um resistor permite, dentre outros, limitar o nível de corrente elétrica em circuitos eletrônicos a um valor seguro. A Figura 1.8 apresenta o símbolo elétrico e o aspecto físico do resistor.

A maioria dos resistores usados em eletrônica possui dimensões bastante reduzidas, os valores de sua resistência normalmente não vêm escritos no corpo (exceto para resistores de maior potência), mas vêm codificados na forma de faixas coloridas.

Os significados de cada faixa bem como o valor de cada cor são mostrados na Figura 1.9.

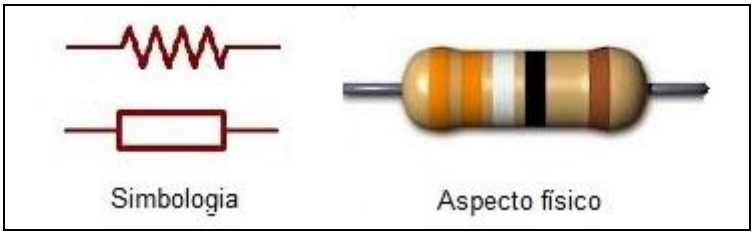
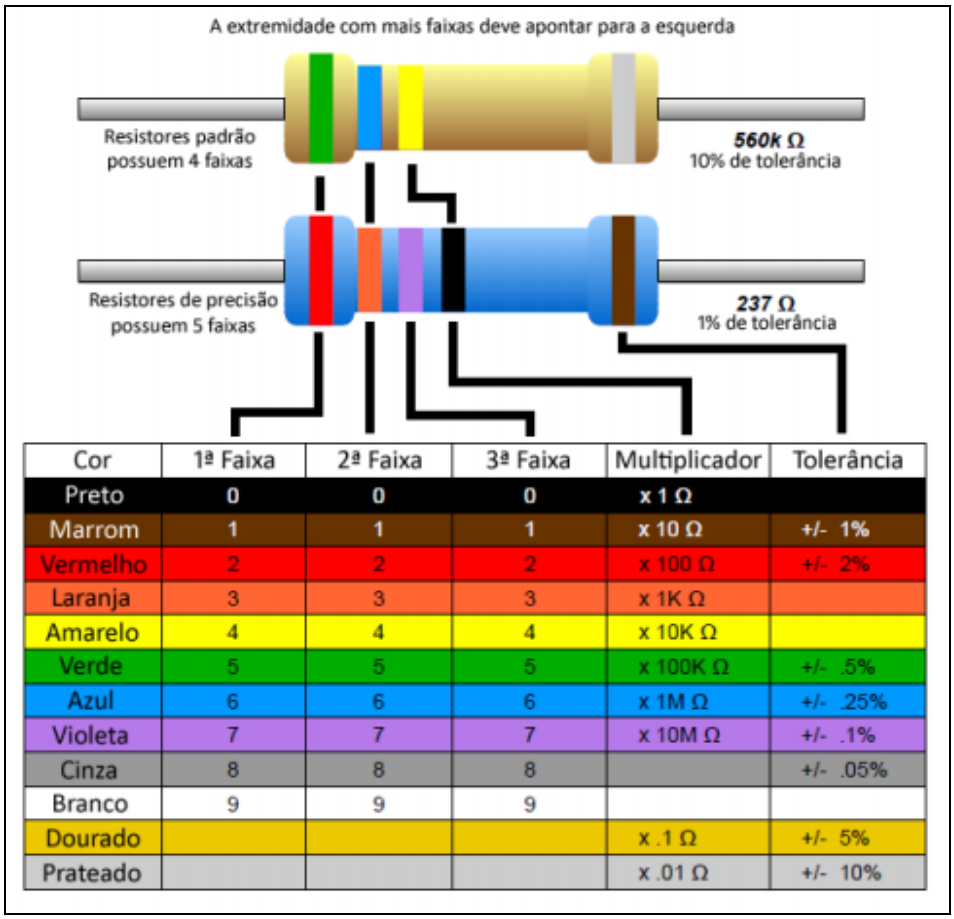



**Figura 1.8: Resistor**

Além dos resistores fixos mostrados anteriormente, existem os resistores que possuem resistência variável. Esses resistores, comumente denominados de *potenciômetros*, são aplicados nos casos onde necessitamos ajustar um dado valor de tensão ou corrente para que o circuito eletrônico funcione de maneira esperada. Os potenciômetros foram projetados para que haja variação da resistência a qualquer momento e, para isto, possuem um eixo para atuação. Os principais formatos de potenciômetros são o rotativo (mais comum) e o deslizante (Figura 1.10.a). O potenciômetro possui três terminais (Figura 1.10.b), onde a resistência variável é obtida entre o terminal do centro e algum terminal dos extremos. Se tomarmos o valor de resistência entre os terminais dos extremos, teremos um valor fixo de resistência.




| Cor | 1ª Faixa | 2ª Faixa | 3ª Faixa | Multiplicador | Tolerância |
|---|---|---|---|---|---|
| Preto | 0 | 0 | 0 | x 1 Ω | |
| Marrom | 1 | 1 | 1 | x 10 Ω | +/- 1% |
| Vermelho | 2 | 2 | 2 | x 100 Ω | +/- 2% |
| Laranja | 3 | 3 | 3 | x 1K Ω | |
| Amarelo | 4 | 4 | 4 | x 10K Ω | |
| Verde | 5 | 5 | 5 | x 100K Ω | +/- .5% |
| Azul | 6 | 6 | 6 | x 1M Ω | +/- .25% |
| Violeta | 7 | 7 | 7 | x 10M Ω | +/- .1% |
| Cinza | 8 | 8 | 8 | | +/- .05% |
| Branco | 9 | 9 | 9 | | |
| Dourado | | | | x .1 Ω | +/- 5% |
| Prateado | | | | x .01 Ω | +/- 10% |

**Figura 1.9: Código de cores**

**Figura 10.a: Tipos de potenciômetro**

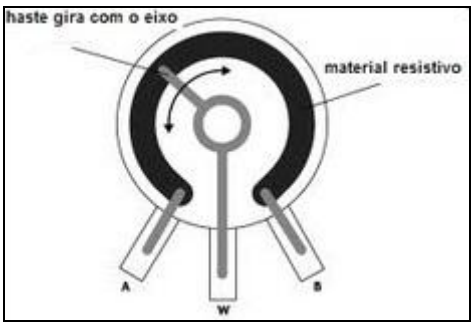



**Figura 10.b: Estrutura de um potenciômetro**

### 1.4.3. Protoboard

A *Protoboard* (Figura 11.a) é um equipamento muito útil na montagem de experimentos. Ela consiste numa placa didática composta de uma matriz de contatos que permite a construção de circuitos experimentais sem a necessidade de efetuar a solda dos componentes, isso permite que seja efetuada uma série de experimentos com os mesmos componentes inserindo ou removendo os mesmos com rapidez e segurança, ideal para projetos educacionais. Na Figura 11.b, as setas indicam o sentido da corrente

na placa e a existência de uma linha tracejada indica que a placa possui uma descontinuidade neste ponto.

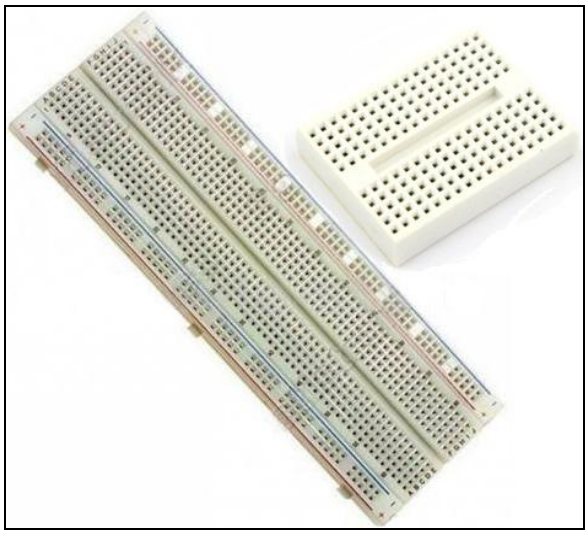


**Figura 11.a: Exemplos de *protoboard***

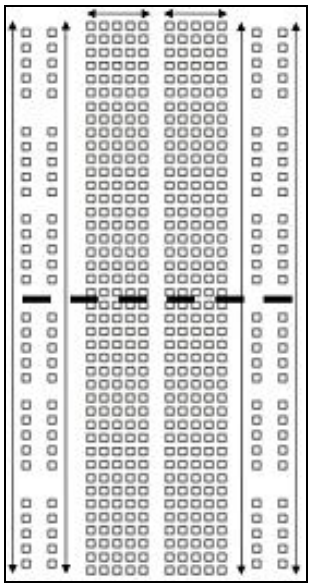


**Figura 11.b: Conexões da protoboard**[8]

A Figura 12 mostra o interior de uma *protoboard*, onde as linhas cinzas indicam uma conexão. Assim, ao encaixarmos fios nos furos imediatamente acima de uma linha cinza, significa dizer que interligamos os fios um ao outro (fio verde e fio vermelho como mostrado na figura).

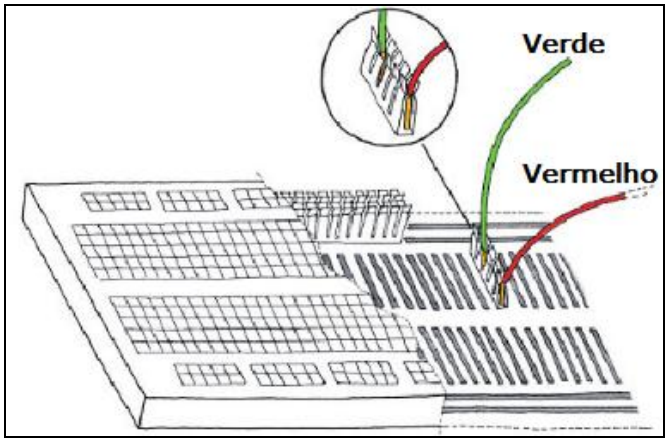



**Figura 1.12: Corte representativo de uma parte da *protoboard***[9]

Agora, após apresentarmos os componentes mais comumente utilizados, iniciamos os nossos experimentos de robótica educacional.

---

[8] http://equipe.nce.ufrj.br/adriano/circlog/bibliografia/introducao.pdf

[9] http://pt.scribd.com/doc/15969149/Aprendendo-a-usar-a-protoboard-ou-matriz-de-contatos-

## 1.5. Experimentos

Os experimentos propostos a seguir podem conter as seguintes seções:

- ***O que é****:* Esta seção fornece uma compreensão geral do que o experimento faz, o que se pretende com ele e as suas possibilidades educacionais.
- ***Material necessário***: Esta seção contém uma lista dos materiais necessários para criação do experimento. Ela pode incluir também materiais alternativos.
- ***Montagem***: Aqui serão fornecidas imagens de esquemas para realização da montagem. Podendo incluir também fotos do experimento real montado ou da sequência de passos.
- ***Código***: Aqui é fornecido o código fonte Arduino, necessário para o experimento, incluindo comentários.
- ***Como funciona***: Nesta seção é explicado como funciona o experimento, incluindo uma descrição de cada um dos componentes utilizados.
- ***Como usar***: Nesta seção é explicado como usar o experimento. Ela também inclui exemplos de utilização em sala de aula.
- ***Aspectos para se notar****:* Esta seção poderá dar algumas idéias de aspectos para o usuário notar durante a montagem, codificação ou utilização do experimento.
- ***Ideias para experimentar***: Esta seção poderá dar algumas idéias de pequenos desafios para o usuário tentar fazer (inverter terminais, trocar componente, alterar algoritmo, etc.) com o experimento.
- ***Estendendo o experimento***: Nesta seção são fornecidas algumas idéias possíveis de serem adicionadas na montagem ou na programação para tornar o experimento mais complexo, detalhado, preciso, etc.
- ***Características educacionais***: Esta seção aponta as características especialmente educacionais que a aplicação do experimento em sala de aula proporciona e em particular as áreas que abrange. Ela também pode indicar referências bibliográficas dos conceitos abordados.
- ***Experimentos relacionados***: Nesta seção são fornecidos os nomes dos experimentos na biblioteca de exemplos do Arduino ou em outros lugares que apresentam conteúdo relacionado.

### 1.5.1. Experimento 1 – Pisca LED

- ***O que é:*** Este experimento aborda aplicações com LEDs, um dos componentes básicos da eletrônica. As aplicações a partir do conhecimento deste experimento são inúmeras.
- ***Material necessário***:
    - 2x LEDs;
    - 2x resitores 330ohms (faixa de cores: laranja, laranja e marrom);

- Arduino, Protoboard e Fios condutores (estes materiais serão utilizados em todos os experimentos).

Os LEDs são relativamente baratos, podendo também ser facilmente sucateados de aparelhos eletrônicos danificados.

- ***Montagem***: A Figura 1.13 mostra como deve ser realizada a montagem dos componentes. Os terminais (+) dos LEDs estão conectados um na porta digital 11 e outro na 12 do Arduino através de fios condutores. Os terminais (-) dos LEDs estão conectados a uma das extremidades dos resistores através da protoboard. As outras extremidades dos resistores estão conectadas entre si através da protoboard, que por sua vez estão conectadas a porta GND (sigla em inglês para *ground*, ou terra em português) do Arduino através de um fio condutor.

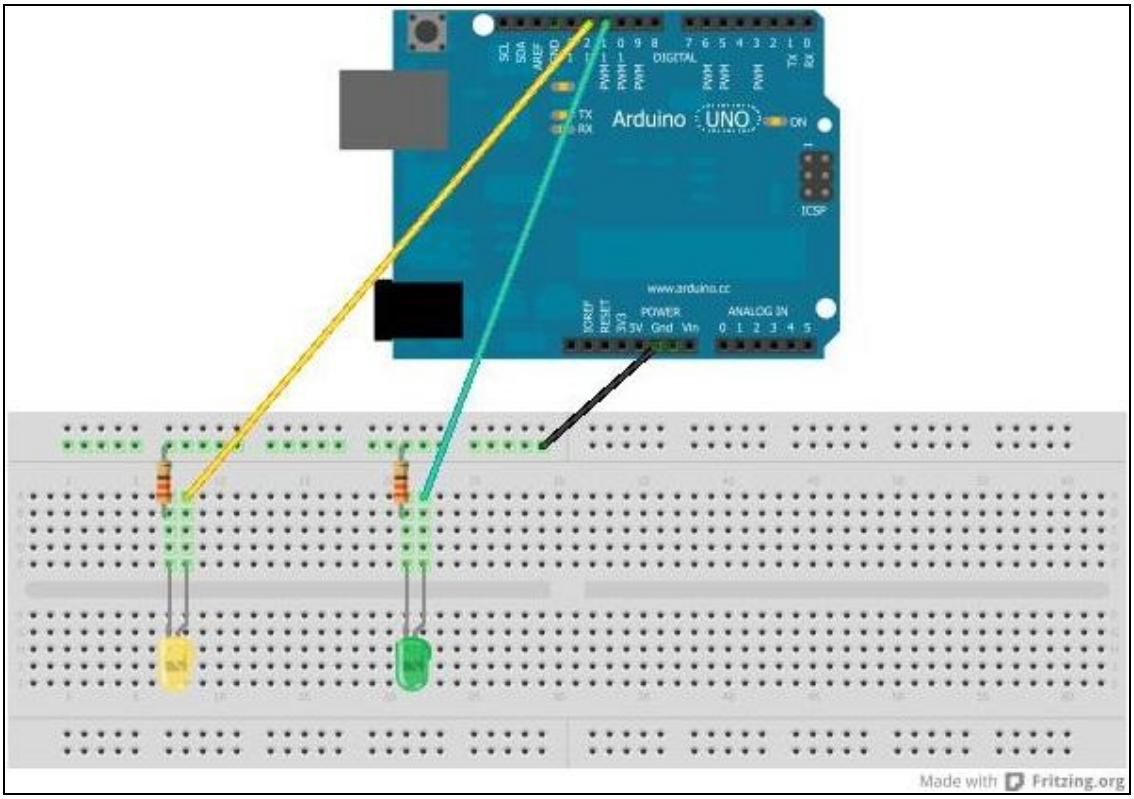



**Figura 1.13: Esquema de montagem Pisca_LED**

- ***Código***: Em nosso programa, precisamos compreender primeiro como funcionam duas estruturas básicas de um código criado no IDE Arduino (Figura 14). A primeira, o método *void setup(){}* é onde inserimos, dentro das chaves, as inicializações das variáveis. Este trecho só é executado uma vez no início do programa. A segunda, o método *void loop(){}* é onde inserimos os comandos do nosso programa que serão executados repetidamente em um loop, enquanto houver alimentação na placa. Este trecho de código é executado logo após o primeiro trecho.

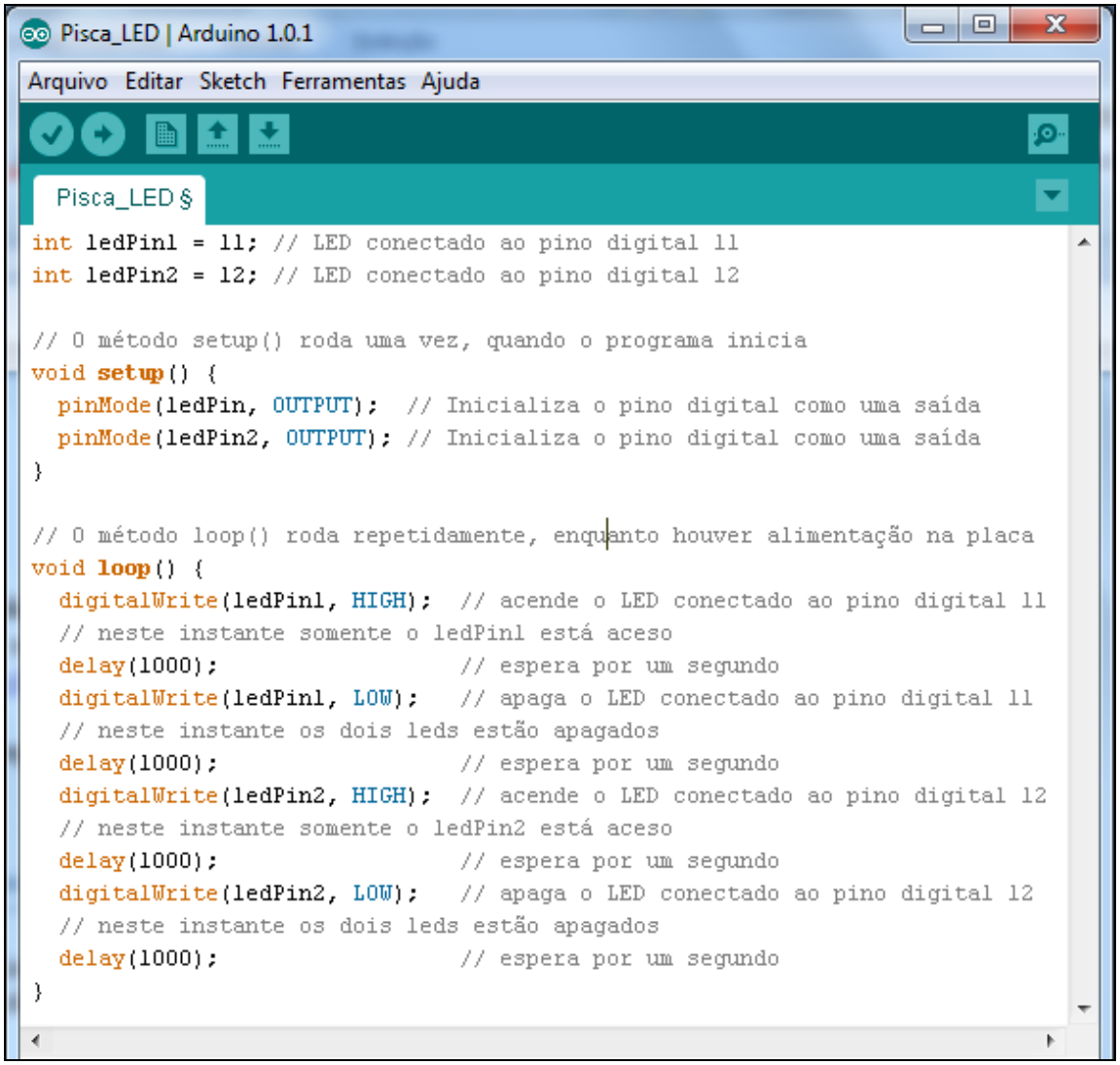



**Figura 1.14: Código fonte Pisca_LED.ino**

- ***Como funciona:*** Primeiro criamos duas variáveis inteiras para guardarem o número da porta em que os LEDs estão conectados. Está prática possibilita uma rápida e fácil mudança para adaptar o programa caso tenhamos que trocar a porta de algum componente, o que é bem comum. Segundo, no método *setup* indicamos que as portas em que estão conectados os LEDs são portas de saída, ou seja, a informação tem o sentido Aduino-LED. O comando *pinMode* requer dois parâmetros: o número da porta e o modo (INPUT ou OUTPUT). Terceito utilizamos o comando *digitalWrite* para acender e apagar um LED. Este comando requer dois parâmetros: o número da porta e o nível lógico (HIGH ou LOW). E quarto, utilizamos o comando *delay* para determinar quanto tempo os LEDs ficam acesos ou apagados. Este comando requer um parâmetro: o tempo em milissegundos em que o fluxo do programa ficará em espera.
- ***Como usar:*** Após digitar o algoritmo clique em *verify* para verificar se o código está correto. Caso estiver com erros será informado uma mensagem em vermelho na parte inferior do software. A seguir, plugue o cabo USB no Arduino

e no computador, configure o tipo de placa e a porta em que está conectada no menu *Tools*. Por fim, clique em *Upload* para carregar o código na placa.

- ***Aspectos para se notar**:* Perceba os LEDs RX e TX piscando após clicar no menu upload (carregar). Eles indicam a comunicação entre o computador e o Arduino. Alguns Arduinos já vêm com um LED instalado na própria placa conectado ao pino digital 13, desta forma é possível fazer um teste inicial sem a necessidade de realizar uma montagem. Nó código, as barras duplas (//) indicam comentários de linha. Os comentários não são levados em consideração pelo IDE, sevem apenas para documentarmos trechos de código para outro usuário. Existe também o bloco de comentário que é iniciado por /* e terminado por */ onde podemos digitar comentários em múltiplas linhas.
- ***Ideias para experimentar***: Um desafio interessante para quem esta iniciando é alterar o código para que os 2 LEDs pisquem de forma alternada. Esta tarefa ajuda a compreender como funciona o fluxo de execução dos comandos. É possível utilizar uma fonte externa para alimentar a placa, desta forma o cabo USB pode ser desplugado e o experimento fica independente do computador e livre para ser transportado.
- ***Estendendo o experimento***: Adicione mais um LED (vermelho) e simule o funcionamento de um Semáforo.
- ***Características educacionais***: Segundo Paulo Freire [Freire, 1970] o aluno se interessa mais pelo problema contextualizado do que quando este é "fabricado" pelo professor. Alguns exemplos de problemas contextualizados que utilizam LEDs são: Trabalhar cidadania com o uso de semáforos em maquetes da própria cidade; Construção de painéis utilizando LEDs para representar gráficos estatísticos e dinâmicos de temas que estiverem em alta, como por exemplo, índice de contaminação por dengue; Mapas com trajetórias de grandes navegações construídas com segmentos de LEDs que vão se acendendo em sequência, para trabalhos sobre História do Brasil; Mapas do Brasil com LEDs em cada região do país que vão sem apagando em um intervalo de tempo para simular o desmatamento na região com o passar dos anos, para trabalhos de Geografia.
- ***Experimentos relacionados***: O IDE Arduino contém em sua lista de exemplos, um exemplo básico chamado Blink (Arquivo -> Exemplos -> Basics -> Blink) que faz um LED piscar. Outro exemplo interessante é o Fade (Arquivo -> Exemplos -> Basics -> Fade) que mostra como controlar a intensidade do brilho de um LED.

### 1.5.2. Experimento 2 – Sensor de Luminosidade

- ***O que é***: Este experimento é baseado na característica do LDR (do inglês *Light Dependent Resistor* ou em português Resistor Dependente de Luz) em alterar o valor de sua resistência elétrica conforme a luz incidente.
- ***Material necessário***:
    - 1x LED (LED1);

- 1x resistor de 47Kohms (R1);
- 1x resistor de 390ohms (R2);
- 1x LDR pequeno (R3).

- ***Montagem***: Um terminal de R3 é conectado ao 5V. O outro terminal de R3 é conectado tanto ao pino analógico 0 quanto a uma das extremidades de R1. A outra extremidade de R1 é conectada ao GND. LED1 tem o terminal (+) conectado ao pino digital 9. O terminal (-) de LED1 esta conectado a uma das extremidades de R2. A outra extremidade de R2 é conectada ao GND (Figura 1.15).

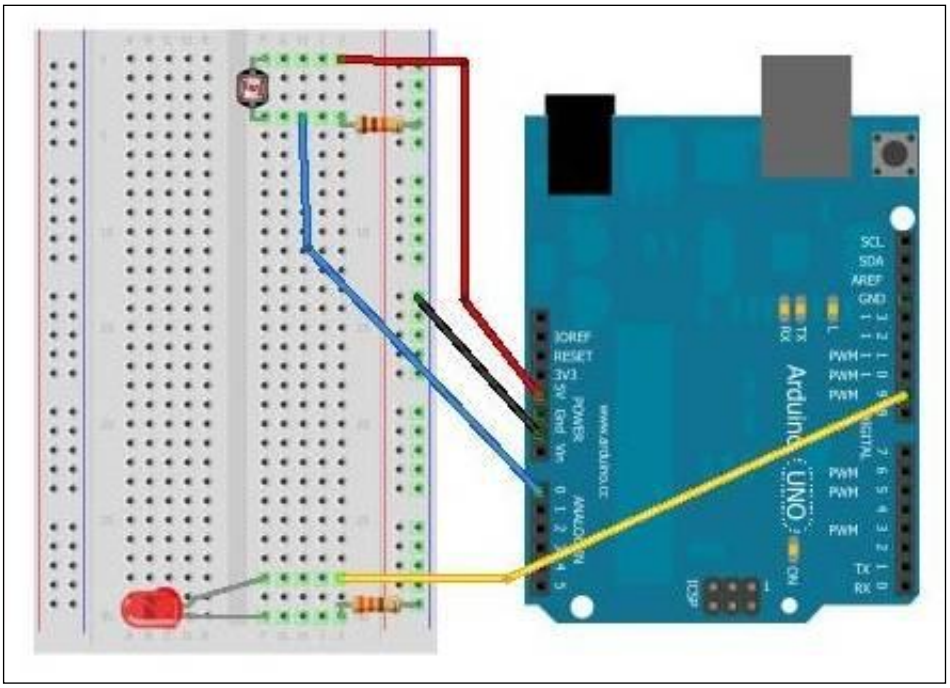



**Figura 1.15: Esquema de montagem Sensor_Luminosidade**

- ***Código***: Carregue no Arduino o código mostrado na Figura 1.16.

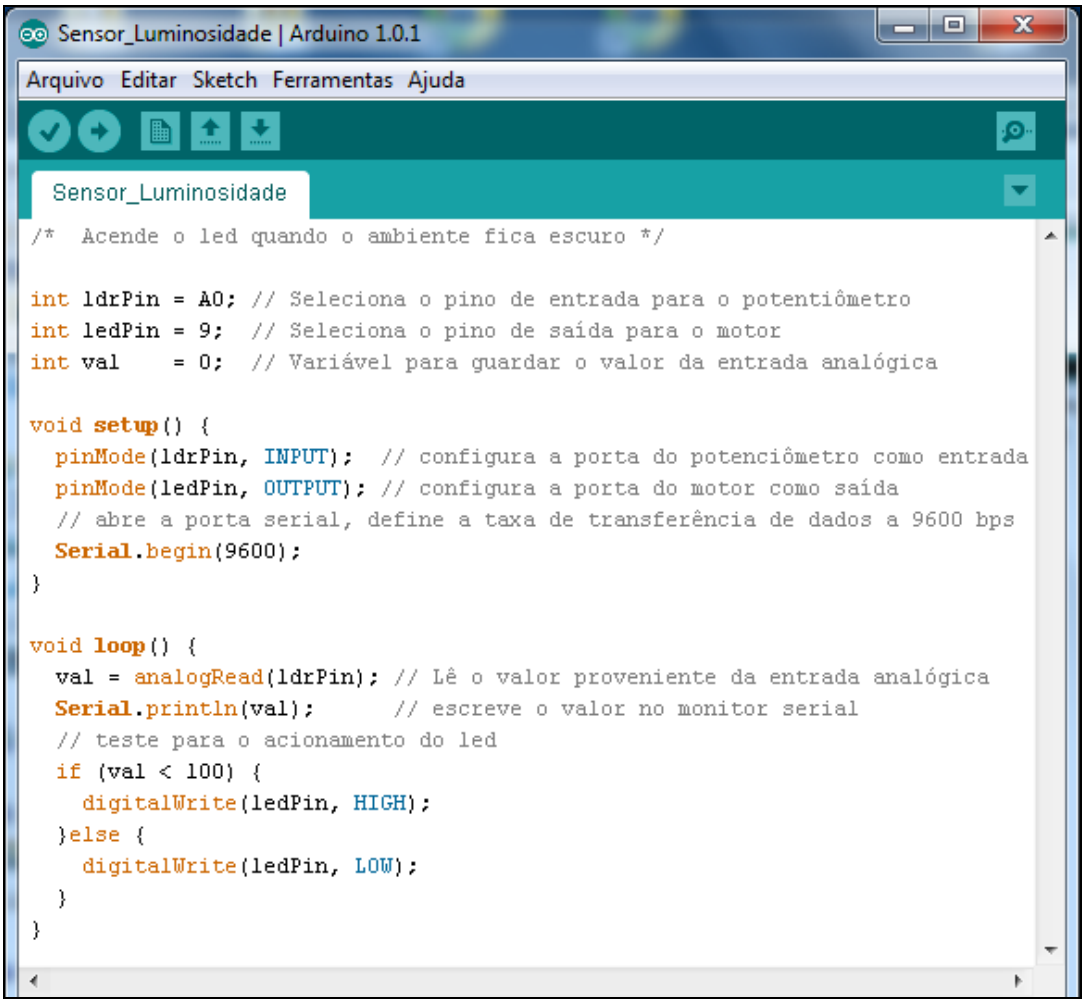



**Figura 1.16: Código fonte Sensor_Luminosidade.ino**

- ***Como funciona***: Quando o ambiente estiver claro, o valor da resistência do LDR (R3) será baixo, fazendo que a energia (tensão) na entrada A0 do Arduino aumente. Com isto, não haverá tensão no pino 9 onde está conectado o LED1 e este não acenderá. Quando o ambiente estiver escuro, o valor da resistência do LDR (R3) aumentará, fazendo com que diminua a tensão na entrada A0 do Arduino. Isso acarretará no aparecimento de uma tensão no pino 9 acendendo o LED1.
- ***Como usar***: Após carregar o programa na placa Arduino, apague a luz do ambiente ou faça sombra com a mão no LDR para perceber o LED1 acendendo.
- ***Aspectos para se notar:*** Se o ambiente já estiver escuro o LED1 já iniciará aceso, então podemos calibrar no código o valor limiar que é 100. Aumentando este valor significa que o experimento estará mais sensível a luz incidente, já que há pouca luz ambiente.
- ***Ideias para experimentar***: Podemos também utilizar um canudo em volta do LDR (R3) e com isso não seria necessário apagar a luz ambiente para testar. O LED1 já estaria aceso, pois o canudo estará fazendo sombra no LDR e assim utilizamos uma lanterna para iluminar dentro do canudo e o experimento reagiria apagando o LED1. Com isso podemos inclusive criar carrinhos acionados por

luz ou qualquer artefato que funcione na presença de luz, mesmo com luz ambiente.

- ***Estendendo o experimento***: Podemos implementar um controle mais preciso da intensidade da luz ambiente utilizando uma sequência de LEDs, onde a medica que se escurece mais LEDs vão se acendendo. Neste caso criaríamos mais valores limiar e assim que eles são ultrapassados acendemos mais um LED.
- ***Características educacionais***: Trabalhar a questão da economia no consumo de energia ao se utilizar os sensores de luminosidade, como acontece nos postes de iluminação pública onde as luzes são acesas automaticamente quando anoitece.

### 1.5.3. Experimento 3 – Sensor de Luminosidade

- ***O que é:*** Neste experimento vamos medir a temperatura ambiente utilizando um termistor como sensor de temperatura e para mostrar o resultado da temperatura aferida vamos utilizar um LED RGB (do inglês *Red, Green and Blue*). O LED RGB tem quatro terminais, um deles é o terra os outros três são os equivalentes as cores vermelho, verde e azul.
- ***Material necessário***:
  - 1x Sensor de Temperatura – Termistor NTC 10kohms – (T1);
  - 2x Resistor – 330ohms – (R1 e R2);
  - 1x Resistor – 10Kohms – (R3);
  - 1x LED RGB – catodo comum – (L1).
- ***Montagem***: T1 tem um terminal conectado ao 5V e o outro terminal esta conectado tanto ao pino analógico 0 quanto a uma extremidade de R1. A outra extremidade de R1 é conectada ao GND (Figura 1.17).

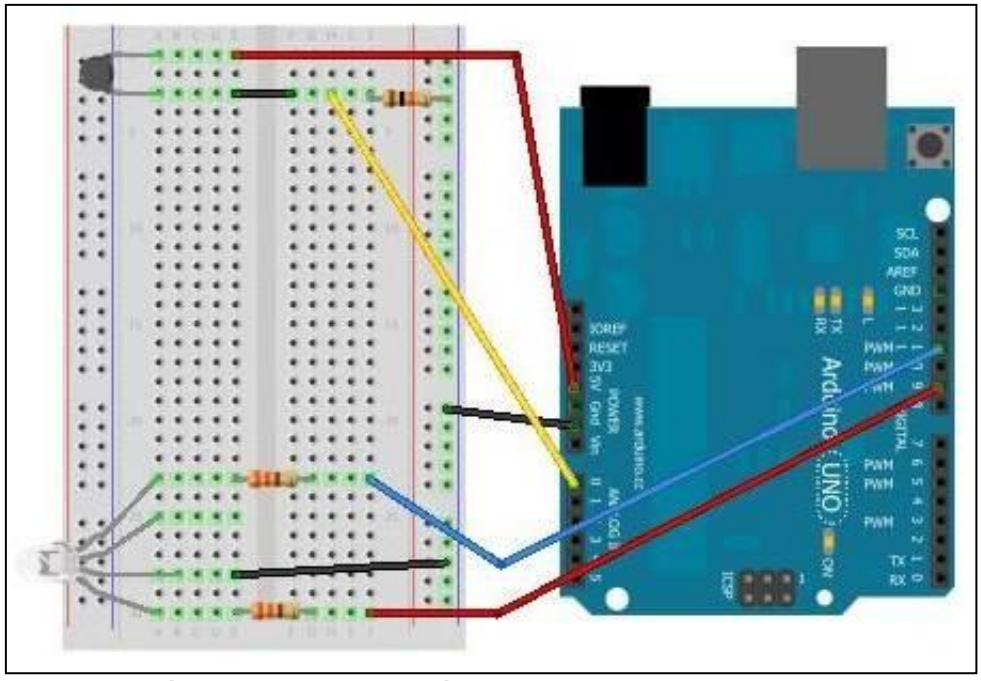



**Figura 1.17: Esquema de montagem Temperatura_RGB**


**Figura 1.18: Código fonte Temperatura_RGB.ino**

- ***Código***: Carregue no Arduino o código mostrado na Figura 1.18.
- ***Como funciona:*** L1 indica a temperatura ambiente medida por T1. Quando a temperatura aumenta a resistência de T1 diminui, fazendo com a tensão na entrada do Arduino aumente. Assim, o Arduino recebe um valor de tensão proporcional a temperatura medida. Para indicação de "quente" e "frio" foi utilizado um LED tipo RGB (L1). Este componente possui três LED's, um para cada cor primária, dentro do mesmo invólucro (corpo). Assumimos neste experimento que a temperatura fria é representada pela luz azul e a temperatura quente pela luz vermelha.
- ***Como usar:*** Apenas esfregando os dedos e depois pegando em T1 com os dedos esquentados, é possível perceber uma pequena variação na tonalidade de L1. Podemos utilizar um ferro de solda para esquentarmos T1. Desta forma, a mudança gradual em L1 da luz azul para a luz vermelha é melhor perceptível.

- ***Aspectos para se notar**:* Utilizamos também o monitor serial para mostrar a temperatura aproximada em graus Celsius. A identificação dos terminais de L1 é realizada através do terminal maior (catodo comum aos outros terminais). Conecte o terminal maior ao GND e, um por vez, conecte os outros terminais ao 5V e note qual luz irá acender. Por precaução conecte R1 em série com o terminal que esta tentando identificar a cor que representa.
- ***Ideias para experimentar***: Para visualizarmos melhor o efeito da luz, sugerimos a criação de um recipiente fosco que envolva L1[10] e também testar em um ambiente escuro.
- ***Estendendo o experimento***: Podemos incrementar o experimento fazendo uma indicação de temperatura máxima. Para isto utilizamos uma Buzina – B1 (*Buzzer* em inglês) que quando a temperatura máxima é atingida B1 emite um sinal sonoro.
- ***Características educacionais***: Podemos trabalhar com este experimento sobre conceito físico de temperatura, sobre meio ambiente e mudanças climáticas, temperatura do corpo humano, etc.

### 1.5.4. Experimento 4 – Motor

- ***O que é:*** Neste experimento iremos controlar a velocidade de um motor com um potenciômetro.
- ***Material necessário***:
  - Motor DC – 5V – (M1)
  - Potenciômetro – 10K – (R1)
- ***Montagem:*** Uma das extremidades de M1 é conectada ao pino digital 9. A outra extremidade de M1 é conectada ao GND. Uma das extremidades de R1 é conectada ao 5V. A outra extremidade de R1 é conectada ao GND. Por fim, o terminal do meio de R1 é conectado ao pino analógico 2 (Figura 1.19).
- ***Como funciona:*** Ao girar o eixo do potenciômetro, dosa-se o valor de tensão na entrada do Arduino (entrada A2) entre 0 e 5 volts. No pino 9 do Arduino está conectado um motor DC que possui sua velocidade dependente da tensão recebida do Arduino. O motor ficará parado quando a tensão no pino 9 for 0 volts e ficará com velocidade máxima quando o pino 9 estiver com 5 volts. Assim, o motor DC terá diferentes velocidades de acordo com a tensão aplicada pelo potenciômetro na entrada do Arduino.
- ***Como usar:*** Girar o eixo do potenciômetro ao longo de todo curso possível e observar a variação da velocidade do motor.
- ***Ideias para experimentar***: Podemos simular um ventilador criando uma hélice e acoplando-a no motor. Além de motores, podemos controlar outros componentes com o potenciômetro, como por exemplo, a intensidade da luz de um LED e o volume do som de uma buzina.

---

[10] http://Arduinobymyself.blogspot.com.br/2012/08/rgb-LED-fading-mood-lights.html

- ***Estendendo o experimento***: Altere o motor DC por um motor Servo. Motores do tipo "servo" não giram de forma contínua como os motores DC. Eles variam seu eixo em um curso de 180º, sendo bastante utilizados para posicionamento de objetos.
- ***Ideias para experimentar***: Podemos simular um ventilador criando uma hélice e acoplando-a no motor. Além de motores, podemos controlar outros componentes com o potenciômetro, como por exemplo, a intensidade da luz de um LED e o volume do som de uma buzina.
- ***Estendendo o experimento***: Altere o motor DC por um motor Servo. Motores do tipo "servo" não giram de forma contínua como os motores DC. Eles variam seu eixo em um curso de 180º, sendo bastante utilizados para posicionamento de objetos.
- ***Características educacionais***: Podemos criar carrinhos com motores DC e trabalhar em sala de aula conceitos de velocidade e aceleração. Com o motor servo podemos criar braços mecânicos e trabalhar conceitos de ângulo. Na robótica educacional há uma forte necessidade de interação em grupo. "Não é impossível, mas um trabalho de robótica educacional levado a cabo apenas por um aluno terá grande chance de insucesso, portanto a colaboração é indispensável" [Silva & Grochocki, 2009]. Assim, também podemos explorar as habilidades e gostos pessoais de cada aluno na distribuição de tarefas.

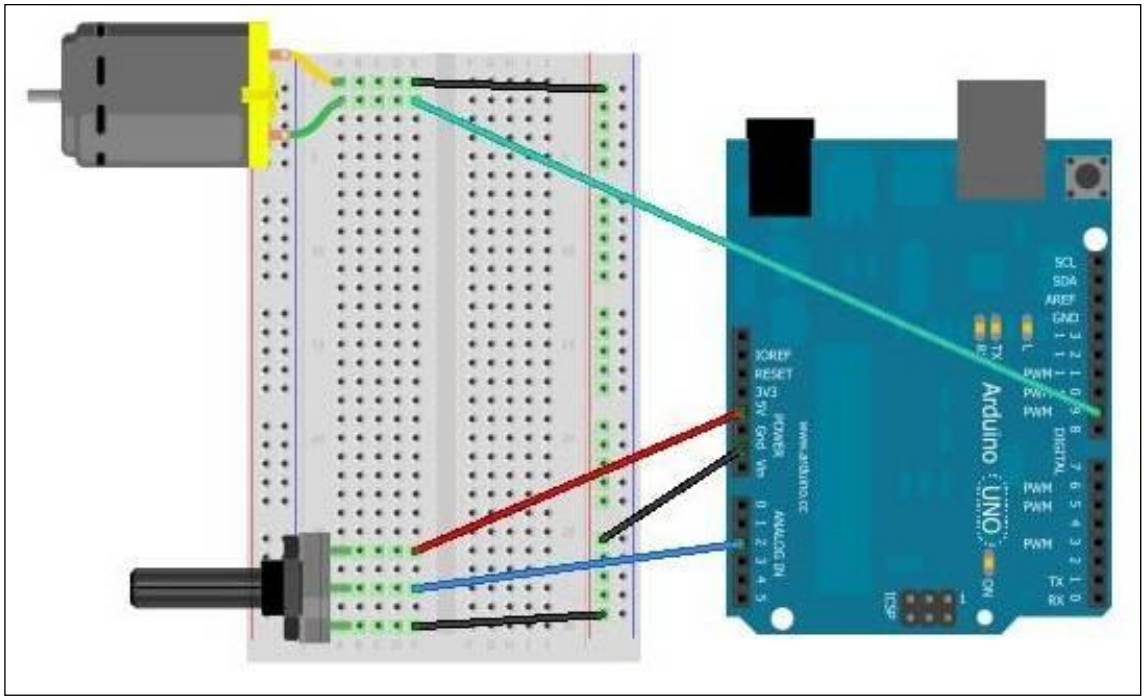



**Figura 1.19: Esquema de montagem MotodDC_Potenciometro**

- ***Código:*** Carregue o código da Figura 1.20 no Arduino.

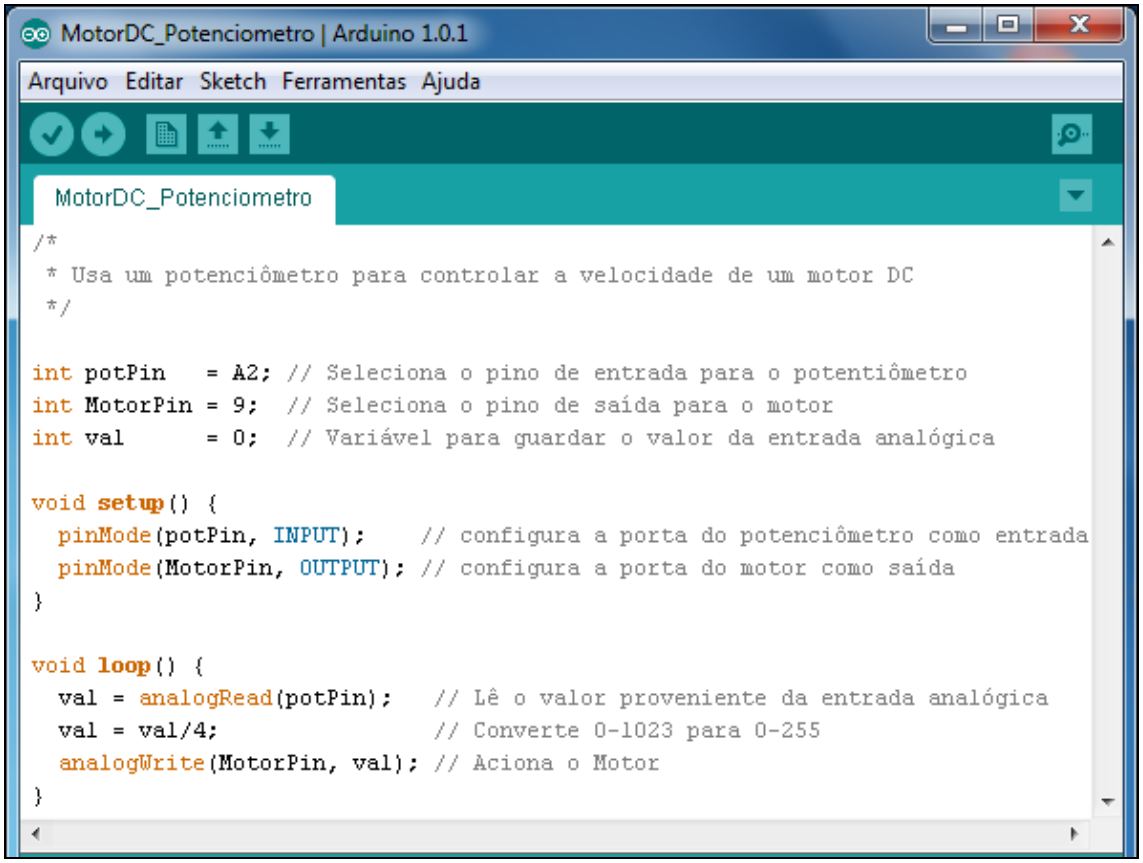



**Figura 1.20: Código fonte MotorDC_Potenciometro.ino**

## 1.6. Linguagem de Programação Visual

A inserção do computador na educação adiciona uma nova dimensão que é a resolução de problemas através de uma linguagem de programação. O objetivo com a programação na robótica educacional não é ensiná-la por si mesma, é usar-la a proveito de algum outro fim educacional. Porém, a linguagem de programação do software Arduino é textual, o que pode ser um obstáculo para os iniciantes. Logo, se a linguagem não é adequadamente adaptada às habilidades de seus usuários, todos os objetivos falharão [Mendelson *et al*., 1990].

Usar uma Linguagem de Programação Visual (ou VPL, sigla em inglês para *Visual Programming Language*) é uma solução para transpor a barreira da programação textual. As VPL fornecem uma metáfora que ajuda o usuário a criar uma ação com o mínimo de treinamento, assim, reduzem a carga cognitiva sobre os estudantes que aprendem sua primeira linguagem de programação [Pasternak, 2009]. A Figura 1.21 mostra do lado esquerdo um exemplo de programação visual e do lado direito o mesmo algoritmo na linguagem textual *Wiring* do software Arduino.

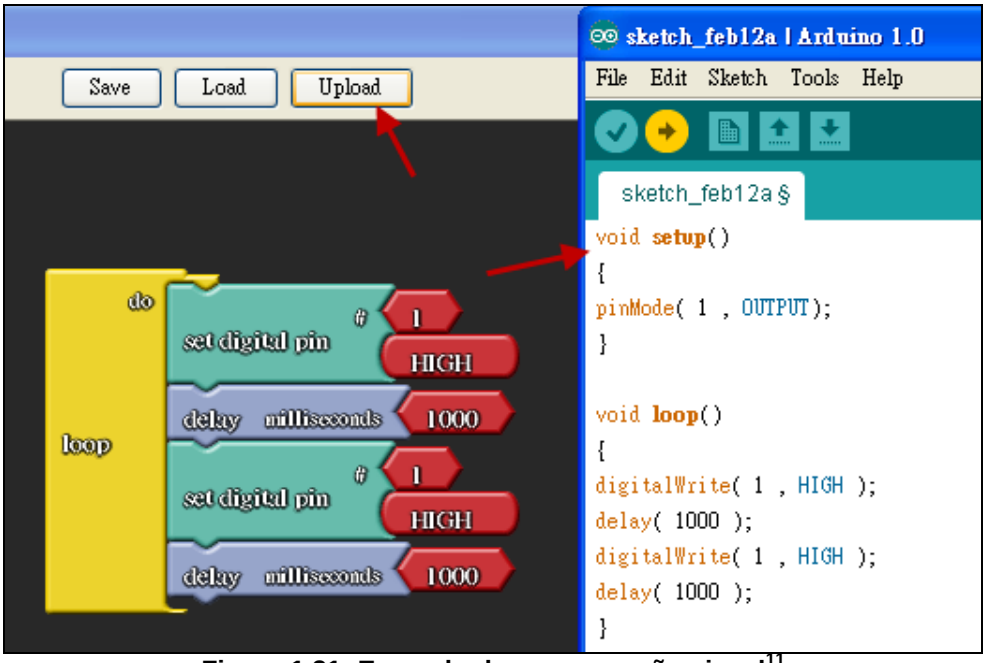



**Figura 1.21: Exemplo de programação visual**[11]

A programação visual é realizada através do mecanismo de arrastar e soltar (*drag and drop*) simplesmente encaixando-se uns nos outros os elementos gráficos, geralmente blocos, formando empilhamentos ordenados. Os blocos serão concebidos para se poderem encaixar apenas de forma que faça sentido sintaticamente, eliminando assim os erros de sintaxe que tanto ocorrem em uma linguagem textual.

## 1.7. Discussão final

Na medida em que os projetos de robótica são aplicados em sala de aula e difundidos pelos grupos de discussão sobre o assunto na web (ex.: fóruns, blogs e wikis), novas demandas vão surgindo. Assim, novas tecnologias são criadas para suprir estas necessidades.

Dois exemplos de soluções sendo desenvolvidas pelo GINAPE/UFRJ no âmbito do Projeto UCA na CUCA (www.nce.ufrj.br/ginape/ucanacuca) financiado pelo CNPq são:

- Ambiente de programação visual que facilita o processo de programação do hardware Arduino por não especialistas.
- Desenvolvimento de um laboratório remoto de robótica educacional com acesso via web possibilitando a realização de experimentos a distância e sem custos iniciais.

Com um pouco de criatividade podemos idealizar aplicações que reúnem um experimento robótico com um conteúdo pedagógico e a partir disso criar projetos educacionais para serem desenvolvidos em sala de aula com os alunos. Nesses ambientes é importante também pensarmos em utilizar elementos do ambiente social em

[11] http://blog.ardublock.com/

que o aluno está inserido como fonte de ideias de problemas a serem resolvidos [Freire, 1970]. Cabe ressaltar ainda a importância de termos professores bem preparados para atuar nestes novos ambientes de ensino-apredizagem.

## 1.8. Referências

Albuquerque, A. P.; Melo, C. M.; César, D. R. e Mill, D. (2007) "Robótica Pedagógica Livre: Instrumento de Criação, Reflexão e Inclusão Sócio-digital". Em *XVIII Simpósio Brasileiro de Informática na Educação. São Paulo.*

Ackermann E. (2001) "Piaget's Constructivism, Papert's Constructionism: What's the difference?" Constructivism: Uses and perspectives in Education, Vol. 1 & 2. Conference Proceedings. Geneva: Research Center in Education / Cahier 8. September. http://learning.media.mit.edu/content/publications/EA.Piaget%20_%20Papert.pdf, Agosto.

Cruz, M. K.; Haetinger, W.; Horn F.; Carvalho, D. V. e Araújo, G. H. (2009) "Controle de Kit de Robótica através de Laboratório Remoto pela Internet: uma Aplicação para a Formação Docente e para a Educação Básica". Em *XX Simpósio Brasileiro de Informática na Educação. Florianópolis.*

D'Abreu, J. V. V. e Chella, M. T. (2001) "Ambiente Colaborativo de Aprendizagem a Distância Baseado no Controle de Dispositivos Robóticos". Em *XII Simpósio Brasileiro de Informática na Educação. Vitória*.

Filho, D. A. M. e Gonçalves, P. C. (2008) "Robótica Educacional de Baixo Custo: Uma Realidade para as Escolas Brasileiras". Em *XXVIII Simpósio Brasileiro de Informática na Educação. Belém do Pará.*

Freire, P. "Pedagogia do oprimido". Rio de Janeiro: Paz e Terra, 1970.

Human Interface Device-HID. Disponível em http://www.usb.org/developers/hidpage/ e http://en.wikipedia.org/wiki/Human_interface_device, Agosto.

Kenski, V. M. Novas tecnologias: o redimensionamento do espaço e do tempo e os impactos no trabalho docente. Informática Educativa, Bogotá, v. 12, n. 1, p.35 - 52, 1999.

Mendelson, P.; Green, T. R. G.; Brna, P. (1990) Programming languages in education: the search for an easy start. In Hoc, J., Green, T., Gilmore, D. & Samway, R. (eds) Psychology of Programming, 175-200, London, Academic Press.

Miranda, L. C.; Sampaio, F. F. e Borges, J. A. dos S. (2007) "ProgrameFácil: Ambiente de Programação Visual para o Kit de Robótica Educacional RoboFácil". Em *XVIII Simpósio Brasileiro de Informática na Educação. São Paulo*.

Miranda, L. C.; Sampaio, F. F. e Borges, J. A. dos S. (2010) "RoboFácil: Especificação e Implementação de um Kit de Robótica para a Realidade Educacional Brasileira". Em *Revista Brasileira de Informática na Educação, Volume 18, Número 3.*

Pasternak, E. "Visual Programming Pedagogies and Integrating Current Visual Programming Language Features". Carnegie Mellon University Robotics Institute.

Thesis Master's Degree. 2009. Disponível em: http://www.ri.cmu.edu/pub_files/2009/8/Thesis-1.pdf. Acesso em: setembro de 2012.

Piaget, J. e Inhelder, B. A. (1970) "Construção do Real na Criança." Trad. Álvaro Cabral. Rio de Janeiro: Zahar. 360 p.

Piaget, J.; Beth, W. E. e Mays,W. (1974) "A Epistemologia Genética e a Pesquisa Psicológica." Rio de Janeiro: Freitas Bastos.

Pinto, M. C. (2011) "Aplicação de arquitetura pedagógica em curso de robótica educacional. com hardware livre" Rio de Janeiro: UFRJ. Dissertação (Mestrado em Informática). Instituto de Matemática, Núcleo de Computação Eletrônica, Universidade Federal do Rio de Janeiro. http://www.nce.ufrj.br/ginape/paginas/teses.html, Agosto.

Santos, F. L., Nascimento, F. M. S., Bezerra, R. M. S. (2010) "REDUC: A Robótica Educacional como Abordagem de Baixo Custo para o Ensino de Computação em Cursos Técnicos e Tecnológicos" Em *XVI Workshop Sobre Informática na Escola – WIE. Belo Horizonte.*

Sasahara, L. R. e Cruz, S. M. S. (2007) "Hajime – Uma nova abordagem em robótica educacional". Em *XVIII Simpósio Brasileiro de Informática na Educação. São Paulo.*

Schons, C.; Primaz, E. e Wirth, G. A. P. (2004) "Introdução a Robótica Educativa na Instituição Escolar para alunos do Ensino Fundamental da disciplina de Língua Espanhola através das Novas Tecnologias de Aprendizagem". Em *Anais do I Workshop de Computação da Região Sul.*

Silva, R. B.; Grochocki, L. R. (2009). "Robótica Educacional". Editora: Barbosa E Silva & Grochocki Ltda.

Souza, M. B.; Netto, J. F. M.; Alencar, M. A. S. e Silva, M. M. (2011) "Arcabouço de um Ambiente Telerobótica Educacional Baseado em Sistemas Multiagentes". Em *XXII Simpósio Brasileiro de Informática na Educação. Aracajú.*

Victorino, L.; Elia, M.; Gomes, A.; Castro, M. e Bastos, C. (2009) "Laboratório Virtual de Atividades Didáticas – LabVad". http://www.lbd.dcc.ufmg.br/colecoes/wie/2009/022.pdf, Agosto.

Vygotsky, L. S. (1984) "Formação Social da Mente". São Paulo, Editora Martins Fontes. (1993) "Pensamento e Linguagem". São Paulo, Editora Martins Fontes.

Zilli, S. R. (2004) "A Robótica Educacional no Ensino Fundamental: Perspectivas e Prática." Dissertação de Mestrado, Universidade Federal de Santa Catarina.