UNIVERSIDADE FEDERAL DE UBERLÂNDIA

FACULDADE DE ENGENHARIA ELÉTRICA

DIVINO ROSA DA SILVA FERREIRA

POTENCIALIZANDO O INTERESSE DOS NOVOS ALUNOS, NATIVOS DIGITAIS, NOS CURSOS DE ENGENHARIA DE COMPUTAÇÃO E AFINS, ATRAVÉS DE UM PROCESSO ATIVO DE EVOLUÇÃO DAS HABILIDADES E COMPETÊNCIAS JÁ EXISTENTES EM VEZ DE UM PROCESSO DE RECONSTRUÇÃO.

UBERLÂNDIA – MG

SETEMBRO - 2016

DIVINO ROSA DA SILVA FERREIRA

POTENCIALIZANDO O INTERESSE DOS NOVOS ALUNOS, NATIVOS DIGITAIS, NOS CURSOS DE ENGENHARIA DE COMPUTAÇÃO E AFINS, ATRAVÉS DE UM PROCESSO ATIVO DE EVOLUÇÃO DAS HABILIDADES E COMPETÊNCIAS JÁ EXISTENTES EM VEZ DE UM PROCESSO DE RECONSTRUÇÃO.

Dissertação de Mestrado apresentada como requisito parcial à obtenção do grau de Mestre em Ciências, pelo programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Uberlândia.

Orientador: Dr. Prof. Luciano Vieira Lima

UBERLÂNDIA – MG

SETEMBRO - 2016

Ficha Catalográfica Online do Sistema de Bibliotecas da UFU
com dados informados pelo(a) próprio(a) autor(a).

F383 Ferreira, Divino Rosa da Silva, 1978-
2016 POTENCIALIZANDO O INTERESSE DOS NOVOS ALUNOS, NATIVOS DIGITAIS, NOS CURSOS DE ENGENHARIA DE COMPUTAÇÃO E AFINS, ATRAVÉS DE UM PROCESSO ATIVO DE EVOLUÇÃO DAS HABILIDADES E COMPETÊNCIAS JÁ EXISTENTES EM VEZ DE UM PROCESSO DE RECONSTRUÇÃO [recurso eletrônico] / Divino Rosa da Silva Ferreira. - 2016.

Orientador: LUCIANO VIEIRA LIMA.
Dissertação (Mestrado) - Universidade Federal de Uberlândia, Pós-graduação em Engenharia Elétrica.
Modo de acesso: Internet.
Disponível em: http://dx.doi.org/10.14393/ufu.di.2019.2211
Inclui bibliografia.
Inclui ilustrações.

1. Engenharia elétrica. I. VIEIRA LIMA, LUCIANO , 1960-, (Orient.). II. Universidade Federal de Uberlândia. Pós-graduação em Engenharia Elétrica. III. Título.

CDU: 621.3

Bibliotecários responsáveis pela estrutura de acordo com o AACR2:
Gizele Cristine Nunes do Couto - CRB6/2091
Nelson Marcos Ferreira - CRB6/3074

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE UBERLÂNDIA
FACULDADE DE ENGENHARIA ELÉTRICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Ata da defesa de DISSERTAÇÃO DE MESTRADO junto ao Programa de Pós-graduação em Engenharia Elétrica da Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia.

Defesa de Dissertação de Mestrado Acadêmico, número 649/2016/PPGEE
Data: 07 de outubro de 2016.
Discente: DIVINO ROSA DA SILVA FERREIRA
Número de matrícula: 11322EEL004
Título do Trabalho: Potencializando o interesse de novos alunos, nativos digitais, nos cursos de Engenharia da Computação e afins, através de um processo ativo de evolução das habilidades e competências já existentes ao invés de um processo de reconstrução.
Área de concentração: Processamento da Informação.
Linha de pesquisa: Inteligência artificial.

As 14:00 horas do dia 07 de outubro de 2016 na Sala de Defesas da Faculdade de Engenharia Elétrica, Campus Santa Mônica da Universidade Federal de Uberlândia, reuniu-se a Banca Examinadora, designada pelo Colegiado do Programa de Pós-graduação em Engenharia Elétrica, assim composta:

Keiji Yamanaka	CPF: 005.225.308-26
Carlos Alberto Lopes Silva	CPF: 603.954.516-04
Luciano Vieira Lima	CPF: 431.696.866-15, orientador do candidato.

Iniciando os trabalhos o presidente da mesa Prof. Dr. Luciano Vieira Lima apresentou a Comissão Examinadora e o candidato, agradeceu a presença do público, e concedeu ao Discente a palavra para a exposição do seu trabalho. A duração da apresentação do Discente e o tempo de argüição e resposta foram conforme as normas do Programa.

A seguir o senhor presidente concedeu a palavra, pela ordem sucessivamente, aos examinadores, que passaram a argüir o candidato. Ultimada a argüição, que se desenvolveu dentro dos termos regimentais, a Banca, em sessão secreta, atribuiu os conceitos finais.

Em face do resultado obtido, a Banca Examinadora considerou o candidato A provado.

Esta defesa de Dissertação de Mestrado Acadêmico é parte dos requisitos necessários à obtenção do título de Mestre. O competente diploma será expedido após cumprimento dos demais requisitos, conforme as normas do Programa, a legislação pertinente e a regulamentação interna da UFU.

Nada mais havendo a tratar, foram encerrados os trabalhos às 15 horas e 00 minutos. Foi lavrada a presente ata que após lida e achada conforme foi assinada pela Banca Examinadora.

Prof. Dr. Luciano Vieira Lima
UFU

Prof. Dr. Carlos Alberto Lopes Silva
IFTM

Prof. Dr. Keiji Yamanaka
UFU

UNIVERSIDADE FEDERAL DE UBERLÂNDIA
CONFERE COM O ORIGINAL
ASS.
FEELT - PPGEE

DEDICATÓRIAS

Ao meu pai Antonio Ferreira de Sousa – In Memoriam. Acompanhou de perto todo o meu esforço, minhas lutas e conquistas. Foi uma excelente pessoa, profissional e pai. Soube educar-me para que eu seguisse sua linha de conduta. Meu exemplo.

Ao meu amigo/irmão Edson Maluf que sempre me inspirou a buscar o conhecimento e colocá-lo em prática a todo custo. É uma referência em todos os sentidos. Tem meu repeito. Meu orgulho.

AGRADECIMENTOS

Ao orientador e professor Dr. Luciano Vieira Lima pela oportunidade oferecida. Foi de grande valor o aprendizado ao longo do tempo. Experiências adquiridas através de práticas executadadas tanto em sala de aula como em laboratório fizeram com que eu encarasse os desafios e assumisse as responsabilidades na posição de um membro docente da instituição. Tive a honra de participar de projetos liderados por ele.

Ao amigo Dr. Carlos Alberto Lopes da Silva o qual foi meu companheiro de laboratório, pesquisas e desenvolvimento. Foram bons tempos trabalhando arduamente para se conseguir bons resultados no projeto em andamento. Época essa que foi de muito aprendizado.

Aos funcionários da instituição que tiveram contato direto comigo e que sempre me atenderam e ajudaram-me no que foi preciso. Por várias vezes se prontificaram a organizar as documentações e orientar-me no que era necessário. Alem de agradecido sou grato a eles.

Aos familiares que apoiaram e estiveram comigo durante todo o tempo, colaborando das mais variadas formas.

A minha noiva Patrícia Aparecida Braz que sempre esteve ao meu lado, apoiando, tendo a devida paciência e superando obstáculos sempre comigo. Obrigado por estar em minha vida.

RESUMO

A diferença entre o que se deseja e o que se encontra ao ingressar em um curso de engenharia faz com que os novos alunos tenham uma desmotivação precoce com o curso. O presente trabalho apresenta uma metodologia que mantem o aluno em seu mundo digital não o afastando de seu dispositivo móvel (celular), mas sim fazendo dele uma ferramenta para o aprendizado. Através de práticas em laboratório o aluno passa a ter desde o inicio do curso o resultado real do que lhe é ensinado. Com isso o aluno mantem o estímulo e se interessa por cada nova experiência apresentada no decorrer da disciplina. Ao final do período o aluno já é capaz de desenvolver projetos úteis de aplicação residencial e/ou comercial. O que se espera de um aluno de engenharia é que ele seja ao final do curso um engenheiro. A proposta do trabalho é torna-lo engenheiro desde seu ingresso.

Palavras-chave: Desmotivação Precoce. Metodologia. Aprendizado. Estímulo.

ABSTRACT

The difference between what is desired and what is to join an engineering course makes new students have an early demotivation with the course. This paper presents a methodology that keeps the student in your digital world not away from your mobile device (cell phone), but making it a tool for learning. Through hands-on lab the student is replaced since the beginning of the course the real result of what you and taught. Thus the student keeps the stimulus and interest in each new experience presented during the course. At the end of the period the student is already able to develop useful projects of residential and / or commercial application. What is expected of a student of engineering is that it is the end of the course an engineer. The proposed work is to engineer it from your ticket.

Keywords: Early Demotivation. Methodology. Learning. Stimulus.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ATMEL:	**Manufaturadora de microncontroladores**
AVR:	**Microcontrolador de chip único com arquitetra harvard desenvolvido pela atmel.**
IDE:	**Integrated development environment ou ambiente de desenvolvimento integrado, programa de computador voltado ao desenvolvimento.**
VBB:	**Virtual breadboard, programa de computador para simulação de funcionamento de circuito.**
PAYPAL:	**Empresa de pagamentos online**
MIT:	**Massachusets institute of technology – Universidade privada de pesquisa localizada em Cambrigde.**

Sumário

1 INTRODUÇÃO

Este trabalho tem o objetivo comprovar que os conceitos aplicados com uma metodologia correta gera resultados satisfatórios no que se refere a transmissão de conhecimento e absorção de conteúdo através de práticas construídas mediante teoria ensinada. O objeto de estudo foram turmas de alunos dos cursos de tecnologia da UFU e culminou para a adoção da metodologia na disciplina de Introdução à Engenharia no curso de Engenharia da Computação.

1.1 Justificativa

As pessoas vêm com bagagem totalmente mobile. A utilização de dispositivos móveis faz parte do dia a dia do aluno e que a utilização de computadores e notebooks está cada vez mais em desuso. O que queremos é manter a rotina e utilização do dispositivo pelo aluno da disciplina de Introdução à Engenharia da Computação sem tirá-lo do conforto, acrescentar uma forma útil de aplicação. Aprenderão o conceito mínimo necessário, pois ofereceerei uma maneira simples de aprendizagem uma vez que posteriormente o aluno terá outras disciplinas com conhecimento mais profundo. Inicialmente o aluno conhecerá os dispositivos com os conceitos corretos desde o primeiro dia.

O aluno de posse de seu dispositivo móvel fará o controle de cargas em sua própria residência. Ele aprenderá sobre questões de segurança e confiabilidade do controle como, por exemplo, saber se o acionamento de um a carga foi realmente feita. Da mesma forma que se deve preocupar com o a resposta do acionamento deve-se preocupar com a resposta equivocada como exemplo podemos citar com o que ocorre em alarmes, onde os vizinhos ao ouvirem o alarme geralmente associam a um disparo acidental, geralmente um gato ou outro animal e neste caso não dando a devida importância para o fato uma vez que se têm mais acionamento equivocados do que os realmente corretos. O correto seria o disparo do alarme por apenas por intruso, sendo assim, um sistema inteligente que distingue a presença causadora do disparo do alarme. Outro exemplo do cotidiano pode ser visto no acionamento de uma lâmpada no carro (faróis, setas ou lanternas). A indicação no painel não retrata a realidade externa ao veículo. Podem-se ter lâmpadas queimadas e mesmo assim o indicativo no painel diz que foi acionado e esta funcionando. Com base nestes critérios o aluno irá aprender a desenvolver sistemas confiáveis com resposta em tempo real tendo a certeza de que o

acionamento oferece a resposta confiável. Para isso foi desenvolvido kits de desenvolvimentos para que os alunos possam desenvolver sistemas de forma prática e com o mínimo de conhecimento necessário. Tais conhecimentos o aluno já traz em sua bagagem educacional como, por exemplo, conceitos sobre tensão, resistência, corrente e potência os quais abordaremos a titulo de revisão, mas com uma metodologia própria voltada para a parte prática com analogias entendíveis. Isso faz com que o aluno aprenda desde o inicio a projetar um sistema de forma correta e não simplesmente reproduzir o que é visto na Internet, evitando assim riscos a saúde e danos materiais.

O objetivo é passar o conhecimento despertando no aluno o interesse pela disciplina e consequentemente pelo curso fazendo com que ele se torne um profissional competente.

1.2 Organização do trabalho

O presente trabalho encontra-se organizado em 10 (dez) capítulos

Capitulo 1 - Justificativa e objetivo explicando o cenário atual e o que se pretende através da aplicação dos métodos contidos no trabalho.

Capítulo 2 - Conceitos básicos utilizando analogias para melhor entendimento e compreensão do aluno.

Capítulo 3 – Sistema de numeração para que o aluno saiba interpretar e diferenciar a principal diferença entre número e numeral, preparando-o para conceitos futuros relativos a lógica de programação.

Capítulo 4 – Escolhendo e justificando a escolha da plataforma Arduino para o aprendizado.

Capítulo 5 – Simulando o Arduino, uma solução econômica e pratica para o aluno que ainda não dispuser do material.

Capítulo 6 – Escolhendo e justificando a escolha da linguagem de programação para desenvolvimento de aplicativo para dispositivo móvel na plataforma Android.

Capitulo 7 – Conceitos introdutórios importantes sobre os conceitos de Cliente/Servidor e Sistema em Tempo Real.

Capítulo 8 – Experiencias práticas com várias tecnologias (bluetooth, ethernet, wifi).

Capítulo 9 – Conclusões sobre todo o trabalho desenvolvido (prática e teoria) com alunos em 2 anos de pesquisa e aplicação.

Capítulo 10 – Anexos importantes citados no trabalho para auxílio e sedmentação do conhecimento.

2 Conceitos Básicos de Eletricidade e Eletrônica com analogia Hidráulica

A melhor forma de entender o funcionamento do fluxo elétrico é compararmos com o fluxo hidráulico uma vez que o fluido é visível e elétrons não são.

Quando pensamos em fluxo devemos entender que existe um elemento que cause o fluxo, ou melhor, a condução. Na hidráulica utilizam-se frequentemente canos ou mangueiras, já na eletricidade utilizam-se fios ou cabos. Vamos entender como funciona o fluxo hidráulico

2.1 Condução Hidráulica Contínua

Tomemos como exemplo duas caixas d'água (Figura 1)

Figura 1 - Caixas d'água

Fonte: Acervo Pessoal

Para transferir água de uma caixa para outra precisaremos de um condutor de água, como por exemplo, um cano (Figura 2).

Figura 2 - Cano

Fonte: Acervo Pessoal

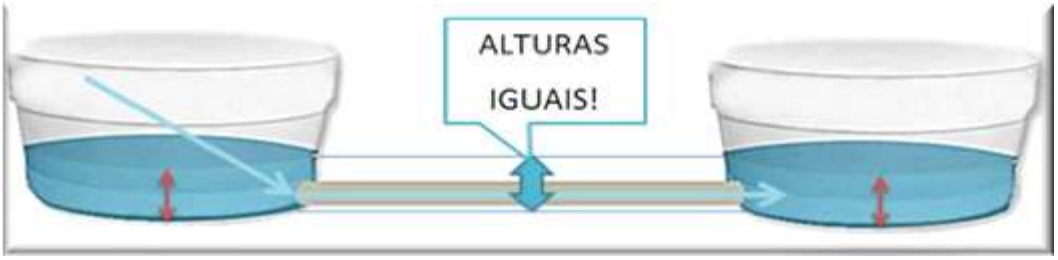
O cano conduz com facilidade a água, pois ele é oco e liso não oferecendo oposição ao fluxo de água. Se colocarmos o cano na parte superior da caixa somente um pouco de água irá passar de uma caixa para a outra e o fluxo ira parar rapidamente fazendo com que o objetivo não seja atingido, pois não se transferirá toda a quantidade de uma caixa para outra (Figura 3).

Figura 3 - Caixas d'água com cano na parte superior

Fonte: Acervo Pessoal

Se colocarmos o cano na parte inferior as duas caixas ficarão com a mesma quantidade de água e quando isso acontecer o fluxo deixara de acontecer e o objetivo não será atingido pois não se transferiu toda a quantidade de uma caixa para outra.

Figura 4 - Caixas d'água com cano na parte inferior



Fonte: Acervo Pessoal

Se colocarmos as caixas em alturas diferentes (caixa com água mais alta que a caixa sem água) a água irá fluir para a caixa mais baixa (figura 5). O tempo gasto neste processo dependerá da espessura interna do cano e da diferença de altura entre as duas caixas.

Figura 5 - Caixas d'água com cano em alturas diferentes

Fonte: Acervo Pessoal

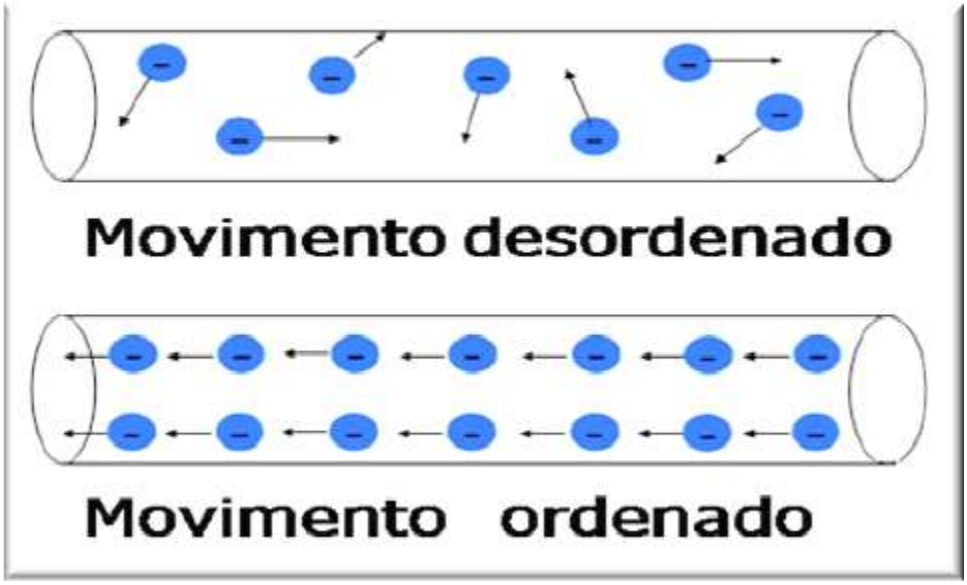
Com isso conclui-se que:

- A água sempre irá da caixa mais alta para a caixa mais baixa desde que haja um condutor interligando-as;
- A agua naturalmente irá do potencial mais alto para o potencial mais baixo;
- Sem diferença de potencial (neste caso a altura) não existe fluxo de água;
- O fluxo acaba quando acabar (esvaziar) a água da caixa de potencial mais alto ou quando for retirado o elemento condutor, neste caso o cano;
- Quanto mais fino o condutor (cano) menor é o fluxo de água e maior é a resistência à passagem de água;
- O fluxo não depende do tamanho das caixas e sim da diferença de potencial (altura) entre elas.

2.2 Condução Elétrica Contínua

Da mesma forma que temos fluxo de água no condutor hidráulico (cano) temos o fluxo ordenado de partículas portadoras de carga elétrica no condutor elétrico (fio). Esse fluxo se dá pela diferença de potencial elétrico nas extremidades do condutor elétrico como pode ser visto na figura 6 a seguir

Figura 6 - Fluxo de partículas portadoras de carga elétrica



Fonte: Acervo Pessoal

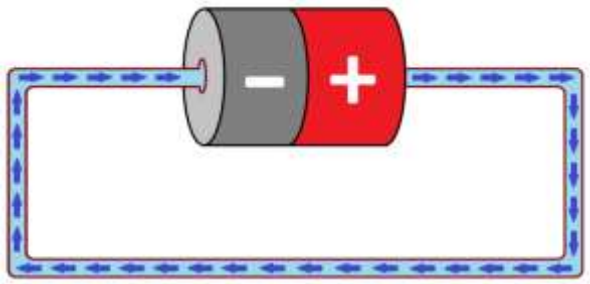
Para que um material seja condutor é preciso que o mesmo atenda uma propriedade física chamada condutividade. Condutividade é a capacidade dos materiais de conduzirem corrente elétrica. Qaunto à condutividade os materiais podem se classificar em:

- **Condutores** – Que conduzem naturalmente corrente elétrica;
- **Semicondutores** – Necessitam de estímulo via diferença de potencial (DDP) para conduzirem;
- **Isolantes** – Não conduzem corrente elétrica.

Nesse capítulo será tratado o funcionamento do material condutor.

Ao se ligar um condutor e uma fonte de energia, como por exemplo, uma pilha, o comportamento dos elétrons no sentido convencional se dará do polo positivo (+) para o negativo (-) conforme figura 2-7 a seguir.

Figura 7 - Sentido convencional de corrente elétrica contínua



Fonte: Acervo Pessoal

A essa forma de movimento ordenado dá-se o nome de corrente elétrica contínua ou corrente direta.

A diferença de potencial elétrico é representada pela tensão e expressa em Volt (V).

O fluxo de elétrons é denominado de corrente elétrica e expressa em Ampere (A)

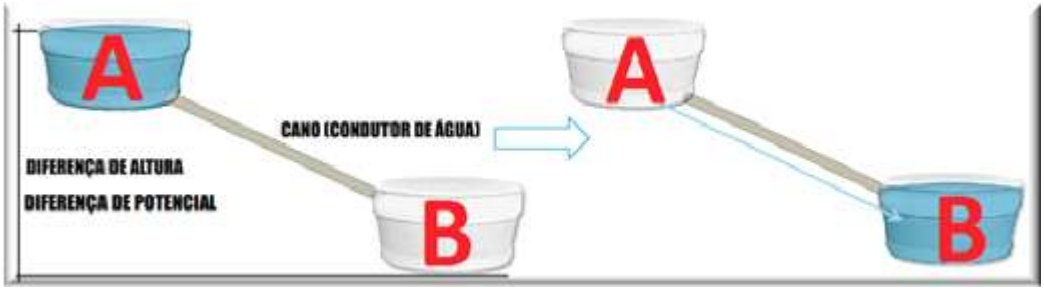
2.3 Condução Hidráulica Alternada

No item "**2.1 – Condução Hidráulica Contínua**" foi demonstrado o fluxo de água de uma caixa d'água A para uma caixa d'água B. Neste caso percebe-se que a água flui apenas em uma direção pois há uma diferença de altura entre as caixas d'água. Agora será demonstrado como se obtém o fluxo nas duas direções.

Supõem-se duas caixas de água conforme apresentadas na figura 8.

Como pode ser visto a caixa d'água A está mais alta que a caixa d'água B. Isso implica no fluxo natural da caixa d'água A para a caixa d'água B.

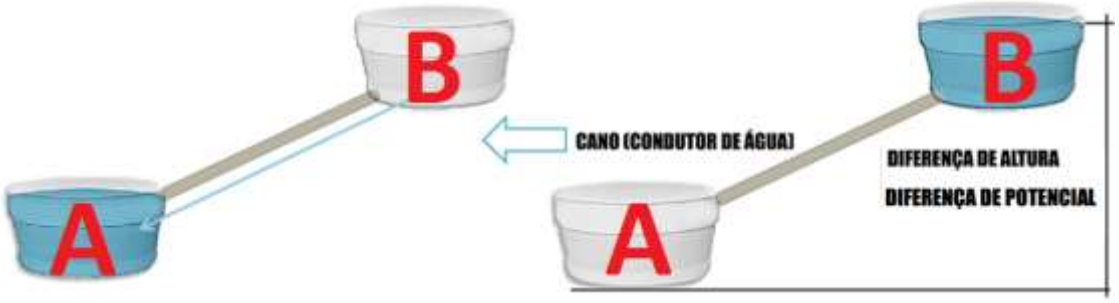
Figura 8 - Fluxo de água da caixa d'água A para a caixa d'água B



Fonte: Acervo Pessoal

Se após a caixa d'água B estiver cheia houver a elevação da altura da mesma em relação a caixa d'água A que vai estar vazia, ocorrerá o processo inverso, o fluxo de água ocorrerá da caixa d'água B para a caixa d'água A conforme figura 9 a seguir.

Figura 9 - Fluxo de água da caixa d'água B para a caixa d'água A



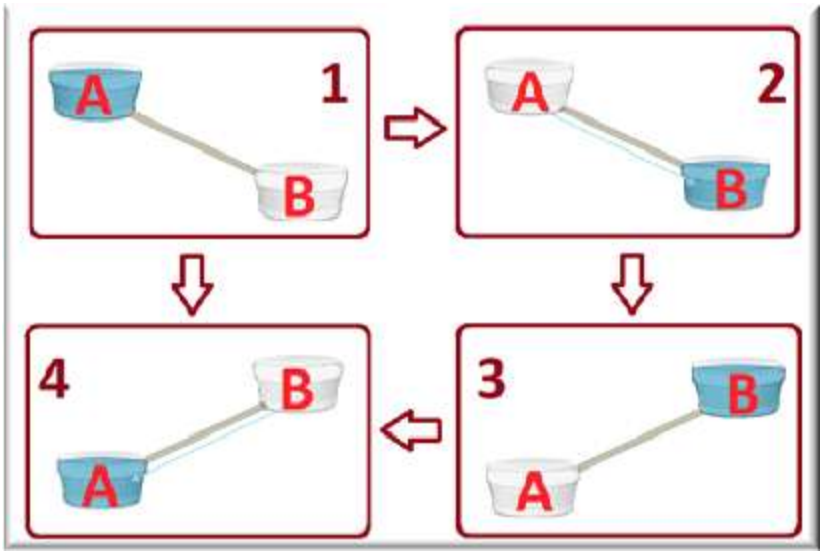
Fonte: Arquivo Pessoal

Se este movimento for constante, a troca de altura entre as caixas assim que a caixa de altura superior estiver vazia, produzirá um fluxo constante ordenado de um lado para o outro conforme as figuras 10 e 11 a seguirem.

Figura 10 - Fluxo de água da caixa d'água A para a caixa d'água B e vice-versa

Fonte: Acervo Pessoal

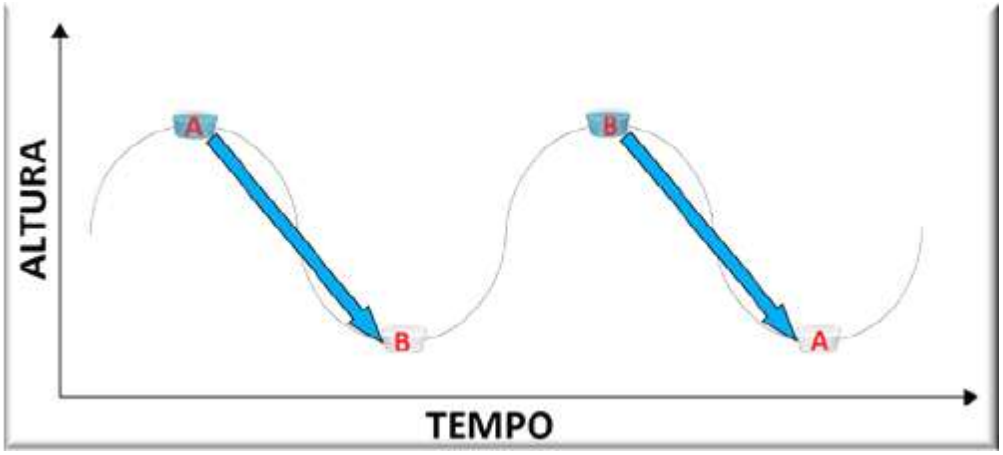
Figura 11 - Ciclo do fluxo de água entre as caixas d'água



Fonte: Acervo Pessoal

Dessa forma é possível imaginar as caixas subindo e descendo continuamente ao longo do tempo conforme figura 12 a seguir.

Figura 12 - Fluxo de água entre as caixas com movimento análogo a senoide



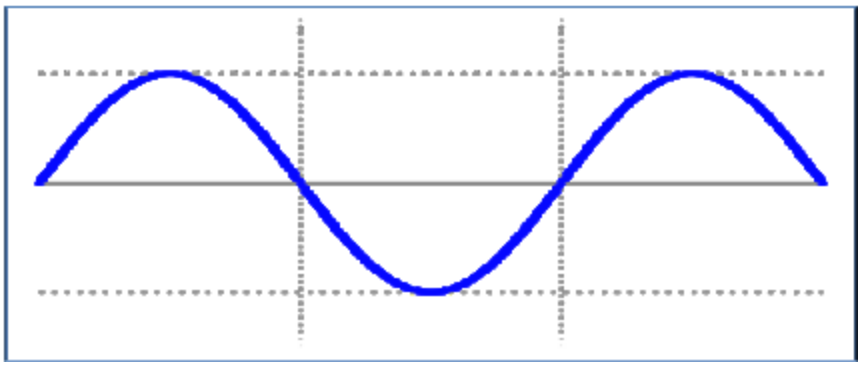
Fonte: Acervo Pessoal

2.4 Condução Elétrica Alternada

Agora que está entendido o fluxo de água de uma caixa d'água A pra outra caixa d'água B e vice-versa em um movimento e vai e vem constante, fica fácil imaginar o movimento dos elétrons quando se trata de corrente alternada.

A corrente elétrica alternada varia o sentido do fluxo dos elétrons ao longo do tempo. A forma de onda mais usual é a senoidal, em forma de senóide conforme figura 13 a seguir.

Figura 13 - Forma de onda senoidal da corrente elétrica



Fonte: http://www.musitec.com.br/revistas/?c=4270

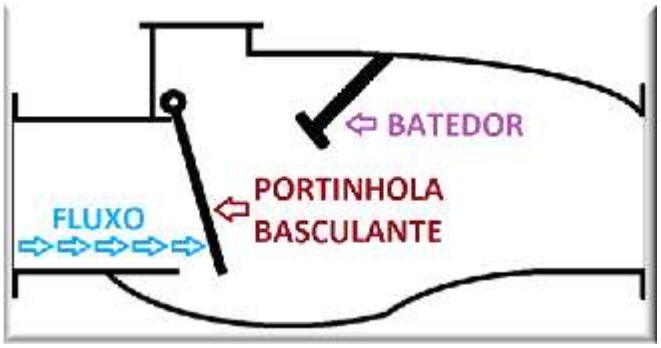
Como os elétrons vão de um potencial mais alto para o potencial mais baixo, fica claro

que essa variação de potencial ocorre constantemente invertendo assim o sentido do fluxo dos elétrons. Na maioria dos países da América, inclusive Brasil e EUA a frequência da rede elétrica é 60 Hz. Isso significa que o fluxo de elétrons inverte seu sentido 60 vezes a cada segundo.

2.5 Fluxo Hidráulico em sentido único

Em instalações hidráulicas às vezes e necessário ordenar o fluxo de água em um único sentido visando com isso a proteção do sistema contra o refluxo da água. Para isso utiliza-se um equipamento chamado válvula de retenção e seu esquema pode ser visto na figura 14 a seguir.

Figura 14 - Esquema mecânico de uma válvula de retenção



Fonte: Acervo Pessoal

O funcionamento é bem simples. Em função do tipo de válvula que possuem permitem o deslocamento da água num só sentido. É necessário observar que a instalação deve ser feita de modo a que a portinhola abra no sentido do fluxo. Caso retorne água no sentido inverso a portinhola se fecha e o fluxo é interrompido.

Esse equipamento pode ser visto na figura 15 a seguir.

Observe que existe uma seta indicando o sentido natural do fluxo e, portanto orientando o modo de instalação.

Figura 15 - Foto de uma válvula de retenção

Fonte: http://buttbrothers-valves.com/index.php?id_category=45&controller=category

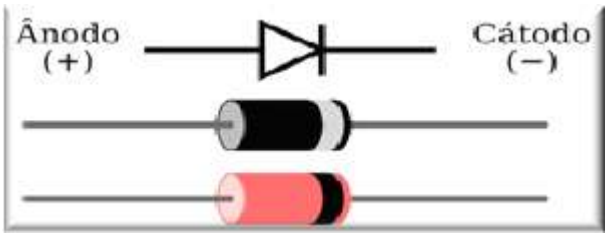
.

2.6 Fluxo de elétrons em sentido único

Em elétrica e eletrônica geralmente se faz necessário a garantia do fluxo de elétrons ou corrente elétrica em um único sentido, ora para proteger o circuito ora para ordenar a corrente elétrica. No primeiro caso temos a proteção e no segundo temos a retificação que nada mais é que a utilização da corrente em um único sentido, descartando assim a corrente no sentido contrário.

Para isso utiliza-se um componente chamado “diodo” e seu símbolo eletrônico bem como a representação de um modelo convencional em eletrônica pode ser visto na figura 2-16 a seguir.

Figura 16 - Simbologia e representação do diodo

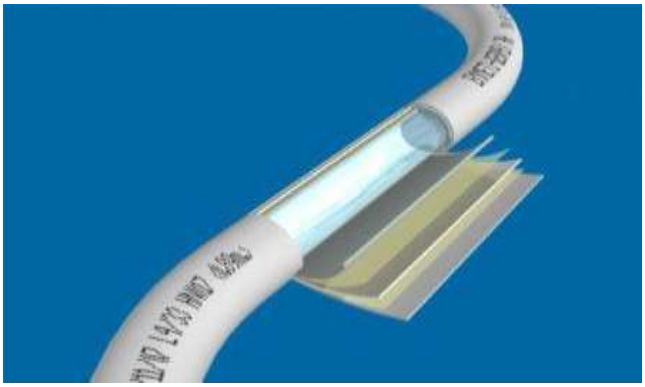


Fonte: https://pt.wikipedia.org/wiki/Diodo_semicondutor

2.7 Resistência Hidráulica

Normalmente uma tubulação não oferece resistência à passagem de fluído, não há obstáculos que dificultam a passagem do fluído em seu interior visto que o material geralmente é liso, sem rugosidades ou imperfeições. Isso quer dizer que a todo o conteúdo será transportado do ponto A ao ponto B conforme figura 2-17 a seguir.

Figura 17 - Passagem de água pelo tudo sem atrito



Fonte: Acervo Pessoal

Dessa forma fica fácil entender que caso haja algum obstáculo no interior do tubo ou mesmo acúmulo de algum tipo de sujeira no mesmo a passagem do fluído ficará comprometida como pode ser visto na figura 18.

Figura 18 - Tubo com obstrução interna

Fonte: http://www.narrowpathplumbing.com/helpful-tips/clogged-drains/

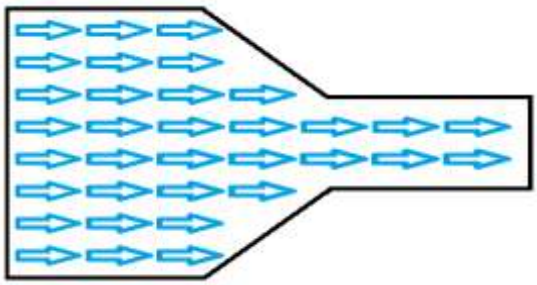
Da mesma forma se o tubo estiver com dano físico, amassado, criará também uma oposição à passagem do fluído conforme figura 19.

Figura 19 - Tubo amassado

Fonte: http://trashandtreasure.weebly.com/latest-finds.html

Às vezes é preciso criar uma resistência hidráulica para que se possa diminuir o fluxo de água em determinadas situações. Para isso emprega-se o uso um componente chamado redução e seu esquema podem ser visto na figura 20 a seguir.

Figura 20 - Esquema de redução hidráulica



Fonte: Acervo Pessoal

Esse dispositivo pode ser visto na figura 21 a seguir.

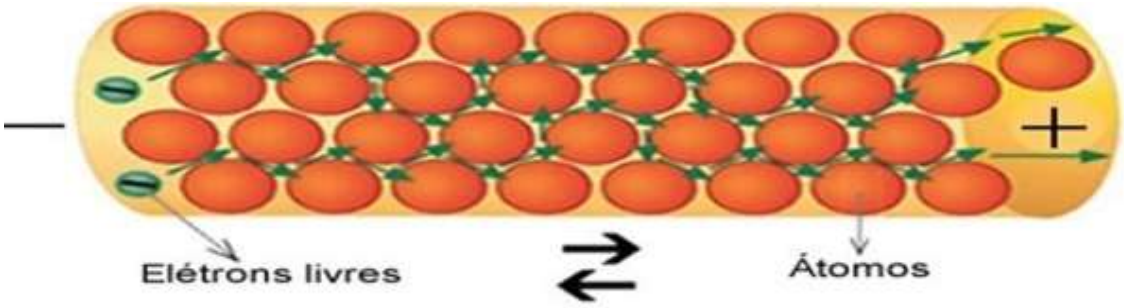
Figura 21 - Redução Hidráulica

Fonte: http://hidraulicapotenza.com.br/produto/predial/buchas-de-reducao

2.8 Resistência Elétrica

Por definição resistência elétrica é a capacidade de um corpo qualquer se opor à passagem de corrente elétrica mesmo quando existe uma diferença de potencial aplicada. Nos condutores o movimento dos elétrons acontece de forma desordenada, o que caracteriza em uma dificuldade de locomoção interna, o que acarreta em colisões com outros elétrons e átomos deste condutor, e quando há essas colisões há também uma dificuldade na passagem dos elétrons, estabelecendo então que a corrente elétrica que ali flui tenha uma resistência, figura 22 a seguir.

Figura 22 - Comportamento dos eletrons no condutor metálico



Fonte: http://biocasantanabloguefq.blogspot.com.br/2014_05_01_archive.html

Em condutores metálicos como o alumínio, cobre e outros, há alguns fatores que determinam o valor de sua resistência, como:

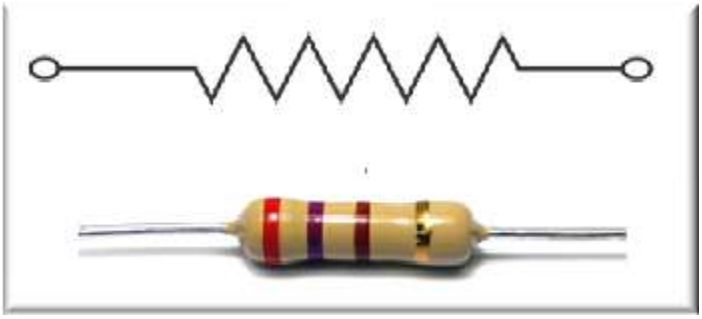
Área do condutor: Quanto maior, menor é a resistência e quanto menor a seção maior é a resistência.

Comprimento: Quanto maior o comprimento do condutor maior será a resistência, pois os elétrons terão um longo caminho a ser percorrido.

Tipo de material: Quanto mais elétrons livres estiverem neste material condutor menor será a resistência elétrica e maior será a passagem de corrente elétrica.

Nos circuitos elétricos e eletrônicos geralmente necessita inserir componentes para promover uma queda de tensão e consequentemente limitação de corrente. Para isso utiliza-se resistências elétricas comumente chamados resistores cujo unidade é dada em ohms (**Ω**)conforme pode ser visto o símbolo e um modelo convencional na figura 23 a seguir.

Figura 23 - Resistor



Fonte: Acervo Pessoal

O seu valor nominal é apresentado por faixas coloridas (código de cores), que obedecem ao seguinte critério: partindo da extremidade, as duas primeiras cores formam um número com dois algarismos; a terceira cor corresponde ao expoente da potência de 10 que multiplica o número inicial; a quarta cor corresponde à tolerância que mostra, percentualmente, a faixa de valores em que pode variar a resistência do resistor de acordo com a tabela 1 a seguir.

Tabela 1 – Código de Resistore

Côr	1ª e 2ª Faixa	3ª Faixa	4ª Faixa	
	1º e 2º Número directo	Factor multiplicador	Tolerância	%
Prata		0.01		+/- 10
Ouro		0.1		+/- 5
Preto	0	x 1	Sem cor	+/- 20
Castanho	1	x 10	Prateado	+/- 1
Vermelho	2	x 100	Dourado	+/- 2
Laranja	3	x 1,000		+/- 3
Amarelo	4	x 10,000		+/- 4
Verde	5	x 100,000		
Azul	6	x 1,000,000		
Violeta	7			
Cinzento	8	x 0.1		
Branco	9	x 0.01		

Dessa forma pode-se calcular facilmente o valor de um resistor.

Tomando-se o resistor da figura 25 a seguir como exemplo pode-se verificar de acordo com a tabela 1 o valor das respectivas.

Figura 24 - Resistor de 47000 Ohms

Fonte: Arquivo Pessoal

As duas primeiras cores: Amarelo (4) e Violeta (7) formam o número 47. A terceira cor, laranja (3), corresponde ao expoente da potência de dez: 10^3; a quarta cor, prata (10%), indica a tolerância. Assim, a resistência elétrica é:

AMARELO	VIOLETA	LARANJA	DOURADO
4	7	3	+/- 10%

Uma vez que foi apresentado todos os modelos torna-se claro suas associações e intuitivas as relações de proporção de acordo com a primeira e segunda lei de Ohm.

A **Primeira Lei de Ohm** postula que um condutor ôhmico (resistência constante), mantido à temperatura constante, a intensidade (i) de corrente elétrica será proporcional à diferença de potencial (ddp) aplicada entre suas extremidades, sua resistência elétrica é **constante**. É representada pela seguinte fórmula:

Equação 1 – Primeira Lei de Ohm

$$R = \frac{U}{I}$$

donde:

R: resistência, medida em Ohm (Ω)
U: diferença de potencial elétrico (ddp), medido em Volts (V)
I: intensidade da corrente elétrica, medida em Ampére (A).

A **Segunda Lei de Ohm** estabelece que a resistência elétrica de um material é diretamente proporcional ao seu comprimento, inversamente proporcional à sua área de secção transversal e depende do material do qual é constituído, sendo representada pela seguinte fórmula:

Equação 2 – Segunda Lei de Ohm

$$R = \frac{\rho . L}{A}$$

donde:

R: resistência (Ω)
ρ: resistividade do condutor (depende do material e de sua temperatura, medida em Ω.m)

L: comprimento (m)
A: área de secção transversal (mm^2)

2.9 Usando modelo de acionamento para produzir trabalho

Uma das máquinas hidráulicas mais simples inventadas pelo homem, o monjolo (figura 2-25) chegou ao Brasil com os portugueses durante o período colonial. E, por muito tempo, tornou-se ferramenta indispensável na lida, pois dispensava o uso de mão-de-obra escrava, que antes socava e moía os grãos em pilões.

Figura 25 - Monjolo

Fonte: http://revistagloborural.globo.com/GloboRural/0,6993,EEC1689831-4528,00.html

A força da queda d'água o impulsiona como se fosse uma gangorra. De um lado, uma concha recebe a água até se encher totalmente. Isso faz com que a outra parte do monjolo, onde há uma estaca, se levante. Ao esvaziar a cuba, o movimento se inverte. E nesse sobe-e-desce, o grão vai sendo socado e moído dentro de um pilão. Obviamente que a tarefa é mais demorada, se comparada ao uso de equipamentos elétricos.

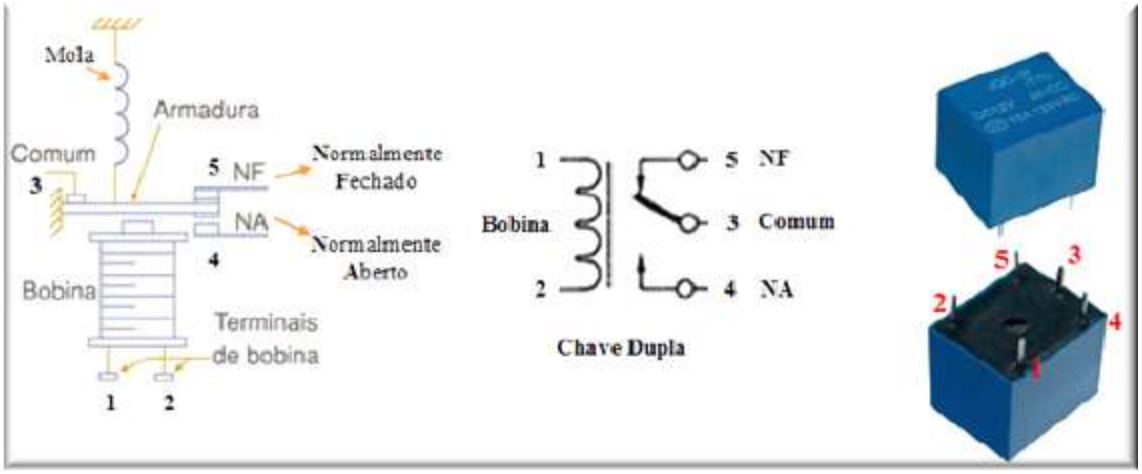
O funcionamento deste equipamento mostra claramente a utilização de um tipo de acionamento (hidráulico) produzindo um trabalho (mecânico).

2.10 Usando um relé para acionamento de carga

Da mesma forma que foi mostrada o monjolo e seu modelo de acionamento separado do trabalho, na eletrônica também são usados componentes com essa função onde o

acionamento é isolado da carga. A esse componente dá-se o nome de relé e pode ser visto na figura 26 a seguir

Figura 26 - Relé



Fonte: http://eletronicaemcasa.blogspot.com.br/2013/12/como-acionar-um-rele-com-arduino-ou-pic.html

A **bobina** é o principal componente do relé. É em torno dela que é gerado um campo eletromagnético quando o relé é energizado. Este campo eletromagnético gera uma força capaz de movimentar um conjunto mecânico (**armadura fixa**) com **contatos móveis** alterando assim seu estado de normalmente aberto para fechado ou de normalmente fechado para aberto de acordo com o tipo de relé, por exemplo.

A ideia de se utilizar um rele é que se pode acionar uma carga com grande potencia a partir de um comando de pequena potencia. Sua aplicação é baseada no acionamento por corrente contínua (5V, 12V ou 24V) para atuar em cargas de corrente contínua que exijam grande fluxo de corrente bem como em corrente alternada (110V e 220V).

2.11 Controle de Fluxo Hidráulico através de Registro

Quando se precisa controlar e/ou dosar a quantidade de fluxo hidráulico em uma tubulação utiliza-se um equipamento chamado registro. Existem registros de vários modelos e para várias aplicações. Um modelo convencional chamado registro de gaveta pode ser visto na figura 27 a seguir.

Figura 27 - Registro de Gaveta

Fonte: http://www.leroymerlin.com.br/registros-de-gaveta

Basicamente o registro irá funcionar de 3 formas:

- **Totalmente Fechado** – neste caso nenhum fluxo de água passará, ele estará impedindo o fluxo completo;
- **Totalmente Aberto** – neste caso todo o fluxo de água passará, ele permitirá o fluxo completo;
- **Ajustável Manualmente** – neste caso o operador dosa a intensidade do fluxo de água que passará pela tubulação através da abertura gradativa até atingir a quantidade desejada.

O sistema de vedação e funcionamento pode ser visto na figura 28 a seguir.

Figura 28 - Registro de gaveta - Constituição interna

Fonte: http://www.leroymerlin.com.br/registros-de-gaveta

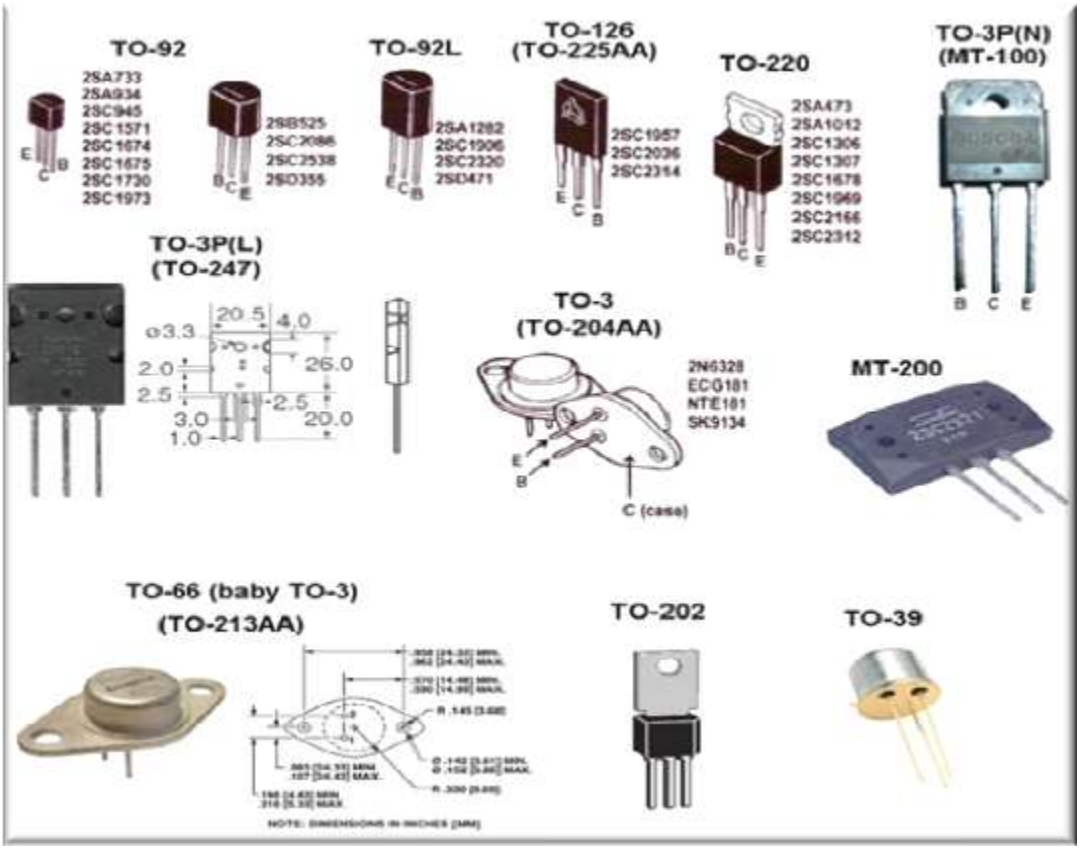
Como pode ser visto, conforme se gira o registro a passagem de água é obstruído ou liberado aos poucos.

2.12 Controle de Fluxo de Elétrons através de Transistor

Nos circuitos eletrônicos geralmente há a necessidade de cortar, liberar ou até mesmo dosar a quantidade do fluxo de elétrons em um determinado ramo do circuito. Para isso emprega-se o componente chamado Transistor.

O transistor é um componente eletrônico que começou a popularizar-se na década de 1950. São utilizados principalmente como amplificadores e interruptores de sinais elétricos, podendo ter variadas funções. Alguns modelos podem ser vistos na figura 29 a seguir.

Figura 29 - Modelos de transistor



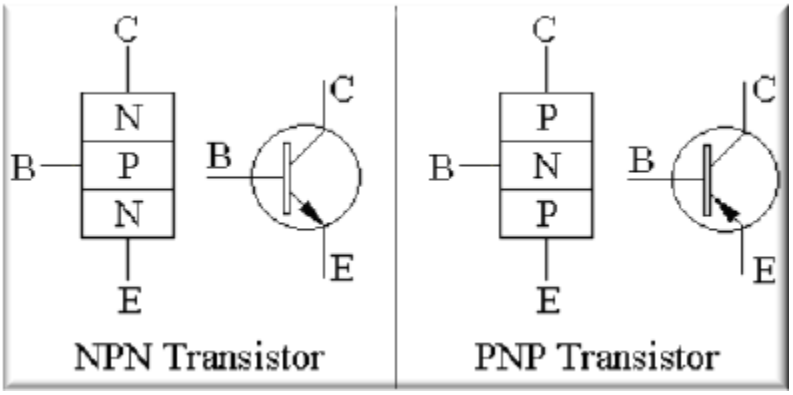
Fonte: http://www.tmgeletronica.com.br/Produto-TRANSISTORESTRANSISTORS-Transistores-2N3439-Trans-GP-BJT-NPN-350V-1A-3-Pin-TO-39-versao-3321-3321.aspx

O recurso consiste em uma pequena partícula de material semicondutor como o silício ou germânio, cuja capacidade de conduzir corrente elétrica se situa entre as dos

materiais condutores e isolantes. Cada transistor tem três terminais, como se fossem canais, que recebem o nome de “emissor”, “coletor” e “base” podendo se apresentar de dois modos distintos baseados na ordem em que os materiais semicondutores se organizam no componente (PNP ou NPN) os quais podem ser vistos na figura 2-30 a seguir.

A classificação NPN e PNP é baseada no tipo do material semicondutor empregado. Observando a letra do meio pode-se dizer qual o tipo do material da base, a qual é o registro do transistor, sendo assim, se for tipo N significa que é controlado por carga negativa, ser for tipo P significa que é controlado por carga positiva (Figura 30).

Figura 30 - Simbologia do transistor



Fonte: Acervo Pessoal

Um dos terminais (coletor) recebe a tensão elétrica, e o outro envia o sinal amplificado (emissor). Já o canal do meio (base) – também conhecido como terminal de controle – é o responsável por fazer a mediação deste processo, uma vez que a corrente elétrica só entra e sai pelos condutores das pontas somente quando o do meio recebe uma tensão elétrica. Portanto, ele que faz o controle do processo. A quantidade de tensão aplicada ao terminal do meio determinará qual será a intensidade da corrente que sairá pelo terminal de saída.

Em computadores e sistemas microcontrolados o transistor não é empregado como amplificado, ele trabalha como chave, pois controlam a chamada “lógica binária” ou “lógica digital” para executarem suas operações internas.

A lógica digital é um ramo peculiar da lógica matemática que por sua vez é um ramo da lógica formal. A peculiaridade consiste no fato de que suas variáveis podem assumir

apenas dois valores mutuamente exclusivos: verdadeiro ou falso, jamais um valor intermediário.

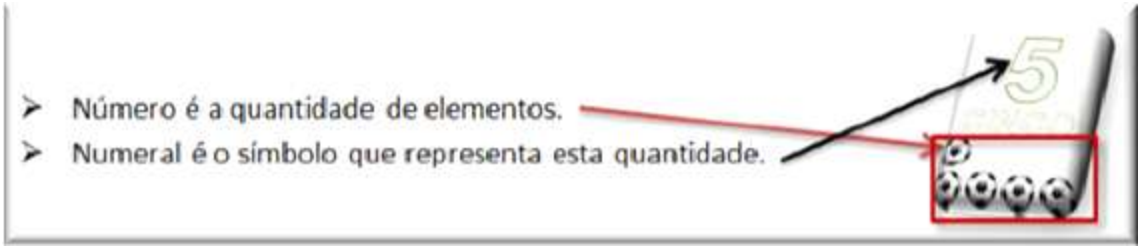
Estes dois valores são usados para representar os algarismos “um” e “zero” empregados nas operações internas dos computadores. Desta forma podemos afirmar que “Verdadeiro = 1” e “Falso = 0” e, apenas com estes dois algarismos, representar todos os números no sistema numérico binário, ou de base 2.

3 Sistema de Numeração

Antes de mais nada, esqueça o que sabe de sistema de numeração, seja decimal ou qualquer outro que conheça, como o romano, egípcio, sumério, etc. Isto é importante para que entenda bem os conceitos de como são formados. Assim, nunca terá problemas ou ter que decorar os mesmos para manipular informações codificadas no sistema binário, hexadecimal e octal utilizados na computação

Primeiro, devemos entender a diferença entre número e numeral

Figura 31 - Diferença entre numero e numeral



Fonte: Acervo Pessoal

Nós estamos acostumados com o sistema decimal, o qual possui, como o próprio nome indica, dez símbolos: os numerais arábicos (indo-arábicos). Inicialmente tinha apenas nove símbolos, já que zero não representa quantidade alguma e um numeral representa graficamente uma quantidade.

Muitos confundem zero com vazio, o que são coisas totalmente diferentes.

- Zero é ausência de alguma coisa que você sabe o que é.
- Vazio é ausência de qualquer coisa.

"A primeira aparição aceita universalmente do numeral 0 ocorreu no ano de 870 dc, em uma inscrição na índia, apesar de já se ter referências na Pérsica e outros documentos indianos desde o século VI" conforme: (Kaplan, 1999) . Assim, o sistema de numeração decimal indu-arábico, da representação gráfica dos números, são:

0 1 2 3 4 5 6 7 8 9

Os quais são derivados e equivalentes à utilizada hoje pelos muçulmanos, conforme a seguir.

٠ ,٩, ٨, ٧, ٦, ٥, ٤, ٣, ٢, ١

Mas nem todo mundo usa o sistema indu-arábico decimal, apesar de ser o mesmo reconhecido mundialmente. Outras dezenas de codificação de numerais podem ser vistas em http://en.wikipedia.org/wiki/Arabic_numerals.

3.1 Sistema de Numeração Decimal – Dez Símbolos

Como são realizadas as representações de quantidade no sistema decimal, quando o valor excede o maior numeral, o 9.

Primeiro, vamos ver como representamos o aumento em uma quantidade (somar uma quantidade a outra quantidade, já representada por outro numeral. Por exemplo:

Se temos uma unidade de alguma coisa e recebemos mais duas unidades, qual é será o numeral que representaria as atuais três unidades?

Um + dois = ?

É simples, observe a seguir.

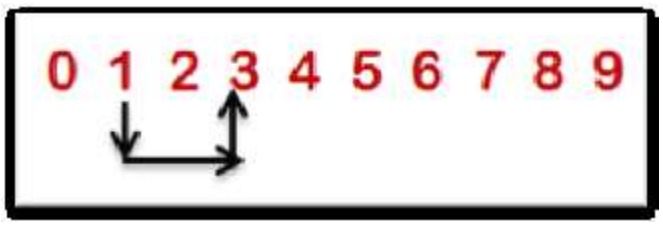
Eis os dez símbolos do sistema decimal por nós utilizados.

0 1 2 3 4 5 6 7 8 9

Temos uma unidade que é representada pelo numeral **1**.

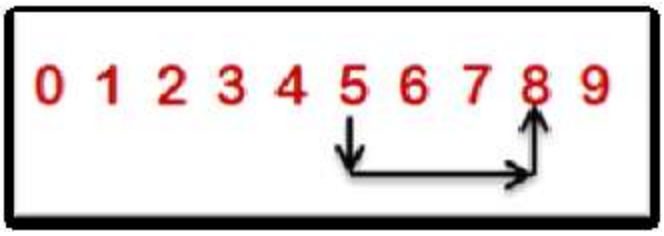
Assim, ao ganharmos mais duas unidades, para saber qual é o símbolo, o numeral que a representa, partindo so símbolo **1**, move-se dois símbolos para frente, encontrando o símbolo **3**.

Desta forma, 1 + dois = 3



E 5 mais três unidades?

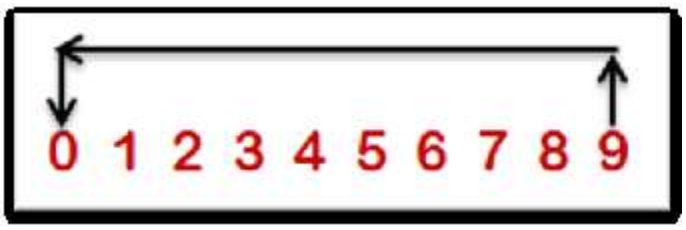
5 + três = 8



E, desta forma, quanto seria 9 + um?

9 + um = ?

Simples, findo os nove símbolos, o próximo é o primeiro da lista. No caso, seria o símbolo **0** (não se assuste, espere um pouco a explicação que segue).



(**9 + um**) significa que percorremos todos os dez símbolos e que paramos no numeral 0. Temos um conjunto de dez símbolos percorridos.

A representação fica, portanto assim:

Temos **1** grupo de dez símbolos percorridos e paramos no símbolo **0**, ou seja:

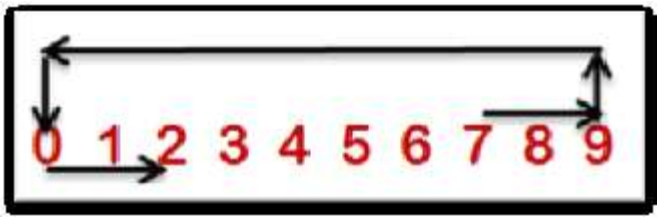
10, onde o primeiro símbolo representa uma dezena de símbolos percorridos e o outro a quantidade que excedeu os dez símbolos, no caso, **0**.

Assim, lemos **10** (um seguido de zero) como representando a quantidade dez. Aprendemos na escola básica, no quadro valor de lugar, que, decompondo este símbolo, temos: uma dezena e zero unidades.

Desta forma, é correto dizer que (**nove (9) + um**) é representado pelos numerais **10 (lê-se: um zero)**, onde nos habituamos e é correto dizer **dez**.

E quanto seria, portanto, sete unidades, representada pelo numeral 7, acrescida de cinco unidades?

7 + cinco = ?



Simples, partindo do símbolo **7**, move-se dois para frente até o símbolo **9**, volta-se ao início e move-se mais três símbolos para frente, parando no símbolo **2**.

Desta forma, temos um grupo completo dez símbolos, parando no símbolo **2**, ou seja: **12**. Assim, **12 (um dois**) representa a quantidade doze, um conjunto de dez unidades, excedendo duas unidades.

Isto pode parecer coisa de criança, mas, com este conceito, e conceito é tudo em ciências, podemos entender todos os demais sistemas de numeração não decimal (sim, existem outros sistemas de numeração que possuem mais e menos que dez símbolos – (https://en.wikipedia.org/wiki/Numeral_system).

Somente baseado neste texto, qual é a quantidade expressa por estes dois símbolos no sistema decimal indu-arábico: **34 ?**

Pelo que aprendemos, o número **3** à direita significa que existem três grupos de dez unidades na contagem, e, portanto, trinta unidades e que ainda sobraram o equivalente

ao numeral **4**, quatro unidades. Portanto **34** indica uma quantidade de **trinta unidades mais quatro unidades**, que totalizam **trinta e quatro unidades**.

Parece que complicou porque você já decorou isto. Mas, se se lembrasse do quanto sofreu para decorar isto, veria que nada foi fácil.

Mas isto simplifica ler a quantidade em qualquer dos um dos sistemas de numeração que veremos e nos interessa, ou seja: o binário, o hexadecimal e octal (menos usado na computação atualmente).

RECORDANDO:

10 representa, portanto a quantidade **dez** no **sistema de numeração decimal**, ou, como comumentemente dizemos, **na base 10**, sendo: $\mathbf{10_{(10)} = dez}$

EXERCITANDO NA BASE 10 (SISTEMA DE NUMERAÇÃO DECIMAL):

- $\mathbf{24_{(10)}}$ = dois grupos de dez unidades mais quatro unidades -> vinte e quatro unidades.
- $\mathbf{50_{(10)}}$ = cinco grupos de dez unidades mais zero unidades -> cincoenta unidades.

3.2 Sistema de Numeração Octal – Oito Símbolos

Utilizando os numerais indu-arábicos, os oito símbolos do sistema octal são:

0 1 2 3 4 5 6 7

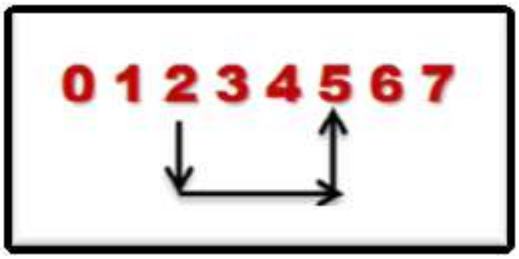
Os primeiros oito símbolos também utilizado no sistema decimal, para simplicidade no processo de assimilá-los.

O conceito de como somar quantidades neste sistema e encontrar o conjunto de símbolos, numerais, que o as representem, é o mesmo visto no sistema decimal. Daí a importância do conceito. Uma vez aprendido, é genérico para aplicações similares.

Assim, na base 8 (sistema de numeração octal), partindo da quantidade representada pelo numeral $\mathbf{2_{(8)}}$, ao acrescentarmos três unidades, qual seria o numeral correspondente à quantidade total que é cinco?

Como visto anteriormente, basta, a partir do numeral $\mathbf{2_{(8)}}$ ver qual é o símbolo, o numeral que está três símbolos após ele, ou seja, o numeral $\mathbf{5_{(8)}}$.

$2_{(8)}$ + três = ?



$2_{(8)}$ + três = $5_{(8)}$

E quanto seria sete mais um, por exempolo estando no numeral **$7_{(8)}$**, ao somarmos uma unidade, qual seria a representação desta quantidade em octal?

$7_{(8)}$ + um = ?

Na base 8, o último símbolo, numeral, é o **7**.

Assim, após ele não tem nenhum outro simbolo. Desta forma, inicia-se novamente a partir do primeiro símbolo, no caso, o numeral **0**.

Com isto, teremos percorrido um grupo completo de oito símbolos e paramos no símbolo(numeral) **0**. Desta forma, sete mais um é igual a **$10_{(8)}$ (lê-se um zero na base 8)**.

$7_{(8)}$ + um = $10_{(8)}$, um conjunto completo de oito símbolos e mais zero elementos.

$10_{(8)}$ = oito

E

$5_{(8)}$ + cinco = ?

Sabemos que cinco mais cinco é dez. Mas como dez é representado na base 8?

Simples, como já explicado. Partindo do símbolo, numeral, 5, avança-se 5 símbolos para a frente. Dois símbolos até chegar no último símbolo, o 7, e, depois, mais três símbolos a partir do símbolo inicial 0, chegando no numeral 2.

Assim, temos:

$5_{(8)}$ + cinco = $12_{(8)}$ (lê-se um dois na base oito)

$12_{(8)}$ = dez unidades.

Um grupo completo de oito elementos e mais duas unidades. O que totaliza a quantidade dez.

IMPORTANTE: É difícil quebrar um paradigma, se adaptar a novos paradignas nunca antes pensados. Se você está se sentindo desconfortável, isto é normal, não se preocupe, você logo se acostuma.

RECORDANDO:

10 representa, portanto a quantidade **oito** no **sistema de numeração octal**, ou, como comumentemente dizemos, **na base 8**: **$10_{(8)}$ = oito.**

EXERCITANDO NA BASE 8 (SISTEMA DE NUMERAÇÃO OCTAL):

- **$24_{(8)}$** = dois grupos de oito unidades mais quatro unidades -> oito + oito + quatro = vinte.
  $24_{(8)}$ = vinte.
- **$50_{(8)}$** = cinco grupos de oito unidades mais zero unidades -> cinco vezes 8 + zero = quarenta.
  $50_{(8)}$ = quarenta.

3.3 Sistema de Numeração Binário – Dois Símbolos

Utilizando os numerais indu-arábicos, os dois símbolos do sistema binário, da base 2, são:

0 1

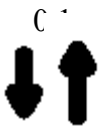
Os dois primeiros dois símbolos também utilizado no sistema decimal, para simplicidade no processo de assimilá-los.

O conceito de como somar quantidades neste sistema e encontrar o conjunto de símbolos, numerais, que o as representem, é o mesmo visto no sistema decimal e octal. Mais uma vez, frizando, daí a importância do conceito. Uma vez aprendido, é genérico para aplicações similares.

Assim, na base 2 (sistema de numeração binário), partindo da quantidade representada pelo numeral $0_{(2)}$, ao acrescentarmos uma unidade, qual seria o numeral correspondente à quantidade total, cuja quantidade é um?

Como visto anteriormente, basta, a partir do numeral **$0_{(2)}$** ver qual é o símbolo, o numeral que está um símbolo após ele, o numeral $1_{(2)}$.

$0_{(2)}$ + um = ?



$0_{(2)}$ + um = $1_{(2)}$

E quanto seria um mais um, por exemplo, estando no numeral $\mathbf{1_{(2)}}$, ao somarmos uma unidade, qual seria a representação desta quantidade em binário?

$1_{(2)}$ + um = ?

0 

Na base 2, o último símbolo, numeral, é o 1.

Assim, após ele não tem nenhum outro simbolo. Desta forma, inicia-se novamente a partir do primeiro símbolo, no caso, o numeral **0**.

Com isto, teremos percorrido um grupo completo de dois símbolos e paramos no símbolo(numeral) **0**. Desta forma, um mais um é igual a **$10_{(2)}$ (lê-se um zero na base 2)**.

$1_{(2)}$ + um = $10_{(2)}$, um conjunto completo de dois símbolos e mais zero elementos.

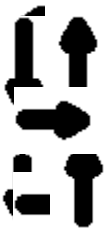
$10_{(2)}$ = dois

E quanto seria

$0_{(2)}$ + 3 = ?

Sabemos que zero mais três são três. Mas como três é representado na base 2?

Simples, como já explicado. Partindo do símbolo, numeral, 0, avança-se três símbolo para a frente, um símbolo até chegar no último símbolo, o 1, e, depois, mais dois símbolos a partir do símbolo inicial 0, chegando no numeral 1.

Assim, temos: 

$0_{(2)}$ + três = $11_{(2)}$ (lê-se um um na base 2)

$0_{(2)}$ + três = três unidades.

Um grupo completo de dois elementos e mais uma unidades. O que totaliza a quantidade três.

RECORDANDO:

10 representa, portanto a quantidade **dois** no **sistema de numeração binário**, ou, como comumentemente dizemos, **na base 2**: $\mathbf{10_{(2)}}$ **= dois.**

FIXANDO O QUE FOI VISTO ATÉ AQUI:

- $\mathbf{10_{(10)}}$ **= dez**
- $\mathbf{10_{(8)}}$ **= oito**
- $\mathbf{10_{(2)}}$ **= dois**

3.4 Sistema Binário e Hardware

Vocês verão em Eletrônica digital, com detalhes, como trabalhar com o sistema binário e a álgebra de Boole (http://en.wikipedia.org/wiki/Boolean_algebra).

O que nos interessa, no momento, para trabalhar com o hardware no Arduíno, é que os computadores, microcontroladores(como o Arduíno), trabalham com a álgebra de Boole e com os símbolos 1 e 0 do sistema binário.

Com os mesmos, podemos criar toda a lógica e aritmética de um computador, e assim foi feito.

IMPORTANTE: O computador internamente só trabalha com o sistema binário, ele não trabalha com o sistema decimal, e, se você não souber disto, ter consciência disto, muita coisa pode dar errada, como veremos posteriormente no treinamento, nas reuniões de grupo dos trainees, ou em sala de aula com os demais alunos.

Assim, os circuitos eletrônicos utilizados nos computadores e nos microcontroladores trabalham com dois níveis de tensão (voltagem) para representar alguma coisa no mesmo.

- O símbolo 1 representa a tensão de alimentação, a tensão de trabalho máxima do componente eletrônico digital. Assim, á tensão máxima de trabalho damos o nome de nível 1 ou HIGH.
- O símbolo 0 representa a tensão de alimentação de referência (0 volts). Damos a este símbolo o nome de nível 0 ou LOW.

No caso do Arduíno:

- o nível lógico 1 = 5Volts (5V)
- o nível lógico 0 = 0 Volts (oV), Grownd ou GND (terra).

4 Escolhendo uma plataforma microcontrolada – Arduino

4.1 O que é Arduino?

Arduino é uma plataforma de prototipagem eletrônica de hardware livre e de placa única, projetada com um microcontrolador Atmel AVR com suporte de entrada/saída embutido, uma linguagem de programação padrão baseada em C/C++. O objetivo do projeto é criar ferramentas que são acessíveis, com baixo custo, flexíveis e fáceis de usar por profissionais e amadores. Principalmente para aqueles que não teriam alcance aos controladores mais sofisticados e de ferramentas mais complexas. A figura 4-1 a seguir mostra o modelo Arduino Uno.

Para mais detalhes sobre modelos de Arduinos e Shields consulte o Anexo I

Figura 32 - Arduino Uno

Fonte: https://pt.wikipedia.org/wiki/Arduino

4.2 Por que utilizar o arduino?

Os argumentos favoráveis são muitos e são citados a seguir:

- **Open Source ou Código Aberto;**

  - A tradução de open source seria Fonte Aberta ou Código Fonte Aberto o que na verdade vai mais além do que isso. Resumidamente podemos citar os dez quesitos que o fazem ser Open Source:

- 01. Distribuição livre;
- 02. Acesso ao código-fonte;
- 03. Permissão para criação de trabalhos derivados;
- 04. Integridade do autor do código-fonte;
- 05. Não discriminação contra pessoas ou grupos;
- 06. Não discriminação contra áreas de atuação;
- 07. Distribuição da licença;
- 08. Licença não específica a um produto;
- 09. Licença não restritiva a outros programas;
- 10. Licença neutra em relação à tecnologia.

Para mais detalhes sobre Open Source consulte o Anexo II:

- **Praticidade;**

  - Por ser uma plataforma de desenvolvimento completa com os circuitos básicos necessários (controladores de tensão, comunicação serial, clock, ...) torna-se prático o desenvolvimento e teste de projetos.

- **Escalabilidade;**

  - Por ser de fácil integração com placas de expansão é possível o empilhamentos de placas e recursos, tornando o projeto adaptável a mudanças e melhorias

- **Diversidade de modelos;**

  - Com um número superior a 20 modelos existentes no mercado, as opções de escolha para um determinado projeto ficam evidentes.

- **Diversidade de Shields;**

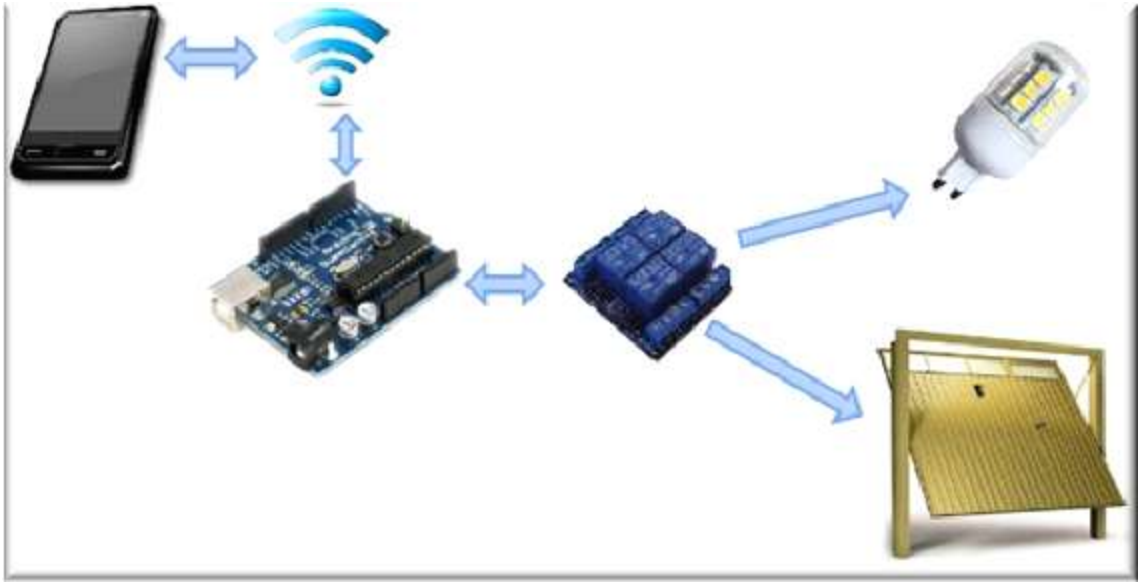
  - Uma grande variedade de placas de expansão estão disponíveis no mercado tais como placa de rede (ethernet shield), placa de bluetooth (bluetooth shield) efim. Isso

facilita na construção de um projeto evitando que o desenvolvedor tenha que construir todo o hardware.

4.3 Controlando Dispositivos Remotamente?

Ha alguns anos atrás a utilização do controle remoto convencional ganhou adeptos pelo mundo inteiro. Sem dúvida é um meio de acionamento muito prático, mas não completamente satisfatório. A maior desvantagem fica por conta da limitação da distancia. Outro fator negativo que também pode ser citado é que o usuário não carrega o controle remoto consigo o tempo todo. Atualmente a utilização de dispositivos móveis vem sendo utilizados em larga escala para esse fim. Os meios de comunicação de dados e a Internet cada vez mais difundida possibilitam a integração de vários dispositivos tais como smartphones, tablets, smartvs, computadores de bordo entre outros. Com isso ha uma facilidade em se efetuar comando a distancia praticamente de qualquer lugar bastando para isso ter acesso a algum tipo de rede ou até mesmo a Internet.

Figura 33 - Controle a distância



Fonte: Acervo Pessoal

4.4 Conhecendo as características principais do Arduino

4.4.1 O Hardware

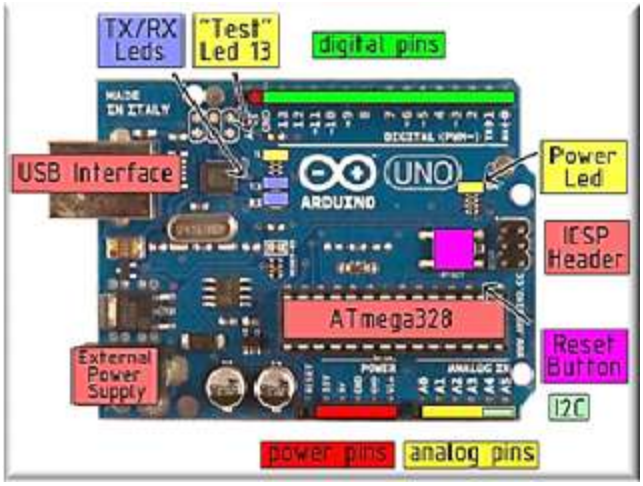
A figura a seguir é do modelo Arduino Uno. Este modelo é o mais comum da linha e oferece os recursos principais para a maioria das montagens e foi utilizado nos experimentos explicados neste material. Por isso será demonstrado através dele as suas características

Figura 34 - Arduino Uno

Fonte: http://www.eletronicacentralpe.com.br/arduino.html

Suas principais características podem ser vistas na figura 4-4 a seguir

Figura 35 - Arduino Uno - Legenda



Fonte: http://digital.csic.es/bitstream/10261/127788/7/D-c-%20Arduino%20uno.pdf

No original italiano o reset muda de lugar, mas o resto praticamente não

O Arduino é um computador como qualquer outro, possuindo:

• Microprocessador (responsável pelos cálculos e tomada de decisão)

• Memória ram (utilizada para guardar dados e instruções, volátil) • Memória flash (utilizada para guardar o sotware, não volátil)

- Temporizadores (timers)
- Contadores
- Clock, e etc.

Pode ser comparada a um computador, porém em menor escala. Possui inclusive menos memória e menor poder de processamento.

O Arduino Uno, por exemplo, possui as seguintes especificações:

- Microcontrolador: ATmega328
- Portas Digitais: 14
- Portas Analógicas: 6
- Memória Flash: 32KB (0,5KB usado no bootloader[2])
- SRAM: 2KB
- EEPROM: 1KB
- Velocidade do Clock: 16MHz

Importante: Bootloader: Para dispensar o uso de um gravador externo, a gravação da Flash é feita por um software pré-gravado, o Bootloader. O Bootloader é o primeiro software executado pelo microcontrolador após um Reset (Boot) e carrega na Flash um software que recebe pela serial (loader).

4.4.2 O Software

O Software é utilizado basicamente para escrever o código do programa, salvá-lo, compilá-lo, e realizar a gravação do código compilado no Arduino (memória flash) através da porta Usb do computador. A IDE do Arduino será utilizada para realizar estes

passos. Este ambiente de desenvolvimento é baseado no Framework Wiring e na linguagem de programação C/C++. Uma vez gravado o programa no Arduino, o computador não é mais necessário. A partir do momento em que se utiliza uma fonte de alimentação externa, o Arduino se torna uma placa totalmente independente. Mas antes de tudo é preciso obter os arquivos de instalação e drivers, que vêm juntos no mesmo pacote, que pode ser obtido no site oficial do Arduino[3]. O download deve ser selecionado de acordo com o sistema operacional utilizado, sendo ele compatível com Windows, Linux e Mac OS no seguinte endereço: http://arduino.cc/en/Main/Software

Após baixar e extrair os arquivos no local desejado execute o programa Arduino.exe, localizado na raiz da pasta principal. Em seguida conecte o seu Arduino ao computador através de uma porta Usb. Ao conectá-lo, um Led de power (pwr) acenderá, isto significa que a placa está energizada. Agora já é possível instalar os Drivers, para isso, será necessário seguir os seguintes passos de acordo com o Sistema Operacional utilizado.

4.4.2.1 Versões para Windows

Será solicitado que um novo driver seja instalado, então deverá selecionar a "escolha manual de drivers", então localize a pasta Drivers dentro do pacote extraído anteriormente.

Obs.: Caso não ocorra a detecção automática de Drivers, será necessário abrir:

- Painel de Controle > Gerenciador de Dispositivos Em seguida selecione os Drivers que estão desatualizados (com uma exclamação) e selecione a opção "Atualizar Driver", logo após selecione a pasta Drivers extraída junto com o pacote anteriormente.

O Windows 8 e 8.1 possui uma particularidade com relação ao Windows 7. Por padrão a instalação de Drivers não assinados é bloqueada no Windows 8 e 8.1, caso não tenha êxito seguindo os passos acima, será necessário desbloquear esta opção.

1. Pressionar a tecla 'windows' + 'R'

2. Digite shutdown.exe /r /o /f /t 00

3. Clique em 'OK'

4. O sistema irá reiniciar, e abrir uma tela azul.

5. Selecione "Solução de Problemas"

6. Selecione "Opções Avançadas"

7. Selecione "Configurações de Inicialização"

8. Clique em "Reiniciar"

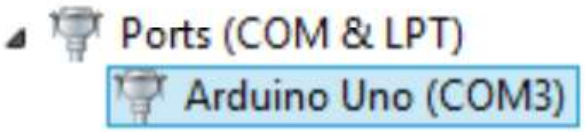
9. O sistema irá reiniciar, então selecione “Desabilitar Imposição de Assinatura de Driver”

10. Pronto! Agora é só seguir o procedimento de instalação novamente

Obs.: 1 - Se você possui um computador que veio com o Windows 8/8.1 e/ou posterior pré-instalado, esta tela opções avançadas (modo de segurança) provavelmente será habilitada na BIOS

2 - Os passos acima provavelmente não funcionarão se o Modo de Segurança estiver desabilitado na BIOS

Figura 36 - Driver do Arduino instalado corretamente



Fonte: https://pt.scribd.com/doc/308848011/Apostila-Arduino-Basico-V1-0-Eletrogate

Obs.: O nº da porta COM pode variar

4.4.2.2 No Linux

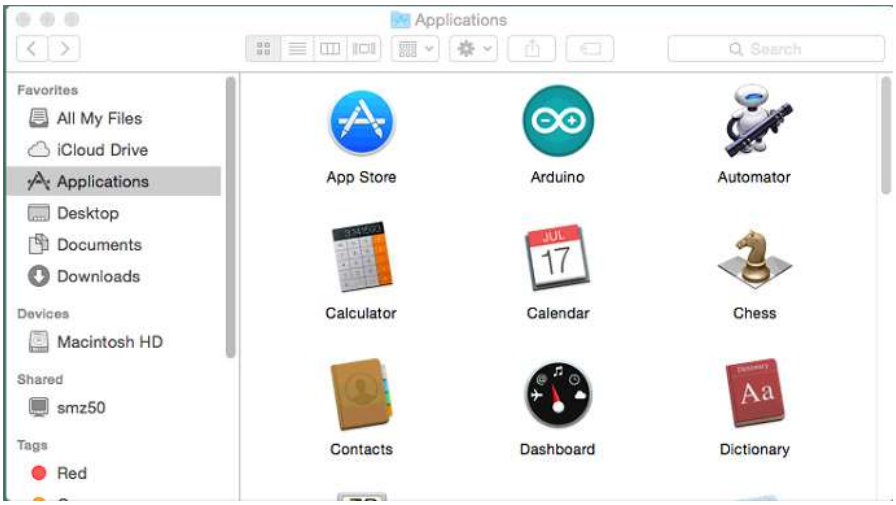
No Linux, abra o terminal e execute o seguinte comando: • sudo aptitude install arduino Ou procure pelo pacote "arduino" no Synaptic: • menu Sistema > Administração > Gerenciador de pacotes Synaptic).

4.4.2.3 No Mac OS

Obter a versão mais recente na página de download. O arquivo está no formato Zip; se você usar o navegador Safari será expandido automaticamente. Se você usar um navegador diferente pode ser necessário extraí-lo manualmente.

Copie o aplicativo Arduino para a pasta Applications (ou em outro lugar no seu computador). Conforme figura 4-6 a seguir

Figura 37 - Arduino no Mac



Fonte: https://pt.scribd.com/doc/308848011/Apostila-Arduino-Basico-V1-0-Eletrogate

4.4.3 – Apresentação do IDE Arduino

Após a instalação será possível abrir a IDE do Arduino, que tem a seguinte aparência:

Figura 38 - IDE Arduino

Fonte: Acervo Pessoal

A IDE do Arduino é muito simples e objetiva, tornando todo o processo de desenvolvimento e gravação bastante intuitivo. Além do espaço em branco destinado ao desenvolvimento do programa, existem 6 botões na parte superior que são respectivamente: Verify, Upload, New, Open, Save e Serial Monitor. Eles são utilizados, respectivamente para Verificar se existem erros no código, enviar (gravar) o programa no Arduino, criar um novo código, abrir um código existente e um monitor de dados da porta serial. Um código desenvolvido para Arduino é chamado de Sketch, traduzindo do inglês ao pé da letra seria algo como “esboço” ou “rascunho”. Isso nos dá uma ideia de que nunca terminamos um código, sempre haverá melhorias e novas funcionalidades. O Sketch possui a extensão ‘.pde’. A partir de agora sempre que for necessário gravar um novo programa no Arduino, basta conectá-lo na porta Usb, Selecionar a placa utilizada em Tools > Board e em seguida selecionar a Porta Serial (COM) associada a ele, neste caso Tools > Serial Port > COM3. Em seguida, após abrir o programa desejado, basta clicar em “Upload”. Após executar estes passos, a IDE deverá exibir uma mensagem no final “Done Uploading”.

4.4.3 Programação:

Obs.: Um programa em Arduino é chamado de sketch

4.4.3.1 Primeiro exemplo:

Programa que faz acender ou apagar um led. Apenas isto.

- Acender um led é colocar a porta em nível alto (HIGH ou 1), onde o mesmo está conectado.
- Apagar um led é colocar a porta em nível baixo (LOW ou 0), onde o mesmo está conectado.

Um Programa em Arduino possui uma estrutura composta por três partes:

1. Declaração de constrantes de variáveis
2. Setup: configuração das portas e outras estruturas e parâmetros internos
3. Loop – o programa será executado pelo Arduino indefinidamente

No programa deste exemplo foram utilizadas somente duas partes: **setup** e **loop**.

Cada parte contém uma função:

- A função **setup**
- A função **loop**

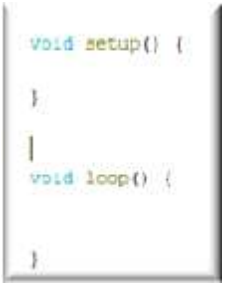
As duas funções possuem argumento vazio, representado por um abrir e fechar de parênteses e são iniciadas pela palavra void.

- **void setup ( )**
- **void loop ( )**

A estrutura fica da seguinte forma:

Um trecho de programa que vai ser executado sempre fica entre colchetes {PROGRAMA}, logo após a chamada das funções.

Figura 39 - Funções do Arduino



Fonte: Acervo Pessoal

Execução do programa pelo Arduino

- O Arduino, quando energizado, primeiro executa a função **setup** e logo após a função **loop**.
- A função **loop**, conforme o próprio nome diz, fica rodando indefinidamente, ou como o próprio nome diz, em loop.

Vamos analisar o programa deste exemplo:

A Função setup.

Obs. O que vem após // é considerado comentário.

Conforme foi dito, a função **setup** é a responsável pela configuração de como o Arduino irá trabalhar, que portas serão saídas, que portas serão entradas, outros que serão explicados momento em que for necessário.

A Função loop.

Obs. O que vem após // é considerado comentário.

O programa entre colchetes {} vai ficar rodando, sendo executado, indefinidamente até que o Arduino seja desligado ou reprogramado.

4.4.3.1.1 O Programa

Acendendo um led na porta 13.

A porta 13 já possui um led instalado na placa, facilitando fazer testes sem ter que adicionar componentes ao arduino.

Figura 40 - Programação

```
1 void setup() {
2   pinMode(13, OUTPUT);//led laranja interno da placa
3 }
4
5 void loop() {
6   digitalWrite(13, HIGH);   // coloca nivel 1 (HIGH) na porta 13
7                             // -> acende o led laranja da placa
8 }
```

Fonte: Arquivo Pessoal

A função **setup.**

Obs. O que vem após // é considerado comentário.

Figura 41 - Trecho da programação - void setup

```
1 void setup() {
2   pinMode(13, OUTPUT);//led laranja interno da placa
3 }
```

Fonte: Acervo Pessoal

Conforme foi dito, a função **setup** é a responsável pela configuração de como o Arduino irá trabalhar.

No caso, temos uma linha de código onde se está configurando a porta 13 do Arduino como porta de saída (OUTPUT): a linha 2.

A função que configura uma porta é a **pinMode**, a qual possui dois argumentos:

1- O número da porta
2- Configuração (**OUTPUT** ou **INPUT**)

Obs. A porta 13 tem uma particularidade: ela já possui conectado à ela, na própria placa do Arduino, um pequeno LED alaranjado, o qual estará sempre mostrando o status desta porta, seja ela configurada como entrada (**INPUT**) ou saída (**OUTPUT**).

Assim, a configuração da porta 13, como saída, linhas 2, utilizando a função **pinMode,** fica assim:

Figura 42 - Trecho da programação - definição do pino

```
2   pinMode(13, OUTPUT);
```

Fonte: Acervo Pessoal

Obs.: Ao final de cada linha de programação, de cada instrução, tem-se que colocar o ponto e vírgula, conforme figura anterior.

A Função **loop.**

Obs. - O que vem após // é considerado comentário.
- O programa entre colchetes {} vai ficar rodando, sendo executado, indefinidamente até que o Arduino seja desligado ou reprogramado.

Figura 43 - Trecho da programação - função void loop

```
5 void loop() {
6   digitalWrite(13, HIGH);   // coloca nível 1 (HIGH) na porta 13
7                             // -> acende o led laranja da placa
8 }
```

Fonte: Acervo Pessoal

Para “escrever” em uma porta, que dizer, colocar um determinado nível lógico na mesma, utiliza-se a função **digitalWrite.**

A mesma possui dois argumentos

1- O número da porta
2- O nível lógico que se deseja colocar na porta que está configurada como saída:
   - O **inteiro 0 ou** a **palavra LOW** para nível baixo.
   - O **inteiro 1 ou** a **palavra HIGH** para nível alto.
   Obs. No caso do exemplo, utilizou-se a palavra HIGH para representar o nível lógico alto, poderia se ter utilizado o inteiro 1.

O programa inicia na linha6 por:

Figura 44 - Trecho da programaçaõ - Pino em nivel alto

```
6   digitalWrite(13, HIGH);
```

Fonte: Acervo Pessoal

Sendo:

- escreve, coloca nível 1 (alto, HIGH) na porta 13, linha 6.

No caso, ao rodar, acenderá o LED que está conectado à porta 13 (o LED interno laranja da placa).

Obs. Para apagar, ou seja, deixar o led apagado, basta na linha 6 trocar a palavra **HIGH** por **LOW** (ou 0)

Figura 45 - Trecho da programação - Pino em nível baixo

```
6   digitalWrite(13,LOW)
```

Fonte: Acervo Pessoal

4.4.3.2 Segundo Exemplo

Programa que faz piscar dois LEDs alternadamente, com tempo de 1 segundo (1000 milissegundos).

4.4.3.2.1 O programa

Figura 46 - Programação Arduino

```
1  void setup() {
2    pinMode(13, OUTPUT); //led laranja
3    pinMode(12, OUTPUT); //led azul
4  }
5
6
7  void loop() {
8    digitalWrite(13, 1);     // coloca nível 1 (HIGH) na porta 13 - acende o led laranja
9    digitalWrite(12,0);      // coloca nível 0 (LOW) na porta 12 - apaga o led azul
10   delay(1000);             // espera 1 segundo (1000 milesegundos)
11   digitalWrite(13, 0);     // muda o nível da porta 13 para 0 (LOW) - apaga o led laranja
12   digitalWrite(12,1);      // muda o nível da porta 12 para 1 (HIGH) - acende o led azul
13   delay(1000);             // espera 1 segundo
14 }
```

Vamos analisar o programa deste exemplo:

A Função setup.

Obs. O que vem após // é considerado comentário.

Figura 47 - Trecho da programação - Função void setup

```
1  void setup() {
2    pinMode(13, OUTPUT); //led laranja
3    pinMode(12, OUTPUT); //led azul
4  }
```

Fonte: Acervo Pessoal

Conforme foi dito, a função **setup** é a responsável pela configuração de como o Arduino irá trabalhar.

No caso, temos duas linhas de código onde se está configurando as portas 12 e 13 do Arduino como porta de saída (OUTPUT): as linhas 2 e 3.

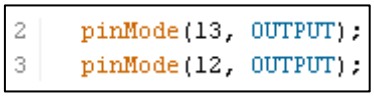
A função que configura uma porta é a **pinMode**, a qual possui dois argumentos:

3- O número da porta
4- Configuração (**OUTPUT** ou **INPUT**)

Obs. A porta 13 tem uma particularidade: ela já possui conectado à ela, na própria placa do Arduino, um pequeno LED alaranjado, o qual estará sempre mostrando o status desta porta, seja ela configurada como entrada (**INPUT**) ou saída (**OUTPUT**).

Assim, a configuração das portas 12 e 13, como saída, linhas 2 e 3, utilizando a função **pinMode,** fica assim:

Figura 48 - Trecho da programação - definição de pinos



```
2    pinMode(13, OUTPUT);
3    pinMode(12, OUTPUT);
```

Fonte: Acervo Pessoal

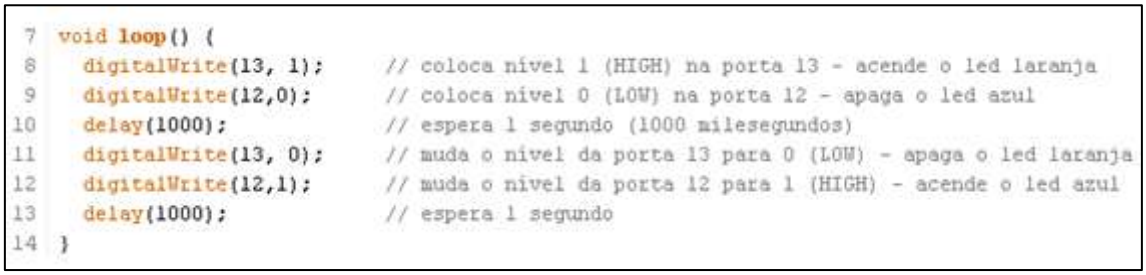
Obs. Ao final de cada linha de programação, de cada instrução, tem-se que colocar o ponto e vírgula, conforme figura anterior.

A Função loop.

Obs. O que vem após // é considerado comentário.

O programa entre colchetes {} vai ficar rodando, sendo executado, indefinidamente até que o Arduino seja desligado ou reprogramado.

Figura 49 - Trecho da programação - função void loop



```
7  void loop() {
8    digitalWrite(13, 1);     // coloca nível 1 (HIGH) na porta 13 - acende o led laranja
9    digitalWrite(12,0);      // coloca nível 0 (LOW) na porta 12 - apaga o led azul
10   delay(1000);             // espera 1 segundo (1000 milesegundos)
11   digitalWrite(13, 0);     // muda o nível da porta 13 para 0 (LOW) - apaga o led laranja
12   digitalWrite(12,1);      // muda o nível da porta 12 para 1 (HIGH) - acende o led azul
13   delay(1000);             // espera 1 segundo
14 }
```

Fonte: Acervo Pessoal

Para “escrever” em uma porta, colocar um determinado nível lógico na mesma, utiliza-se a função **digitalWrite.**

A mesma possui dois argumentos

3- O número da porta
4- O nível lógico que se deseja colocar na porta que está configurada como saída:
   - O **inteiro 0 ou** a **palavra LOW** para nível baixo.
   - O **inteiro 1 ou** a **palavra HIGH** para nível alto.

Obs.: No caso do exemplo, utilizou-se os inteiros 0 e 1 para representar os níveis lógicos baixo e alto.

O programa inicia nas linhas 7 e 8 por:

Figura 50 - Trecho da programação - Configurando niveis dos pinos

```
8    digitalWrite(13, 1);
9    digitalWrite(12,0);
```

Fonte: Acervo Pessoal

Sendo:

- escreve, coloca, nível 1 (alto, HIGH) na porta 13, linha 8.

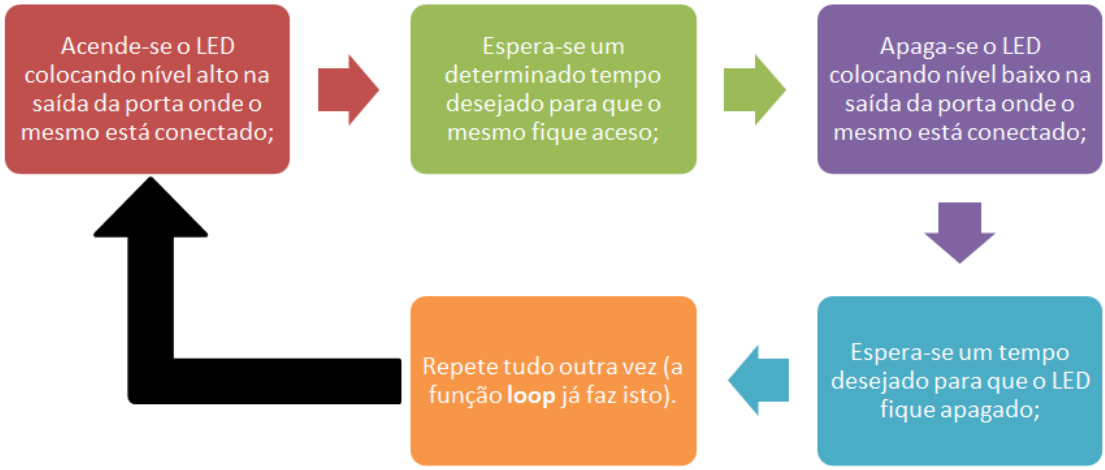
- escreve, coloca, nível 0 (baixo, LOW) na porta 12, linha 9.

No caso, ao rodar, acenderia o LED que está conectado à porta 13 (incluindo o LED interno laranja da placa) e apagaria o LED que estiver conectado à porta 12.

Fazendo o LED piscar

Não existe uma instrução para fazer um LED piscar, você tem que construir um programa que faça isto. No caso, o programa é simples:

Figura 51 - Ciclo da programação para piscar um led



Fonte: Acervo Pessoal

Já sabemos como colocar nível alto e baixo em uma porta, portanto, só falta saber como programar o Arduino para esperar um tempo antes de executar a próxima linha de programa.

A função que faz isto é a função **delay**.

A função **delay** possui apenas um argumento: o tempo de espera em milissegundos (1000 milissegundos = 1 segundo)

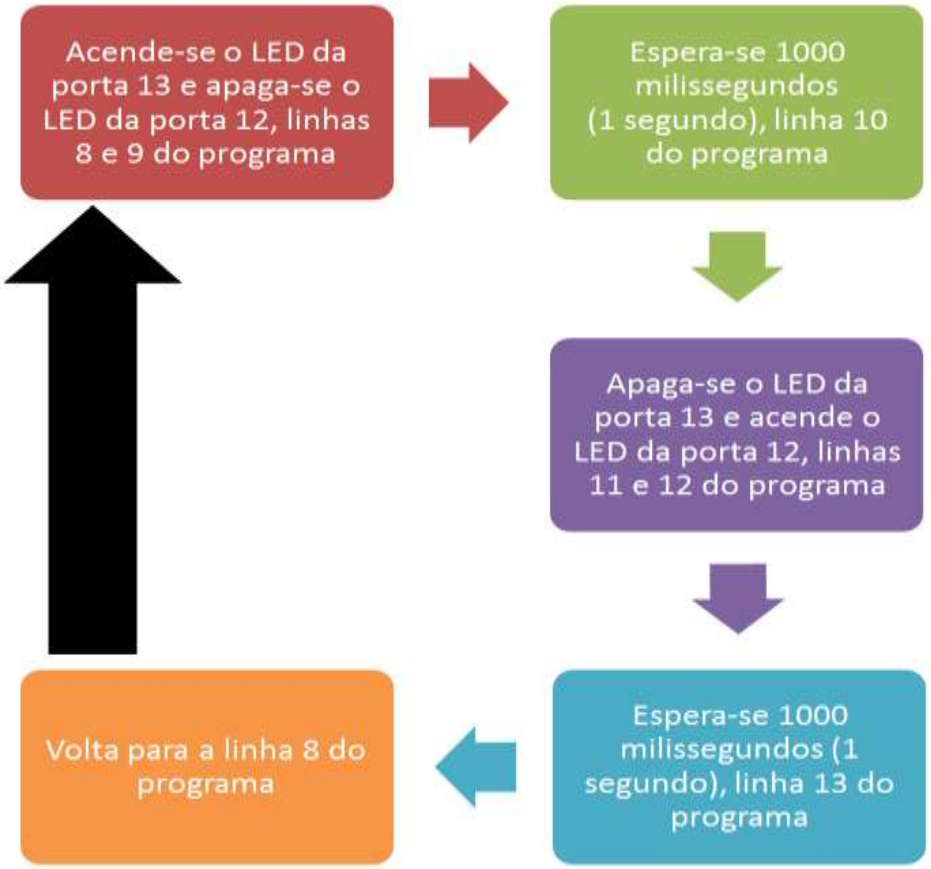
As linhas 10 e a 13 fazem com que o Arduino fique esperando 1000 milissegundos (1 segundo) antes de executar a próxima instrução.

No caso da linha 10, a próxima instrução é a da linha 11.

No caso da linha 13, a próxima instrução a ser executada é a da linha 8, a primeira do **loop**, já que a linha 13 é a última do **loop**.

Assim, o programa é executado da seguinte forma:

Figura 52 - Ciclo da programação Arduino para piscar dois leds



Fonte: Acervo Pessoal

Devido à função **loop**, os LEDs ficarão piscando em intervalos alternados de um segundo cada.

Obs.: Pode-se colocar tempos diferentes para os delas das linhas 10 e 13. Os mesmos não precisam ser iguais.

4.4.3.3 Terceiro exemplo:

O mesmo programa, agora introduzindo o conceito de constante e mostra de que utilizar lables (rótulos, nomes), em vez de números para portas e estados (0,1), facilitam a programação e evita-se confusões e erros.

Figura 53 - Prograamação arduino

```
1  const int ledAzul = 12;    // ledAzul    - constante com valor 12
2  const int ledLaranja = 13; // ledLaranja - constante com valor 13
3
4  void setup() {
5    pinMode(ledLaranja, OUTPUT); //led laranja
6    pinMode(ledAzul   , OUTPUT); //led azul
7  }
8
9
10 void loop() {
11   digitalWrite(ledLaranja, HIGH); // coloca nivel 1 (HIGH) na porta 13 - acende o led laranja
12   digitalWrite(ledAzul,LOW);      // coloca nivel 0 (LOW) na porta 12 - apaga o led azul
13   delay(1000);                    // espera 1 segundo (1000 milesegundos)
14   digitalWrite(ledLaranja, LOW);  // muda o nivel da porta 13 para 0 (LOW) - apaga o led laranja
15   digitalWrite(ledAzul,HIGH);     // muda o nivel da porta 12 para 1 (HIGH) - acende o led azul
16   delay(1000);                    // espera 1 segundo
17 }
```

Fonte: Acervo Pessoal

Fica mais simples programar dando nomes às coisas. Por exemplo, em vez de chamar a porta de **13**, já que a mesma está ligada ao LED laranja, seria mais fácil chamar a porta de **ledLaranja**, por exemplo. Fica mais claro chamar o nível alto lógico de **HIGH** em vez de **1**. O valor **1** para nível poderia ser confundido com a **porta 1**, por exemplo. O mesmo para o nível lógico **0**, que passaremos a utilizar como **LOW**, pelo mesmo motivo.

Para Declarar uma constante, deve-se utilizar a palavra **const**, seguida do tipo de dado (no caso **int** – inteiro) seguido do nome da constante seguido do símbolo = seguido do valor da constante.

Isto pode ser visto nas linhas 1 e 2 do programa (sketch).

Figura 54 - Trecho da programação - Declarando constantes

```
1  const int ledAzul = 12;
2  const int ledLaranja = 13;
```

Fonte: Acervo Pessoal

O restante do programa é mera substituição dos valores numéricos pelas constantes criadas e pela substituição do nível alto 1 por HIGH e o nível lógico baixo 0 por LOW.

Criando constantes para os delays

Vamos criar também constantes para os **delays**.

Vamos chamar de **delay1** e **delay2**.

Vamos colocar valores diferentes para cada um.

- delay1 = 1 segundo
- delay2 = 0,5 segundo

O programa fica assim:

Figura 55 - Programação Arduino

```
const int ledAzul = 12;    // ledAzul    - constante com valor 12
const int ledLaranja = 13; // ledLaranja - constante com valor 13
const int delay1 = 1000;   // delay1 = 1 segundo (1000 milissegundos)
const int delay2 = 500;    // delay2 = 0,5 segundo (500 milissegundos)

void setup() {
  pinMode(ledLaranja, OUTPUT); //led laranja
  pinMode(ledAzul   , OUTPUT); //led azul
}


void loop() {
  digitalWrite(ledLaranja, HIGH); // coloca nível 1 (HIGH) na porta 13 - acende o led laranja
  digitalWrite(ledAzul,LOW);      // coloca nível 0 (LOW) na porta 12 - apaga o led azul
  delay(delay1);                    // espera 1 segundo (1000 milesegundos)
  digitalWrite(ledLaranja, LOW);  // muda o nível da porta 13 para 0 (LOW) - apaga o led laranja
  digitalWrite(ledAzul,HIGH);     // muda o nível da porta 12 para 1 (HIGH) - acende o led azul
  delay(delay2);                    // espera 1 segundo
}
```

Fonte: Acervo Pessoal

4.4.3.4 Quarto exemplo:

Programa que faz piscar dois leds alternadamente, com tempo de 1 segundo (1000 milissegundos) e, conforme o nível lógico (status) de uma chave (HIGH ou LOW) faz um buzzer dar um bip de 50 milissegundos.

Obs. Este programa utiliza as três estruturas do scketch. Também utiliza a função if, que será explicada neste texto.

Figura 56 - Programação Arduino

```
1  const int botao = 10;       // porta 10 = botao
2  const int buz =11;          // porta 11 = buz
3  const int ledAzul = 12;     // porta 12 = LED Azul
4  const int ledLaranja = 13; // porta 13 = LED Laranja
5  const int delay1 = 1000;    // delay1 = 1 segundo
6  const int delay2 = 500;     // delay2 = 0,5 segundo (1/2 segundo)
7  const int delay3 = 100;     // delay3 = 0,1 segundo (1/10 de segundo)
8  int chave = 0;              // variável do tipo int com valor inicial 0 que é
9                              // equivalente a LOW. Pode-se usar LOW que não dá
10                             // erro no programa.
11 void setup() {
12   pinMode(ledLaranja, OUTPUT);
13   pinMode(ledAzul, OUTPUT);
14   pinMode(buz, OUTPUT);
15   pinMode(botao, INPUT);
16 }
17
18 void loop() {
19   chave=digitalRead(botao);        // declara que a variável chave assume o mesmo
20                                    // valor da entrada do botao (10)
21   digitalWrite(ledLaranja, HIGH);
22   digitalWrite(ledAzul,LOW);
23   delay(delay1);
24   digitalWrite(ledLaranja, LOW);
25   digitalWrite(ledAzul,HIGH);
26   delay(delay2);
27   if (chave==HIGH){               // início da estrutura do if
28       digitalWrite(buz,HIGH);
29       delay(delay3);
30       digitalWrite(buz,LOW);
31     }
32     else{
33       digitalWrite(buz,LOW);
34     }                             // fim da estrutura do if
35 }
```

Fonte: Acervo Pessoal

Este programa apresenta 3 novidades em relação ao anterior:

1- Criação de uma variável: chave. No caso, a variável que assumirá o valor lógico da chave, e, como a chave pode mudar constantemente de valor, estaremos armazenando o mesmo em uma variável. Em linguagem procedural, aceita-se que uma variável mude de valor, o que matematicamente não é aceito e nem em linguagens funcionais, como CLEAN, por exemplo.
A variável não inicia com **const**, conforme pode-se ver na linha 8. **const** é para constantes.

Figura 57 - Trecho da programação - criação de variável

```
8  int chave = 0;
```

Fonte: Acervo Pessoal

2- Associação do valor da variável ao nível lógico de uma porta. Isto pode ser visto na linha 19.

Figura 58 - Trecho da programação - Associação ao nivel lógico

```
19    chave=digitalRead(botao);
```

Fonte: Acervo Pessoal

Associou-se a variável **chave** porta **botao** (porta 10), através da leitura da porta. A função que faz a leitura é a **digitalRead**.
Esta função só possui um argumento: a porta, no caso, **botao** – porta 10..

3- A função **if**.

Figura 59 - Trecho da programação - Função if

```
if ( condição ) {                          // inicio da estrutura do if
      ações se a condição for verdadeira
   }
   else  {
      ações se a condição for falsa
   }                                       // fim da estrutura do if
```

Fonte: Acervo Pessoal

Assim, a lógica para disparar o buzzer (sirene) ou não, está descrita nas linhas de número 27 à 34 através da estrutura de um if.

Figura 60 - Trecho da programação - Lógica do buzzer (sirene)

```
27    if (chave==HIGH){                 // início da estrutura do if
28        digitalWrite(buz,HIGH);
29        delay(delay3);
30        digitalWrite(buz,LOW);
31      }
32      else{
33        digitalWrite(buz,LOW);
34      }                                 // fim da estrutura do if
```

Fonte: Acervo Pessoal

O if funciona assim:

1- A linha 27 testa se o valor da variável chave é HIGH (nível alto).

Figura 61 - Trecho da programação - Condição if

```
27    if (chave==HIGH)
```

Fonte: Acervo Pessoal

2- Se (if) for verdade, as instruções entre chaves {} que seguem a condição são executadas, as linhas de número 28 a 30.

Figura 62 - Trecho da programação - Condição satisfeita

```
28        digitalWrite(buz,HIGH);
29        delay(delay3);
30        digitalWrite(buz,LOW);
```

Fonte: Acervo Pessoal

Ativa o buz, porta 10, espera o tempo do delay3 (100 milissegundos) e desativa o buz.
Assim, se o nível da porta da chave for alto, a sirene, buzzer, soará por 100 milissegundos.

3- Caso a condição não for verdadeira, a estrutura do **if** leva o programa à condição else.

Figura 63 - Trecho da prograamação - Condição não satisfeita (else)

```
32      else{
```

Fonte: Acervo Pessoal

Isto faz com que as instruções entre as chaves que a segue sejam executadas. No caso, a linha 33.

Figura 64 - Trecho da programação - Desligando o buzzer (sirene)

```
33        digitalWrite(buz,LOW);
```

Fonte: Acervo Pessoal

4- O programa procede no loop, piscando os LEDS e soando ou não a sirene conforme a leitura da porta 10 (buz) onde uma chave aciona nível alto ou baixo (HIGH ou LOW).

No próximo trata da simulação do Arduino. É uma maneira interessante para quem não dispõe do hardware e mãos.

5 Simulando o arduino

Simulação é a técnica de estudar o comportamento e reações de determinados sistemas através de modelos. Um bom exemplo de simulação é aquele usado na indústria aeronáutica, onde a aerodinâmica dos aviões em projeto é testada em túneis de vento através de pequenas maquetes que apresentam o mesmo formato do avião, ou seja, é o "modelo" do avião real. Esta técnica é aplicada, pois seria completamente inviável construir todo o avião e tentar fazê-lo voar com pilotos de prova. A perda de vidas e investimentos seria enorme e certamente nossos aviões não seriam como hoje os conhecemos se não fosse usada a simulação.

A evolução vertiginosa da informática nos últimos anos tornou o computador um importante aliado da simulação. A simulação por computador é usada nas mais diversas áreas, citando como exemplos as análises de previsão meteorológica, dimensionamento de call centers/contact centers, treinamento de estratégia para militares e pilotagem de veículos ou aviões. Até mesmo o estudo aerodinâmico, antes feito por maquetes, pode ser realizado agora pelo computador.

Isso é possível, pois o computador é alimentado com as propriedades e características de sistemas reais, criando um ambiente "virtual", que é usado para testar as teorias desejadas. O computador efetua os cálculos necessários para a interação do ambiente virtual com o objeto em estudo e apresenta os resultados do experimento no formato desejado pelo analista.

A simulação de processos permite que se faça uma análise de sistemas sem a necessidade de interferir no mesmo. Todas as mudanças e consequências, por mais profundas que sejam, ocorrerão apenas com o modelo computacional e não com o sistema real.

Trata-se de um estudo de baixo custo, visto que todo o trabalho de implementação é testado no computador, permitindo ainda o teste de inúmeros cenários e alternativas de solução para os sistemas em estudo.

Iremos demonstrar a simulação com dois softwares: Virtual Breadboard e o Isis Proteus.

5.1 Virtual Breadboard

Virtual Breadboard é um ambiente virtual para simulação de circuitos eletrônicos que possui um Toolkit para a plataforma Arduino possibilitando a emulação das placas UNO, MEGA e NANO. Com esse Toolkit você poderá simular diversas aplicações, com interação de diversos componentes virtuais e depois enviar para um Arduino real.

É um ambiente inspirado na plataforma Microsoft Visual Studio, que faz o hardware ser orientado a objetos, transformando as tarefas de hardware para simulação em software.

Possui uma grande variedade de componentes e instrumentos virtuais para serem utilizados durante o desenvolvimento que são representados de uma forma realística na interface. Existem vários exemplos disponíveis que auxiliam no estudo e aprendizado da ferramenta.

O Virtual Breadboard pode ser adquirido gratuitamente em
http://www.virtualbreadboard.com/Main.aspx?TAB=Downloads.

O VBB está disponível apenas para Windows, e você deve escolher o link para a plataforma 32 bits ou 64 bits, antes de fazer o download. Também será necessário instalar o Microsoft Visual J#® 2.0 Redistributable Package (disponível no mesmo endereço citado acima), pois a aplicação é baseada neste componente.

O Software VirtualBreadboard é licenciado com módulos que ampliam os recursos do núcleo do solftware.

5.1.1 Como licenciar:

1. Compre uma Licença VBB com PayPal

Figura 65 - Compra do Virtual Breadboard

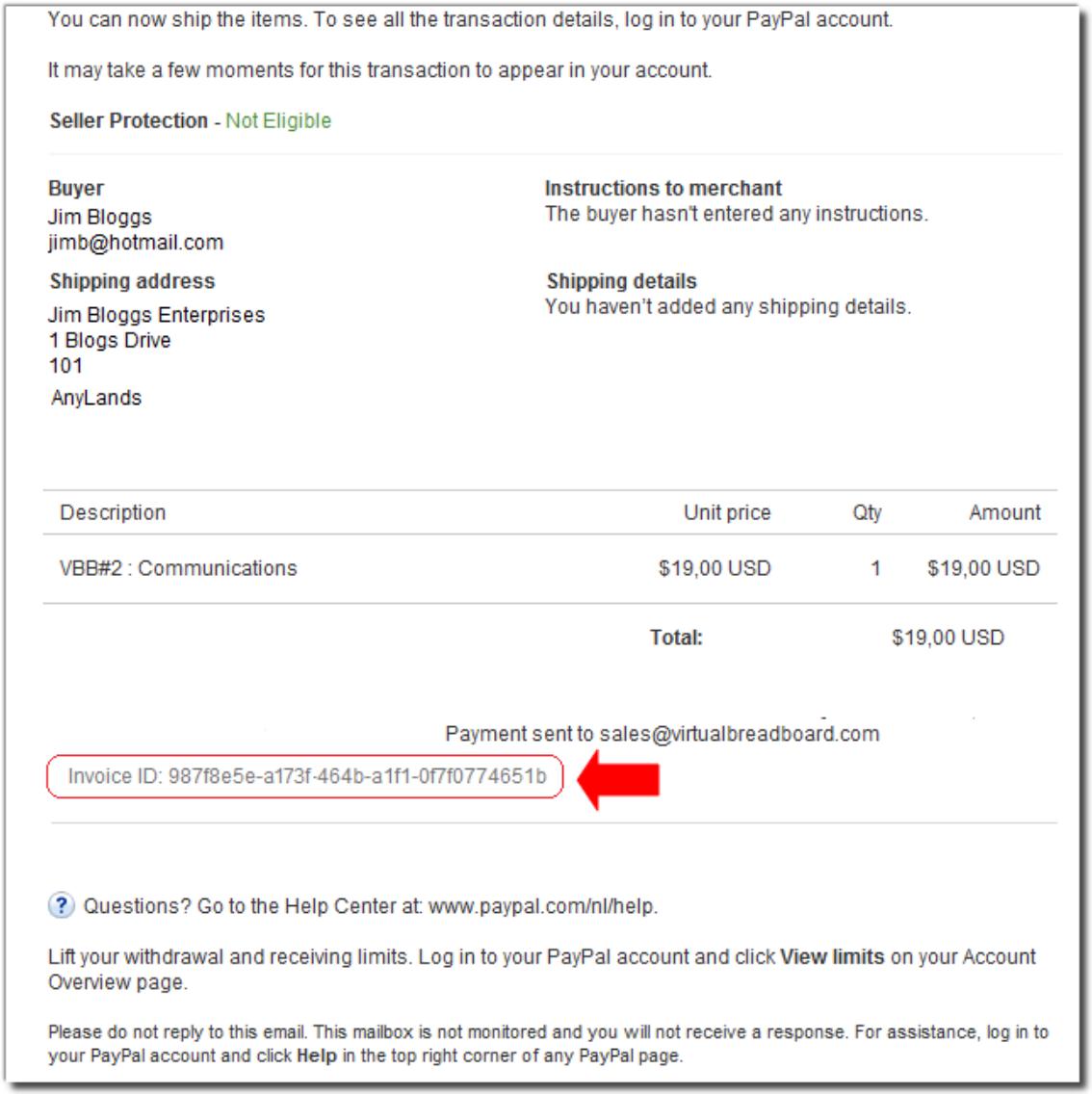
Fonte: http://www.virtualbreadboard.com/

2. Localize a chave de licença

A chave de licença é anexado ao recibo do PayPal com o InvoiceID.

DICA: Você sempre pode consultar a sua chave de licença em sua conta do PayPal

Figura 66 - Adquirindo a licença



Fonte: http://www.virtualbreadboard.com/

3. Instale a Licença

Copie para o VBB-STUDIO MarketPlace e clique em Adicionar

Figura 67 - Inserindo a licança

Fonte: http://www.virtualbreadboard.com/

O Ticket verde é mostrado quando corretamente ativado

Figura 68 - Finalizando a compra

Fonte: http://www.virtualbreadboard.com/

A licença permite o acesso aos serviços de Internet necessários ao VBB, para funcionar você também precisa de uma conexão de internet em tempo integral para utilizar a licença.

As licenças podem ser tanto uma licença vitalícia ou uma assinatura anual.

A licença é intransferível, periodicamente, uma nova chave de licença é emitida

A licença pode ser usada em 3 de seus próprios computadores, tipicamente PC de casa, PC de Trabalho e Laptop

Você pode atualizar sua licença dentro do período de dois meses por reembolso PayPal

1. A aquisição da Licença Atualizada;

2. Pedir um reembolso para a licença anterior.

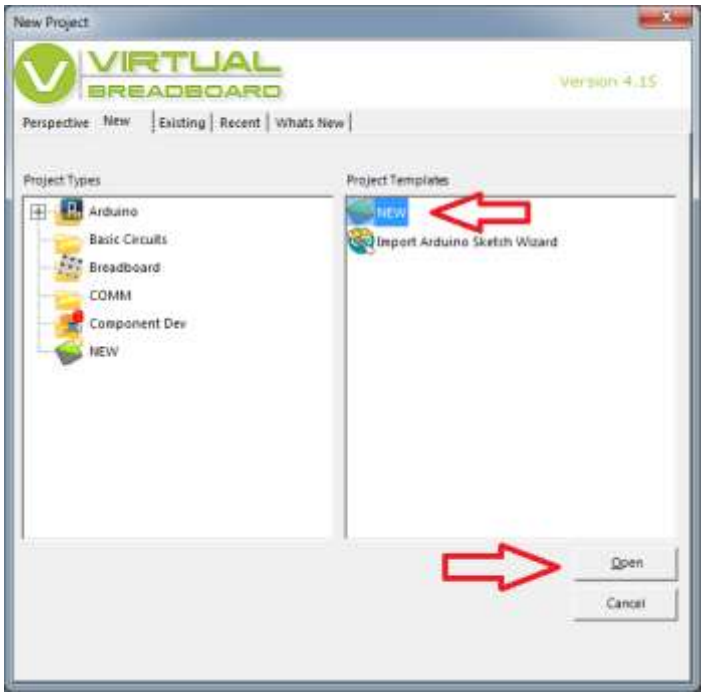
5.1.2 Simulando com Virtual Breadboard

Nesta experiência faremos um exemplo clássico. Utilizaremos uma plataforma Arduino, um protoboard, um led e um resistor com a versão 4.15 do Virtual Breadbord.

Ao abrir o Virtual BreadBoard será mostrado uma janela conforme a figura a seguir.

Para iniciar o projeto clique em NEW e em seguida no botão [Open]

Figura 69 - Iniciando um novo projeto

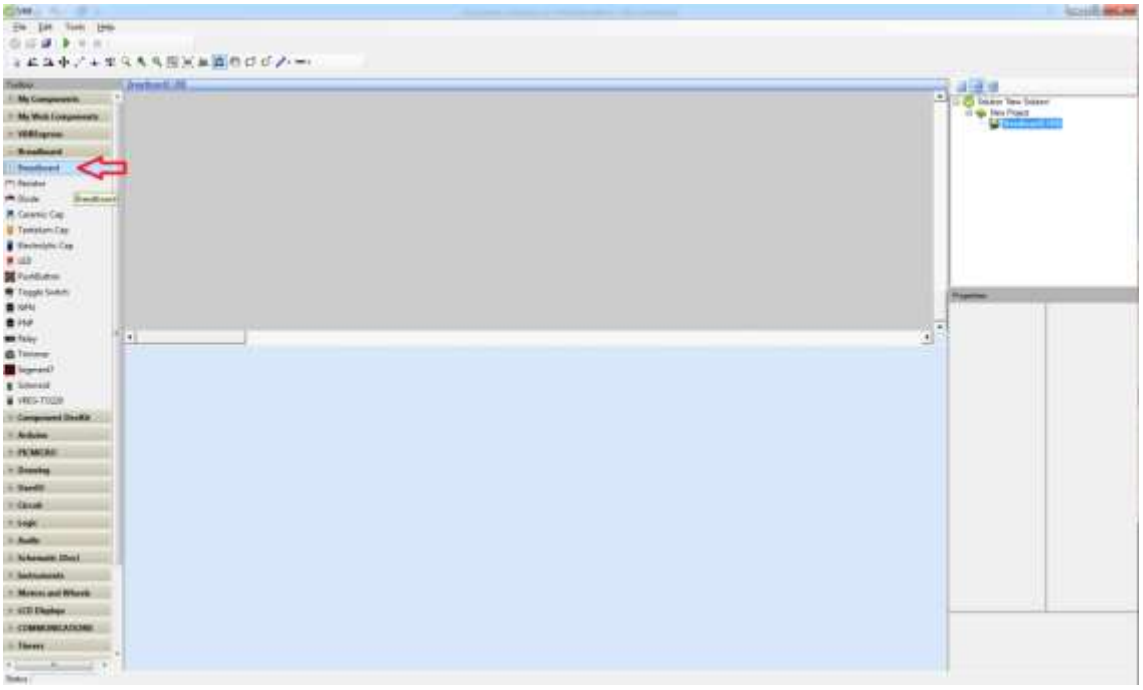


Fonte: Acervo Pessoal

Vamos colocar um protoboard (matriz de contato) para acomodar as ligações do projeto.

No menu do lado esquerdo vá até a opção Breadboard, selecione o componente breadboard e clique na área cinza.

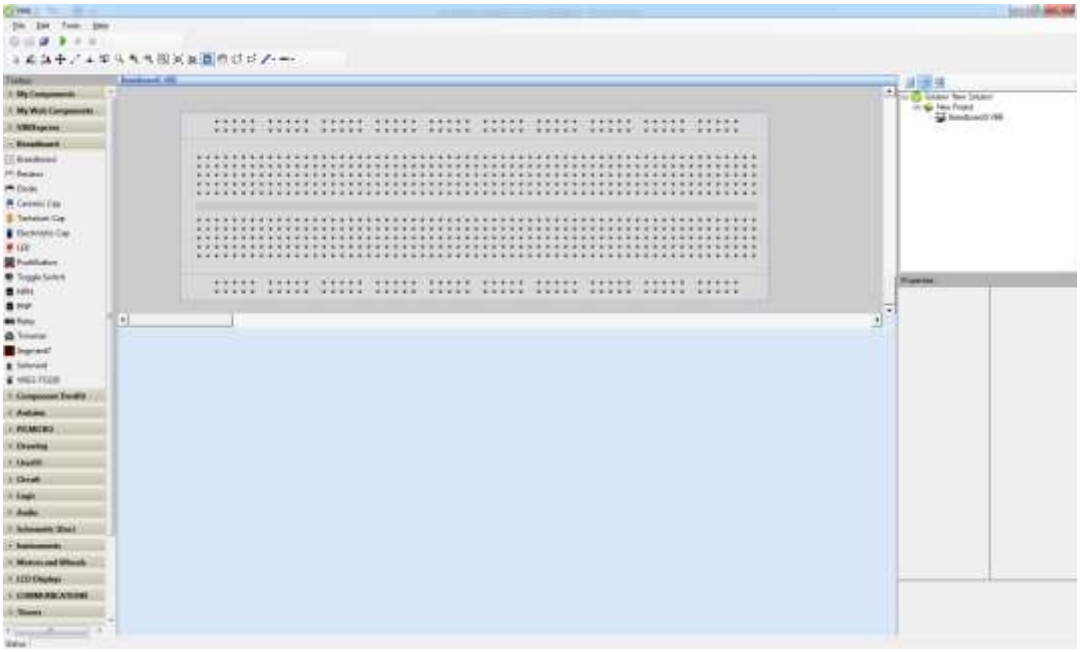
Figura 70 - Inserindo uma matriz de contato



Fonte: Acervo Pessoal

Aparecerá o componente conforme figura a seguir

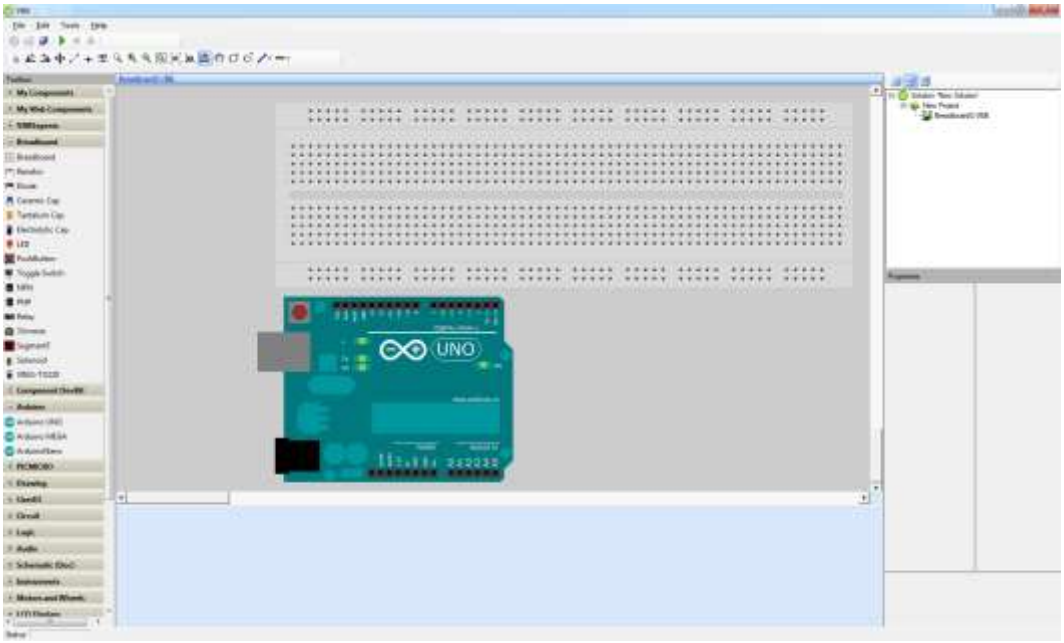
Figura 71 - Matriz de contatos inserida



Fonte: Acervo Pessoal

Volte ao menu, vá até a opção Arduino, selecione o modelo desejado (utilizamos o modelo UNO) e clique na área cinza.

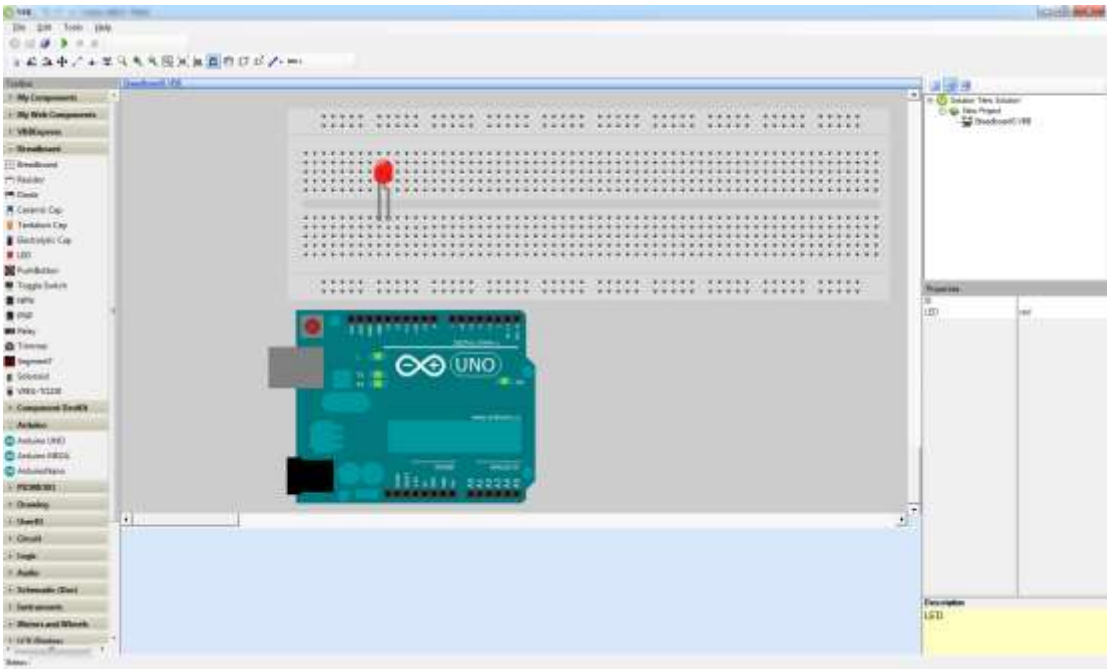
Figura 72 - Inserindo um arduino



Fonte: Acervo Pessoal

Volte ao menu, selecione o componente Led, clique na matriz de contato e organize o componente conforme a necessidade.

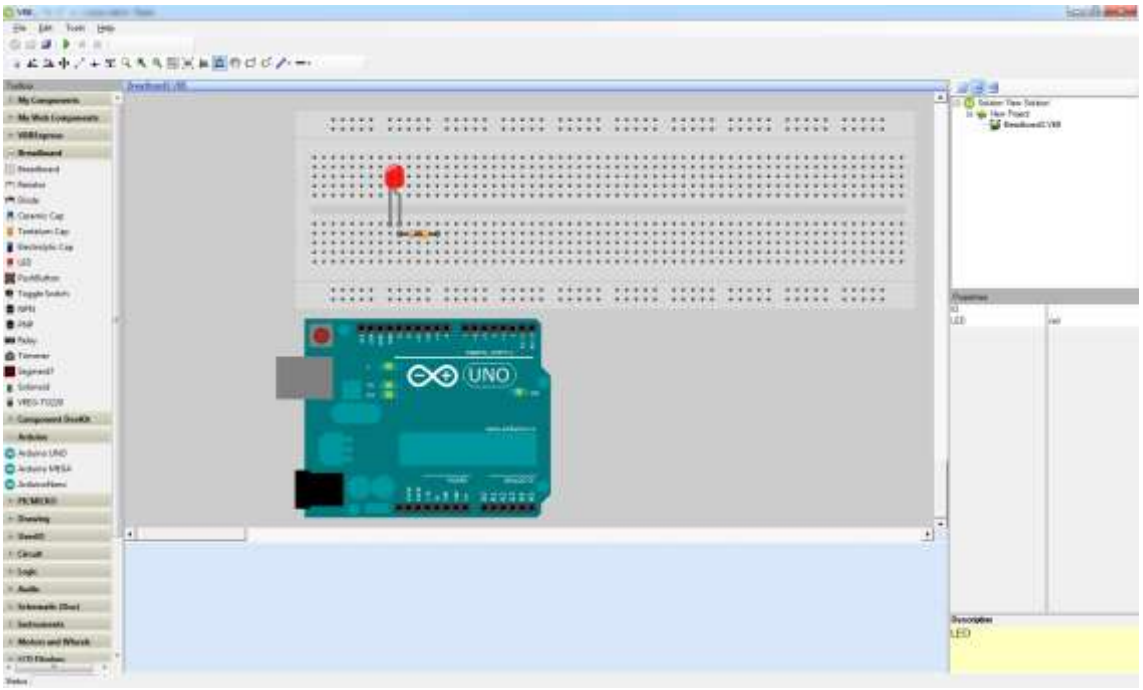
Figura 73 - Inserindo um led



Fonte: Acervo Pessoal

Volte ao menu, selecione o componente Resistor, clique na matriz de contato e organize o componente conforme a necessidade ligando-o ao componente Led respeitando a ordenação das trilhas do protobord.

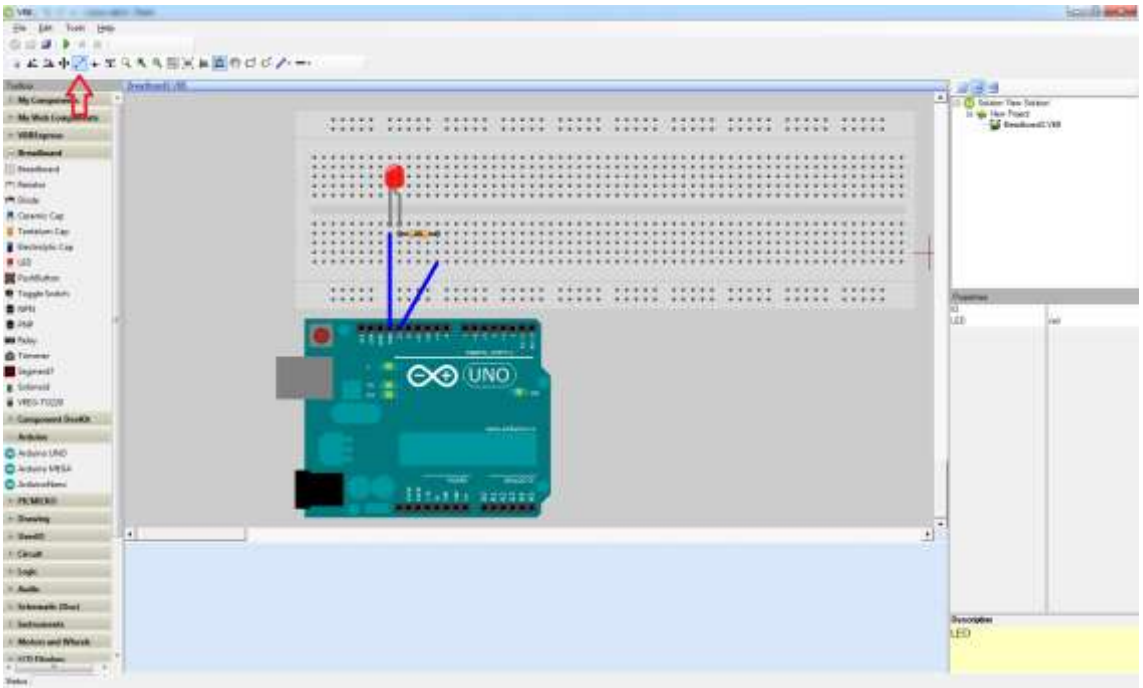
Figura 74 - Inserindo resistor



Fonte: Acervo Pessoal

Clique no menu superior e selecione o componente Trilha conforme figura a seguir. Faça as devidas ligações conforme figura a seguir. Nesta experiência ligamos o pino 13 do arduino ao positivo do led, o negativo do led em um dos lados do resistor e o outro lado do resistor ao GND do arduino.

Figura 75 - Ligando os componentes

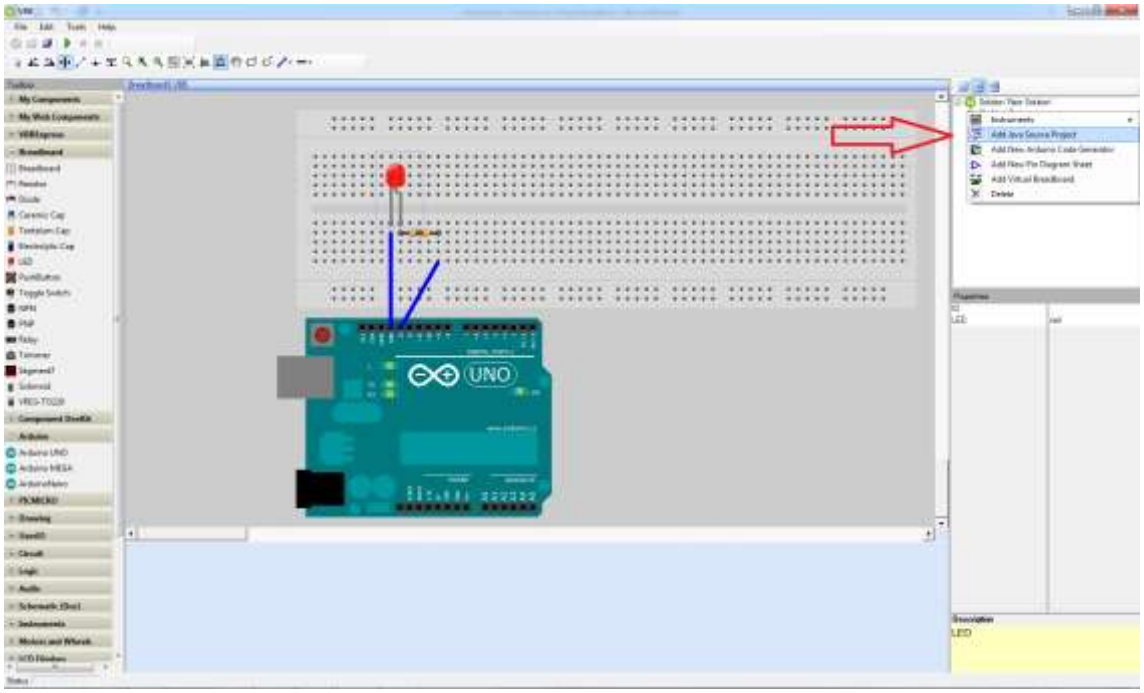


Fonte: Acervo Pessoal

Depois do circuito montado é hora de iniciar a programação.

Do lado direito, clique com o botão direito em New Project e escolha a opção “Add Java Source Project”.

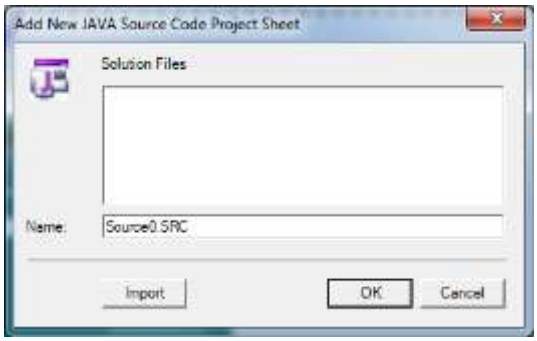
Figura 76 - Iniciando a programação



Fonte: Acervo Pessoal

Digite um nome ou deixe como o padrão Source0.SRC

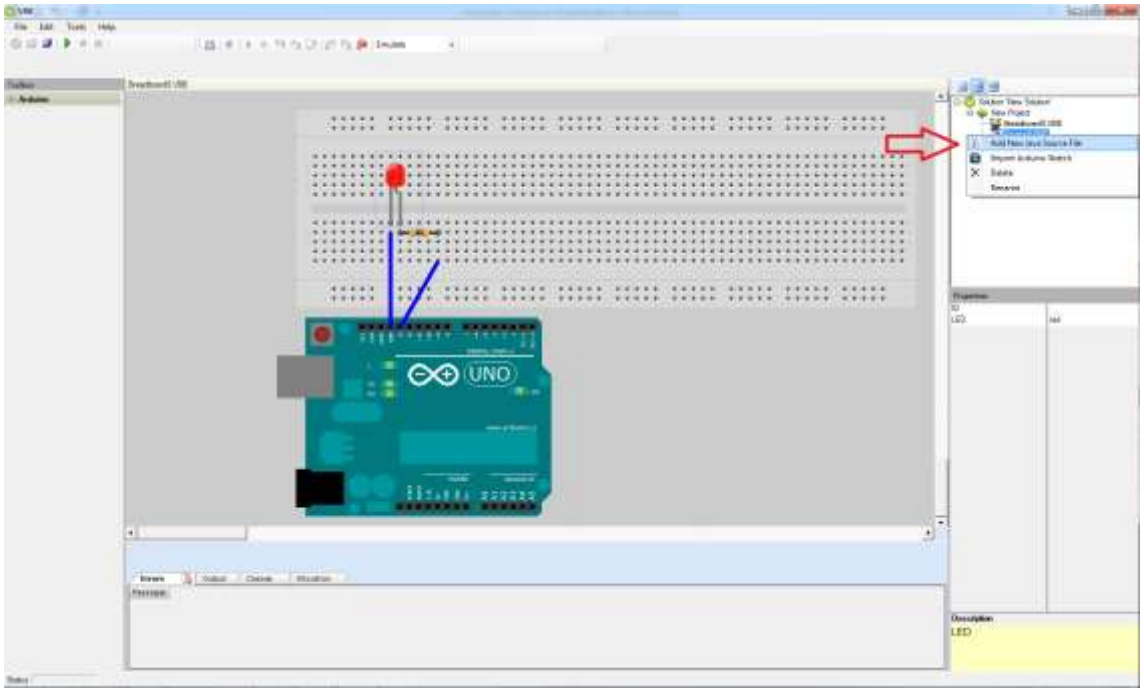
Figura 77 - Criando o arquivo de programação



Fonte: Acervo Pessoal

Clique com o botão direito no nome anteriormente criado “Source0.SRC e selecione a opção “Add New Java Source File”

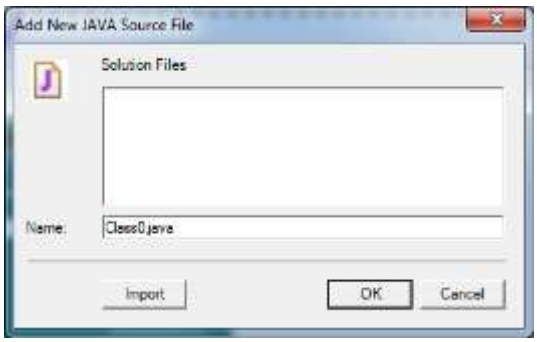
Figura 78 - Criando um arquivo de componente java



Fonte: Acervo Pessoal

Digite um nome ou deixe como o padrão Class0.SRC.

Figura 79 - Nomeando o arquivo java

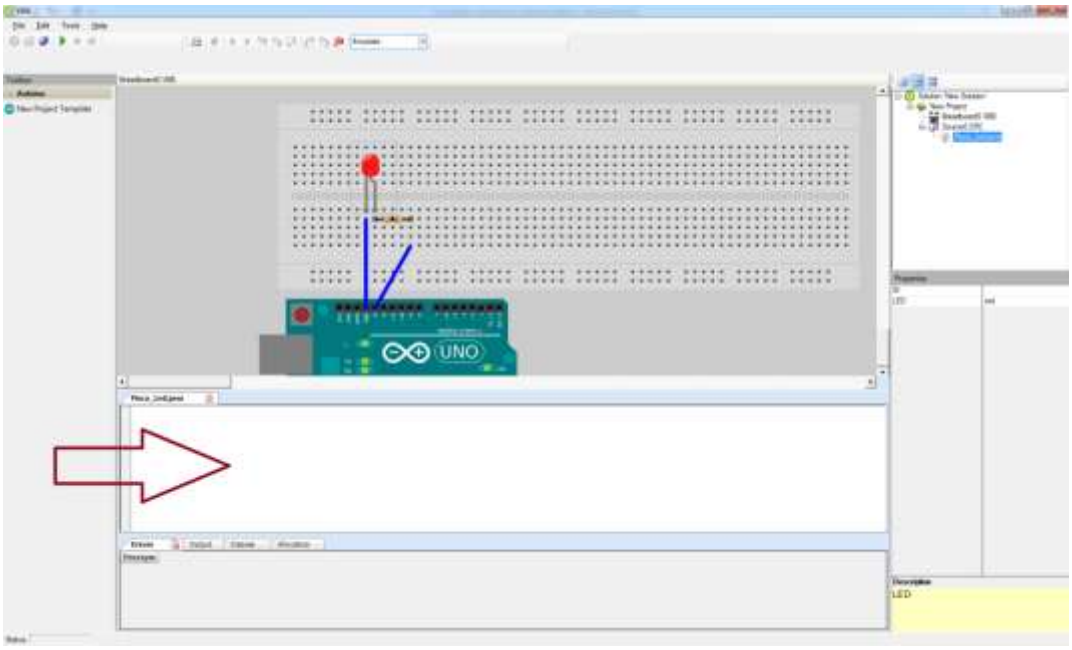


Fonte: Acervo Pessoal

Dica: Coloque o nome sugestivo de acordo com nosso projeto. Como exemplo usaremos o nome Pisca-Led.

Com isso será criada a área da programação conforme figura 80 a seguir

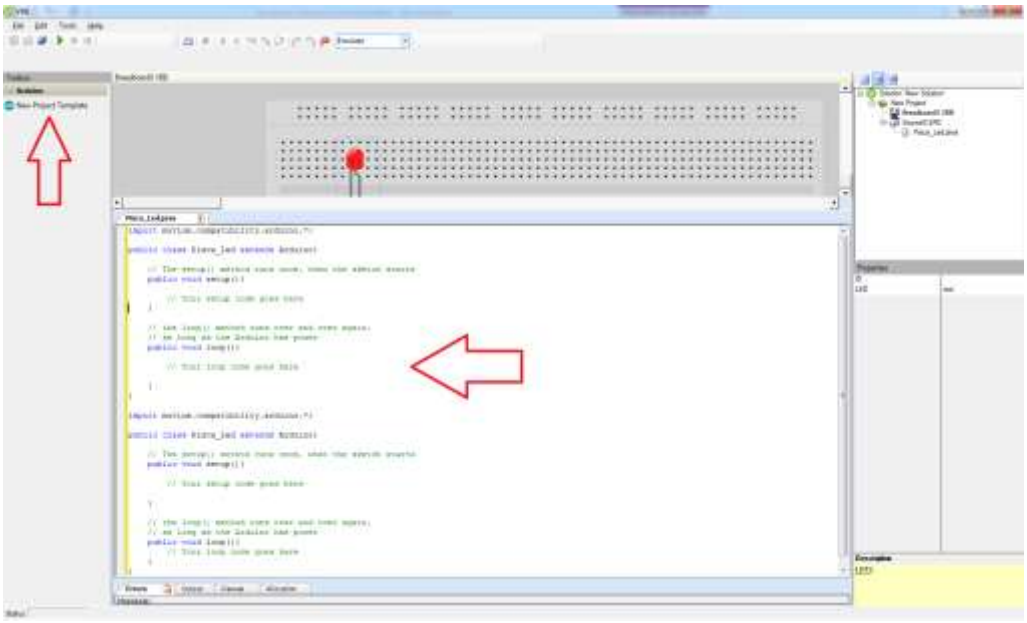
Figura 80 - Iniciando a área de programação



Fonte: Acervo Pessoal

Expanda a opção “Arduino” no menu esquerdo e em seguida dê um duplo clique em New Project Template. Isso fará o preenchimento do corpo da programação com as funções Void Setup e Void Loop de acordo com a linguagem Java conforme figura a seguir.

Figura 81 - Expandindo a área de programação



Fonte: Acervo Pessoal

Insira a programação desejada. Nosso exemplo é mostrado na figura a seguir.

Figura 82 - Programação

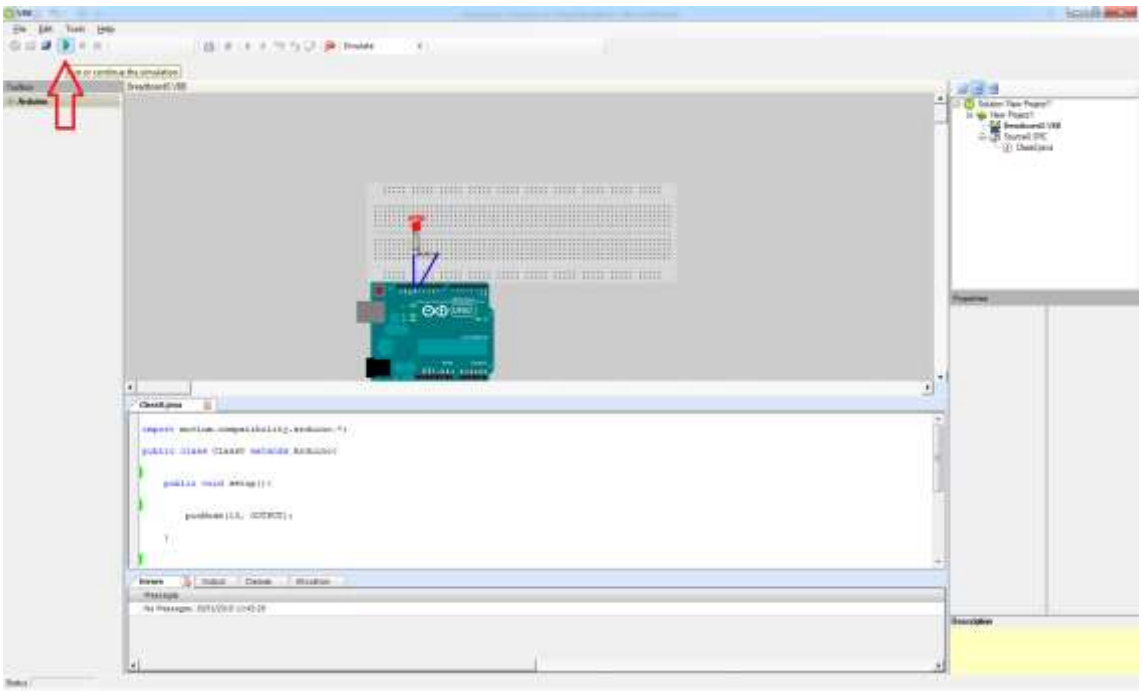
Class0.java

```
import muvium.compatibility.arduino.*;

public class Class0 extends Arduino{


    public void setup(){


        pinMode(13, OUTPUT);

    }


    public void loop(){



        digitalWrite(13, HIGH);
        delay(500);
        digitalWrite(13, LOW);
        delay(500);


    }
}
```

Fonte: Acervo Pessoal

Para testar a aplicação clique no botão com símbolo de seta na parte superior conforme figura a seguir

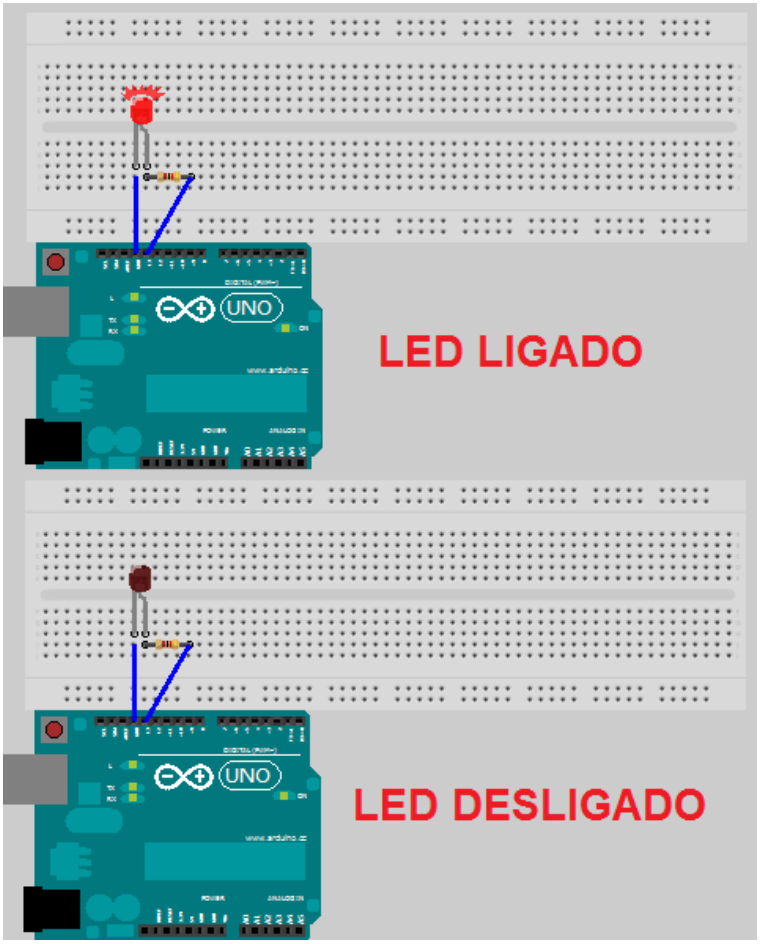
Figura 83 - Testando a aplicação



Fonte: Acervo Pessoal

Com isso poderá ser visto o LED piscando de ½ em ½ segundo conforme a programação.

Figura 84 - Testando o projeto



Fonte: Acervo Pessoal

5.2 Isis Proteus

O Proteus atualmente é o um dos melhores simuladores de circuitos eletrônicos. Apesar de ser em inglês possui uma interface extremamente fácil de se trabalhar, é altamente recomendado para a criação de projetos eletrônicos, tanto para teste de circuitos quanto para elaborar placas de circuito impresso, para tal missão acompanha o programa duas interfaces o ARES e o ISIS.

- **ARES:** Utilizado para a elaboração de placas de circuito impresso, é bastante fácil de mexer.
- **ISIS:** Este é simulador de circuitos, possui uma vasta biblioteca de componentes e é o mais indicado para tal finalidade. Para as versões instaláveis que são mais completas e se pode atualizar bibliotecas e tudo mais, você deve executar todos os cracks como administrador se não nem funciona.

A nova versão do Suite Poteus 8 representa mais de três anos de desenvolvimento contínuo e inclui melhorias em cada área do conjunto de software. Um grande trabalho sobre a estrutura do aplicativo em conjunto com o introdução de um banco de dados que possibilita um fluxo de trabalho muito mais estável para os usuários enquanto os novos conjunto de recursos economiza tempo e esforço no ciclo de vida do projeto.

5.2.1 Configurando o Isis Proteus para utilização e simulação

Uma vez adquirido e instalado o software Proteus, siga os procedimentos a seguir.

- Baixe o arquivo no seguinte endereço:

http://playground.arduino.cc/uploads/Main/LiquidCrystal.zip

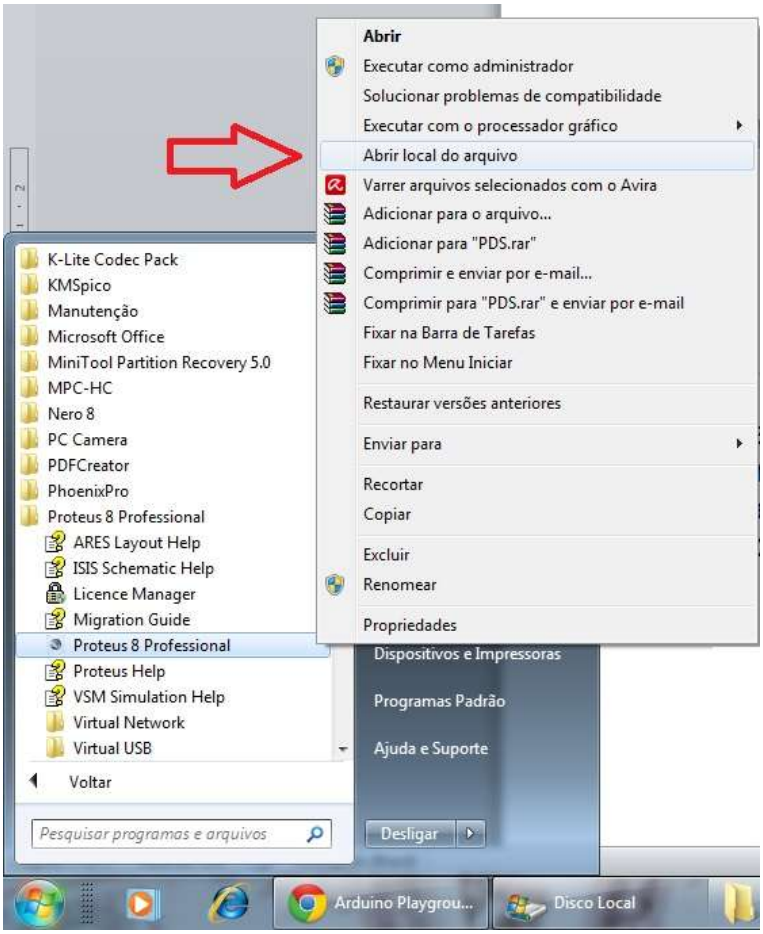
Este arquivo LiquidCrystal.zip contem os seguintes arquivos compactados:
- LiquidCrystal.h , LiquidCrystal.cpp e keywords

Estes arquivos fazem parte do componente Display que será utilizado na simulação de um display no arduino.

Descompacte estes arquivos, copie-os e cole-os dentro da pasta Library do software Proteus.

Caso não saiba o caminho vá até o ícone do software Proteus, clique com o botão direito e vá até a opção “abrir local do arquivo” conforme figura 85 a seguir.

Figura 85 - Configurando o Isis Proteus

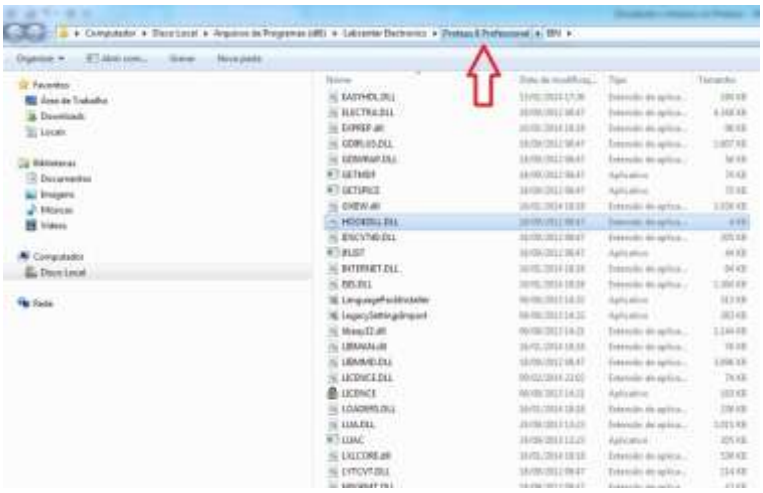


Fonte: Acervo Pessoal

Em seguida será aberta a localização da pasta.

A pasta BIN estará aberta. Volte uma pasta “C:\Program Files (x86)\Labcenter Electronics\Proteus 8 Professional” conforme figura 86 a seguir.

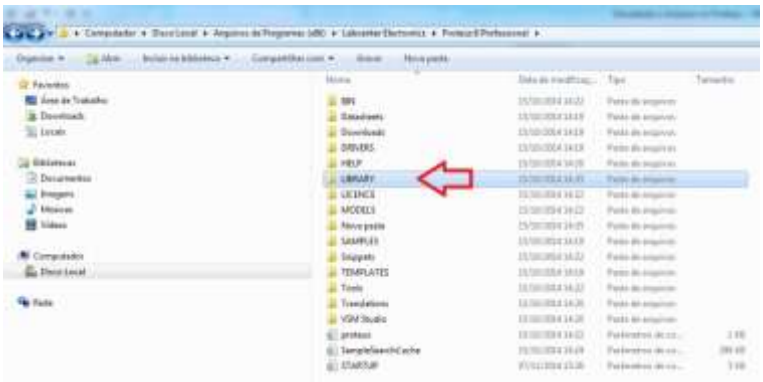
Figura 86 - Localizando a pasta de instalação



Fonte: Acervo Pessoal

Abra a pasta LIBRARY indicada na figura 87 a seguir.

Figura 87 - Pasta de Biblioteca de Componentes



Fonte: Acervo Pessoal

- Baixe o arquivo no seguinte endereço:

http://www.theengineeringprojects.com/DownloadSoftware/Proteus%20Arduino%20Library.rar

Este arquivo Proteus Arduino Library.rar contem os seguintes arquivos compactados:

Arduino.idx e Arduino.lib

Estes arquivos fazem parte do componente Arduino Básico Uno R3, Mega 1280 e Mega 2560 que serão utilizados para simular estes Arduinos.

- Baixe o arquivo no seguinte endereço:

https://github.com/blogembarcadobr/Library/archive/master.zip

Este arquivo máster.zip contem os seguintes arquivos compactados:

Blogembarcados.lib e UltraSonicSensor.hex
Estes arquivos fazem parte de outros modelos de Arduino que serão utilizados para simular tais componentes.

5.2.2 Iniciando um projeto no IDE Arduino

Abra a IDE do Arduino.
Abra o exemplo Blink em:
Menu File(Arquivo) / Examples (Exemplos) / Basics / Blink

Figura 88 - Abrindo um projeto exemplo

Fonte: Acervo Pessoal

Clique no botão Upload (Carregar) para gerar o arquivo.hex

Figura 89 - Botão Upload (Carregar)

Fonte: Acervo Pessoal

Como o arduino não está conectado à porta USB o software informará o seguinte erro:
Não se preocupe pois mesmo com esse erro o arquivo .hex já foi gerado com sucesso.

5.2.3 Copiando o arquivo de programação para a pasta do Isis Proteus

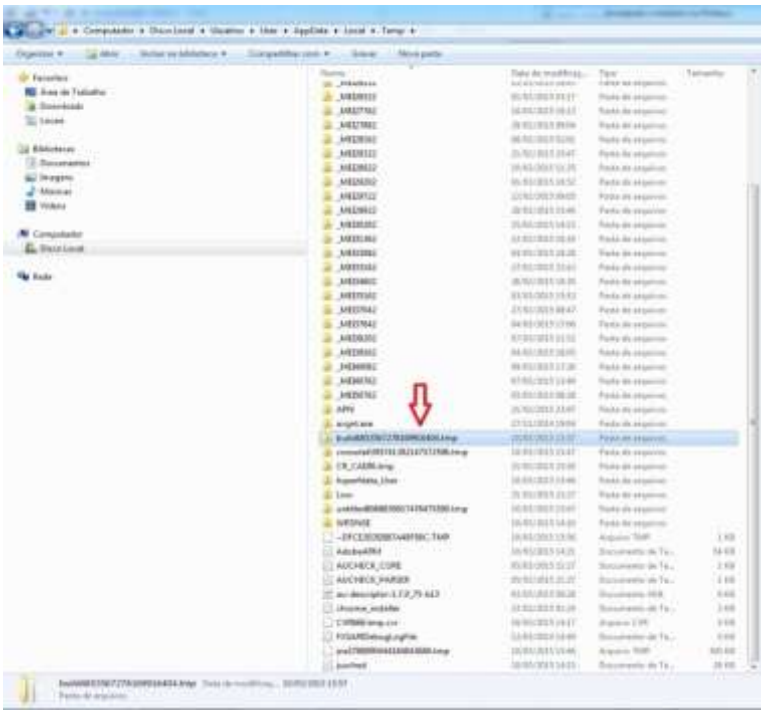
Explore “Meu Computador” e navegue até a seguinte pasta:

C:\Users\User\AppData\Local\Temp

Obs.: Lembrando que “User” é o nome do usuário do sistema do computador.

Uma vez dentro desta pasta localize uma pasta chamada “Buildxxxxxxx” onde “x” será uma sequencia de números.

Figura 90 - Caminho da Pasta Build



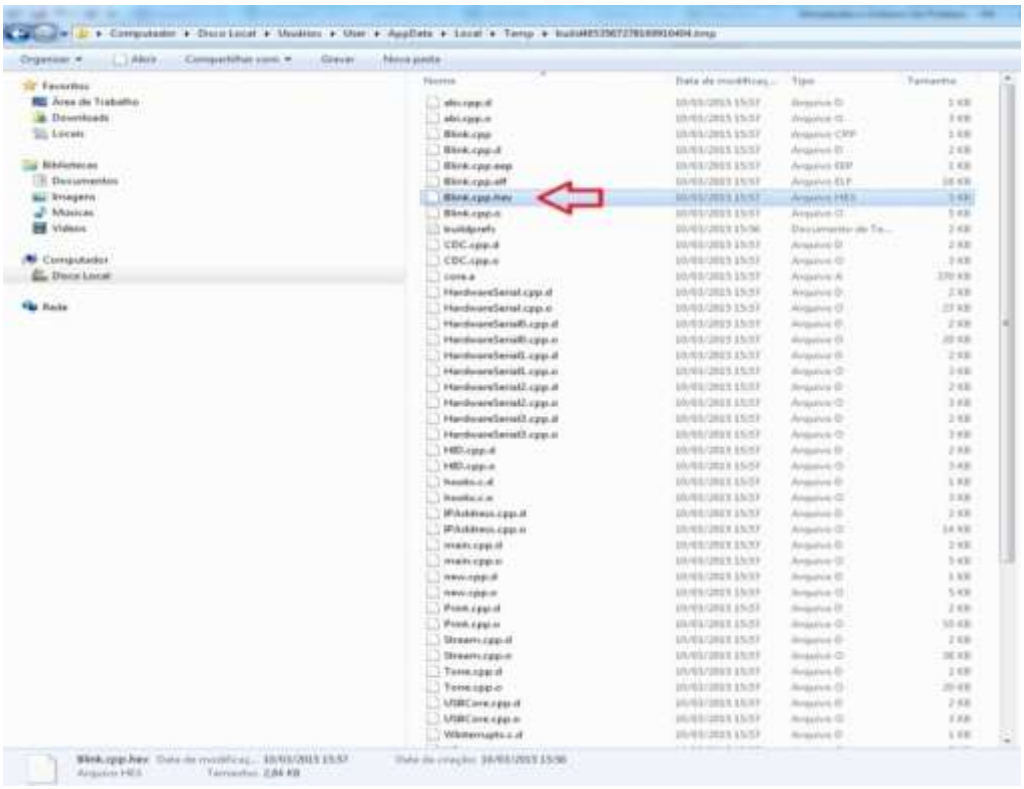
Fonte: Acervo Pessoal

Abra essa pasta e localize o arquivo Blink.hex

Esse arquivo será necessário para alimentar a programação do simulador no software Proteus.

Obs.: Lembrando que este arquivo assume o nome do projeto do IDE do Arduino.

Figura 91 - Arquivo .hex

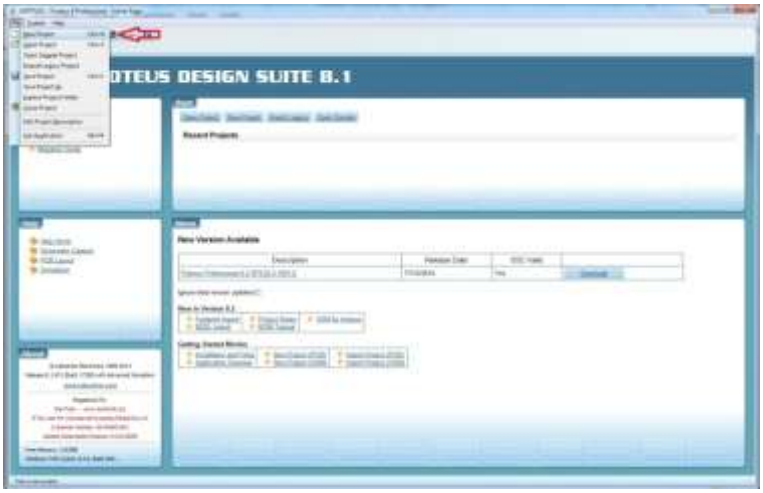


Fonte: Acervo Pessoal

5.2.4 Simulando com Isis Proteus

Vá no menu File / New Project

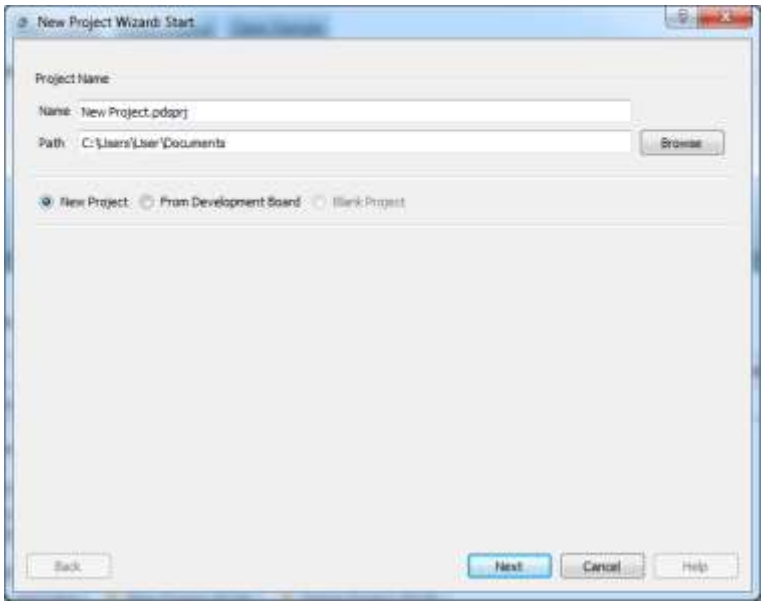
Figura 92 - Iniciando um novo projeto



Fonte: Acervo Pessoal

Dê um nome para o projeto ou deixe como New Project.pdsprj conforme a figura 93 a seguir.

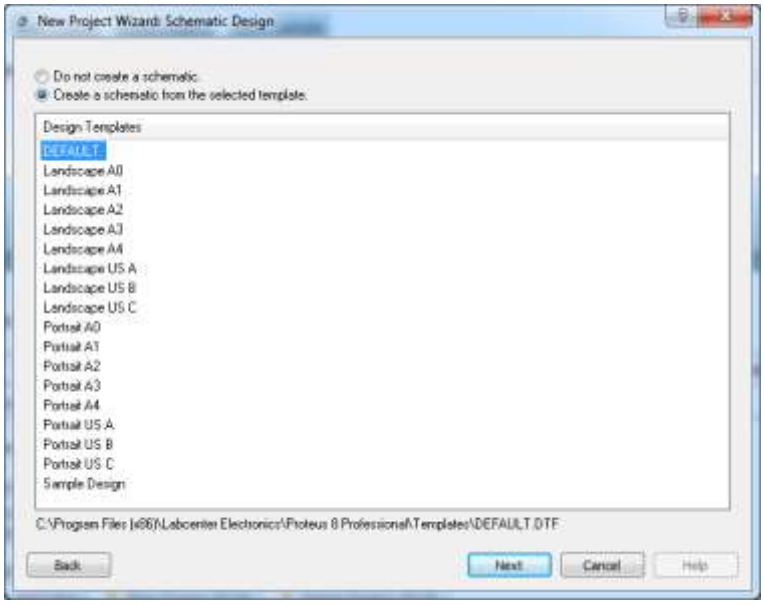
Figura 93 - Nome do projeto



Fonte: Acervo Pessoal

A próxima janela (figura 94) refere-se ao projeto esquemático. Clique em “Create a Schematic from the selected template. Deixe a primeira opção marcada “Default”.

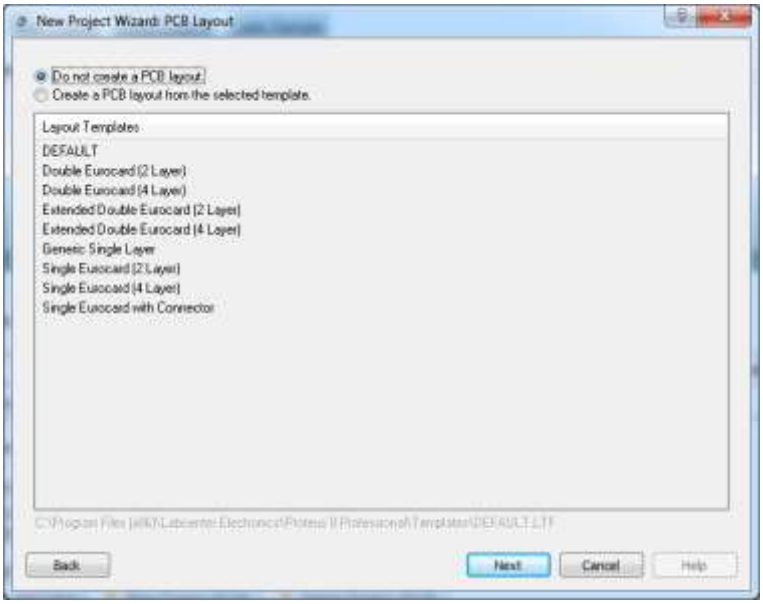
Figura 94 - Projeto esquemático



Fonte: Acervo Pessoal

A próxima janela (figura 95) refere-se ao projeto PCB e pode ser ignorado deixando selecionado a opção “do not create a PCB Layout” pois neste caso iremos fazer apenas a simulação.
Clique em Next para continuar

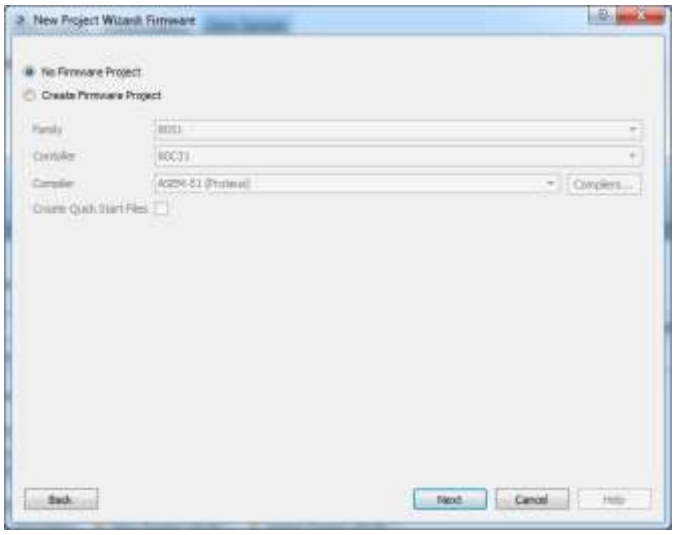
Figura 95 - Projeto PCB Layout



Fonte: Acervo Pessoal

A próxima janela (figura 96) refere-se ao projeto Firmware e pode ser ignorado deixando selecionado a opção “No Firmware Project” pois neste caso iremos fazer apenas a simulação.
Clique em Next para continuar

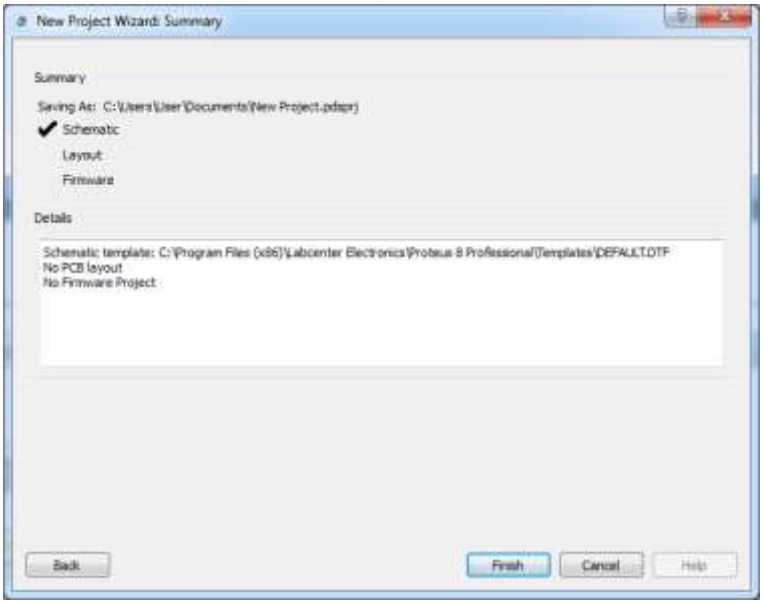
Figura 96 - Projeto de Firmware



Fonte: Acervo Pessoal

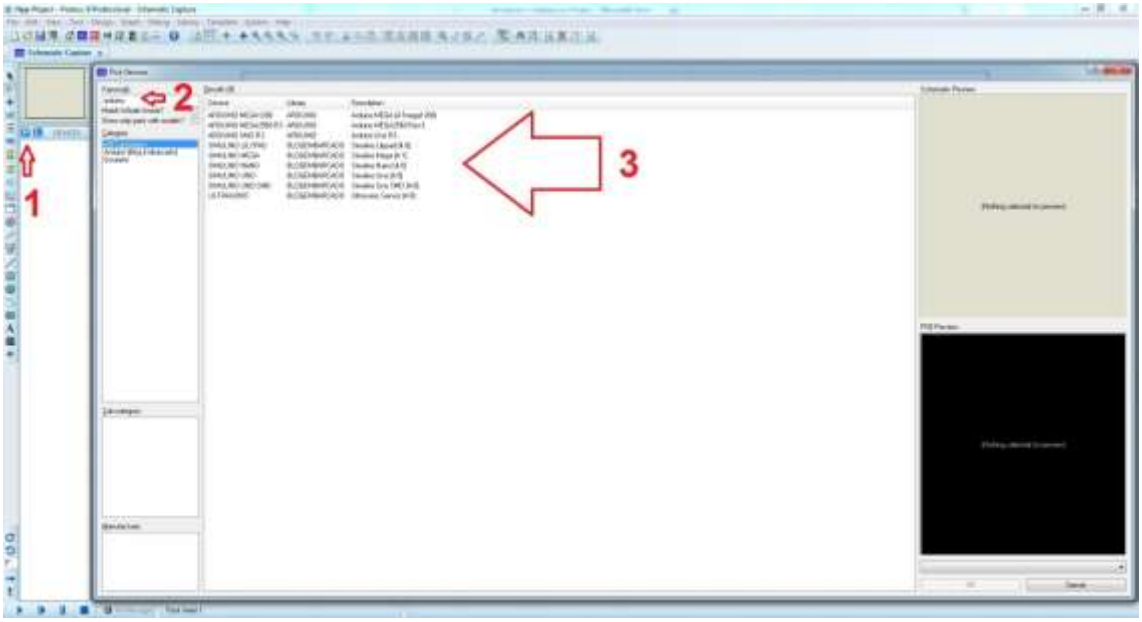
A próxima janela (Figura 97) dá o resumo dos projetos que serão iniciados. Clique em “Finish”.

Figura 97 - Resumo dos projetos a serem iniciados



Fonte: Acervo Pessoal

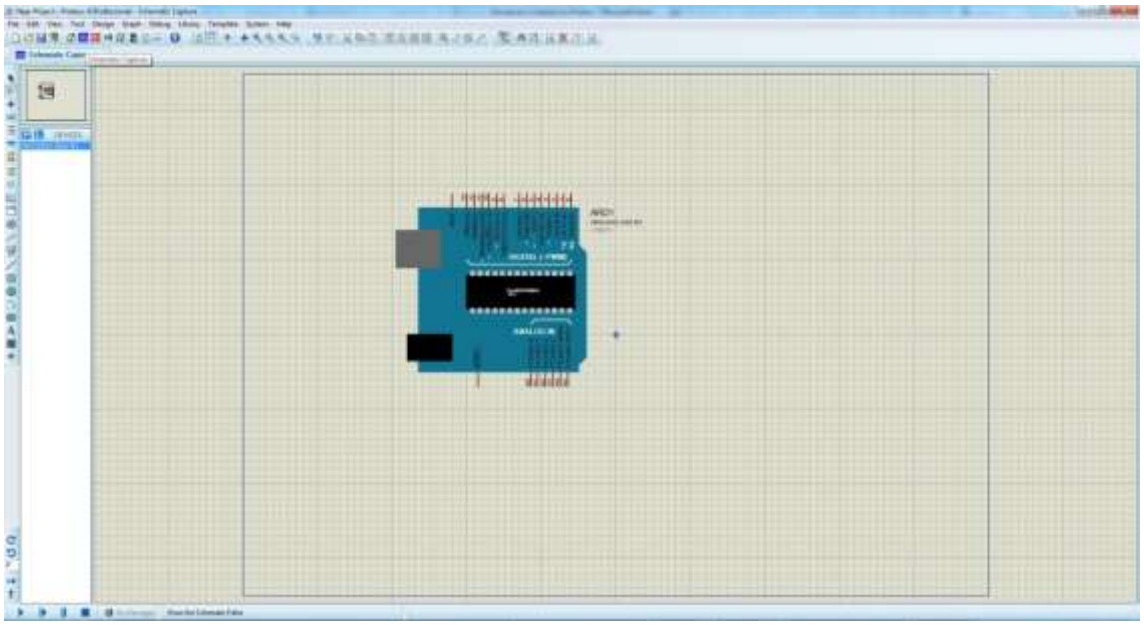
Figura 98 - Selecionando o modelo do Arduino



Fonte: Acervo Pessoal

Selecione o modelo desejado. Em seguida clique no nome do modelo no menu esquerdo e depois clique na área quadriculada. O componente irá aparecer conforme figura 99 a seguir.

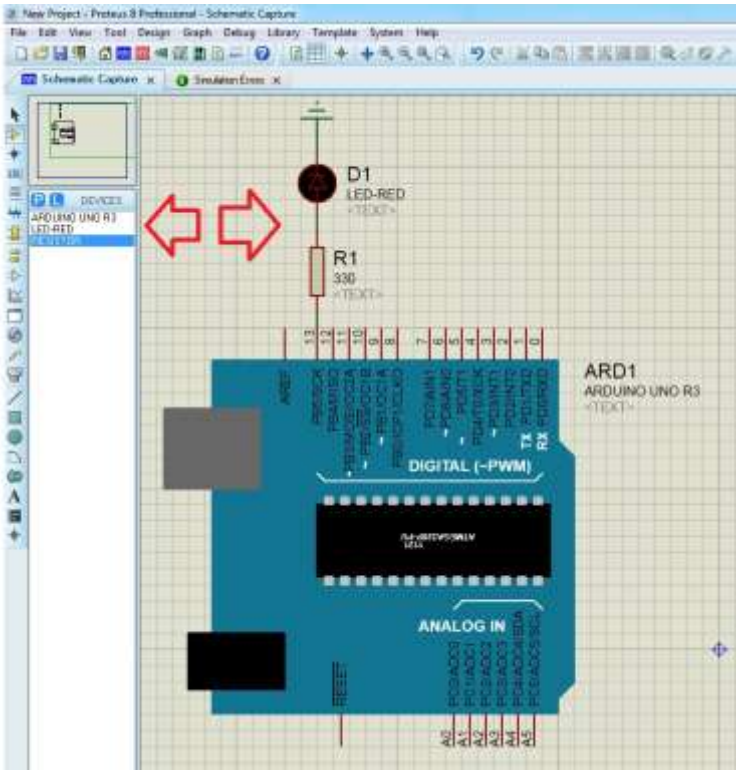
Figura 99 - Componente Arduino



Fonte: Acervo Pessoal

Repita esse procedimento para os demais componentes: Resistor e Led conforme figura 100 a seguir.

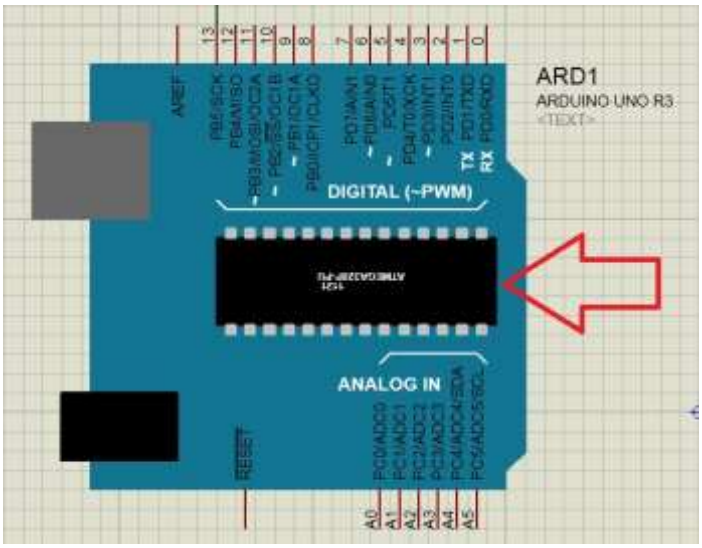
Figura 100 - Componentes do projeto



Fonte: Acervo Pessoal

Duplo clique no componente Arduino para abrir a janela de pesquisa do código .hex conforme figura 101 a seguir.

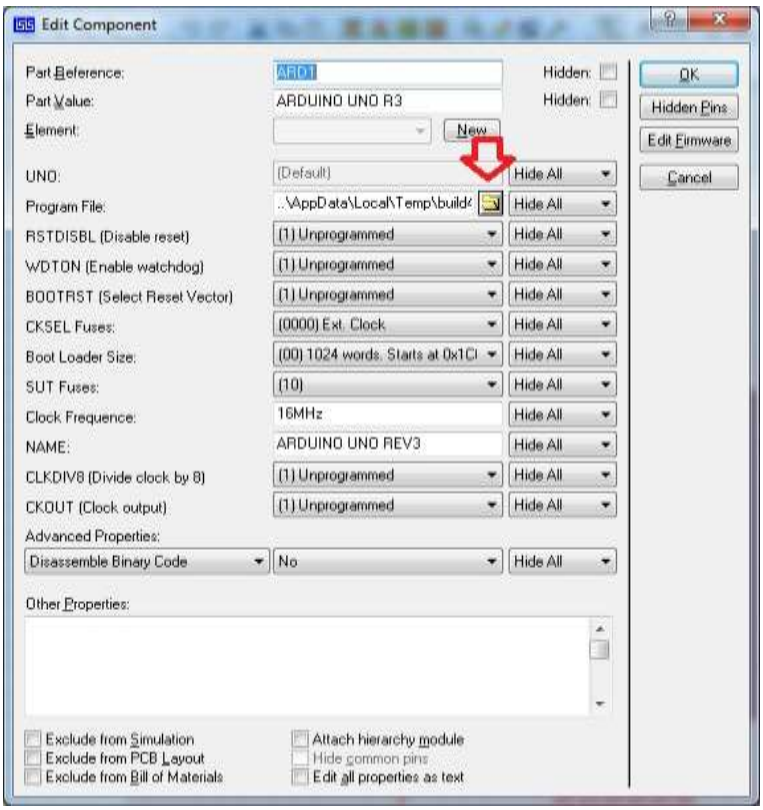
Figura 101 - Clicando no microcontrolador



Fonte: Acervo Pessoal

Em seguida na janela que abrir (figura 102) clique na pasta para localizar o arquivo.

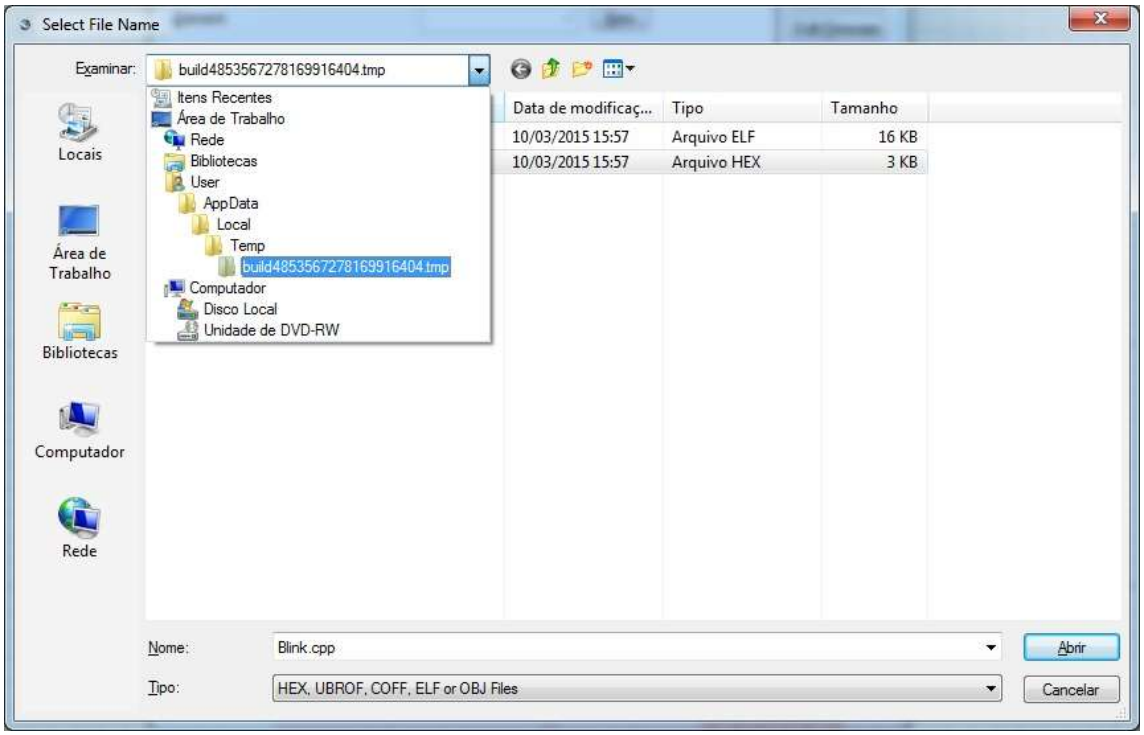
Figura 102 - Localizando o arquivo .hex



Fonte: Acervo Pessoal

Navegue até a pasta do arquivo anteriormente encontrado conforme mostra a figura 103.

Figura 103 - Encontrando a pasta do arquivo



Fonte: Acervo Pessoal

Selecione o arquivo Blink.hex e clique em abrir

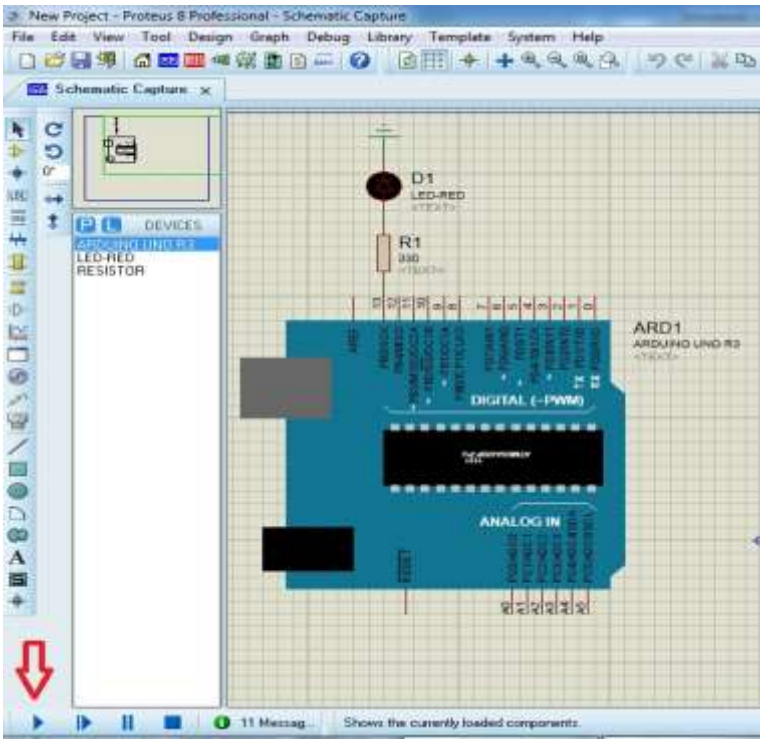
Figura 104 - Selecionando o arquivo .hex

Fonte: Acervo Pessoal

Com esse procedimento carregamos a programação para o microcontrolador.

Agora clique no botão para rodar a simulação indicado na figura 21 a seguir.

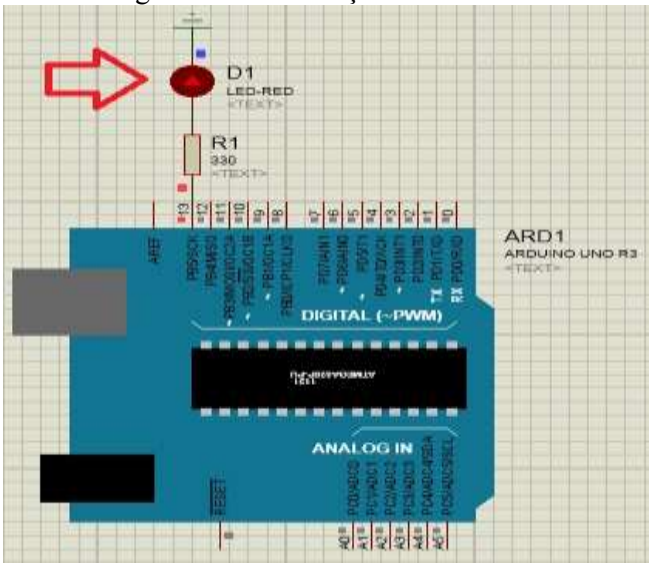
Figura 105 - Testando a simulação



Fonte: Acervo Pessoal

Isso fara a simulação rodar figura 106. Note que a cor do led mudou indicando que o mesmo está energizado. Como essa experiência faz o led piscar com a frequência de 1 segundo, poderá ser observada a mudança de cor frequente enquanto a simulação estiver rodando.

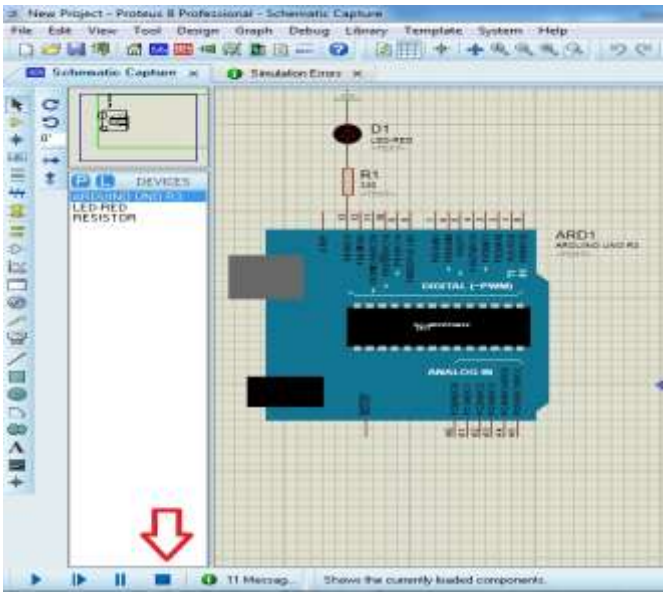
Figura 106 - Simulação funcionando



Fonte: Acervo Pessoal

Caso queira parar a simulação é só clicar no botão STOP figura 107.

Figura 107 - Parando a simulação



Fonte: Acervo Pessoal

6 App Inventor: Uma linguagem especializada em Android

App Inventor é uma ferramenta desenvolvida pela Google e mantida pela MIT que permite a criação de aplicativos para smartphones e tablets que rodam o sistema operacional Android, sem que seja necessário conhecimento em programação. Seu objetivo é permitir que utilizadores sejam também criadores e não apenas consumidores.

Com uma interface simples e fácil de usar, o programa foge das linhas de programação normal e possibilita até mesmo usuários comuns de lançarem seus aplicativos. Graças ao recurso "*drag and drop"*, a programação das aplicações acontece de forma simples e intuitiva.
Nas interfaces gráficas de computadores, ***drag-and-drop*** (arrastar e largar) é a ação de clicar em um objeto virtual e "arrastá-lo" a uma posição diferente ou sobre outro objeto virtual.

6.1 Por quê utilizar App Inventor?

Primeiramente a programação por blocos facilita o desenvolvimento uma vez que é mais intuitivo que os modelos convencionais. Outro argumento significativo é a facilidade de desenvolvimento e testes em tempo real. Sem a necessidade de instalar a **IDE (Ambiente de Desenvolvimento Integrado)**, uma vez que o processo ocorre todo na nuvem, torna-se prático a execução do mesmo e foge da limitação do equipamento, sendo desenvolvido através de qualquer computador que atenda os requisitos mínimos, pois a aplicação funciona de modo inteiramente online, diretamente do navegador e para utilizá-la, basta ter uma conta de usuário da Google.

Na educação, a utilização desse modelo de programação vem ganhando força. Programadores e celebridades como Bill Gates (Microsoft) e Mark Zuckerberg (Facebook) se juntaram à organização Code.org para incentivar crianças e jovens a aprender programação. A Code.org é uma organização sem fins lucrativos que visa aumentar o ensino de programação de computadores em escolas e direcionar pessoas através de ferramentas online.

Para mais detalhes sobre o que é IDE consulte o Anexo III.

6.2 Conhcecendo o Ambiente de Desenvolvimento App Inventor

Inicialmente iremos configurar o ambiente de desenvolvimento.

App Inventor é um ambiente de desenvolvimento online. Atualmente tipos de serviços que são executados online são denominados de “computação em nuvem” (Cloud Computing).

O requisitos e passos para se configurar o App Inventor são os descritos a seguir:

- Ter ou criar uma conta de email Gmail

➔ Entrar em http://www.gmail.com e clicar no botão [Criar uma conta] no canto superior direito conforme figura 108 e seguir os passos para criação do email.

Figura 108 - Bem vindo ao Gmail

Fonte: http://gmail.com

Acessar através do PC/Notebook o site http://appinventor.mit.edu

- Clicar no botão CREATE indicado na figura 109 a seguir

Figura 109 - Iniciando a aplicação web do App Inventor

Fonte: http://appinventor.mit.edu

***Obs.:** Pode ocorrer se estiver utilizando o navegador Google Chrome, de aparecer uma mensagem sugerindo traduzir a página em cima do botão CREATE ocultando o mesmo. Neste caso feche a mensagem e prossiga.

- Será direcionado para o site Gmail para efetuar login conforme figura 110 a seguir. Faça o login.

Figura 110 - Tela de Login

Fonte: http://gmail.com

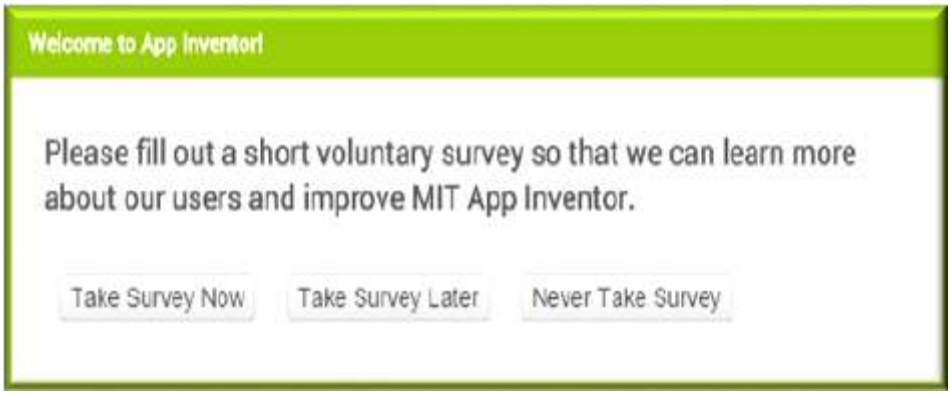
Permitir que o App Inventor tenha acesso a sua conta Google clicando no botão "Allow" conforme figura 111 a seguir.

Figura 111 - Clicando no botão Allow

Fonte: http://appinventor.mit.edu

- Aparecerá uma tela para coleta de dados para aprimoramento, figura 112 a seguir. Caso queira colaborar com informações no momento clique no botão **Take Survey Now**, se desejar colaborar depois clique no botão **Take Survey Later** e se não desejar colaborar clique no botão **Never Take Survey**.

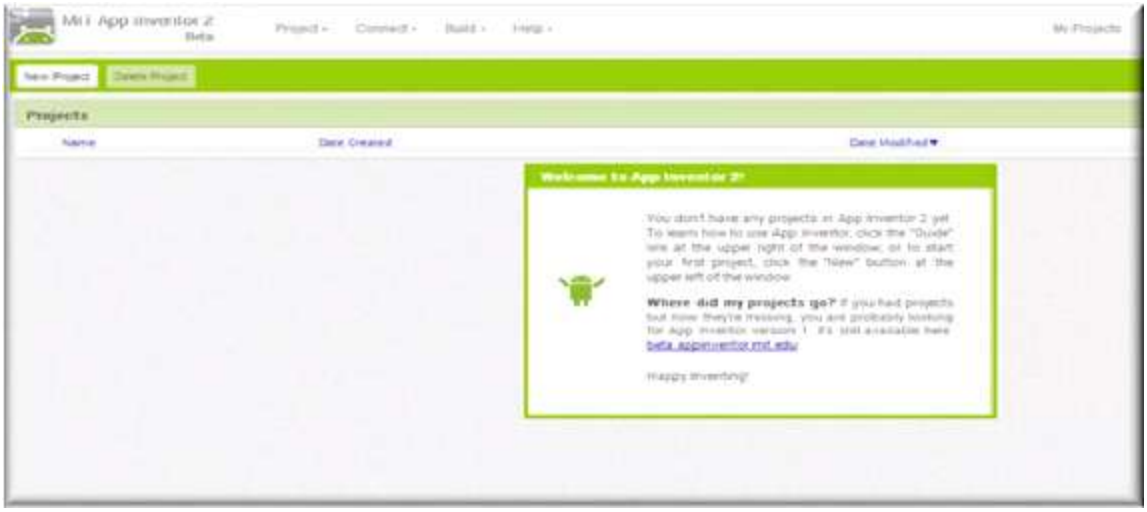
Figura 112 - Coleta de dados para aprimoramento do produto



Fonte: http://appinventor.mit.edu

- Após essa etapa abrirá o ambiente de desenvolvimento App Inventor com a mensagem de boas vindas conforme figura 113 abaixo. Clique em qualquer parte fora da mensagem para dar inicio ao projeto.

Figura 113 – Janela de Boas Vindas



Fonte: http://appinventor.mit.edu

- Para criar um novo projeto, clicar no botão “Start New Project” conforme figura 114 abaixo.

Figura 114 - Iniciando um novo projeto

Fonte: http://appinventor.mit.edu

6.3 Instalando o software MIT AI2 Companion no dispositivo móvel com Android.

Para se utilizar o software MIT AI2 COMPANION deve-se seguir os procedimentos descritos abaixo:

- Abrir o play store acessando o ícone mostrado na figura 115.

Figura 115 - Ícone Google Play

Fonte: https://play.google.com/store

- Pesquisar pelo MIT AI2 COMPANION

Figura 116 - MIT AI2 Companion no Play Store

Fonte: https://play.google.com/store

- **Instalar o MIT AI2 Companion clicando no botão instalar conforme figura 5-10**

Figura 117 - Instalar MIT AI2 Companion no Play Store

Fonte: https://play.google.com/store

6.4 Testando a aplicação

Existem 3 formas de testar os resultados das experiências, via WiFi, via Emulador ou via Cabo USB. Em nossas experiências utilizaremos a opção de rede WiFi por ser mais prático. O requisito necessário é que o PC/Notebook e o dispositivo móvel esteja na mesma rede WiFi conforme figura 118 a seguir.

Figura 118 - Notebook e dispositivo móvel na mesma rede WiFi

Fonte: http://appinventor.mit.edu

Para mais detalhes sobre a como testar a aplicação consulte:

- Anexo IV – Como configurar via WIFI.

- Anexo V – Como configurar via emulador.

- Anexo VI – Como configurar via cabo USB.

6.5 Iniciando as Experiências

As experiências são divididas nas seguintes etapas:

- Programação do Aplicativo que irá ser utilizado pelo usuário no dispositivo móvel;

- Programação do Arduino para efetuar as tarefas de acordo com a experiência;

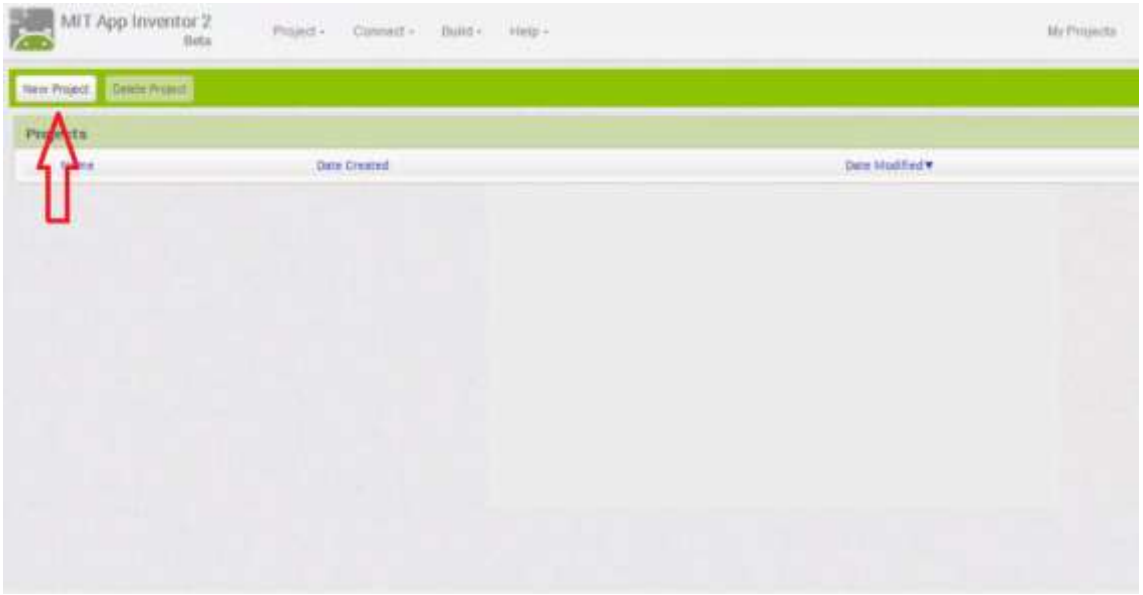
- Montagem do circuito eletrônico.

6.5.1 Primeiro Projeto

Nesse primeiro projeto iremos criar um aplicativo que escreverá o texto “Olá” ao ser clicado o botão para escrever e apagará o texto ao ser clicado o botão para limpar o texto.

- 1º - Para criar um novo projeto clicar no botão new Project conforme figura 119.

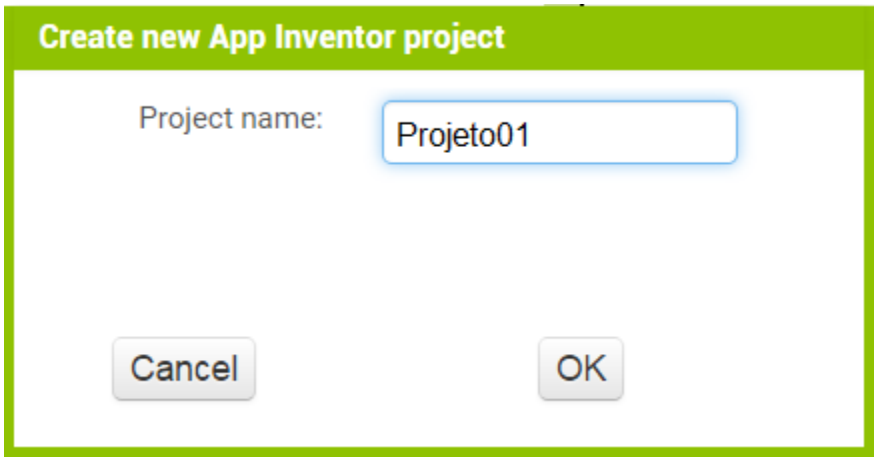
Figura 119 - Novo Projeto



Fonte: http://appinventor.mit.edu

2º - Dar um nome ao projeto sem espaço em branco (figura 120). Em nosso exemplo demos o nome de “Projeto01”.

Figura 120 - Nome do Projeto



Fonte: http://appinventor.mit.edu

Com isso será mostrado a imagem simbolizando o dispositivo móvel (figura 121).

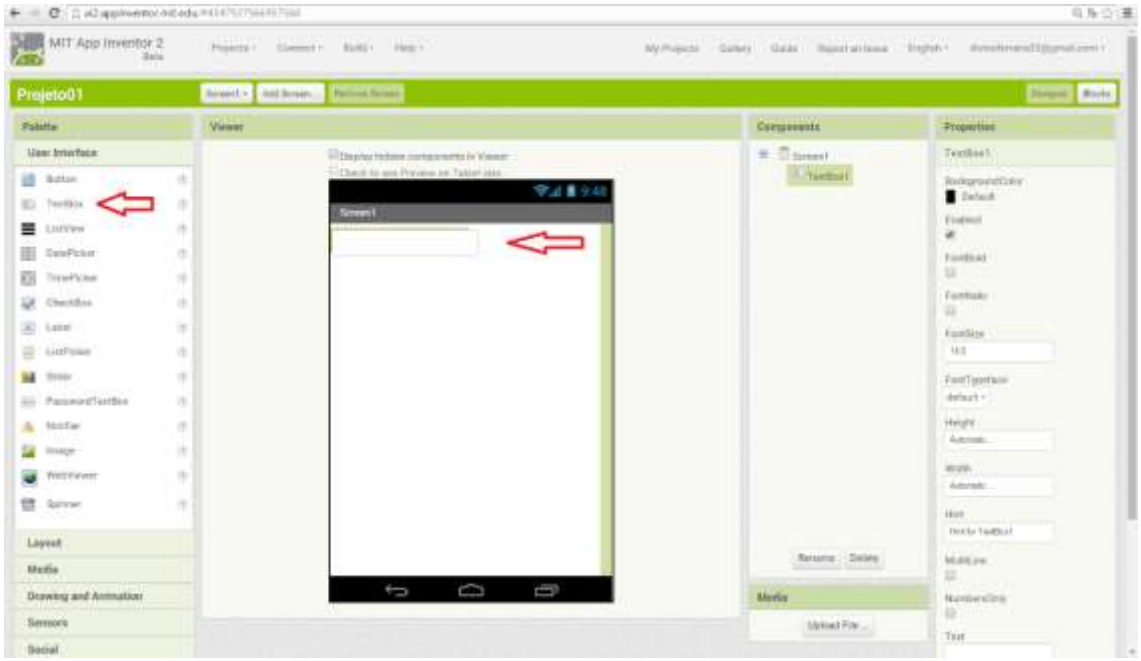
Figura 121 - Layout do projeto

Fonte: http://appinventor.mit.edu

É preciso criar um local onde o texto sera mostrado. Para isso deve-se inserir uma caixa de texto

• 3º - Clique e arraste para a área do dispositivo móvel o componente TextBox (figura 122).

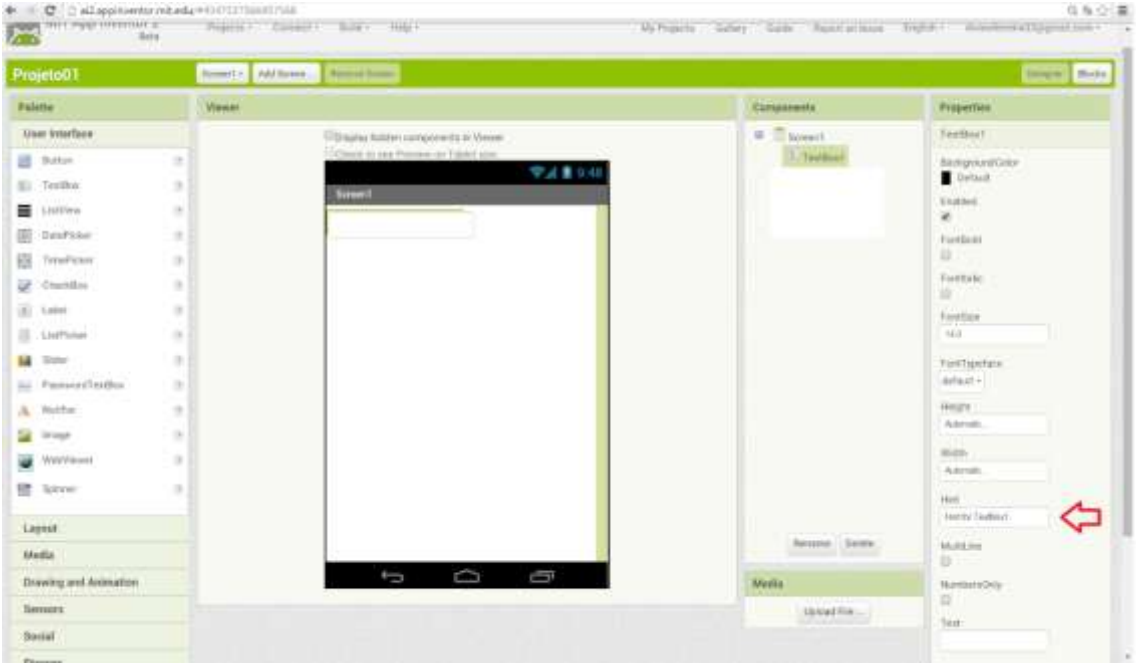
Figura 122 - Inserindo um componente “caixa de texto”



Fonte: http://appinventor.mit.edu

4º - Apague o texto Hint for TextBox1 para que não seja mostrado na caixa de texto quando utilizar a aplicação (figura 123).

Figura 123 - Apagando o texto de rótulo da caixa de texto

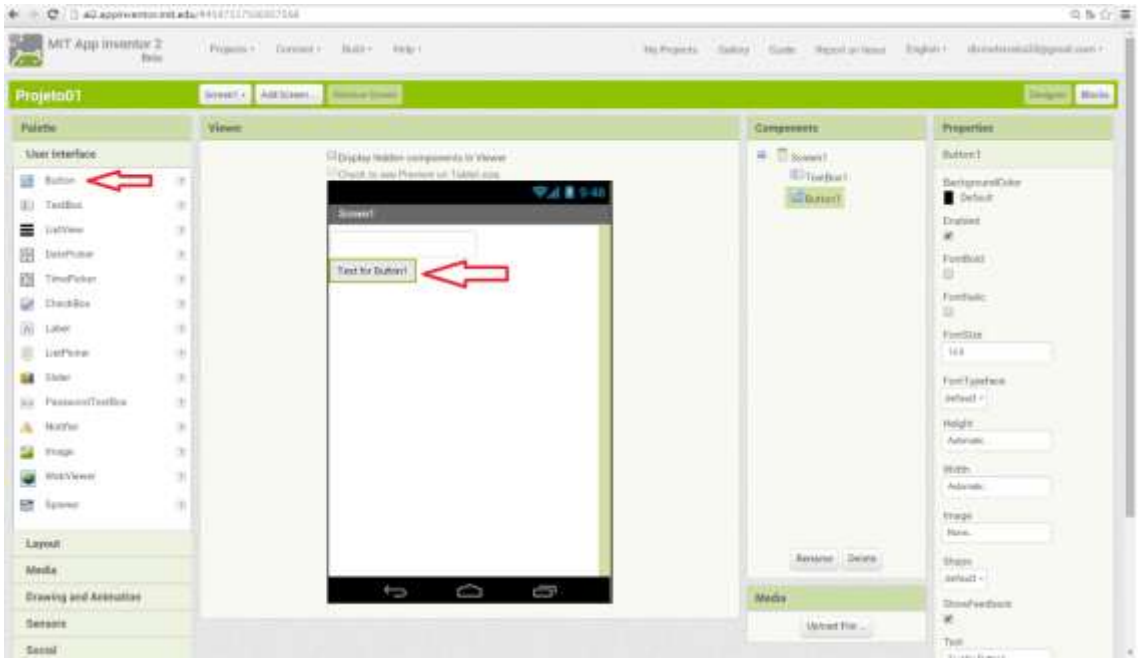


Fonte: http://appinventor.mit.edu

É preciso criar um componente para fazer a escrita na caixa de texto com o texto deseado. Para isso deve-se inserir um botão

- 5º - Clique e arraste para a área do dispositivo móvel o componente Button (figura 124).

Figura 124 - Inserindo um componente botão

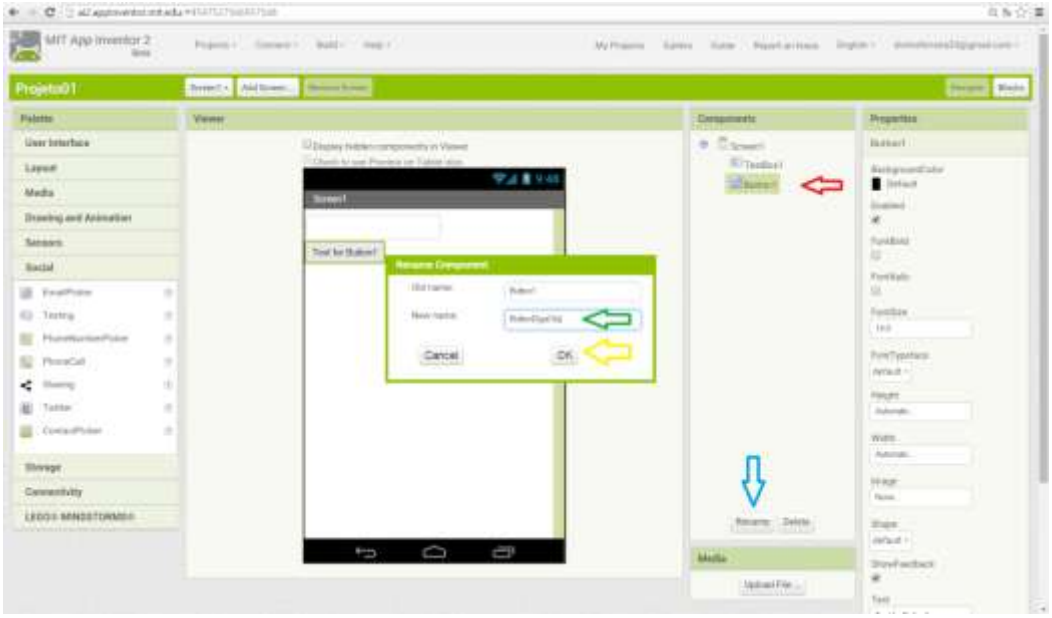


Fonte: http://appinventor.mit.edu

Obs.: É importante renomear os componentes com nomes sugestivos afim de tornar o projeto mais organizado e facilitar na hora de programar. Neste caso o botão que acaba de ser inserido terá a função de escrever o texto “Diga Olá” então devemos colocar um nome que lembra a ação deste componente.

- 6º - Clique ao lado esquerdo do projeto no componente Button1 (seta vermelha fig. 125). Clique no botão [Rename] (seta azul figura 125). Clique no campo de texto “New Name” (seta vermelha figura 125) e digite o novo nome para o componente Button1. Em nosso exemplo utilizamos o nome “ButtonDigaOla”. O App Inventor não aceita nomes de componentes com espaço em branco por isso colocamos tudo junto sem espaço. Caso o programador se esqueça desse detalhe o próprio ambiente de desenvolvimento do App Inventor se encarregará de preencher os espaços em branco com “underline” (traço baixo “_”).

Figura 125 - Renomeando o componente “botão”



Fonte: http://appinventor.mit.edu

É preciso fazer com que apareça escrito no botão o nome da tarefa que será executada por ele. Neste caso como ele será usado para escrever o texto “Diga Olá”, nada mais sugestivo do que escrevermos nele o texto “Diga Olá.

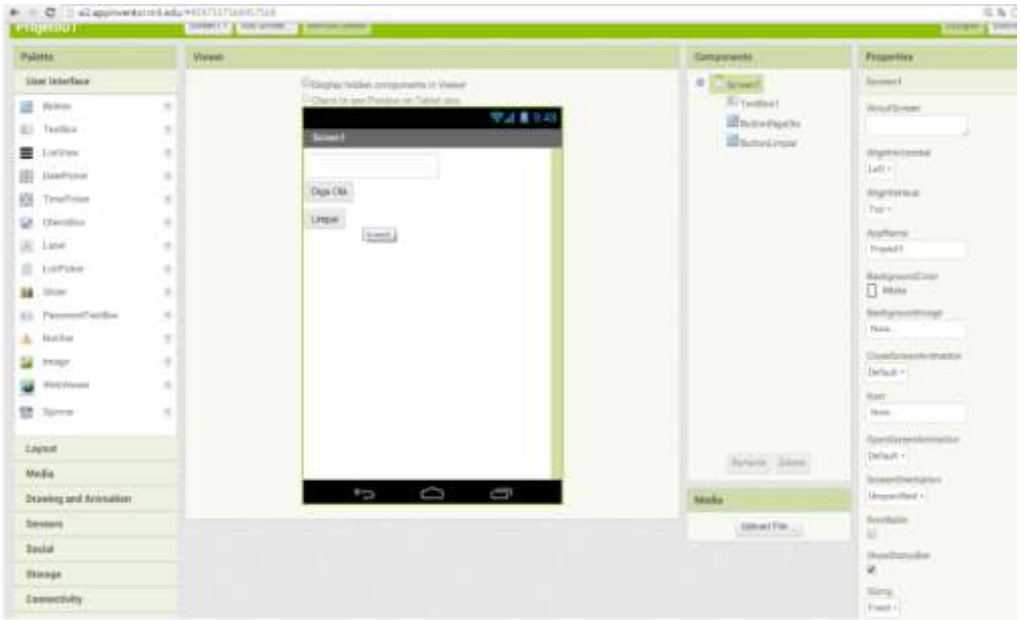
- 7º - Clique ao lado esquerdo do projeto no componente ButtonDigaOla (seta vermelha figura 126). Clique na caixa de texto “Text” (seta azul figura 126). Digite o texto que aparecerá no botão.

Figura 126 - Escrevendo o texto do botão

Fonte: http://appinventor.mit.edu

Repita os procedimentos 5º, 6º e 7º para a criação do botão “Limpar” conforme figura 127.

Figura 127 - Inserção do botão Limpar

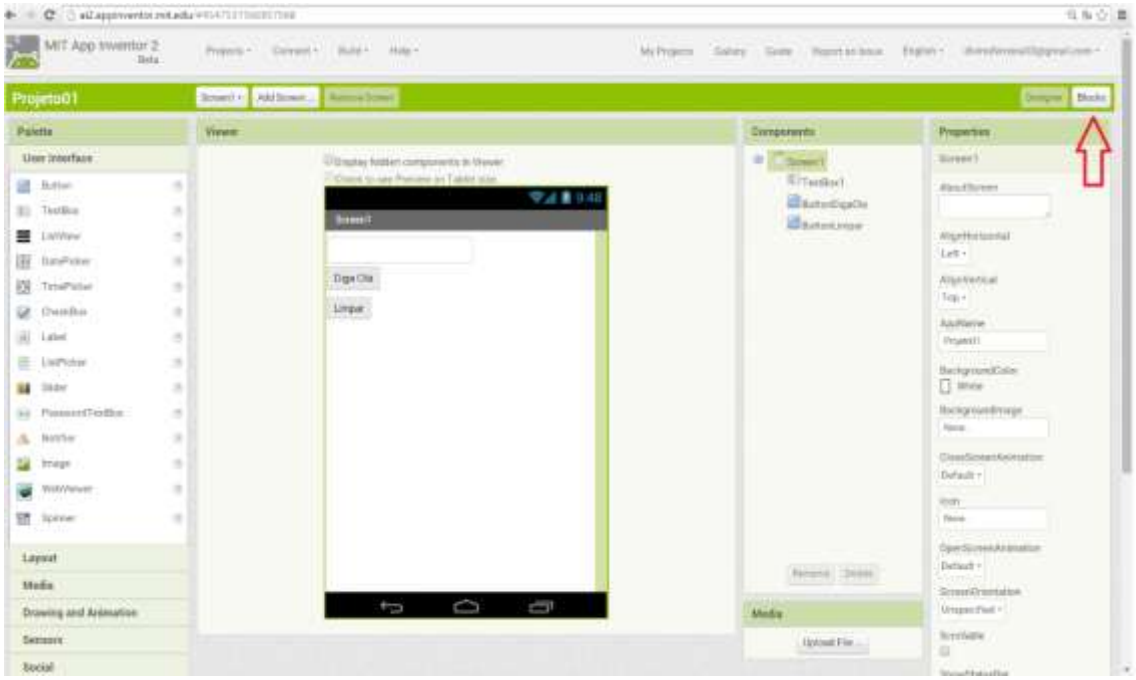


Fonte: http://appinventor.mit.edu

Com a parte do Layout concluída pode-se iniciar a programação.

- 8º - Clique no botão Blocks no canto superior direito conforme figura 128

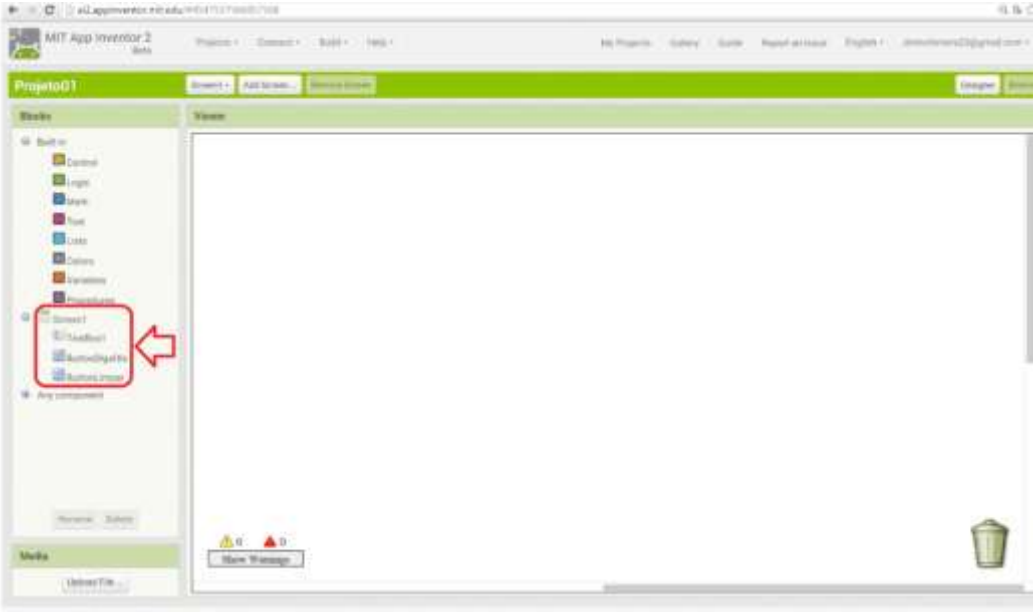
Figura 128 - Começando a programar



Fonte: http://appinventor.mit.edu

Com isso será mostrado o ambiente de programação em blocos. Note do lado esquerdo os componentes anteriormente adicionados (figura 129). Veja que com os nomes sugestivos fica mais fácil identificar os componentes e assim atribuir funções aos mesmos.

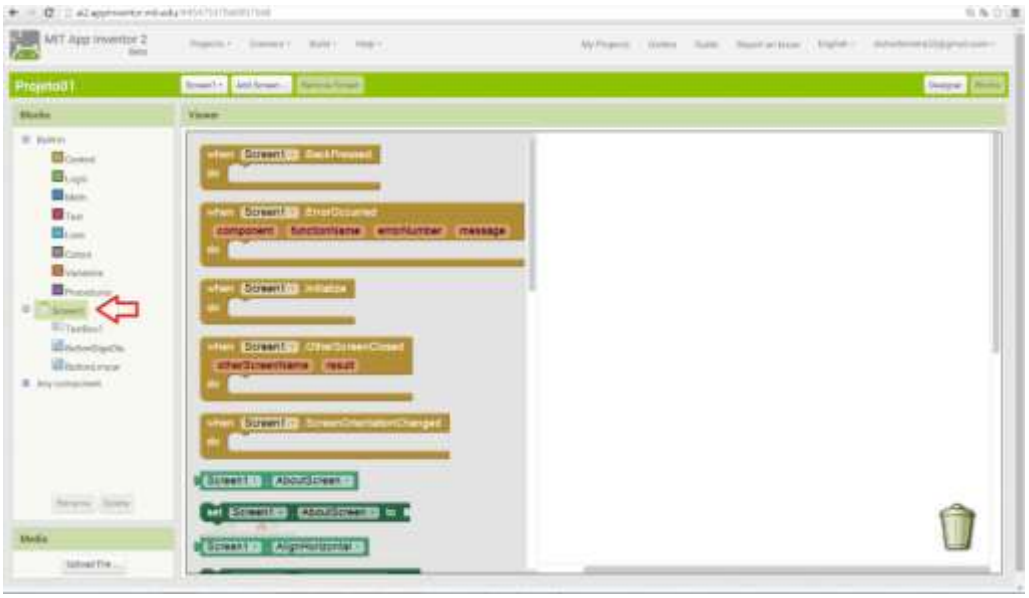
Figura 129 - Ambiente de programação em blocos



Fonte: http://appinventor.mit.edu

O procedimento para programar é o seguinte: Clicar no componente e selecionar o comando desejado na lista que aparece ao lado (figura 130).

Figura 130 - Lista de comandos por componente

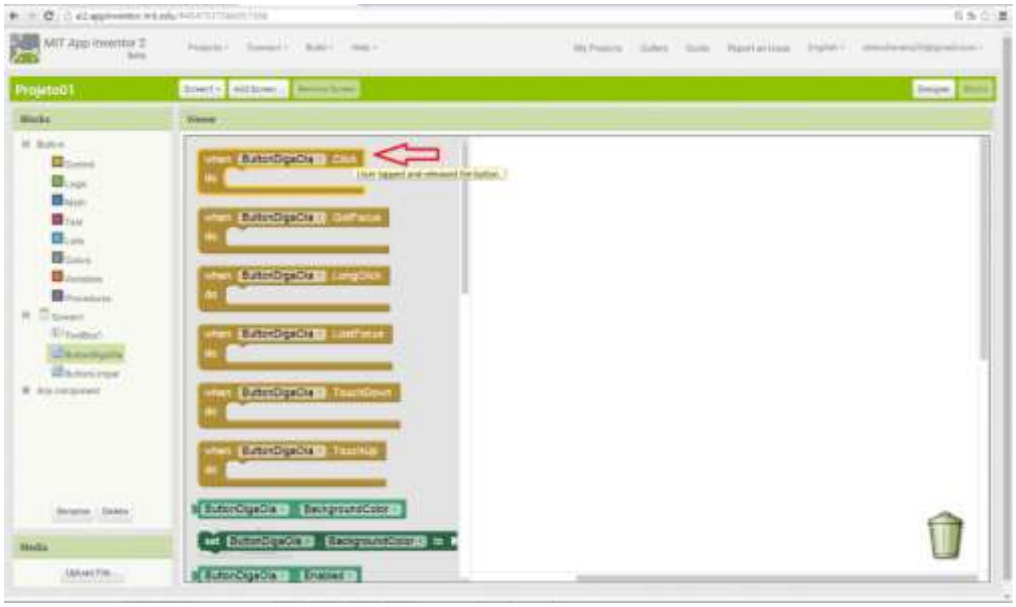


Fonte: http://appinventor.mit.edu

Basicamente utiliza-se comandos para os dois componentes botões anteriormente inseridos.

- 9º - Clique no componente botão “Diga Olá” e em seguida no bloco de comando “When ButtonDigaOla.Click do” (figura 131).

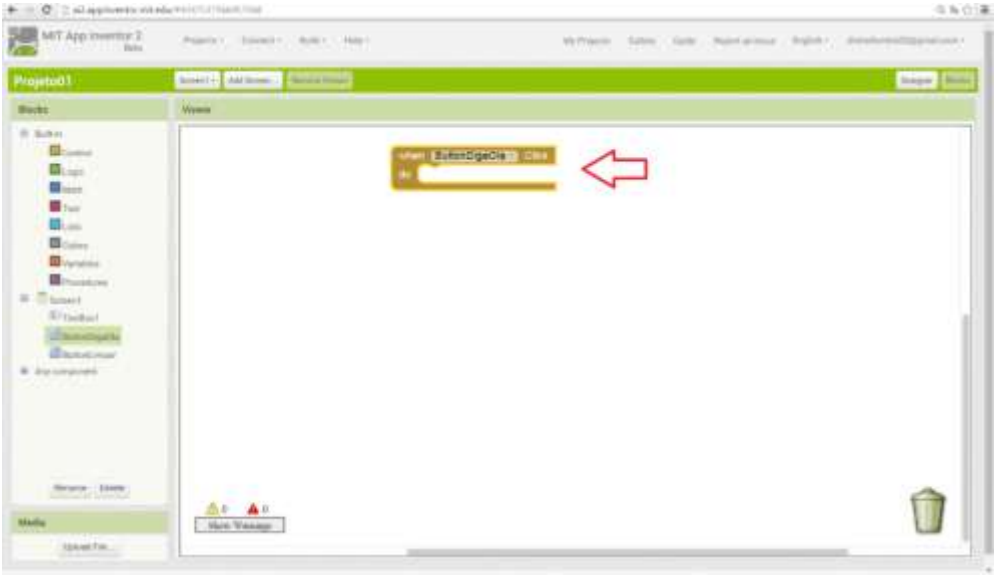
Figura 131 - Selecionando o comando para o componente



Fonte: http://appinventor.mit.edu

10º - Depois de clicado arraste e solte para a área em branco (figura 132).

Figura 132 - Inserindo o bloco

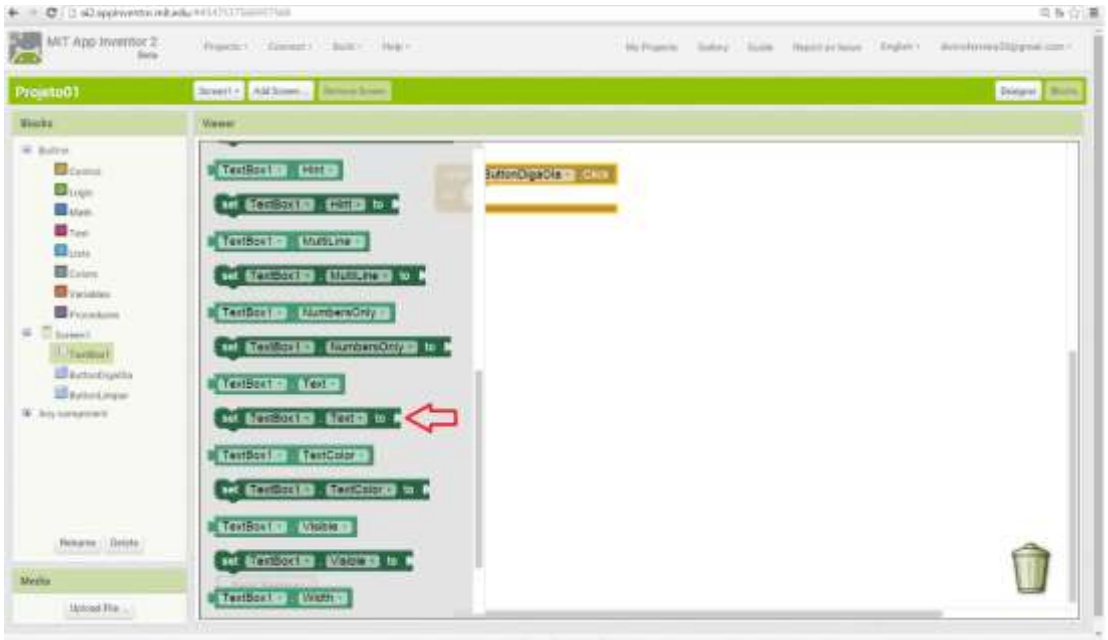


Fonte: http://appinventor.mit.edu

Esse comando recém-selecionado tem a função executar uma ação quando o componente botão “ButtonDigaOlá” for clicado. Neste caso teremos que fazer com que o texto “Diga Olá” aparece no componente “TextBox1” que é a caixa de texto onde aparecerá escrito o que desejarmos.

- 11º - Clique no componente “TextBox1” e em seguinda no bloco de comando “ Set TextBox1.text to” (figura 133).

Figura 133 - Selecionando o comando para o componente



Fonte: http://appinventor.mit.edu

12º - Depois de clicado arraste e solte para a área em branco (figura 134).

Figura 134 - Construindo a programação

Fonte: http://appinventor.mit.edu

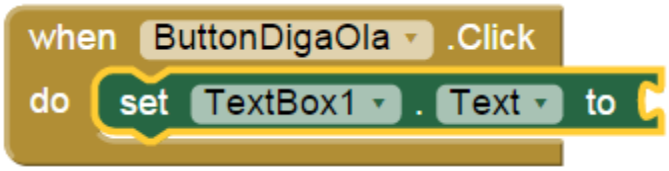
Perceba que os formatos dos blocos são desenhados de forma a encaixar uns nos outros (figura 135).

Figura 135 - Encaixando os blocos

Fonte: http://appinventor.mit.edu

- 13º - Clique e arraste o bloco “Set TextBox1.Text to” para dentro do bloco “When ButtonDigaOla.Click do” (figura 136).

Figura 136 - Blocos devidamente encaixados



Fonte: http://appinventor.mit.edu

É necessário escrever o texto que será escrito na caixa de texto.

- 14º - Clique no componente “Text” (seta vermelha figura 137) e em seguida no bloco de texto (seta azul figura 137)

Figura 137 - Selecionando bloco de lógica

Fonte: http://appinventor.mit.edu

15º - Arraste e encaixe no componente “set TextBox1.Text to” (figura 138).

Figura 138 - Bloco de logica encaixado

Fonte: http://appinventor.mit.edu

16º - Digite o texto “Olá” no bloco de campo de texto recém inserido (figura 139).

Figura 139 – Inserindo o texto

Fonte: http://appinventor.mit.edu

É preciso criar a lógica para o botão “Limpar”.

Repita os procedimentos 9º ao 16º deixando vazio o texto do campo de texto (figura 140). Dessa forma ao se clicar no botão “Limpar” será apagado o texto da caixa de texto ou melhor dizendo, o texto será preenchido com vazio.

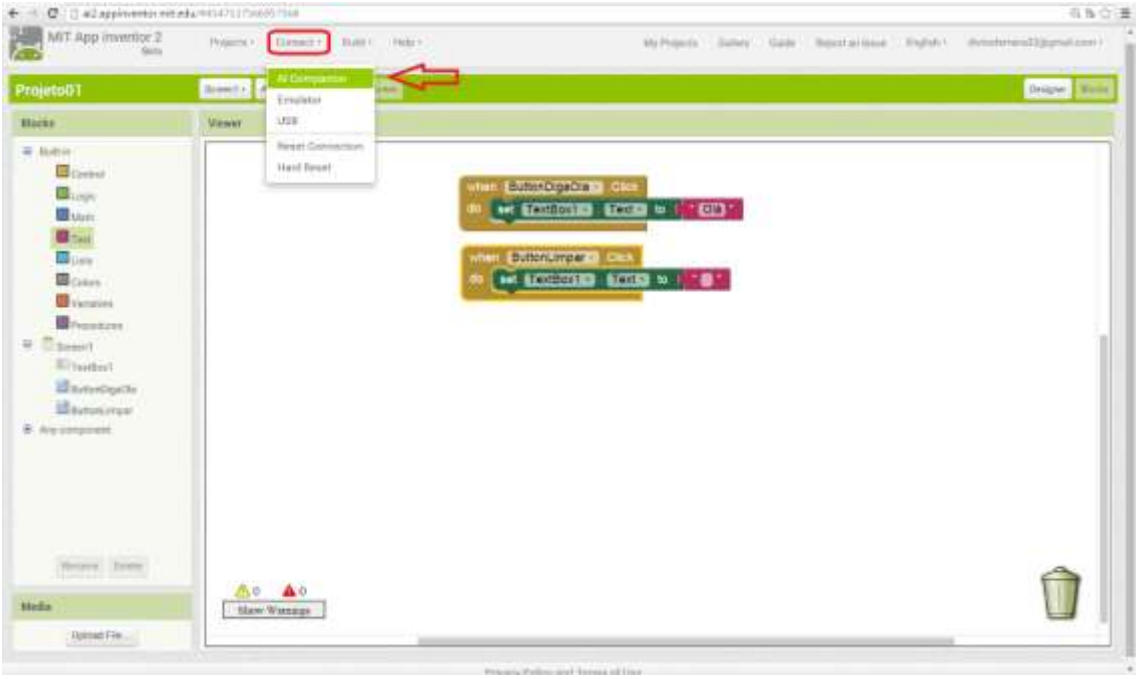
Figura 140 - Finalizando a lógica

Fonte: http://appinventor.mit.edu

Com o layout e a programação prontos é hora de testar a aplicação.

- 17º - Uma vez que o pc/notebook e o dispositivo móvel esteja na mesma rede wifi, clique no menu conect e na opção al companion (figura 141)

Figura 141 - Carregando a aplicação temporária para o dispositivo móvel



Fonte: http://appinventor.mit.edu

18º - Será mostrada uma janela com um Código e um QR Code. Pode-se utiliizar qualquer um dos dois (figura 142).

Obs.: Código QR (sigla do Inglês Quick Response) é um código de barras bidimensional que pode ser facilmente esquadrinhado usando a maioria dos telefones celulares equipados com câmera**.**

Figura 142 - Code e QR Code

Fonte: http://appinventor.mit.edu

- 19º - Abra o software **MIT AI2 Companion** no dispositivo móvel. Será mostrado a tela com opções para o carregamento do aplicativo (figura 143).

Figura 143 - Inserção do Code ou captura do QR Code

Fonte: http://appinventor.mit.edu

- 20º - Preencha o código da seguinte forma:
- Digite o código no campo Six Digit Code e depois em connect with code

Ou

- Clique em scan QR code e aproxime a camera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Se tudo correr bem será mostrado na tela do dispositivo móvel a seguinte tela:

Figura 144 - Aplicação na tela do Dispositivo Móvel

Fonte: Acervo Pessoal

Testando a aplicação

Clique no botão [Diga Olá] para escrever o texto “Olá” (figura 145) e clique no botão [Limpar] para apagar o texto (figura 146).

Figura 145 - Escrevendo Ola

Fonte: Acervo Pessoal

Figura 146 - Apagando o texto

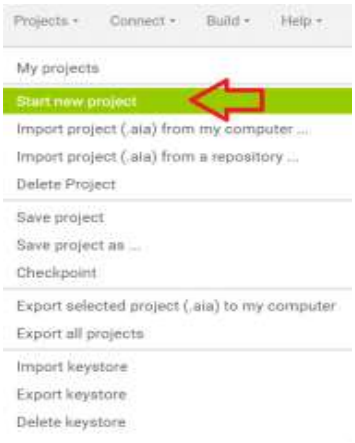
Fonte: Acervo Pessoal

6.5.2 Segundo Projeto

Nesse projeto iremos criar um aplicativo emitirá um som referente a uma figura na tela. Ao ser clicado em cima da figura desejada será ouvido o som proveniente do mesmo, se for um gato ouvir-se-á o miado, se for um cachorro ouvir-se-á um latido.

1º - Inicie um novo projeto com o nome de “Projeto2” clicando no menu Projects / Star New Projetct (figura 147).

Figura 147 - Começando um outro novo projeto



Fonte: http://appinventor.mit.edu

2º - Insira dois botões com os seguintes nomes sugestivos: ButtonCachorro e ButtonGato (figura 148).

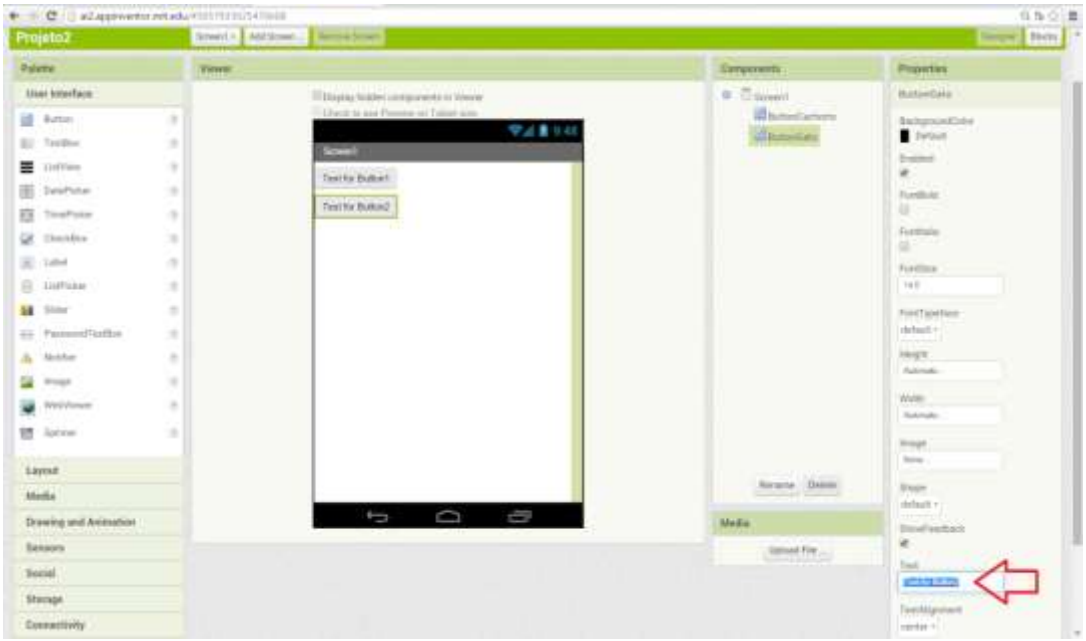
Figura 148 - Começando um outro novo projeto

Fonte: http://appinventor.mit.edu

Como serão colocadas figuras nos botões não há a necessidade de os mesmos terem rótulos então neste caso deve-se apagar os nomes dos mesmos.

3º - Clique nos componentes botões recém-inseridos e depois Text do lado esquerdo (figura 149). Apague o que estiver escrito, geralmente “Text for Button1, Text for Button2 e assim por diante conforme se tem mais botões.

Figura 149 - Apagando os rótulos dos botões

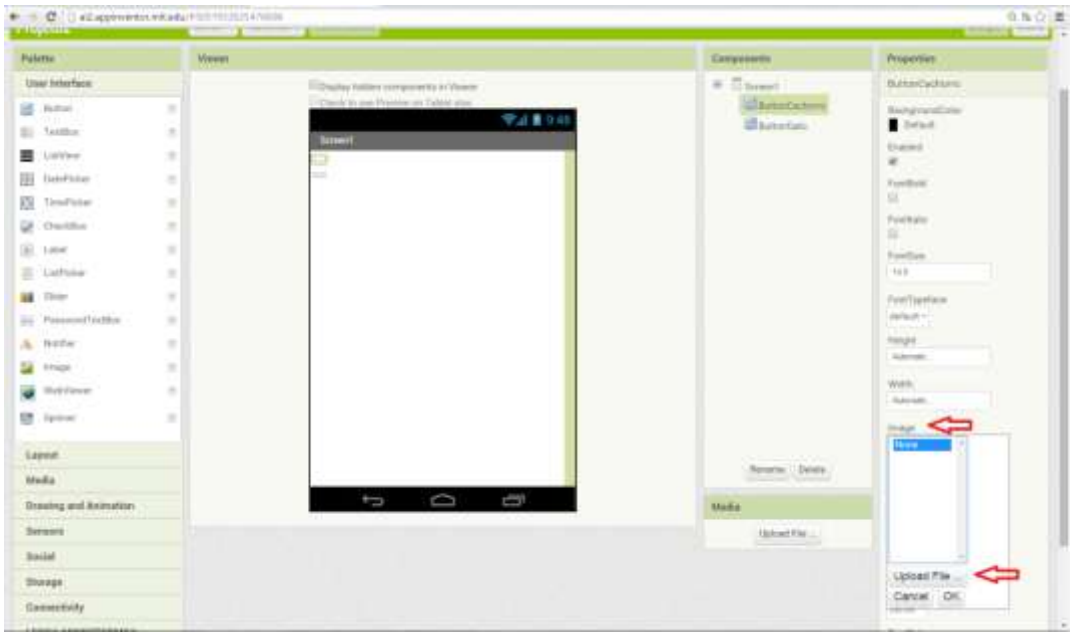


Fonte: http://appinventor.mit.edu

Deve-se inserir as figuras nos botões para que os mesmos tenha o aspecto desejado.

4º - Clique no componente “ButtonCachorro” e em seguida no item Image (figura 150).

Figura 150 - Fazendo upload do arquivo de imagem

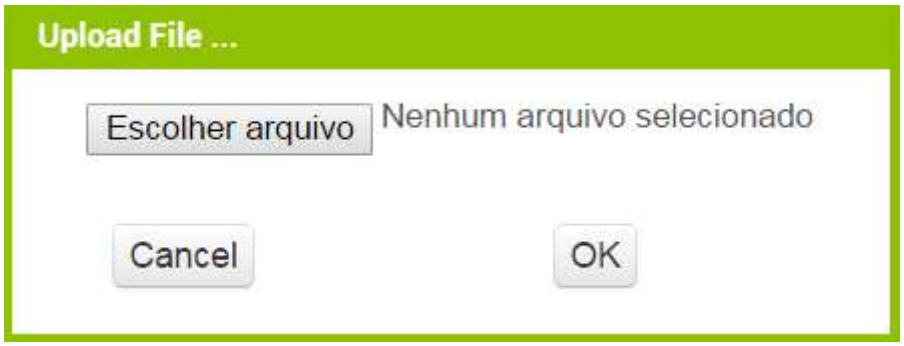


Fonte: http://appinventor.mit.edu

Será mostrado uma tela para escolha do arquivo (figura 151).

5º - Clique o botão [Escolher arquivo] (figura 151). Selecione o arquivo “cachorro.jpg” disponível no CD que acompanha este material. Clique no botão [OK] depois no botão [OK] novamente.

Figura 151 - Escolher o arquivo de imagem



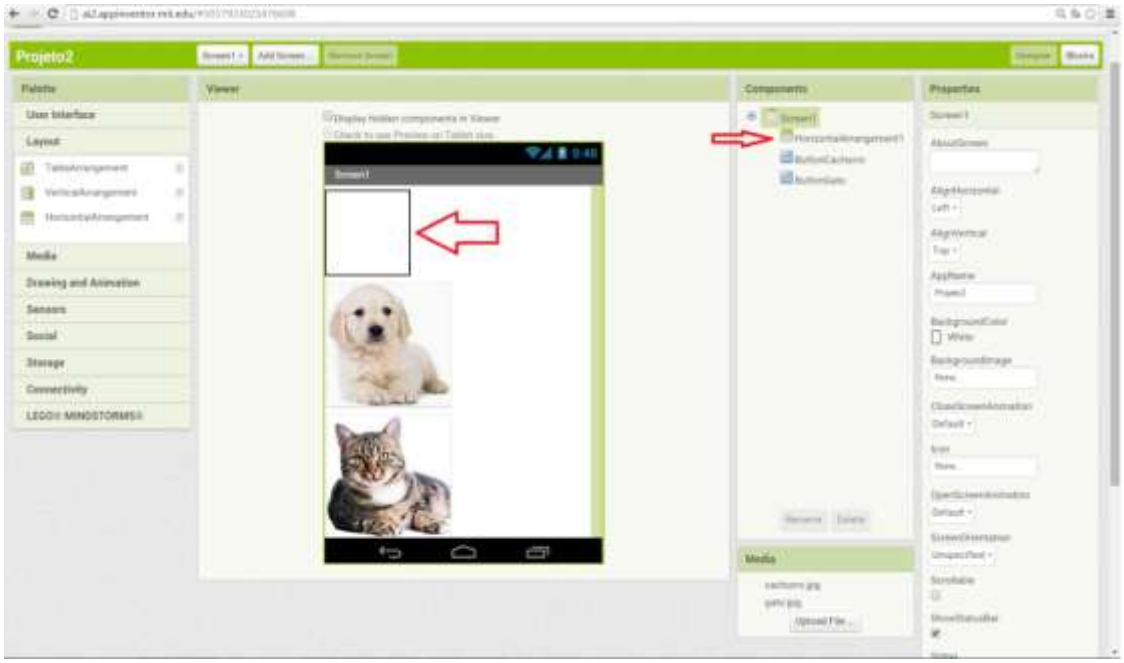
Fonte: http://appinventor.mit.edu

Repita os procedimentos 4º e 5º para a escolha do arquivo “gato.jpg”.

É preciso organizar os botões lado a lado.

6º - Clique no menu do lado esquerdo no item “Layout” e em seguida no componente “Horizontal Arrangement”. Clique e arrast para a tela do projeto (figura 152).

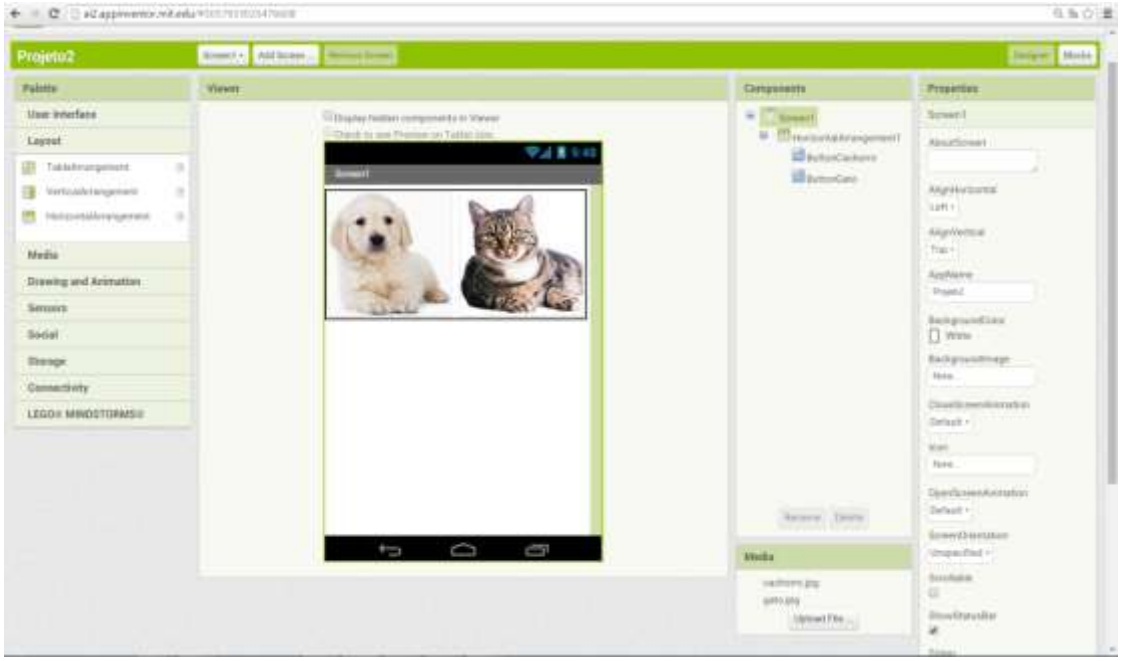
Figura 152 - Inserindo o componente para organização horizontal



Fonte: http://appinventor.mit.edu

7º - Clique nos botões, arraste e solte dentro do componente Horizontal Arrangement na tela do projeto (figura 153).

Figura 153 - Botões organizados horizontalmente

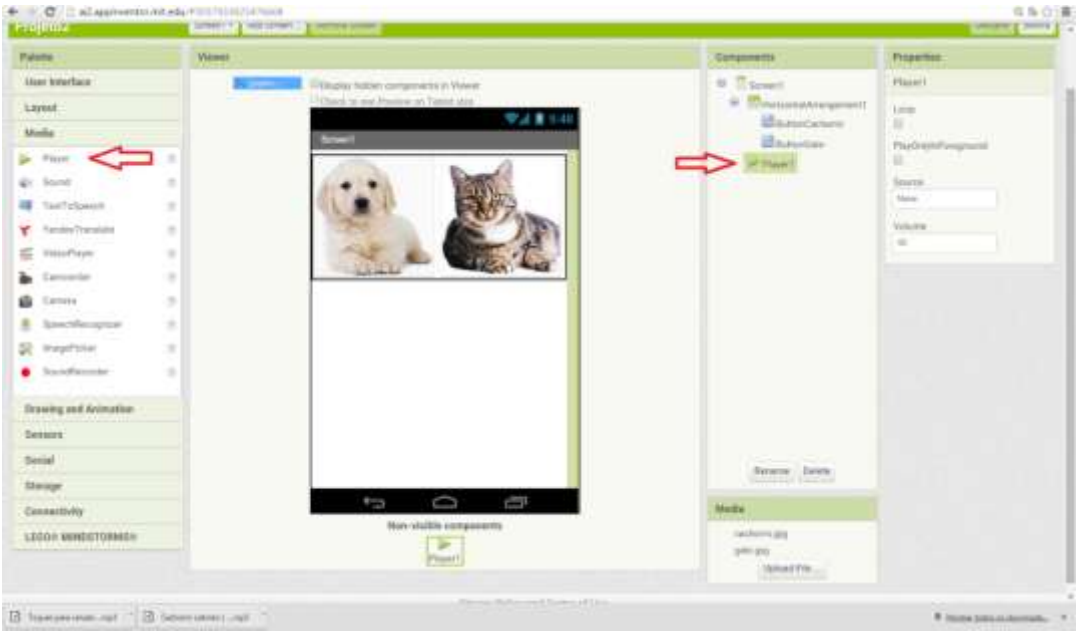


Fonte: http://appinventor.mit.edu

É preciso um componente para associar ao áudio que será produzido pelos botões

8º - Clique no menu do lado esquerdo no item “Media” e em seguida no componente “Player”. Clique e arraste para a tela do projeto (figura 154). Note que esse componente não ficará dentro do layout do projeto e sim na parte inferior. Isso porque ele não é um objeto visível ao utilizador e sim um vínculo para arquivo.

Figura 154 - Inserindo um componente player



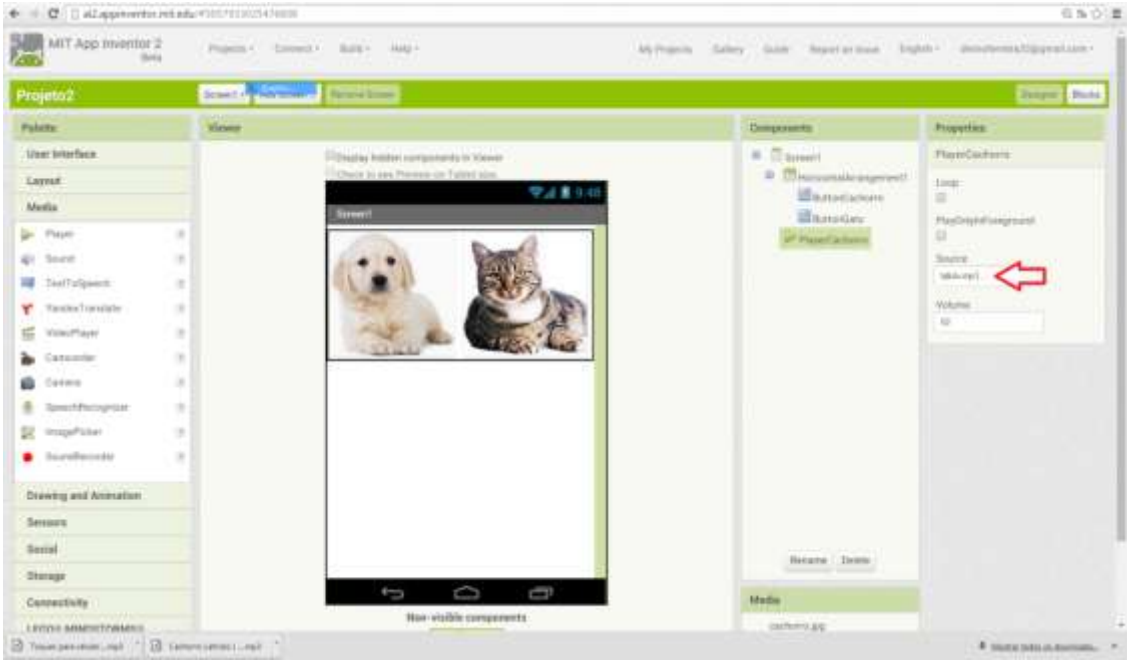
Fonte: http://appinventor.mit.edu

Como iremos utilizar dois sons (um para o cachorro e outro para o gato) sugere-se nomeá-los de forma sugestiva.

9º - Clique no componente Player1 e o renomeie para PlayerCachorro.

10º - Clique no componente PlayerCachorro e depois no item “Source” (figura 155). Siga os procedimentos descritos no item

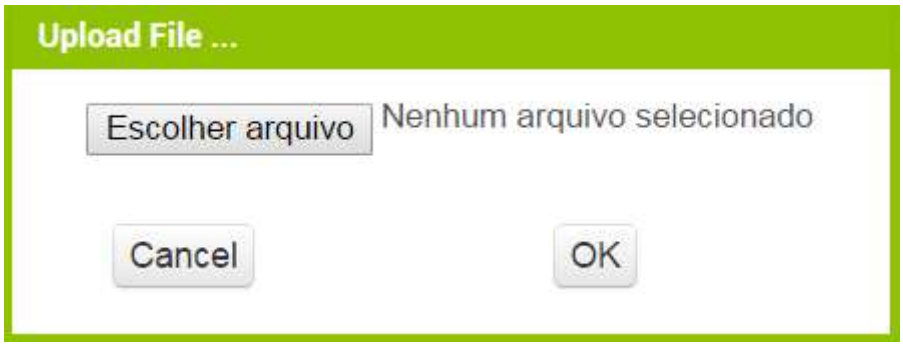
Figura 155 - Buscando um arquivo de áudio



Fonte: http://appinventor.mit.edu

11º - Clique o botão [Escolher arquivo] (figura 156). Selecione o arquivo "latido.mp3" disponível no CD que acompanha este material no laboratório ou providencie um arquivo em mp3 com este nome . Clique no botão [OK] depois no botão [OK] novamente.

Figura 156 - Escolher o arquivo de imagem



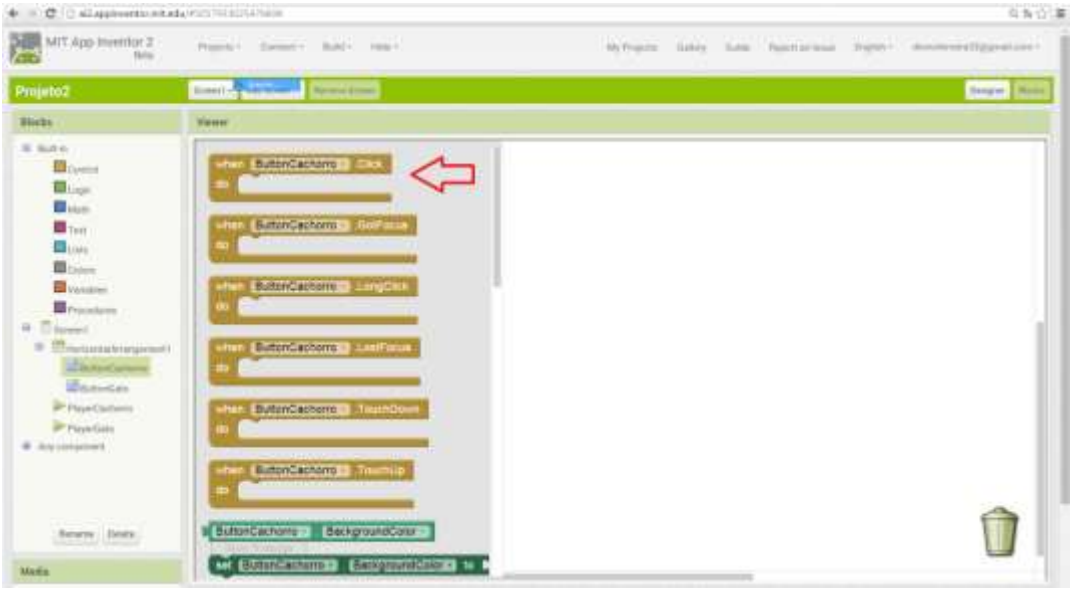
Fonte: http://appinventor.mit.edu

Repita os procedimentos 8º, 9º, 10º e 11º para a escolha do arquivo "miado.mp3".

Com a parte do Layout concluída deve-se iniciar a programação.

12º - Clique no componente botão “ButtonCachorro” e em seguida no bloco de comando “When ButtonCachorro.Click do” (figura 157).

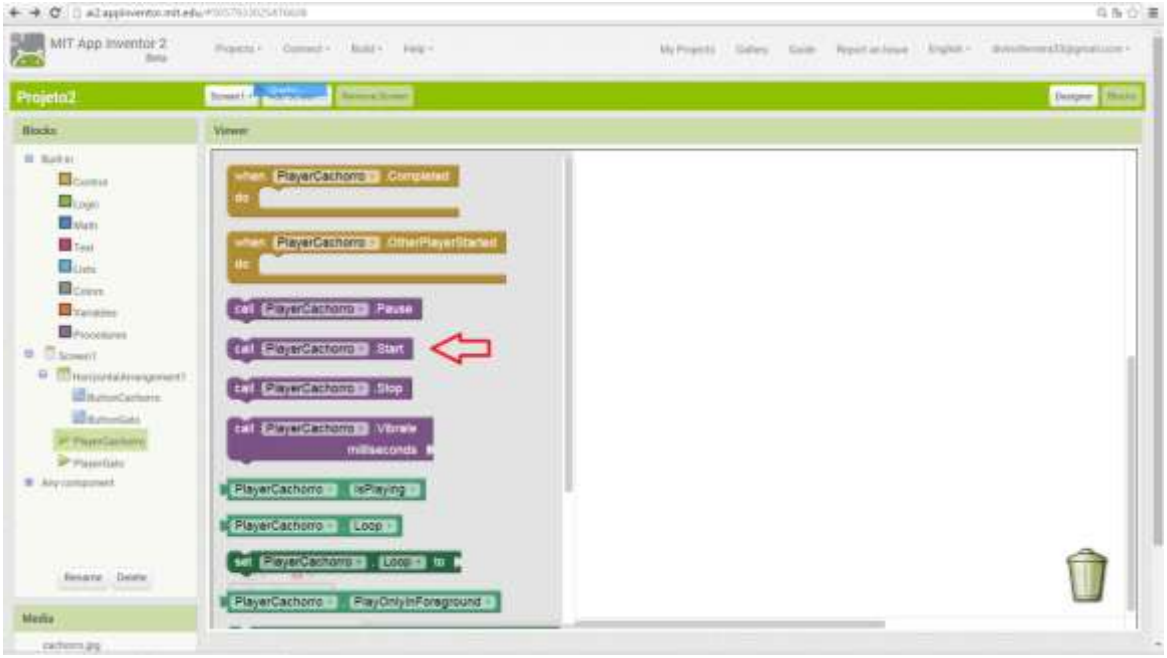
Figura 157 - Inserindo o bloco de comandos do componente ButtonCachorro



Fonte: http://appinventor.mit.edu

13º - Clique no componente botão “PlayerCachorro” e em seguida no bloco de comando “CallPlayerCachorro.start” (figura 158).

Figura 158 - Inserindo o bloco de comandos do componente PlayerCachorro



Fonte: http://appinventor.mit.edu

14º – Encaixe o bloco PlayerCachorro no bloco "When ButtonCachorro.Click do" (figura 159)

Figura 159 - Encaixando os blocos

Fonte: http://appinventor.mit.edu

Repetir os procedimentos 12º, 13º e 14º para o componente ButtonGato e Player Gato (figura 160)

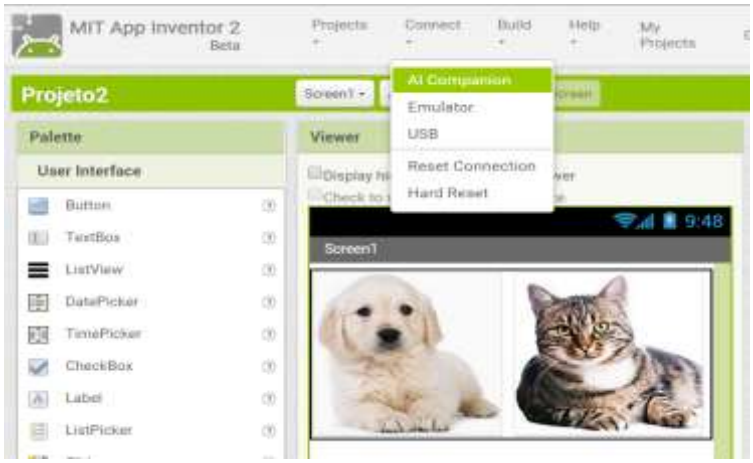
Figura 160 - Blocos dos botões e players

Fonte: http://appinventor.mit.edu

Com o layout e a programação prontos deve-se testar a aplicação.

• 15º - Uma vez que o pc/notebook e o dispositivo móvel esteja na mesma rede wifi, clique no menu conect e na opção al companion (figura 161)

Figura 161 - Carregando a aplicação temporária para o dispositivo móvel



Fonte: http://appinventor.mit.edu

16º - Será mostrada uma janela com um Código e um QR Code. Pode-se utiliizar qualquer um dos dois (figura 162).

Figura 162 - Code e QR Code

Fonte: http://appinventor.mit.edu

17º - Abra o software **MIT AI2 Companion** no dispositivo móvel. Será mostrado a tela com opções para o carregamento do aplicativo (figura 163).

Figura 163 - Inserção do Code ou captura do QR Code

Fonte: http://appinventor.mit.edu

- 18º - Preencha o código da seguinte forma:
- Digite o código no campo Six Digit Code e depois em connect with code

Ou

- Clique em scan QR code e aproxime a camera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Se tudo correr bem será mostrado na tela do dispositivo móvel a seguinte tela (figura 164).

Figura 164 - Aplicação na tela do Dispositivo Móvel

Fonte: Acervo Pessoal

Testando a aplicação

Ao clicar em cima da figura do cachorro será produzido o som do latido; ao clicar em cima da figura do gato será produzido som de miado.

6.5.3 Terceiro Projeto

Nesse terceiro projeto iremos criar um aplicativo que reconhecerá o som produzido pelo utilizador e o reproduzirá em forma de texto. Terá um botão para iniciar a gravação da voz e um campo de texto para mostrar a mensagem anteriormente falada.

1º - Inicie um novo projeto com o nome de "Projeto2" clicando no menu Projects / Star New Projetct (figura 165).

Figura 165 - Começando novo projeto

Fonte: http://appinventor.mit.edu

2º - Insira um componente Button no projeto conforme visto nos capítulos 1 e 2.

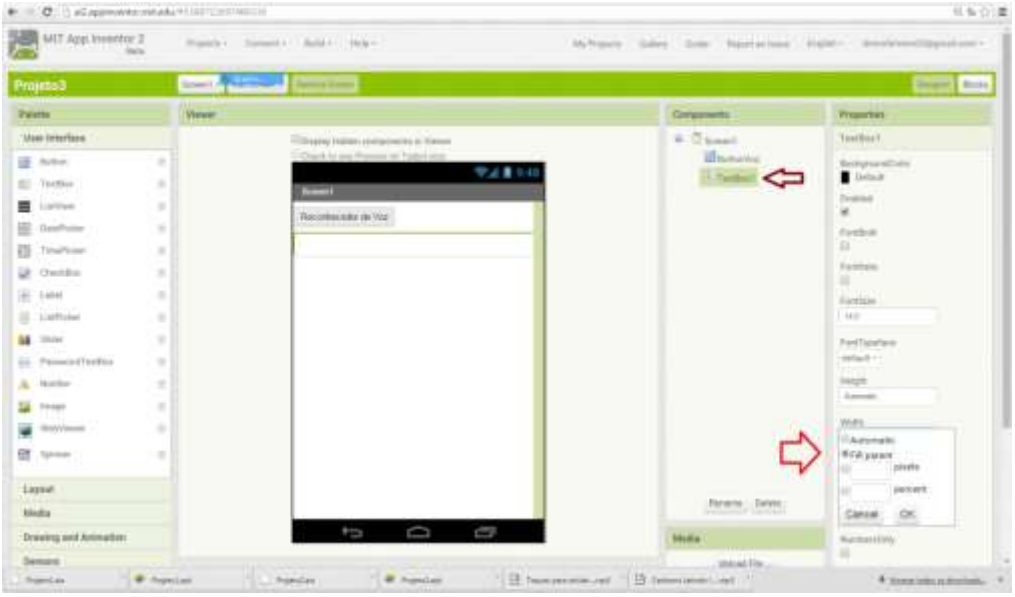
3º - Renomeie o botão com um nome sugestivo. Em nossa experiência utilizamos "ButtonVoz".

4º - Coloque o seguinte texto no componente ButtonVoz: "Reconhecedor de Voz"

5º - Insira um componente TextBox no projeto conforme visto no capitulo 1.

6º - Clique no componente TextBox1 e em seguida no item Width marque a opção "Fill Parent" fig.166. Isso fará com que o componente TextBox1 assuma toda a extensão horizontal da tela.

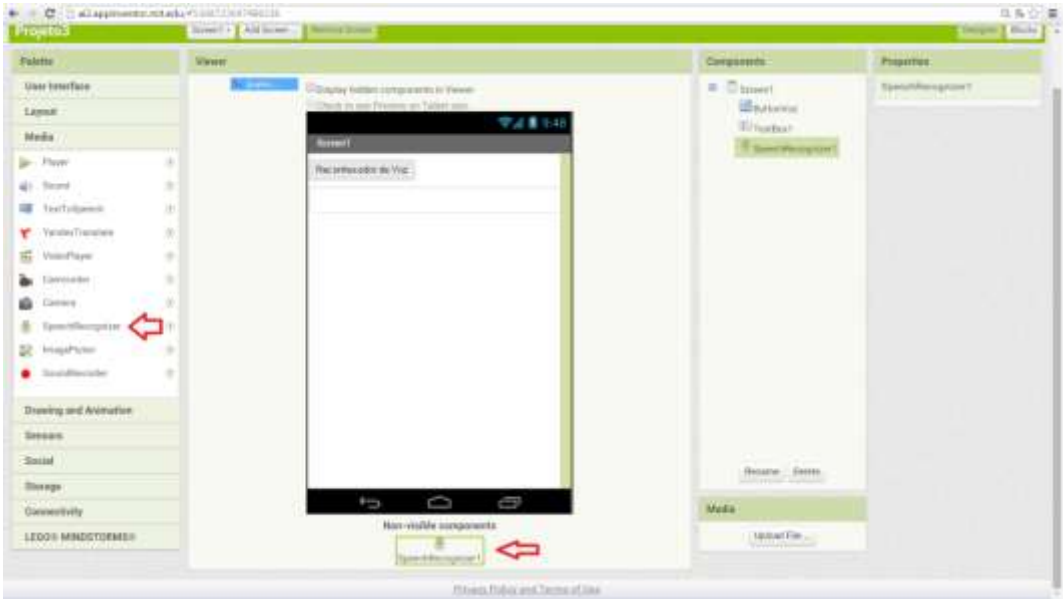
Figura 166 - Expandindo a área do componente



Fonte: http://appinventor.mit.edu

7º - Clique no menu Media e em seguida SpeechRecognizer (figura 167). Clique, arraste e solte o componente na área do projeto. Note que o mesmo não ficará na área do projeto, alojando-se na parte inferior pois o mesmo não é utilizado como um objeto e sim um vínculo para uma função.

Figura 167 - Inserindo o componente SpeechRecognizer



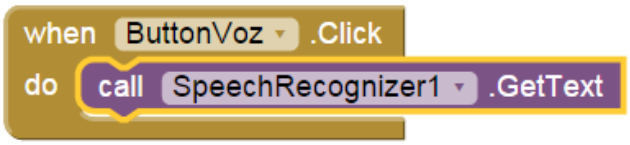
Fonte: http://appinventor.mit.edu

Com o layout criado pode-se dar inicio à programação.

8º - Clique no componente botão “ButtonVoz” e em seguida no bloco de comando “When ButtonVoz.Click do”. Clique, arraste e solte na área do projeto conforme visto nos capítulos 1e 2.

9º - Clique no componente SpeechRecognizer1 e selecione o bloco de comando “call SpeechRecognizer1.getText”. Clique, arraste, solte e encaixe o bloco desse componente no bloco “When ButtonVoz.Click do” figura 168.

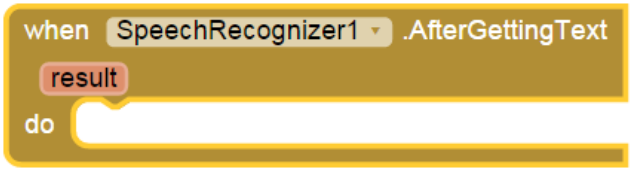
Figura 168 - Encaixando componentes



Fonte: http://appinventor.mit.edu

10º - Clique no componente SpeechRecognizer1 e selecione o bloco de comando “when SpeechRecognizer1.AfterGetingText do”. Clique, arraste, solte na área do projeto (figura 169)

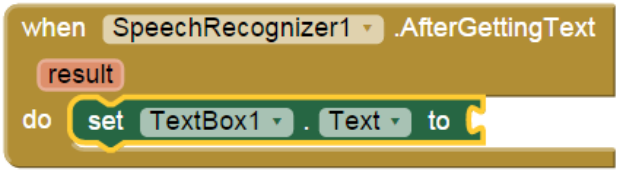
Figura 169 - Inserindo o bloco SpeechRecognizer1.AfterGetingText do



Fonte: http://appinventor.mit.edu

11º - Clique no componente TextBox1 e selecione o bloco de comando “Set TextBox1.text do”. Clique, arraste, solte e encaixe o bloco desse componente no bloco “when SpeechRecognizer1.AfterGetingText do” figura 170.

Figura 170 - Encaixando os blocos

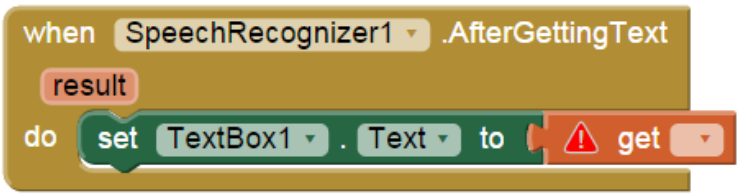


Fonte: http://appinventor.mit.edu

É preciso criar uma variável para retornar o valor ditado em forma de texto.

12º - Clique no componente Variables e selecione o bloco de comando “get”. Clique, arraste, solte e encaixe o bloco desse componente no bloco “Set TextBox1.text do” figura 171.

Figura 171 - Encaixando o bloco “get” no bloco “ set TextBox1.Text do”



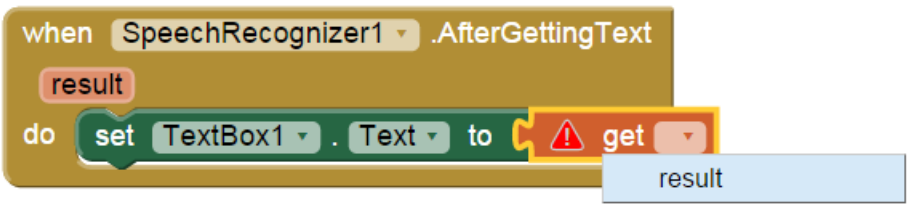
Fonte: http://appinventor.mit.edu

Note que o bloco “get” mostra um triangulo vermelho com ponto de exclamação. Isso significa que existe uma inconsistência de informação, a programação não está sequenciada. Geralmente ocorre por duplicação de blocos, componentes ou mesmo por falta de link entre eles.

Neste caso o que ocorre é que o bloco “when SpeechRecognizer1.AfterGetingText do” precisa de um valor retornado chamado “result”. Precisaremos selecionar essa variável.

13º - Clique na seta do bloco “Get” e selecione “result” (figura 172).

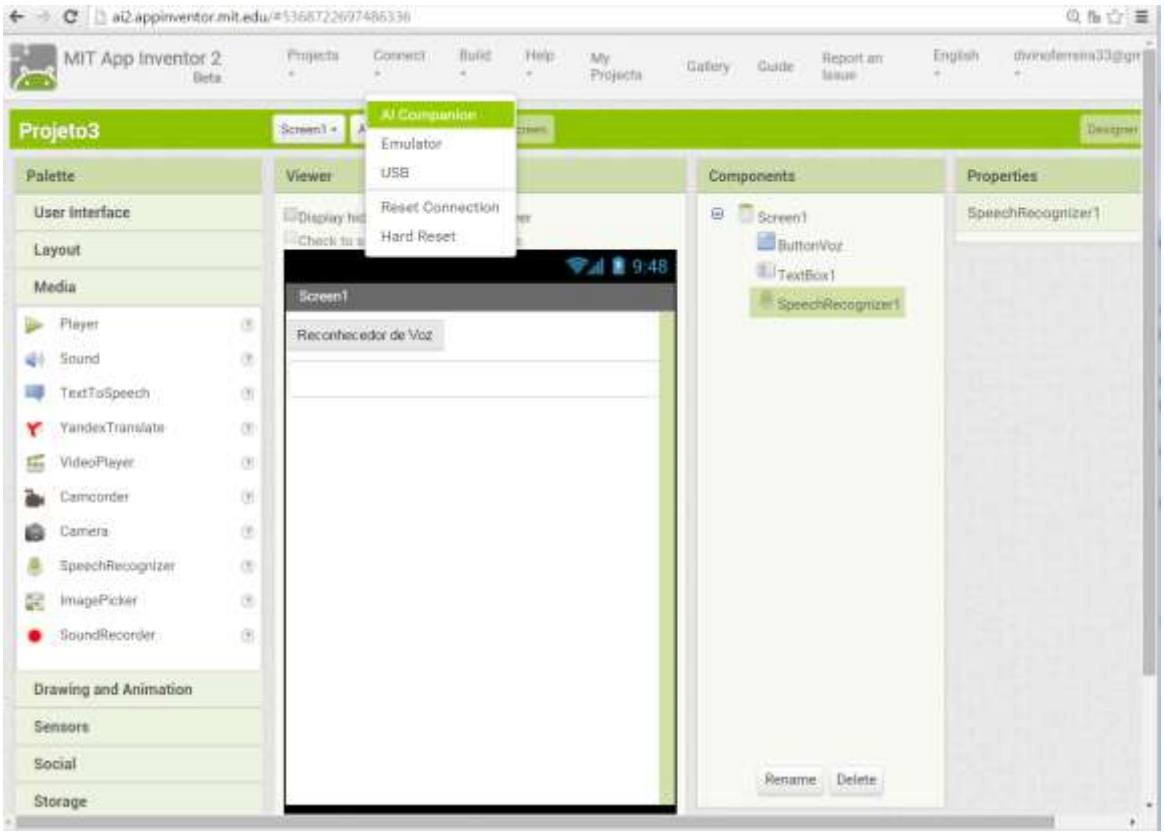
Figura 172 - Selecionando “result” no bloco “get”



Fonte: http://appinventor.mit.edu

- 14º - Uma vez que o pc/notebook e o dispositivo móvel esteja na mesma rede wifi, clique no menu conect e na opção al companion (figura 173)

Figura 173 - Carregando a aplicação temporária para o dispositivo móvel



Fonte: http://appinventor.mit.edu

15º - Será mostrada uma janela com um Código e um QR Code. Pode-se utiliizar qualquer um dos dois (figura 174).

Figura 174 - Code e QR Code

Fonte: http://appinventor.mit.edu

16º - Abra o software **MIT AI2 Companion** no dispositivo móvel. Será mostrado a tela com opções para o carregamento do aplicativo (figura 175).

Figura 175 - Inserção do Code ou captura do QR Code

Fonte: http://appinventor.mit.edu

17º - Preencha o código da seguinte forma:

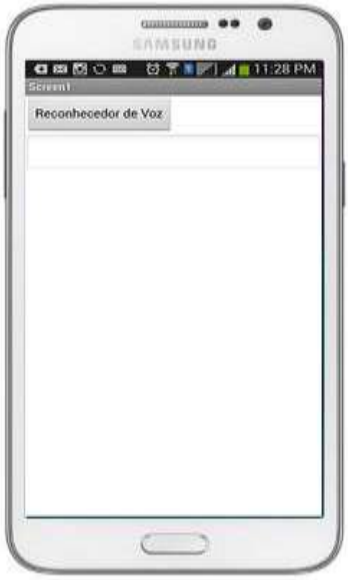
- Digite o código no campo Six Digit Code e depois em connect with code

Ou

- Clique em scan QR code e aproxime a câmera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Se tudo correr bem será mostrado na tela do dispositivo móvel a seguinte tela (figura 176).

Figura 176 - Aplicação na tela do Dispositivo Móvel



Fonte: Acervo Pessoal

Testando a aplicação

18º - Clique no botão Reconhecedor de Voz. Será mostrada uma tela pedindo "fale agora" (figura 5-70). Fale o texto desejado. A aplicação irá reconhecer o que foi ditado e irá escrever na caixa de texto (figura 5-71).

Figura 177 - Pedindo "Fale agora"

Fonte: Acervo Pessoal

Figura 178 - Texto Reconhecido

Fonte: Acervo Pessoal

7 Conceitos importantes necessários para o desenvolvimento prático das experiências

Agora que o aluno compreendeu de forma segmentada, passo a passo e com a metodologia correta aplicada em cada uma das etapas do aprendizado, é hora de partir para a parte prática. Será ensinado como controlar remotamente uma carga através de um dispositivo móvel de forma que o ensinamento obedecerá duas situações distintas: controle sem resposta e com resposta. É importante demostrar as duas formas para que o aluno e futuro profissional saiba controlar de forma efetiva um comando enviado, relacionando o que existe no mercado e o que se espera de um produto produzido por um profissional com nível superior de qualidade.

7.1 Cliente / Servidor

Será explicado agora o conceito de cliente e servidor para que fique clara a forma de comunicação existente entre um comando e uma ação quando executado em um contexto no qual o comando e ação estão separados fisicamente.

O principal conceito que deve ser aprendido é sobre Cliente e Servidor. Sempre que se fala em cliente /servidor vem à mente redes de computadores. Isso não está errado, mas é preciso ir mais fundo para entender o que realmente vem a ser um cliente e um servidor.

A tecnologia cliente/servidor é uma arquitetura na qual o processamento da informação é dividido em módulos ou processos distintos. Um processo é responsável pela manutenção da informação (servidores) e outros responsáveis pela obtenção dos dados (os clientes).

Os processos cliente enviam pedidos para o processo servidor, e este por sua vez processa e envia os resultados dos pedidos.

Dessa forma podemos utilizar uma analogia para entender o principio de cliente e servidor. Imaginemos a ação de ligar um aparelho de TV pelo controle remoto. Ao se apertar o botão no controle o televisor irá ligar ou não. Nesse contexto o controle se comporta como um Servidor e o televisor se comporta com um Cliente.

- Se ao pressionar o botão o televisor não ligar teremos uma resposta áudio/visual da situação, ou seja, o televisor não emitiu som e nem imagem e neste caso saberemos por outros meios que o televisor não ligou. Se o televisor ligar
- Se ao pressionar o botão o televisor ligar teremos uma resposta áudio/visual da situação, ou seja, o televisor emitiu som e imagem e neste caso saberemos por outros meios que o televisor ligou.

Essa situação mostra que a comando não obtém resposta pelo mesmo meio de transmissão. No controle não chega nenhuma informação de que o televisor ligou.

Para tornar o caso mais inteligível suponha-se que estivéssemos de olhos vendados e ouvidos abafados e quiséssemos executar a mesma ação de ligar o televisor. Neste caso não é sabido após se executar o comando se o televisor ligou ou não ligou. Com isso conclui que o Cliente é sempre Cliente e o Servidor é sempre Servidor.

Para resolver esse problema pode-se desenvolver um controle com algum tipo de dispositivo vibratório e a cada comando enviado ao televisor e executado com sucesso uma vibração ocorra no controle e o usuário saberia se a ação foi concluída. Uma vez que isso ocorra fica claro que o Cliente e Servidor invertem seus papéis mediante a necessidade.

7.2 Resposta em tempo real

O controle a distância com resposta em tempo real é o personagem principal deste trabalho. É preciso entender o verdadeiro significado do termo “tempo real” evitando equívocos de interpretação.

No primeiro momento imaginamos como sendo tempo real algo instantâneo. Mas isso não e verdade e pode causar efeitos indesejáveis quando se trata de desenvolvimento de projeto. O tempo real na resposta é o tempo necessário para que a resposta chegue ao seu destino levando-se em consideração o tempo necessário para a execução da tarefa e o meio de transmissão utilizado no envio e recebimento da informação. Podemos exemplificar de duas formas para melhorar o entendimento

- **Primeiro exemplo** - um sistema que simule uma gestação humana com todo o desenvolvimento do bebê. Tal sistema não poderá fornecer a resposta do nascimento do bebê antes dos nove meses previstos.
    - **Segundo exemplo** - Precisamos saber sobre a situação de alguém que mora distante. Podemos escrever uma carta e aguardar a resposta que levará dias ou semanas (e no passado era assim) ou podemos ligar e saber instantaneamente as notícias desejadas.

Fica nítido que a velocidade da resposta depende diretamente do processo a ser monitorado e do meio de transmissão utilizado.

Nas experiências que se seguem será demonstrada a diferença entre acionamento convencional e acionamento com resposta.

8 Experiências Práticas

8.1 Ligar lâmpada através do celular utilizando Bluetooth (sem resposta)

Bluetooth é uma especificação industrial para áreas de redes pessoais sem fio (*Wireless personal area networks* – PANs). O Bluetooth provê uma maneira de conectar e trocar informações entre dispositivos como telefones celulares, notebooks, computadores, impressoras, câmeras digitais e consoles de videogames digitais através de uma frequência de rádio de curto alcance globalmente licenciada e segura. As especificações do Bluetooth foram desenvolvidas e licenciadas pelo "Bluetooth Special Interest Group".

Nesta iremos ligar e desligar 1 Lâmpada utilizando um dispositivo móvel através do bluetooth com a plataforma arduino.

Iniciaremos abrindo o ambiente de desenvolvimento App Inventor no site http://www.appinventor.mit.edu e executando os procedimentos a seguir.

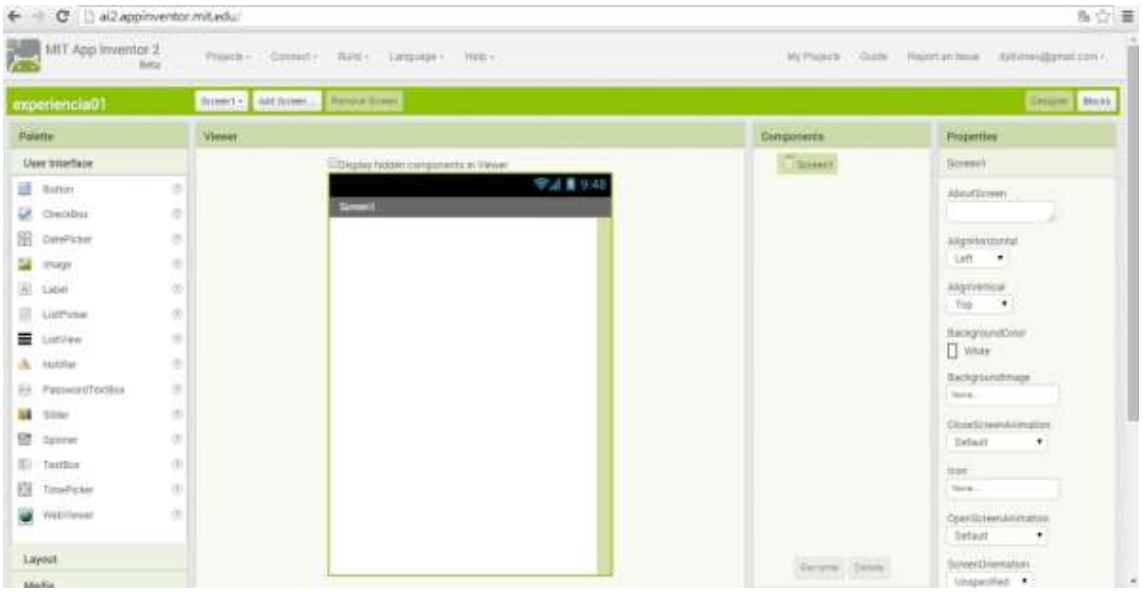
- Clique no botão Start New Project (figura 179) e dê um nome ao projeto. O nome não deve conter espaço em branco.

Figura 179 - Nome do projeto

Fonte: http://appinventor.mit.edu

- Abrirá automaticamente o ambiente de desenvolvimento na guia Designer conforme figura 180.

Figura 180 - Ambiente de desenvolvimento

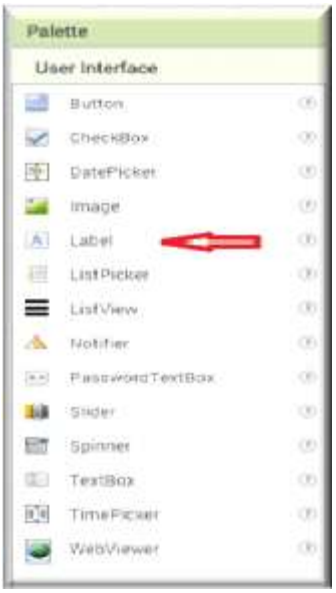


Fonte: http://appinventor.mit.edu

8.1.1 Desenvolvendo o Layout (Designer) da Aplicação

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente Label conforme indicado na figura 181 e 182.

Figura 181 - Palette/User Interface/Label



Fonte: http://appinventor.mit.edu

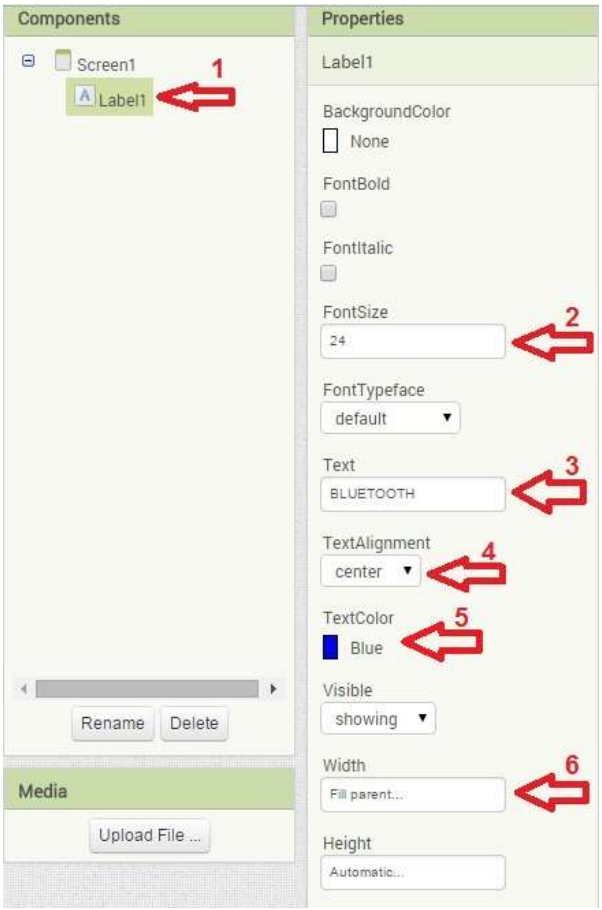
Figura 182 - Componente Label

Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente Label1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 183 a seguir indicados pelas setas (1 – componente de texto, 2 – tamanho da fonte, 3 – texto sugerido para o título, 4 – alinhamento do texto, 5 – cor do texto, 6 – largura.

  Obs.: A opção 6 selecionada como Fill Parent define que o componente irá ocupar toda a largura da aplicação.

Figura 183 - Configurando o componente Label1



Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto o texto “BLUETOOTH” centralizado na tela conforme figura 184 a seguir.

Figura 184 - Label

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **Layout** clique, arraste e solte na janela Sreen1 um componente **HorizontalArrangement** conforme indicado na figura 185. Esse componente possibilita o posicionamento de objetos lado a lado horizontalmente e será utilizado para acomodar os botões **Conectar** e **Desconectar.**

Figura 185 - Palette/Layout/HorizontalArrangement

Fonte: http://appinventor.mit.edu

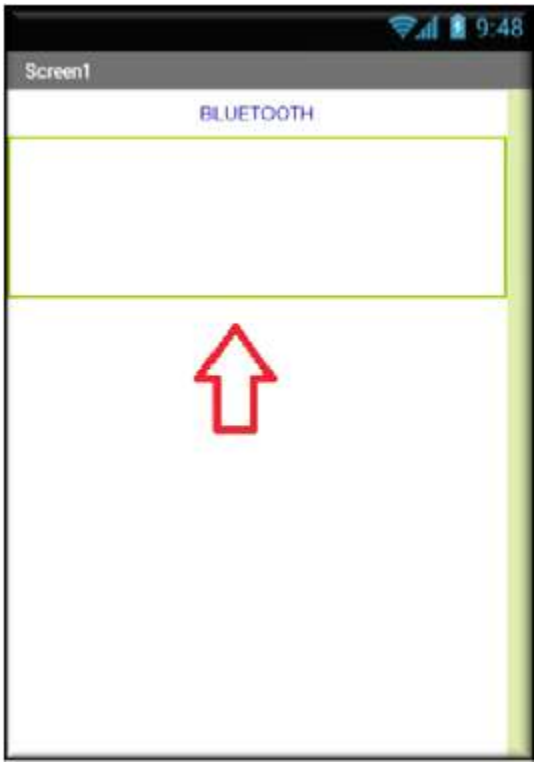
- Na coluna Component selecione o componente HorizontalArrangement1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 186 a seguir indicados pelas setas.

Figura 186 - Configuração do HorizontalArrangement

Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto um retângulo verde centralizado na tela conforme figura 187 a seguir.

Figura 187 - Horizontal Arrangement



Fonte: http://appinventor.mit.edu

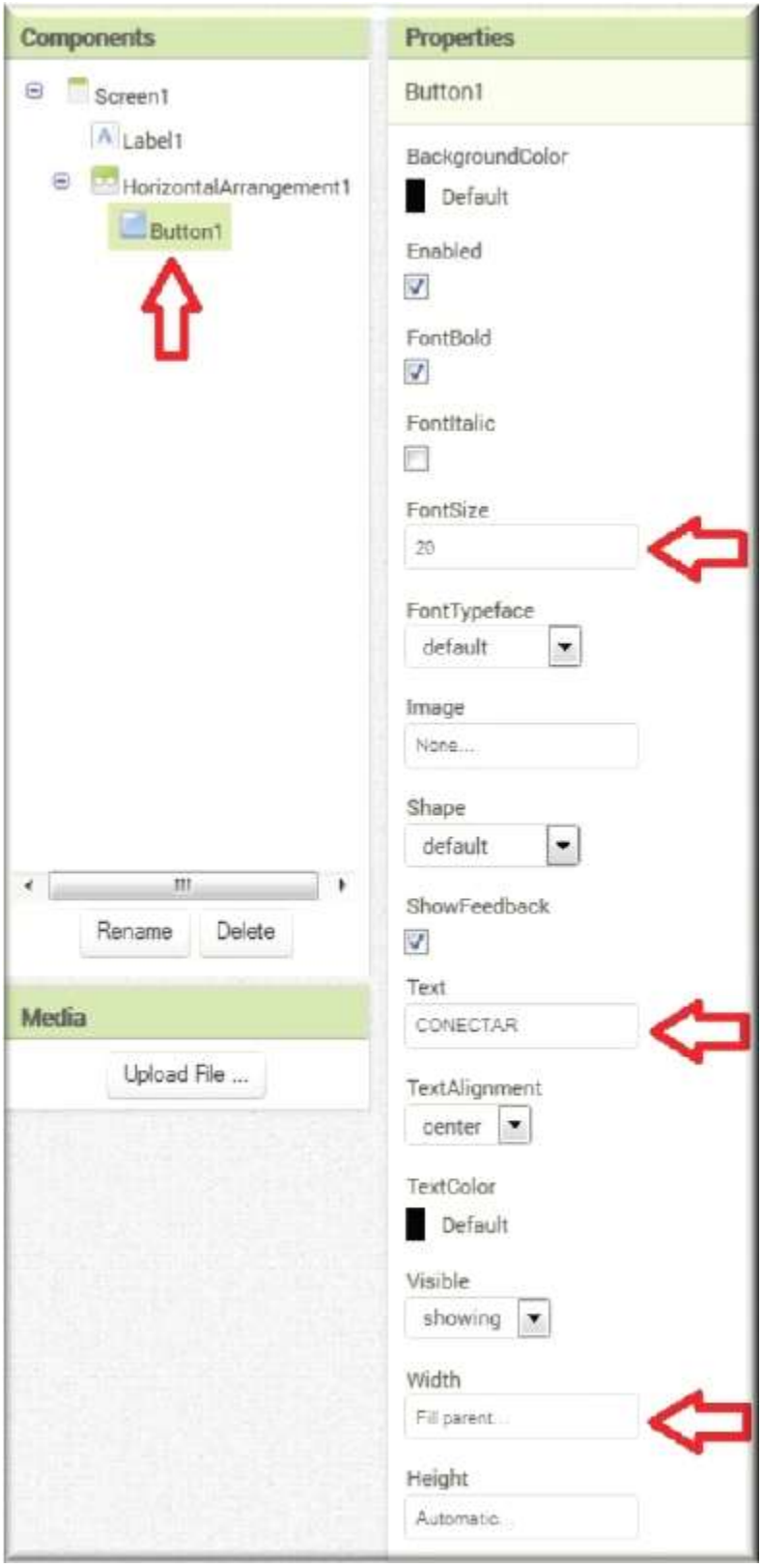
- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Button** conforme indicado na figura 188.

Figura 188 - Botão Conectar

Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente button1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 189 a seguir indicados pelas setas.

Figura 189 - Configuração do Botão Conectar



Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente button1 renomeie o objeto para btnConectar conforme figura 190 a seguir. Esse procedimento facilita a identificação do componente durante o processo de criação da aplicação.

Figura 190 - Renomeando Botão Conectar

Fonte: http://appinventor.mit.edu

Figura 191 - Renomeando o botão

Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto um botão em retângulo verde centralizado e estendido na tela conforme 192.

Figura 192 - Botão Conectar

Fonte: http://appinventor.mit.edu

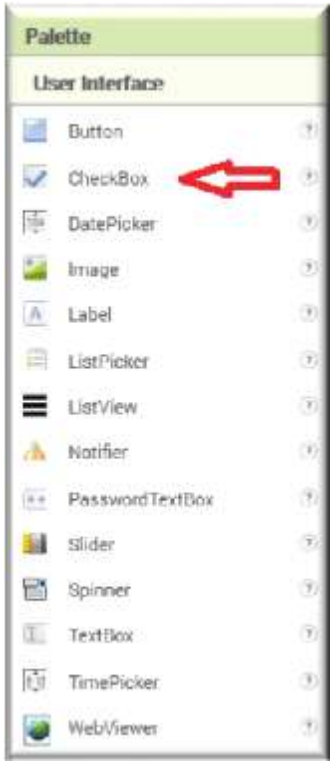
- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Button** e faça o mesmo procedimento do componente anterior com as seguintes alterações:
  - texto do componente: Desconectar;
  - nome do componente : btnDesconectar.
- Com isso poderá ser visto os dois botões em retângulo verde centralizado e extendido na tela conforme 193.

Figura 193 - Botões Conectar e Desconectar

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Checkbox** conforme indicado na figura 194.

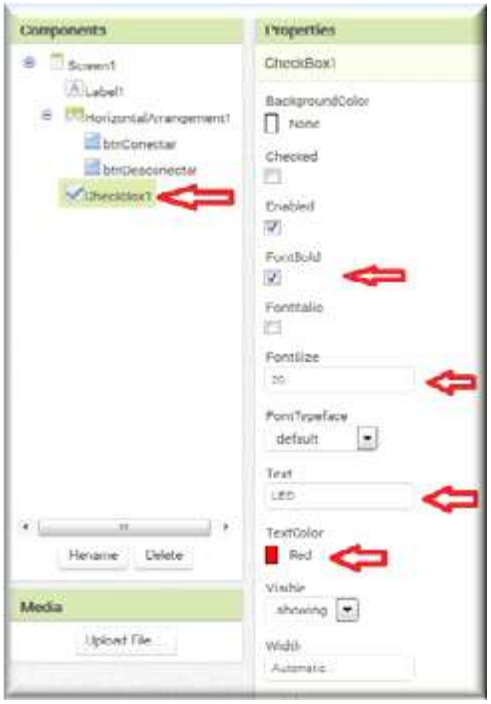
Figura 194 - Checkbox



Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente **checkbox** e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 195 a seguir indicados pelas setas.

Figura 195 - Configurando Checkbox



Fonte: http://appinventor.mit.edu

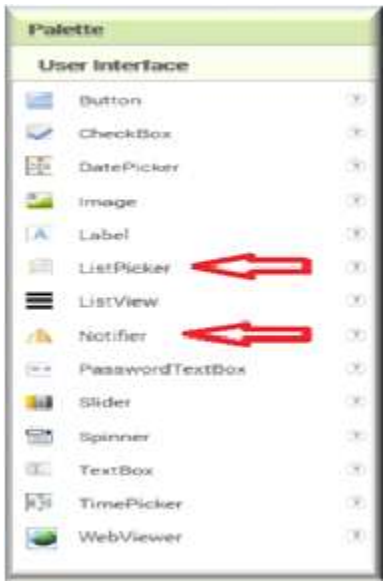
- Com isso poderá ser visto uma caixa de marcação na tela conforme figura 196.

Figura 196 - Checkbox

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 os componentes **ListPicker** e **Notifier** conforme indicado na figura 197.

Figura 197 - Componentes ListPicker e Notifier



Fonte: http://appinventor.mit.edu

- O componente Notfier é um componente visível no tela da aplicação e por isso ele automaticamente se posicionará abaixo da tela conforme figura 198.

Figura 198 - Localização do componente notifier

Fonte: http://appinventor.mit.edu

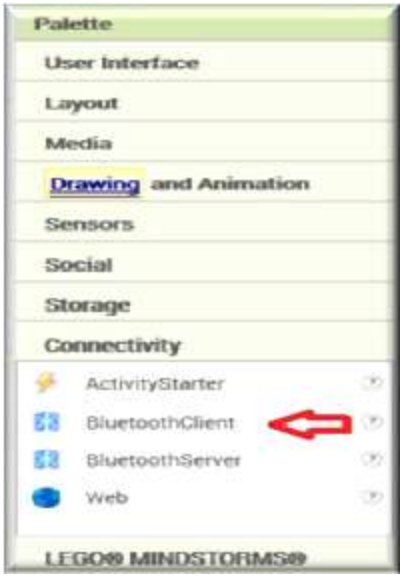
- O componente ListPicker não necessita ficar visível na tela por isso pode-se configurá-lo como invisível. Na coluna Component selecione o componente **ListPicker** e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 199 a seguir indicado pela seta.

Figura 199 - Configuração do componente ListPicker

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **Connectivy** clique, arraste e solte na janela Sreen1 o componente Bluetooth Client conforme indicado na figura 200.

Figura 200 - Componente Bluetooth Client



Fonte: http://appinventor.mit.edu

- O componente Bluetooth Client é um componente visível no tela da aplicação e por isso ele automaticamente se posicionará abaixo da tela conforme figura 7-21.

Figura 201 - Localização do componente Bluetooth Client

Fonte: http://appinventor.mit.edu

8.1.2 Desenvolvendo a Lógica (Blocks) da Aplicação

Nesta etapa iniciaremos o desenvolvimento da lógica de programação da aplicação. Clicar no botão **Blocks** no canto superior direito conforme indicado na figura 7-22.

Figura 202 - Iniciando a lógica da programação

Fonte: http://appinventor.mit.edu

A programação por blocos é bastante intuitiva. Conta com dois recursos bastante práticos que auxiliam no processo de montagem da estrutura. São eles: formato de quebra-cabeça e definição de grupo por cor.

- Clique no componente na coluna Blocks. Aparecerá uma coluna Viewer com o grupo de opções (funções) que podem ser atribuídas ao componente selecionado. Clique na opção desejada e arraste para a área de desenvolvimento.

Figura 203 - Escolhendo o componente e selecionando a ação

Fonte: http://appinventor.mit.edu

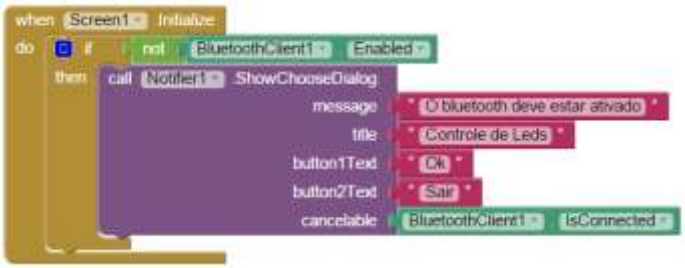
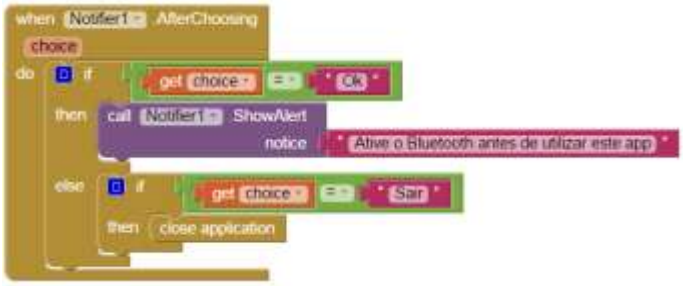
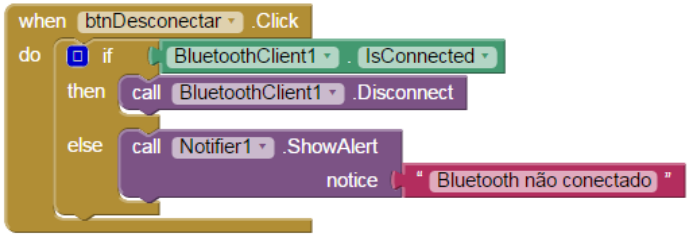
Cada bloco indica no campo de cor mais clara (figura 7-24 indicado pela seta vermelha) a qual componente ou objeto ele pertence. A lacuna indicada pela seta verde na figura 34 informa o local para inserção de outros blocos respeitando o formato do encaixe.

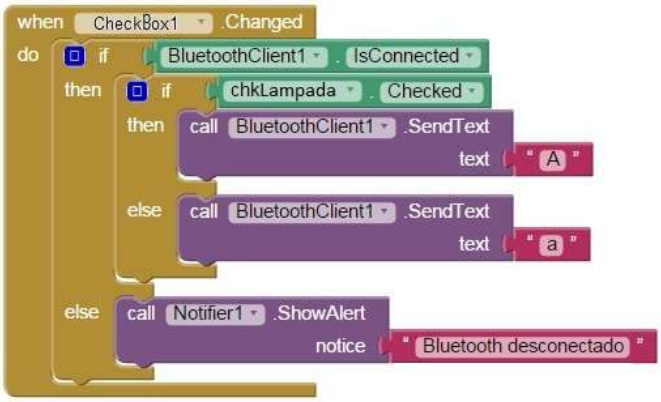
Figura 204 - Entendendo os blocos

Fonte: http://appinventor.mit.edu

A lógica ficará estruturada da seguinte forma:

Tabela 2 - Lógica AppInventor (Experiência 8.1)

Bloco	Comentário
	Este bloco trata o erro quando a aplicação for iniciada caso o bluetooth do dispositivo não esteja habilitado.
	Este bloco trata o erro quando o usuário tentar executar a aplicação sem o bluetooth ativado e avisa via janela de notificação
	Caso o usuário clicar no botão conectar sem o bluetooth habilitado, fornece mensagem de erro. Se o bluetooth tiver habilitado chama o componente ListPicker
	O componente ListPicker retorna uma lista de nomes e endereços de dispositivos bluetooth disponíveis para conexão.
	O cliente bluetooth se conecta ao dispositivo desejado. Se a conexão não for feita com sucesso mostra uma mensagem de erro.
	Quando o usuário clicar no botão Desconectar, verifica se está conectado. Se tiver conectado será desconectado. Se não tiver conectado mostra mensagem de notificação de que o dispositivo não está conectado.

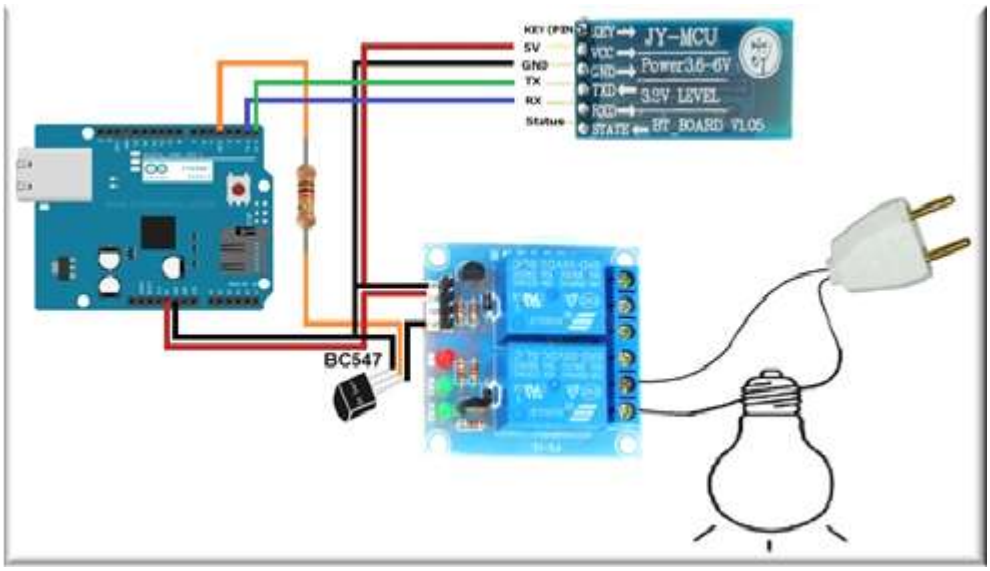
	Faz a verificação se o Bluetooth está conectado. Quando o usuário marcar a caixa de seleção envia a letra "A" para o hardware e a LâmpadaLiga. Quando o usuário desmarcar a caixa de seleção envia a letra "a" para o hardware e a Lâmpada desliga.

8.1.3 O Hardware

Essa experiência utiliza Arduino Uno R3 e o módulo Bluetooth HC-05 mas pode ser realizada com qualquer versão do arduino e outro módulo Bluetooth.

O esquema de ligação pode ser visto na figura 205 a seguir.

Figura 205 - Ligação do Hardware



Fonte: Acervo Pessoal

- Vcc do Bluetooth no 5V
- GND do Bluetooth no 0V
- TX do Bluetooth no RX do arduino – Pino 0
- RX do Bluetooth no TX do arquino – Pino 2
- In Rele Shield → Pino 4 do Arduino

Lista de Materiais

1 Arduino Uno ou Similar
1 Rele Shield (1, 2 ou 4 reles)
1 Bluetooth Shield HC-05
1 Transistor PNP 547 ou similar
1 Resistor 1k Ohm x ¼ W (Marron, Preto, Vermelho)
1 Bocal
1 Lâmpada
1 Protoboard
1 Kit Jumpers

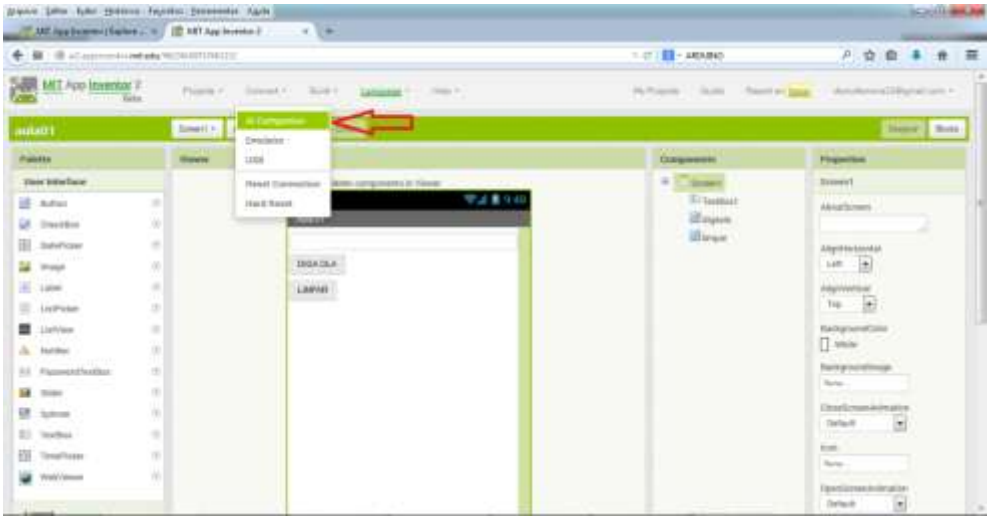
8.1.4 Programação Arduino

```
void setup()
{
  pinMode(4,OUTPUT); //Definindo o Pino 4 como saída
  Serial.begin(9600);//Iniciando a comunicação serial
}
void loop()
{
  char c;//variável para armazenar o caracter informado
  c = Serial.read();//recebendo o caracter via comunicação serial
  if (c =='A') digitalWrite(4,HIGH);// Se o caracter for “A” liga a Lâmpada
  if (c =='a') digitalWrite(4,LOW);//Se o caracter for “a” desliga a Lâmpada
  delay(500);
}
```

8.1.5 Testando

Uma vez que o PC/Notebook e o dispositivo móvel estejam na mesma rede WiFi, clique no menu CONECT e na opção AL COMPANION conforme figura 206 a seguir.

Figura 206 - Conectando Via WiFi



Fonte: http://appinventor.mit.edu

- Será mostrada uma janela com um código e um querycod. pode-se utiliizar qualquer um dos dois conforme figura 207 a seguir

Figura 207 - Conectando Via WiFi

Fonte: http://appinventor.mit.edu

- Abra o software MIT AI2 Companion no dispositivo móvel. Será mostrado a tela da figura 208.
- Digite o código no campo Six Digit Code
- Depois em connect with code

Ou

- Clique em scan QR code e aproxime a Camera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Figura 208 - Acessando a aplicação

Fonte: http://appinventor.mit.edu

Caso tenha ocorrido tudo certo até o momento deverá aparecer a seguinte tela no dispositivo móvel.

Figura 209 - Acessando a aplicação

Fonte: Acervo Pessoal

Obs.: Se o bluetooth não estiveja ativado vá em configurações e ative-o movendo o controle deslizante da esquerda para a direita conforme figura 7-30 e 7-31 a seguir

Figura 210 - Ativando o Bluetooth

Fonte: Acervo Pessoal

Figura 211 - Bluetooth Ativado

Fonte: Acervo Pessoal

Selecione o shield HC-05 para fazer o pareamento entre os dispositivo móvel e o dispositivo bluetooth. Para isso clique em procurar conforme figura 212.

Figura 212 - Procurar Bluetooth

Fonte: Acervo Pessoal

Selecione o dispositivo conforme figura 233.

Figura 213 - Localizando o Bluetooth

Fonte: Acervo Pessoal

Com isso será mostrado um campo para digitar a senha de acesso ao dispositivo

Figura 214 - Pedindo senha

Fonte: Acervo Pessoal

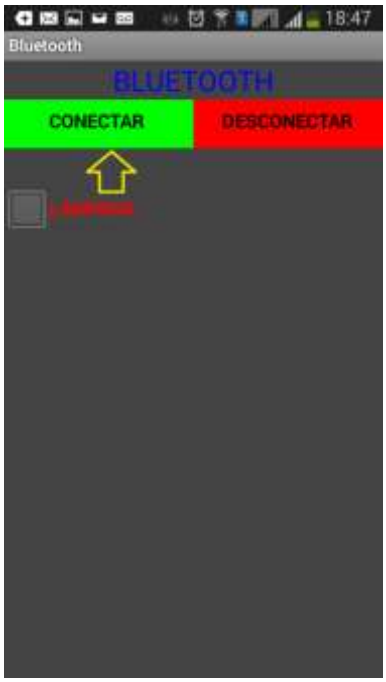
Entre com a senha 1234 e em seguida clique no botão [OK]

Figura 215 - Inserindo senha

Fonte: Acervo Pessoal

Volte à interface do programa e clique no botão verde **CONECTAR**.

Figura 216 - Conectando ao hardware



Fonte: Acervo Pessoal

Selecione o dispositivo anteriormente pareado. Observe, pois geralmente o nome vem composto primeiramente pelo endereço MAC e posteriormente pelo nome do modelo então observe sempre o final do nome do item a ser selecionando conforme figura 217 a seguir.

Figura 217 - Localizando

Fonte: Acervo Pessoal

Clique no nome “Lâmpada” para marcar o quadrinho ao lado esquerdo. Com o quadrinho marcado (selecionado) a lâmpada irá ligar; com o quadrinho desmarcado a lâmpada irá desligar conforme figuras 218, 219, 220 e 221.

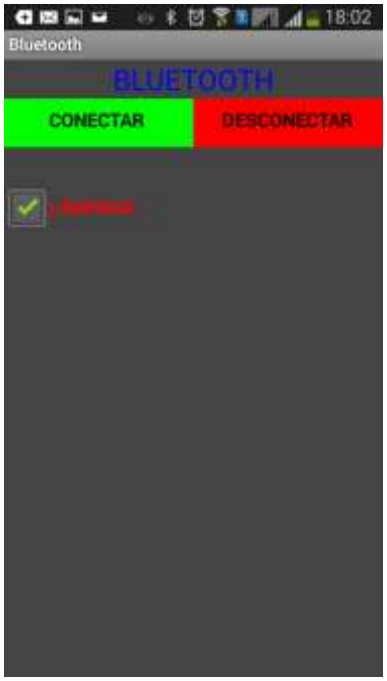
Figura 218 - Tela do Aplicativo

Fonte: Acervo Pessoal

Figura 219 - Lâmpada de desligada

Fonte: Acervo Pessoal

Figura 220 - Comando ligar



Fonte: Acervo Pessoal

Figura 221 - Lampada ligada

Fonte: Acervo Pessoal

8.2 Ligar lâmpada através do celular utilizando Bluetooth (com resposta)

Até agora fizemos os acionamento à distância sem termos a certeza de que a ação foi executada. Nessa experiência iremos obter resposta no dispositivo se a Lâmpada realmente ligou e desligou conforme o comando dado.

O circuito e a programação é praticamente os mesmos da experiência 1. O que muda no circuito é a adição de um componente para receber a informação do estado da luz, informar ao microcontrolador e o mesmo enviar resposta ao dispositivo móvel. Com isso inserimos na programação do arduino uma verificação em uma determinada porta de entrada de nível de sinal enviado pelo sensor e na programação do dispositivo móvel o complemento para receber a resposta de forma completa.

Para isso utilizaremos um sensor de luminosidade com resistência variável ou LDR figura 222 a seguir.

Figura 222 - LDR

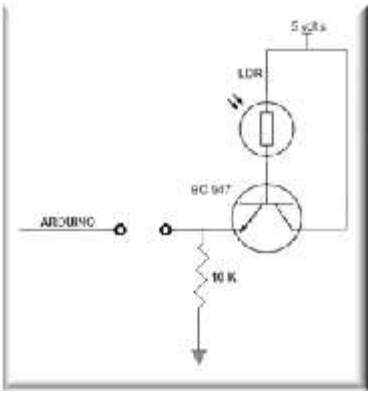
Fonte: https://meetarduino.wordpress.com/2012/05/03/arduino-ldr-led/

LDR (do inglês ***L**ight **D**ependent **R**esistor*), em português Resistor Dependente de Luz ou Fotoresistência é um componente eletrônico passivo do tipo resistor variável, mais especificamente, é um resistor cuja resistência varia conforme a intensidade da luz que incide sobre ele. Tipicamente, à medida que a intensidade da luz aumenta, a sua resistência diminui.

O LDR é construído a partir de material semicondutor com elevada resistência elétrica. Quando a luz que incide sobre o semicondutor tem uma frequência suficiente, os fótons que incidem sobre o semicondutor libertam elétrons para a banda condutora que irão melhorar a sua condutividade e assim diminuir a resistência.

O circuito que emitirá a resposta ao arduino de que a luz está realmente acesa ou apagada pode ser visto na figura 223 a seguir.

Figura 223 - Circuito do LDR



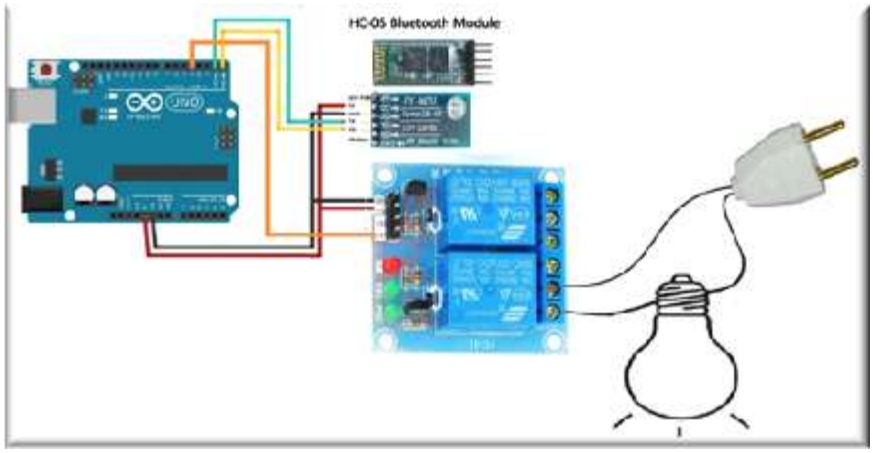
Fonte: Acervo Pessoal

O princípio de funcionamento deste circuito é bem simples. O transistor trabalhando como chave irá saturar quando o **LDR** gerar uma resistência o suficiente para que haja uma tensão de 0,69v no VBE do transistor, assim uma tensão de 5 volts chega na porta do arduino.

Para se obter resposta satisfatória do conjunto deve-se montar o LDR posicionado para receber a luz da lâmpada quando a mesma estiver acessa de tal forma que quando a lâmpada estiver apagada não chegue luminosidade suficiente para que o LDR conduza corrente para alimentar a base do transistor. De tal forma é importante que o LDR esteja ao abrigo da luz natural do ambiente.

A montagem completa pode ser vista na figura 224 a seguir.

Figura 224 - Montagem do Hardware

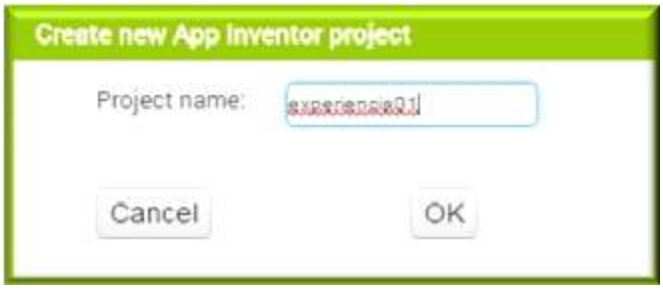


Fonte: Acervo Pessoal

Iniciaremos abrindo o ambiente de desenvolvimento App Inventor no site http://www.appinventor.mit.edu e executando os procedimentos a seguir.

- Clique no botão Start New Project (figura 225) e dê um nome ao projeto. O nome não deve conter espaço em branco.

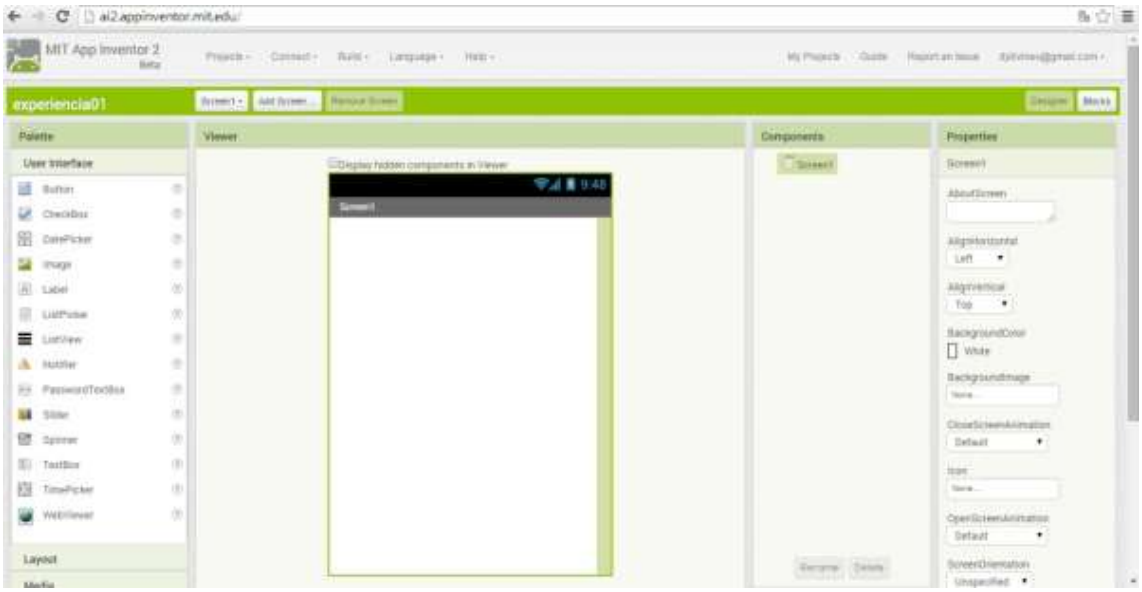
Figura 225 - Nome do projeto



Fonte: http://appinventor.mit.edu

Abrirá automaticamente o ambiente de desenvolvimento na guia Designer conforme figura 226.

Figura 226 - Ambiente de desenvolvimento

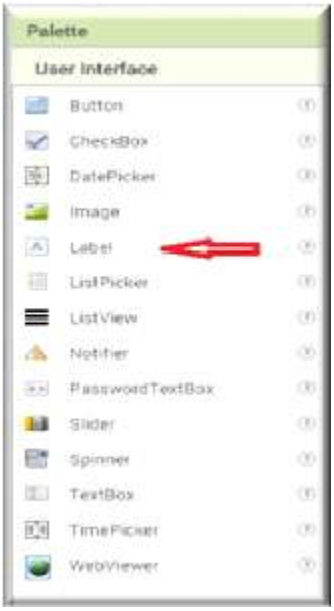


Fonte: http://appinventor.mit.edu

8.2.1 Desenvolvendo o Layout (Designer) da Aplicação

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente Label conforme indicado na figura 227.

Figura 227 - Palette /User Interface /Label

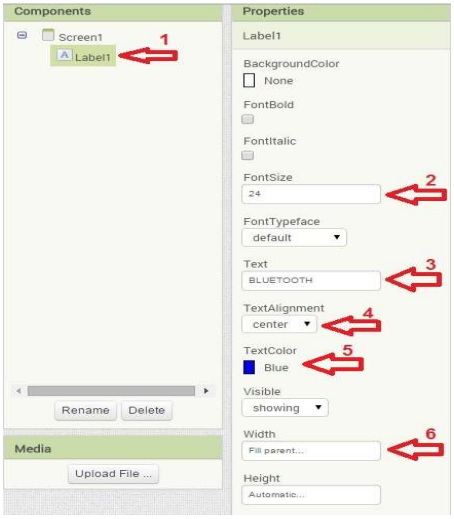


Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente Label1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 228 a seguir indicados pelas setas (1 – componente de texto, 2 – tamanho da fonte, 3 – texto sugerido para o título, 4 – alinhamento do texto, 5 – cor do texto, 6 – largura.

  Obs.: A opção 6 selecionada como Fill Parent define que o componente irá ocupar toda a largura da aplicação.

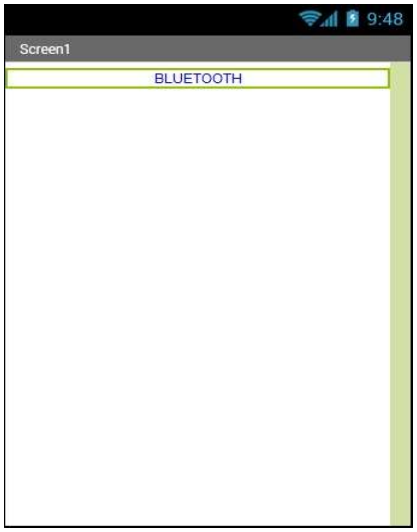
Figura 228 - Configurando o componente Label1



Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto o texto “BLUETOOTH” centralizado na tela conforme figura 299 a seguir.

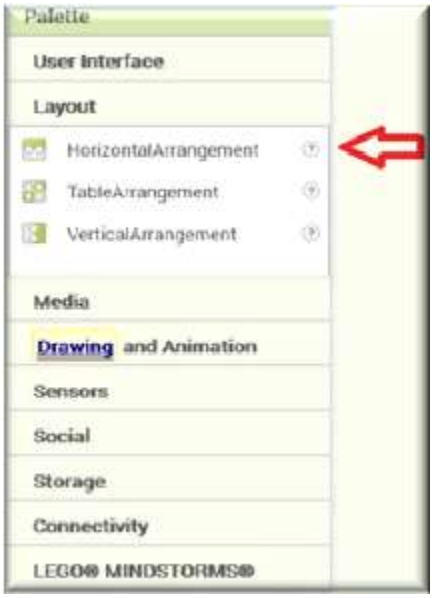
Figura 229 - Label



Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **Layout** clique, arraste e solte na janela Sreen1 um componente **HorizontalArrangement** conforme indicado na figura 230. Esse componente possibilita o posicionamento de objetos lado a lado horizontalmente e será utilizado para acomodar os botões **Conectar** e **Desconectar.**

Figura 230 - Palette /Layout /HorizontalArrangement



Fonte: http://appinventor.mit.edu

Na coluna Component selecione o componente HorizontalArrangement1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 231 a seguir indicados pelas setas.

Figura 231 - Configuração do HorizontalArrangement

Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto um retângulo verde centralizado na tela conforme figura 232 a seguir.

Figura 232 - Horizontal Arrangement

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Button** conforme indicado na figura 233.

Figura 233 - Botão Conectar

Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente button1 e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 234 a seguir indicados pelas setas.

Figura 234 - Configuração do Botão Conectar

Fonte: http://appinventor.mit.edu

- Na coluna Component selecione o componente button1 renomeie o objeto para btnConectar conforme figura 235 a seguir. Esse procedimento facilita a identificação do componente durante o processo de criação da aplicação.

Figura 235 - Renomeando Botão Conectar

Fonte: http://appinventor.mit.edu

- Com isso poderá ser visto um botão em retângulo verde centralizado e extendido na tela conforme 236.

Figura 236 - Botão Conectar

Fonte: http://appinventor.mit.edu

Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Button** e faça o mesmo procedimento do componente anterior com as seguintes alterações:

- texto do componente: Desconectar;
- nome do componente : btnDesconectar.

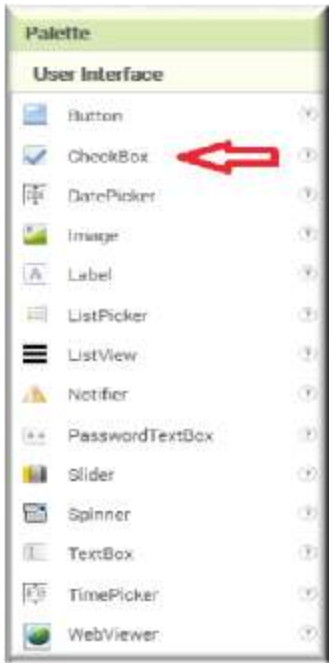
- Com isso poderá ser visto os dois botões em retângulo verde centralizado e extendido na tela conforme 237.

Figura 237 - Botões Conectar e Desconectar

Fonte: http://appinventor.mit.edu

Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 um componente **Checkbox** conforme indicado na figura 238.

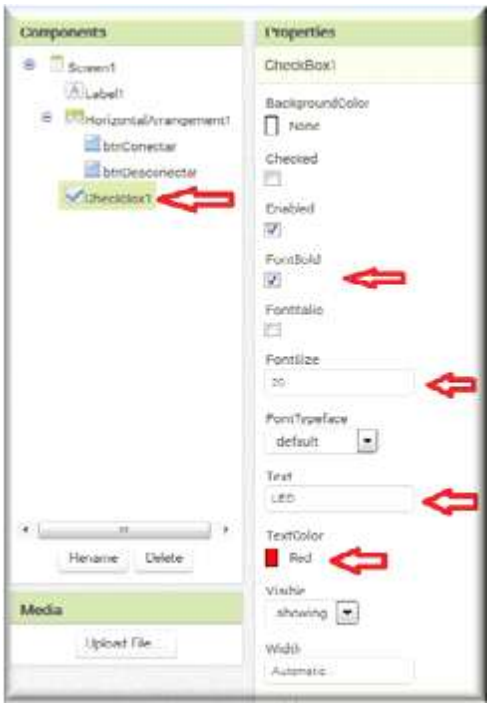
Figura 238 – Componente checkbox



Fonte: http://appinventor.mit.edu

Na coluna Component selecione o componente **checkbox** e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 239 a seguir indicados pelas setas.

Figura 239 - Configurando Checkbox



Fonte: http://appinventor.mit.edu

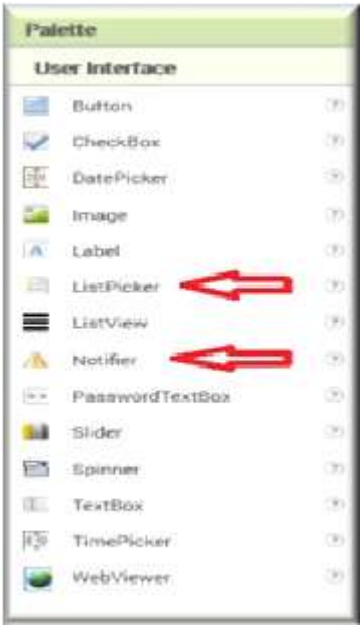
- Com isso poderá ser visto uma caixa de marcação na tela conforme figura 240.

Figura 240 - Checkbox

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **User Interface** clique, arraste e solte na janela Sreen1 os componentes **ListPicker** e **Notifier** conforme indicado na figura 241.

Figura 241 - Componentes ListPicker e Notifier



Fonte: http://appinventor.mit.edu

- O componente Notfier é um componente visível no tela da aplicação e por isso ele automaticamente se posicionará abaixo da tela conforme figura 242.

Figura 242 - Localização do componente notifier

Fonte: http://appinventor.mit.edu

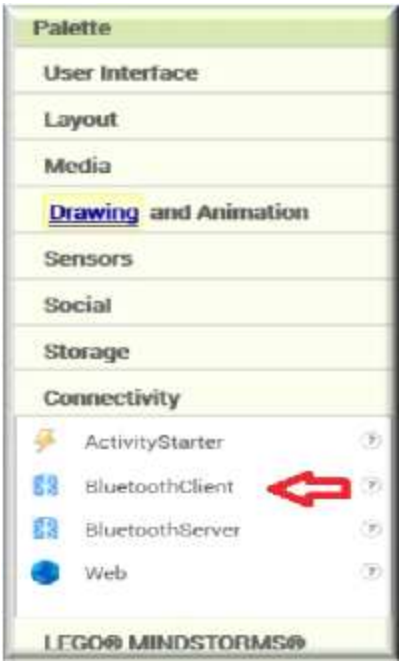
- O componente ListPicker não necessita ficar visível na tela por isso pode-se configurá-lo como invisível. Na coluna Componente selecione o componente **ListPicker** e faça as alterações nos itens da coluna Properties (Propriedades) conforme figura 243 a seguir indicado pela seta.

Figura 243 - Configuração do componente ListPicker

Fonte: http://appinventor.mit.edu

- Na coluna **Palette**, em **Connectivy** clique, arraste e solte na janela Sreen1 o componente Bluetooth Client conforme indicado na figura 244.

Figura 244 - Componente Bluetooth Client



Fonte: http://appinventor.mit.edu

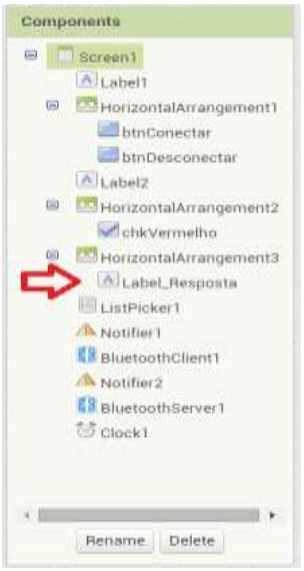
- O componente Bluetooth Client é um componente visível no tela da aplicação e por isso ele automaticamente se posicionará abaixo da tela conforme figura 245.

Figura 245 - Localização do componente Bluetooth Client

Fonte: http://appinventor.mit.edu

Insira também um texto “STATUS ...” que será utilizado para informar o estado do dispositivo controlado (figuras 246 e 247 a seguirem)

Figura 246 - Label Resposta



Fonte: http://appinventor.mit.edu

Figura 247 - Layout com Resposta

Fonte: http://appinventor.mit.edu

8.2.2 Desenvolvendo a Lógica (Blocks) da Aplicação

Nesta etapa iniciaremos o desenvolvimento da lógica de programação da aplicação. Clicar no botão **Blocks** no canto superior direito conforme indicado na figura 248.

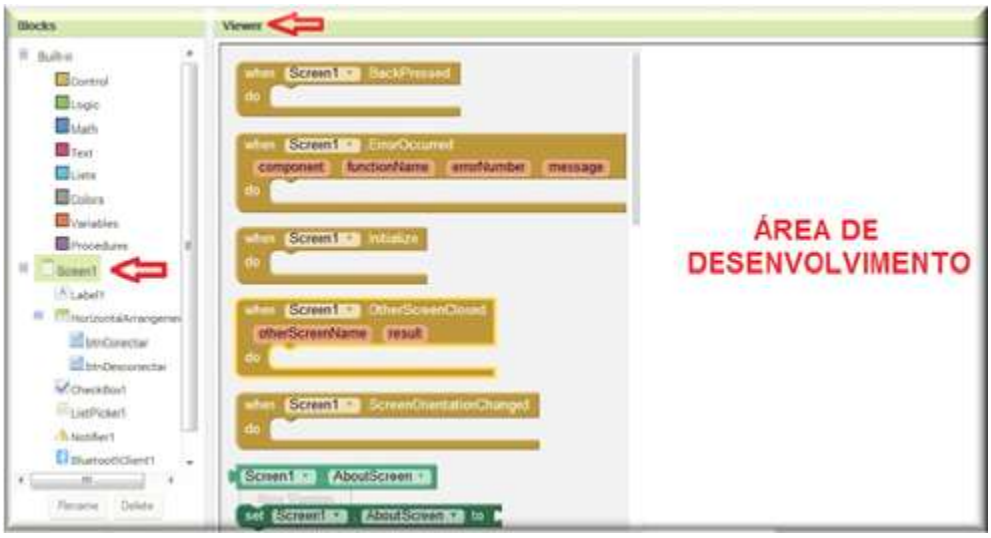
Figura 248 - Iniciando a lógica da programação

Fonte: http://appinventor.mit.edu

A programação por blocos é bastante intuitiva. Conta com dois recursos bastante práticos que auxiliam no processo de montagem da estrutura. São eles: formato de quebra-cabeça e definição de grupo por cor.

- Clique no componente na coluna Blocks. Aparecerá uma coluna Viewer com o grupo de opções (funções) que podem ser atribuídas ao componente selecionado. Clique na opção desejada e arraste para a área de desenvolvimento conforme figura 249.

Figura 249 - Escolhendo o componente e selecionando a ação



Fonte: http://appinventor.mit.edu

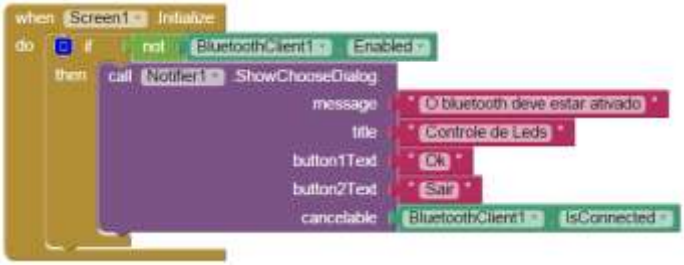
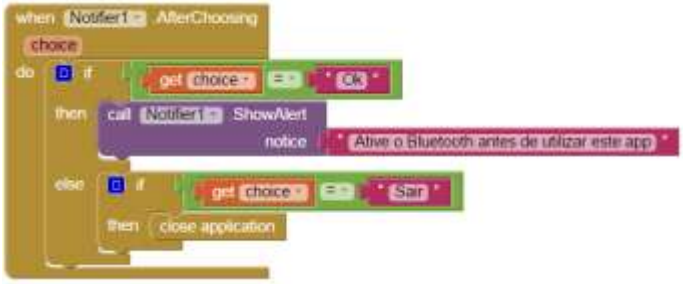
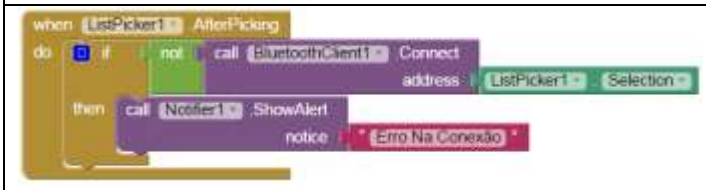
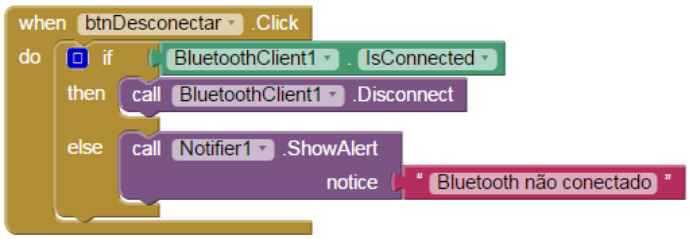
Cada bloco indica no campo de cor mais clara (figura 250 indicado pela seta vermelha) a qual componente ou objeto ele pertence. A lacuna indicada pela seta verde na figura 34 informa o local para inserção de outros blocos respeitando o formato do encaixe.

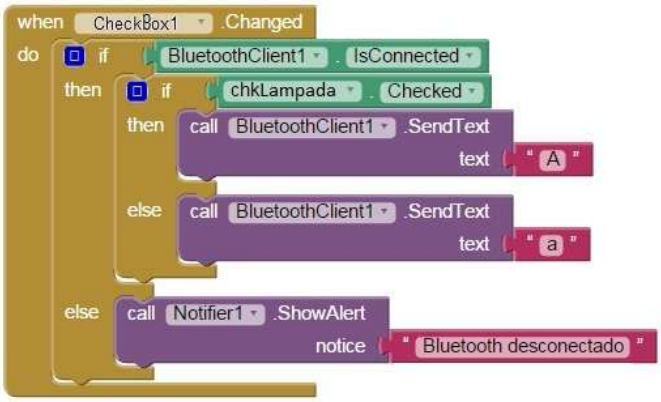
Figura 250 - Entendendo os blocos

Fonte: http://appinventor.mit.edu

A lógica ficará estruturada da seguinte forma:

Tabela 3 - Lógica AppInventor (Experiência 8.2)

Bloco	Comentário
	Este bloco trata o erro quando a aplicação for iniciada caso o bluetooth do dispositivo não esteja habilitado.
	Este bloco trata o erro quando o usuário tentar executar a aplicação sem o bluetooth ativado e avisa via janela de notificação
	Caso o usuário clicar no botão conectar sem o bluetooth habilitado, fornece mensagem de erro. Se o bluetooth tiver habilitado chama o componente ListPicker
	O componente ListPicker retorna uma lista de nomes e endereços de dispositivos bluetooth disponíveis para conexão.
	O cliente bluetooth se conecta ao dispositivo desejado. Se a conexão não for feita com sucesso mostra uma mensagem de erro.
	Quando o usuário clicar no botão Desconectar, verifica se está conectado. Se tiver conectado será desconectado. Se não tiver conectado mostra mensagem de notificação de que o dispositivo não está conectado.

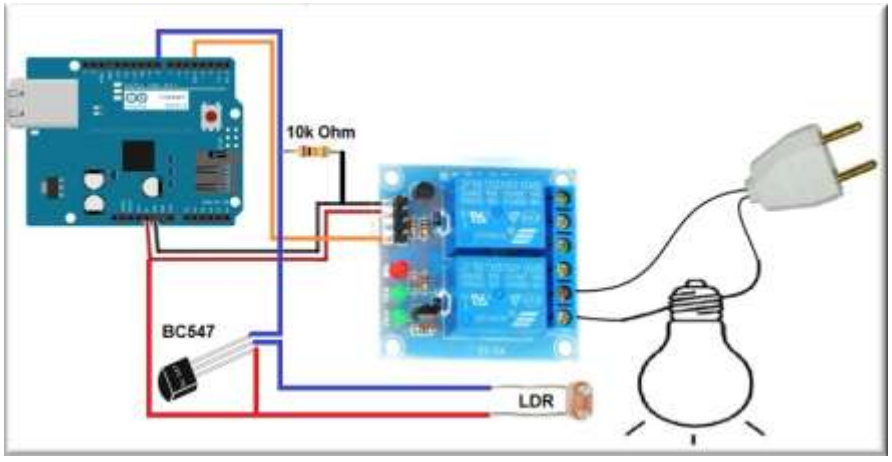
	Faz a verificação se o Bluetooth está conectado. Quando o usuário marcar a caixa de seleção envia a letra "A" para o hardware e a LâmpadaLiga. Quando o usuário desmarcar a caixa de seleção envia a letra "a" para o hardware e a Lâmpada desliga.
	Monitora o tempo todo o estado do dispositivo controlado (lâmpada) informando com as letras "L" para ligado e "D" para desligado emitido pelo arduino através do bluetooth.

8.2.3 O Hardware

Essa experiência utiliza Arduino Uno R3 e o módulo Bluetooth HC-05 mas pode ser realizada com qualquer versão do arduino e outro módulo Bluetooth.

O esquema de ligação pode ser visto na figura 251 a seguir.

Figura 251 - Ligação do Hardware



Fonte: http://appinventor.mit.edu

- Vcc do Bluetooth no 5V
- GND do Bluetooth no 0V
- TX do Bluetooth no RX do arduino – Pino 0
- RX do Bluetooth no TX do arquino – Pino 2
- In Rele Shield → Pino 4 do Arduino
- LDR → Pino 8 do Arduino

Lista de Materiais

1 Arduino Uno ou Similar
1 Rele Shield (1, 2 ou 4 reles)
1 Bluetooth Shield HC-05
2 Transistor PNP 547 ou similar
1 Resistor 1k Ohm x ¼ W (Marron, Preto, Vermelho)
1 Resistor 10k Ohm x ¼ W (Marron, Preto, Laranja)
1 LDR
1 Bocal
1 Lâmpada
1 Protoboard
1 Kit Jumpers

8.2.4 Programação Arduino

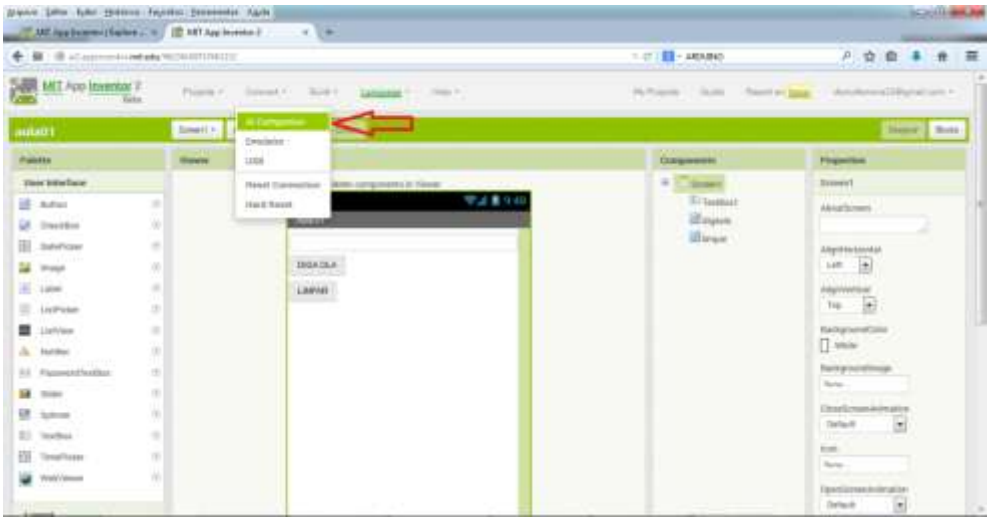
```
const int led = 13;
const int ldr = 8;
int estado = 0;
void setup()
{
  pinMode (4,OUTPUT); //Definindo o Pino 4 como saída
  pinMode (8,INPUT);
  pinMode (13,OUTPUT);
  Serial.begin(9600); //Iniciando a comunicação serial
}
void loop()
```

```
{
  char c;//variável para armazenar o caracter informado
   char l; //variável para armazenar estado ligado
   char d; //variável para armazenar estado desligado
   l = 'L'; //atribui a letra L à variável l
   d = 'D'; //atribui a letra D à variável d
  if (Serial.available())  //verifica se tem dados diponível para leitura
  {
 c = Serial.read(); //recebendo o caracter via comunicação serial
     if (c =='A') //se rececer caracter A liga o LED
     {
     digitalWrite(4,HIGH); // Se o caracter for “A” liga a Lâmpada
     }
     if (c =='a')
     {
     digitalWrite(4,LOW); //Se o caracter for “a” desliga a Lâmpada
     }
  delay(500);
  }
  estado = digitalRead(ldr);
  if (estado == HIGH) { //se o LDR condizir envia o caracter L de ligado
    digitalWrite(led,HIGH); //Botão pressionado, acende o led.
    Serial.write(l);
  } else {//se o LDR nao conduzir envia o caracter L de desligado
    digitalWrite(led,LOW);  //Botão não pressionado, apaga o led.
    Serial.write(d);
  }
}
```

8.2.5 Testando

Uma vez que o PC/Notebook e o dispositivo móvel estejam na mesma rede WiFi, clique no menu CONECT e na opção AL COMPANION conforme figura 252 a seguir.

Figura 252 -Conectando Via WiFi



Fonte: http://appinventor.mit.edu

- Será mostrada uma janela com um código e um querycod. pode-se utiliizar qualquer um dos dois conforme figura 253 a seguir

Figura 253 - Conectando Via WiFi

Fonte: http://appinventor.mit.edu

- Abra o software MIT AI2 Companion no dispositivo móvel. Será mostrado a tela da figura 254.

- Digite o código no campo Six Digit Code
- Depois em connect with code

Ou

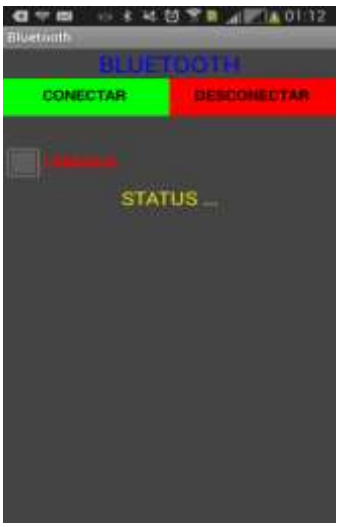
- Clique em scan QR code e aproxime a Camera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Figura 254 - Acessando a aplicação

Fonte: http://appinventor.mit.edu

Caso tenha ocorrido tudo certo até o momento deverá aparecer a seguinte tela no dispositivo móvel (figura 255).

Figura 255 - Acessando a aplicação



Fonte: http://appinventor.mit.edu

Se o bluetooth não estiveja ativado vá em configurações e ative-o movendo o controle deslizante da esquerda para a direita conforme figura 256 e 257 a seguir

Figura 256 - Ativando o Bluetooth

Fonte: http://appinventor.mit.edu

Figura 257 - Bluetooth Ativado

Fonte: http://appinventor.mit.edu

Selecione o shield HC-05 para fazer o pareamento entre os dispositivo móvel e o dispositivo bluetooth. Para isso clique em procurar conforme figura 258.

Figura 258 - Procurando Bluetooth

Fonte: http://appinventor.mit.edu

Selecione o dispositivo conforme figura 259.

Figura 259 - Selecionando Bluetooth

Fonte: http://appinventor.mit.edu

Com isso será mostrado um campo para digitar a senha de acesso ao dispositivo

Figura 260 - Inserindo o PIN

Fonte: http://appinventor.mit.edu

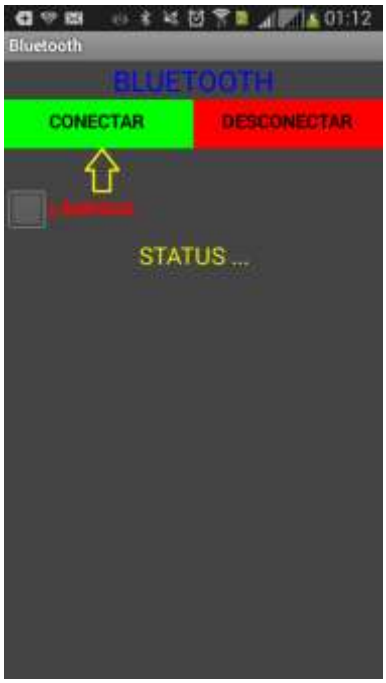
Entre com a senha 1234 e em seguida clique no botão [OK]

Figura 261 - Pareando o dispositivo

Fonte: http://appinventor.mit.edu

Feito isso volte a interface do programa e clique no botão verde CONECTAR.

Figura 262 - Conectando o Bluetooth



Fonte: http://appinventor.mit.edu

Selecione o dispositivo anteriormente pareado. Observe pois geralmente o nome vem composto primeiramente pelo endereço MAC e posteriormente pelo nome do modelo então observe sempre o final do nome do item a ser selecionando conforme figura 263 a seguir.

Figura 263 - Selecionando Bluetooth Pareado

Fonte: http://appinventor.mit.edu

Com isso o software já irá informar o estado da lâmpada. Como a lâmpada ainda não está ligada, será mostrada a letra “D” de “desligado” no lugar da palavra STATUS. Essa amostragem permanecerá e preencherá a tela até que o estado da lâmpada mude para ligado e neste caso aparecerá a letra “L” de “ligado”.

Figura 264 - Utilizando o Aplicativo

Fonte: Acervo Pessoal

Clique no nome “Lâmpada” para marcar o quadrinho ao lado esquerdo. Com o quadrinho marcado (selecionado) a lâmpada irá ligar; com o quadrinho desmarcado a lâmpada irá desligar conforme figuras 265, 266, 267 e 268.

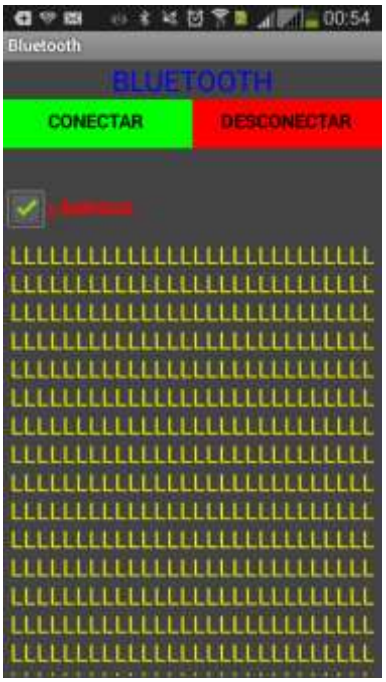
Figura 265 - Informação Desligado

Fonte: Acervo Pessoal

Figura 266 - Lâmpada Desligada

Fonte: Acervo Pessoal

Figura 267 - Informação Ligado



Fonte: Acervo Pessoal

Figura 268 - Lâmpada Ligada

Fonte: Acervo Pessoal

8.3 Ligar lâmpada através do celular utilizando Ethernet e Roteador (sem resposta)

Um roteador é um elemento intermediário em uma rede de computadores que permite o roteamento de pacotes entre redes separadas. Em outras palavras o roteador permite que vários computadores utilizem recursos da mesma rede como também o uso da Internet. Atualmente os roteadores oferecem dois tipos de conexão de rede, com fio e sem fio (wireless). Geralmente a comunicação com fio é destinada a interligação com rede cabeada e a comunicação wireless é destinada a computadores desktops, notebooks e dispositivos móveis tais como smartphones e tablets. Dispositivos como tvs, impressoras, vídeo games estão sendo vendidos com essa tecnologia que já é tendência.

Figura 269 - Rede com roteador

Fonte: http://www.chapteron.com.br

Um shield ethernet possui um preço muito menor um shield wifi. Levando em consideração que os roteadores estão popularizados optamos por fazer essa configuração.

Nesta experiência iremos ligar e desligar 1 lâmpada utilizando um dispositivo móvel através da rede WiFi proveniente de um roteador com a plataforma arduino.

8.3.1 Desenvolvendo o Layout (Designer) da Aplicação

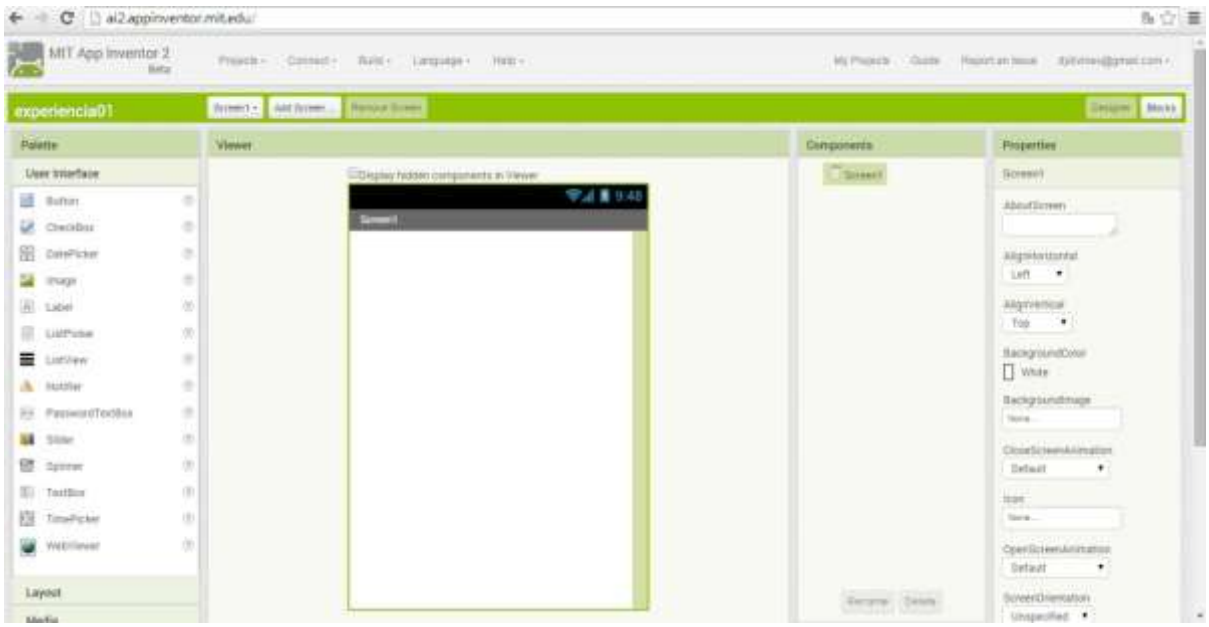
- Clique no botão Start New Project (figura 270) e dê um nome ao projeto. O nome não deve conter espaço em branco.

Figura 270 - Nome do projeto

Fonte: http://appinventor.mit.edu

Abrirá automaticamente o ambiente de desenvolvimento na guia Designer conforme figura 271.

Figura 271 - Ambiente de desenvolvimento



Fonte: http://appinventor.mit.edu

- Siga os procedimentos descritos no item 7.1.1 adicionando os seguintes componentes:

    - Horizontal Arrangement – Para acomodar os botões lado a lado;
      Palette → User Interface
    - Dois botões “Ligar” e “Desligar” (btnLigar e btnDesligar respectivamente);
      Palette → User Interface
    - Um Label com o texto “STATUS”;
      Palette → User Interface
    - Um Label com o nome Label_Resultado com o texto “Aguardando ...”;
      Palette → User Interface
    - Um Text Box sem texto;
      Palette → User Interface
    - Um Web.
      Palette → Connectivity

Organize o layout conforme desejar. O layout de nossa experiência pode ser visto na figura 272

Figura 272 - Interface da Aplicação

Fonte: http://appinventor.mit.edu

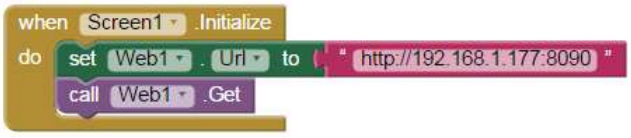
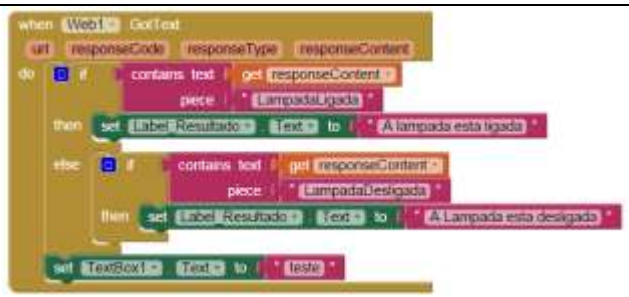
8.3.2 Desenvolvendo a Lógica (Blocks) da Aplicação

Nesta etapa iniciaremos o desenvolvimento da lógica de programação da aplicação. Clicar no botão **Blocks** no canto superior direito conforme indicado na figura 273 e seguir os procedimentos baseados no item 7.1.2.

Figura 273 - Iniciando a lógica da programação

Fonte: http://appinventor.mit.edu

Tabela 4 - Lógica AppInventor (Experiência 8.3)

Bloco	Comentário
	Este bloco conecta a aplicação no endereço de rede pré definido. Este endereço refere-se ao endereço IP do adaptador ethernet conectado ao Arduino. Pode assumir qualquer valor de endereço dependendo da configuração da rede a ser utilizada. É configurada na programação do Arduino. *Para mais detalhes sobre configuração de rede consulteo Anexo VII.*
	Esse bloco verifica o texto de resposta da página web e compara com o valor de texto armazenado. De acordo com o estado da lampada (ligada ou desligada) a pagina retornará uma resposta em forma de texto e a aplicação replica essa informação. É uma maneira de se ter o feedback do acesso.

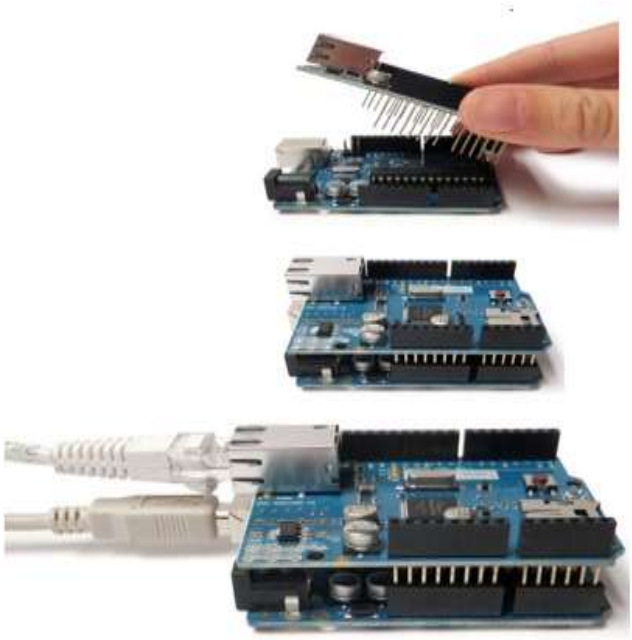
	Esse bloco envia para o Arduino/AdaptadorEthernet um comando em forma de url (link) que liga a lampada.
	Esse bloco envia para o Arduino/AdaptadorEthernet um comando em forma de url (link) que liga lampada.

8.3.3 O Hardware

Essa experiência utiliza Arduino Uno R3 e o Adaptador de rede ethernet W5100. Outros adaptadores sugerem outras programações.

Esse shield de adaptador pode ser conectado diretamente ao arduino conforme figura 274 a seguir.

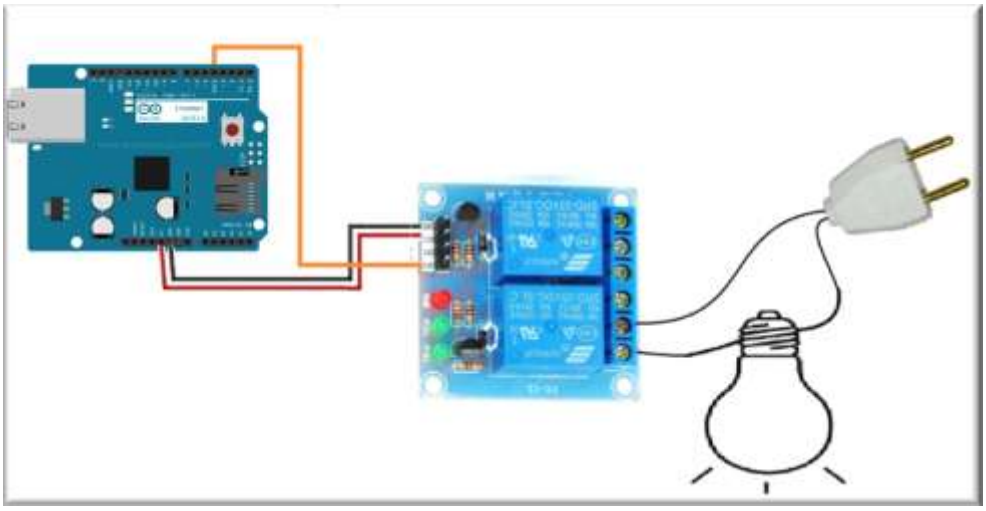
Figura 274- Arduino + Ethernet Shield



Fonte: http://pt.slideshare.net/maganus/arduino-smtp-en

O diagrama de ligação pode ser visto na figura 275 a seguir

Figura 275 – Ligação



Fonte: Acervo Pessoal

Lista de Materiais

1 Arduino Uno ou Similar
1 Rele Shield (1, 2 ou 4 reles)
1 Ethernet Shield
1 LDR
1 Bocal
1 Lâmpada
1 Protoboard
1 Kit Jumpers

***Obs.:** O adaptador ethernet deve ser ligado via cabo de rede com conector RJ45 ao roteador. A configuração de endereço IP deve estar dentro do range da rede local. O acesso poderá ser feito também através da Internet por meio de uma configuração no roteador.

Para mais detalhes sobre configuração IP consulte o Anexo VIII.

8.3.4 Programação Arduino

```
#include <SPI.h>
#include <String.h>
#include <Ethernet.h>

byte mac[] = { 0x90, 0xA2, 0xDA, 0x00, 0x9B, 0x36 }; // Endereço Mac
```

```
byte ip[] = { 192, 168, 1, 177 }; // Endereço de Ip da sua Rede
EthernetServer server(8090); // Porta de serviço
int lampada = 4; // Pino onde deve ser ligado o shield de rele
String readString = String(30); // string para buscar dados de endereço
boolean statusLampada = false; // Variável para o status da lampada
void setup(){
  // Inicia o Ethernet
  Ethernet.begin(mac, ip);
  // Define o pino como saída
  pinMode(lampada, OUTPUT);
  // Inicia a comunicação Serial
  Serial.begin(9600);
}
void loop(){
  // Criar uma conexão de cliente
  EthernetClient client = server.available();
   if (client) {
    while (client.connected())
    {
     if (client.available())
     {
       char c = client.read();
       // ler caractere por caractere vindo do HTTP
       if (readString.length() < 30)
       {
         // armazena os caracteres para string
         readString += (c);
       }
        //se o pedido HTTP terminou
```

```
if (c == '\n')
  {
  // vamos verificar se a lampada deve ser ligada
  // Se a string possui o texto L=Ligar
  if(readString.indexOf("LigarLampada")>=0)
  {
    digitalWrite(lampada, HIGH); // A Lampada vai ser ligada
    statusLampada = true;
  }
  // Se a string possui o texto L=Desligar
  if(readString.indexOf("DesligarLampada")>=0)
  {
    // A lampada vai ser desligado
    digitalWrite(lampada, LOW);
    statusLampada = false;
  }
// dados HTML de saída começando com cabeçalho padrão
client.println("HTTP/1.1 200 OK");
client.println("Content-Type: text/html");
client.println();
client.print("<font size='20'>");
if (statusLampada) {
client.print("LampadaLigada");
} else {
  client.print("LampadaDesligada");
}
readString="";//limpa string para a próxima leitura
client.stop();        // parar cliente
}   }   }   }    }
```

8.3.5 Testando

Teste a aplicação baseado no procedimento do item 7.1.5.

Caso tenha ocorrido tudo certo até o momento deverá aparecer a seguinte tela no dispositivo móvel (figuras 276, 277 e 278).

Figura 276 - Tela1

Figura 277 - Tela2

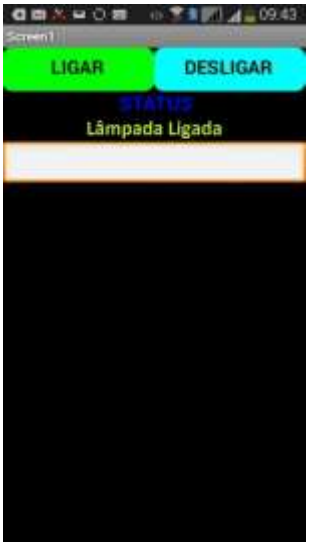


Figura 278 - Tela3

Fonte: http://appinventor.mit.edu

Aguardando: Quando a conexão ainda não estiver sido estabelecida

Lampada Ligada: Quando pressionado o botão Ligar

Lampada Desligada: Quando pressionado o botão Desligar

8.4 Ligar lâmpada através do celular utilizando Ethernet e Roteador (com resposta)

A experiência anterior foi demonstrado o acionamento à distância sem termos a certeza de que a ação foi executada. Nessa experiência iremos obter resposta no dispositivo se a Lâmpada realmente ligou e desligou conforme o comando dado.

O circuito, o layout e a programação são praticamente os mesmos da experiência 3. O que muda no circuito é a adição de um componente para receber a informação do estado da luz, informar ao microcontrolador e o mesmo enviar resposta ao dispositivo móvel.

Com isso inserimos na programação do arduino uma verificação em uma determinada porta de entrada de nível de sinal enviado pelo sensor e na programação do dispositivo móvel o complemento para receber a resposta de forma completa.

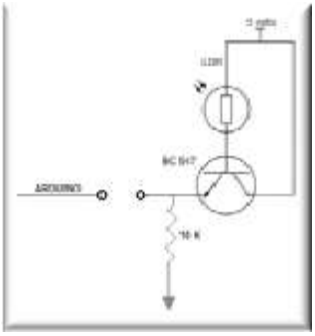
Para isso utilizaremos um sensor de luminosidade com resistência variável ou LDR figura 279 a seguir.

Figura 279 – LDR

Fonte: https://meetarduino.wordpress.com/2012/05/03/arduino-ldr-led/

O circuito que emitirá a resposta ao arduino de que a luz está realmente acesa ou apagada pode ser visto na figura 280 e seu funcionamento já foi descrito na experiência anterior a seguir.

Figura 280 - Circuito do LDR

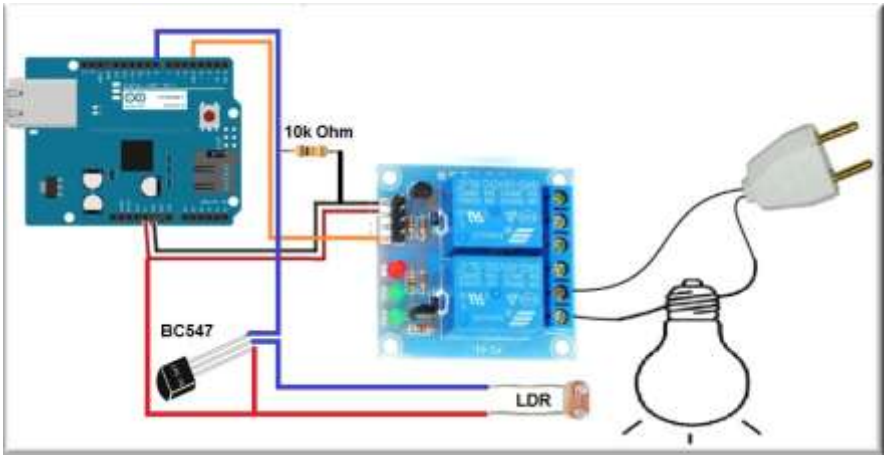


Fonte: Acervo Pessoal

8.4.1 O hardware

A montagem completa pode ser vista na figura 281 a seguir.

Figura 281 - Circuito Completo



Fonte: Acervo Pessoal

Lista de Materiais

1 Arduino Uno ou Similar
1 Rele Shield (1, 2 ou 4 reles)
1 Ethernet Shield HC-05
1 Transistor PNP 547 ou similar
1 Resistor 1k Ohm x ¼ W (Marron, Preto, Vermelho)
1 Resistor 10k Ohm x ¼ W (Marron, Preto, Laranja)
1 LDR
1 Bocal
1 Lâmpada
1 Protoboard
1 Kit Jumpers

8.4.2 Programação Arduino

```
#include <SPI.h>

#include <String.h>

#include <Ethernet.h>

byte mac[] = { 0x90, 0xA2, 0xDA, 0x00, 0x9B, 0x36 }; // Endereço Mac

byte ip[] = { 192, 168, 1, 177 }; // Endereço de Ip da sua Rede

EthernetServer server(8090); // Porta de serviço


int led = 4; // Pino onde deve ser ligado o shield de rele

String readString = String(30); // string para buscar dados de endereço

boolean statusLed = false; // Variável para o status do led
```

```
const int sensor1 = 8;


void setup() {


  Ethernet.begin(mac, ip); // Inicia o Ethernet
  pinMode(led, OUTPUT);  // Define o pino como saída - lampada
  Serial.begin(9600); // Inicia a comunicação Serial
  pinMode (sensor1, INPUT); // Define o pino como entrada - sensor
   }
void loop(){
  EthernetClient client = server.available(); // Criar uma conexão de cliente


  if (client) {
    while (client.connected())
    {
      if (client.available())
      {
        char c = client.read();
        // ler caractere por caractere vindo do HTTP
        if (readString.length() < 30)
        {
          // armazena os caracteres para string
          readString += (c);
        }
       // Se o pedido HTTP terminou
        if (c == '\n')
          {
          // vamos verificar se a lampada deve ser ligado
```

```
  // Se a string possui o texto LigarLampada
  if(readString.indexOf("LigarLampada")>=0)
  {
    // O Led vai ser ligado
    digitalWrite(led, HIGH);
    statusLampada = true;
    delay(500);
  }
  // Se a string possui o texto DesligarLed
  if(readString.indexOf("DesligarLampada")>=0)
  {
    digitalWrite(led, LOW); // A Lampada vai ser desligada
    statusLampada = false;
    delay(500);
  }
 // dados HTML de saída começando com cabeçalho padrão
 client.println("HTTP/1.1 200 OK");
 client.println("Content-Type: text/html");
 client.println();
int sensor1 = digitalRead(8);
 client.print("<font size='20'>");
 if (statusLed)
 {
  client.print("ComandoLigar");
     if (sensor1 ==0)
       {
        client.print("LuzDesligada");
        }
```

```
    else
    {
        client.print("LuzLigada");
    }
  }
    else
  {
   client.print("ComandoDesligar");
 if (sensor1 ==0)
      {
       client.print("LuzDesligada");
      }
    else
    {
    client.print("LuzLigada");
     }
  }
  readString="";//limpa string para a próxima leitura
  client.stop();// parar cliente
 }}}}}
```

8.4.3 Programação App Inventor - Alterando a mensagem da resposta da página.

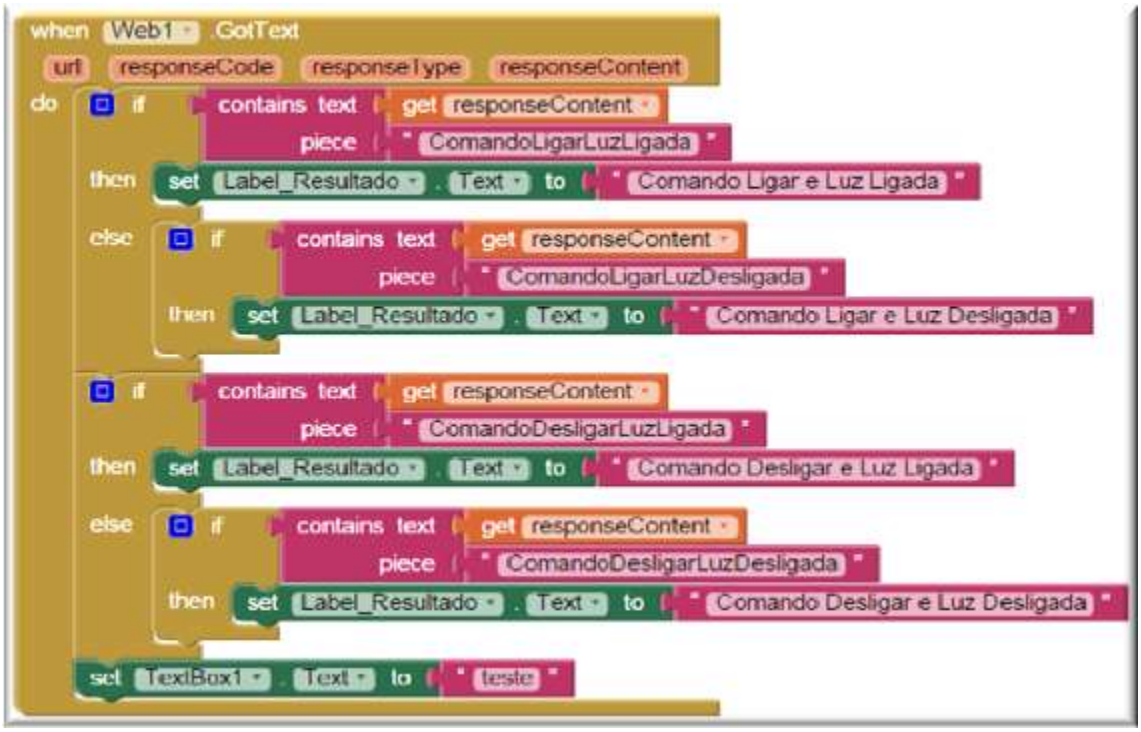
Com auxílio do circuito de realimentação baseado no LDR, a resposta no dispositivo deverá retornar o nome do comando (Ligar ou Desligar) e o estado da lâmpada (Ligada ou Desligada).

Para isso deverá ser alterado no bloco "When Web1.GotText o texto da resposta conforme figura 282 a seguir.

Alteração: set Label_Resultado.Text to:

- **Comando Ligar e Luz Ligada / Comando Ligar e Luz Desligada**
- **Comando Desligar e Luz Ligada / Comando Desligar e Luz Desligada**

Figura 282 - Alteração na programaçao App Inventor



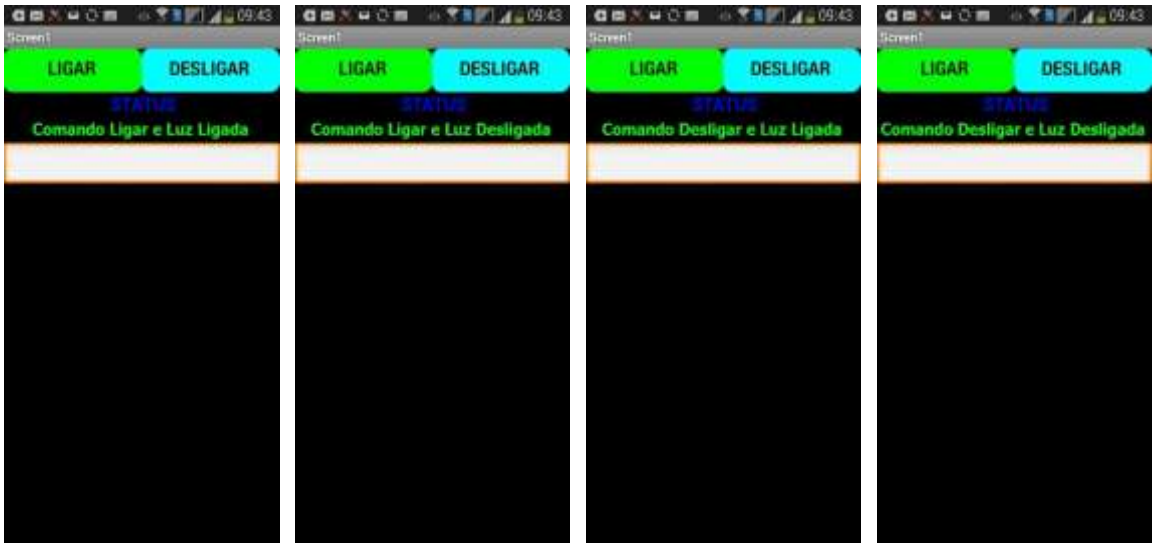
Fonte: http://appinventor.mit.edu

8.4.4 Testando

Teste a aplicação baseado no procedimento do item 1.5.

Caso tenha ocorrido tudo certo até o momento deverá aparecer a seguinte tela no dispositivo móvel.

Correto	Incorreto	Incorreto	Correto
Figura 283 - Tela1	**Figura 284 - Tela2**	**Figura 285 - Tela3**	**Figura 286 - Tela4**



Fonte: http://appinventor.mit.edu

Aguardando: Quando a conexão ainda não estiver sido estabelecida

Comando Ligar e Luz Ligada: Quando pressionado o botão Ligar e a Lâmpada Ligar

Comando Ligar e Luz Desligada: Quando pressionado o botão Ligar e a Lâmpada não Ligar

Comando Desligar e Luz Ligada: Quando pressionado o botão Desligar e a Lâmpada não Desligar

Comando Desligar e Desligada: Quando pressionado o botão Desligar e a Lâmpada Desligar

8.5 Ligar lâmpada através do celular utilizando Ethernet+Roteador+PLC

Nesta experiência iremos apenas acrescentar um item ao conjunto anteriormente apresentado para aumentar o alcance do acionamento. Trata-se de um PLC (Power Line Comunication) dispositivo que permite utilizar a rede elétrica como mídia para utilização de transmissão de dados.

Para mais detalhes sobre PLC consulte o Anexo IX.

Em casa, temos em todos os aparelhos elétricos cabos que passam pelo caminho até chegarem às tomadas. E, para completar, ainda usamos cabos de rede para conectar os PCs entre si e à internet. Isso porque nem tudo que é sem fio (*wireless*) funciona perfeitamente. Nem sempre o sinal sem fio trabalha sem falhar.

Se a opção wireless fosse perfeita, seria a ideal, já que o não uso de fios sempre é a prioridade. Na procura por melhorar a qualidade e não ter ainda mais fios foi criada a tecnologia PLC (*Power line Communication*).

Através de um adaptador conectado em sua tomada (figura 287), você transforma todo o sistema elétrico da sua casa em uma rede. Assim, cada tomada é um ponto de acesso.

Figura 287 - PLC (Power Line Comunication)

Fonte: http://www.xataka.com/otros/d-link-powerline-dhp-303-plc-pensado-para-la-alta-definicion

Quando se fala em utilizar a rede elétrica para algo que não seja abastecer eletrônicos, já pensamos que pode dar errado e queimar os aparelhos. O fato é que a energia elétrica funciona na frequência entre 50 e 60 Hz, enquanto a conexão PLC usa de 1 a 30 MHz. Dessa maneira, um sinal supostamente não deve causar interferências ao outro.

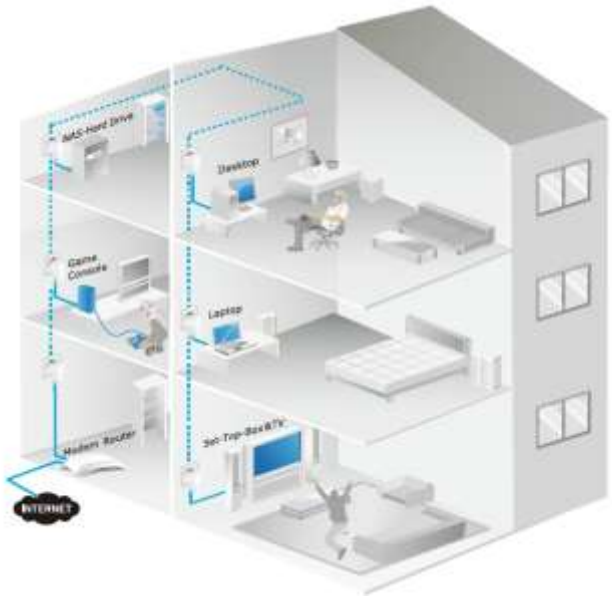
A tecnologia funciona tanto de maneira interna, com a transmissão de dados usando a rede elétrica do prédio, apartamento ou casa, quanto externa, em que é usada a rede pública de energia para transmitir. A maior vantagem é que, com o Powerline, a velocidade se mantém bastante alta e dificilmente tem quedas.

8.5.1 Como Usar

A instalação é muito fácil e acaba sendo um dos pontos mais positivos de se usar a tecnologia PLC. Tudo de que você precisa é de um adaptador (comumente chamado de *Powerline Adapter*). Você só precisa ligar o adaptador na tomada e conectar a ele o modem e/ou o roteador. Depois disso, qualquer tomada em sua casa vira um ponto de acesso de rede.

Você precisa de um adaptador, que será o distribuidor de sinal, e de um capaz de receber. Sendo assim, o segundo *Powerline Adapter* vai à outra tomada da residência, bastando então conectar um cabo de rede nele e no PC. Você pode usar quantos adaptadores quiser, ligando na tomada e conectando o cabo até o PC.

Figura 288 - PLC instalado



Fonte: https://images-na.ssl-images-amazon.com/images/G/01/aplus/detail-page/tplinkamz_home_lg.jpg

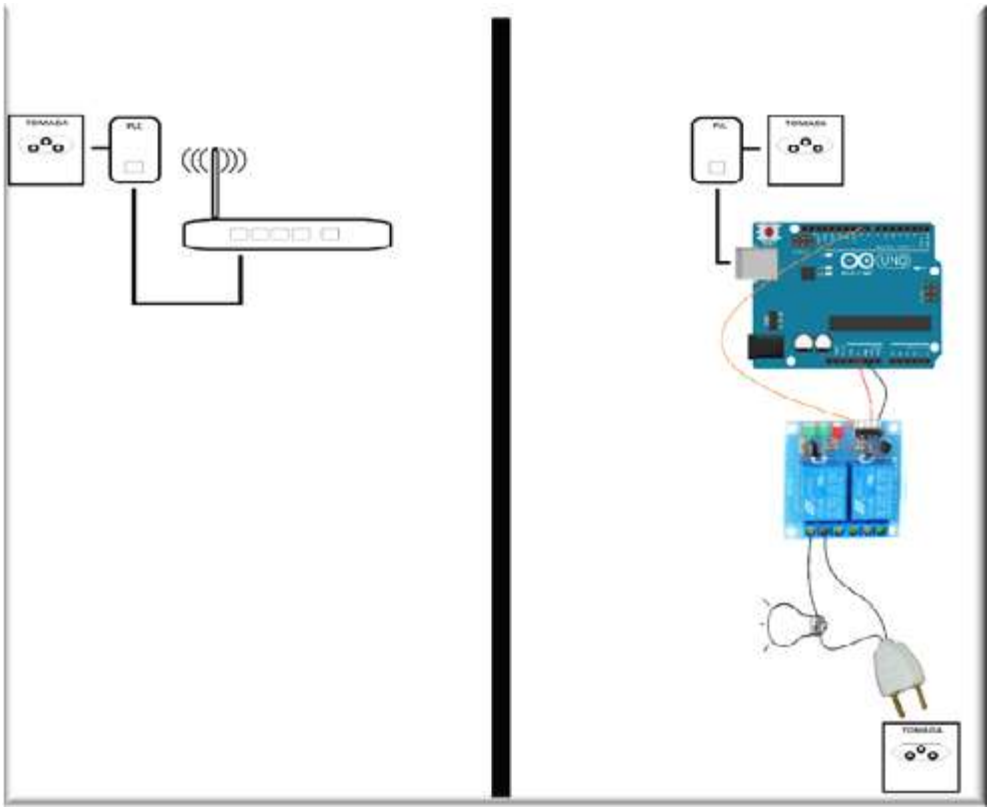
A maior desvantagem da tecnologia é o fato de a conexão PLC ser prejudicada por filtros de linha, estabilizadores e no-breaks, sendo assim, você só pode usar a tomada para o adaptador PLC. Além disso, o PLC é *half-duplex*: cada adaptador pode transmitir e receber dados através dele, mas não simultaneamente. Assim, os pontos funcionam um de cada vez.

8.5.2 Testando

A figura 289 mostra a forma de ligação do plc ao projeto. Coloque um PLC na tomada e ligue um cabo UTP com conector RJ45 do conector do PLC ao conector LAN do

roteador. Instale o outro PCL em outra tomada conforme a necessidade e ligue um cabo UTP com conector RJ45 do conector do PLC ao shield ethernet do projeto. Feito isso é só proceder conforme o item 6.3.

Figura 289 - Teste com PLC



Fonte: Acervo Pessoal

Geralmente os PLCs comercializados possuem leds indicadores de energia e atividade. Caso ligue seu PLC e nenhum led estiver ligado deverá ser verificado se na tomada existe tensão e se o PLC esteja realmente funcionando.

8.6 Comunicação direta wireless utilizando o shield Yun

Nessa experiência iremos conectar o dispositivo móvel diretamente ao hardware sem passar por um roteador mas utilizando a tecnologia sem fios. Para isso iremos utilizar um shield que contem o recurso de transmissão wireless. Existem vários modelos no mercado. Em nossa experiência utilizamos o modelo YUN por ser mais completo em termos de recursos. Vamos conhecer esse componente.

Existe no mercado um modelo de arduino chamado de Yun Arduino. A semelhança mais próxima a esse modelo seria utilizar o Arduino Leonardo + Yun Shield. Nesse contexto o Shield seria mais viável por ser mais flexível sendo compatível com outros modelos de arduino.

O shield Yun vem com o sistema Linux OpenWRT embutido. Além de possibilitar a comunicação via cabo e wireless, esse shield disponibiliza uma porta USB para acesso a internet via 3G Dongle ou armazenamento.

8.6.1 Especificações

- Processor de 400MHz, 24K MIPS
- Flash de 16MBytes
- Memória de 64MBytes
- Tensão de Entrada de 4.75v a 23v via Arduino VIN pin
- 1 x 10M/100M Conector RJ45
- 150M WiFi 802.11 b/g/n
- Antena Externa para conexão I-Pex
- 1 x Conexão USB 2.0 usada para armazenamento ou conexão 3G
- 1 x Botão de Reset
- Compatível com 3.3v ou 5v I/O Arduino

8.6.2 Características

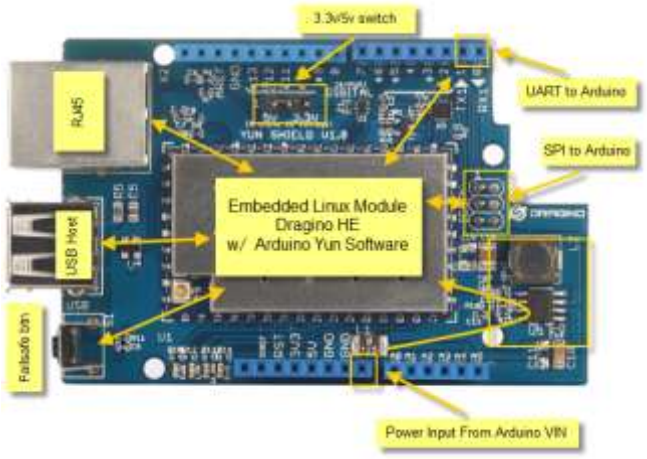
- Open Source Linux (OpenWrt) embutido (mini sistema operacional)
- Baixo consumo de energia
- Compatível com Arduino IDE 1.5.4 ou posterior
- Gerenciável por GUI Web, SSH via LAN ou Wi-Fi
- Software atualizável via rede
- Servidor web integrado
- Conexão de internet via porta LAN, Wi-Fi ou 3G Dongle.
- Flash USB Suporte para fornecer armazenamento para projetos Arduino.

- Compatível com Arduino Leonardo, Uno, Duemilanove, Diecimila, o Mega

8.6.3 Estrutura física

A estrutura física do Yun Shield pode ser vista na figura 290 a seguir.

Figura 290 - Yun Shield – Estrutura Física

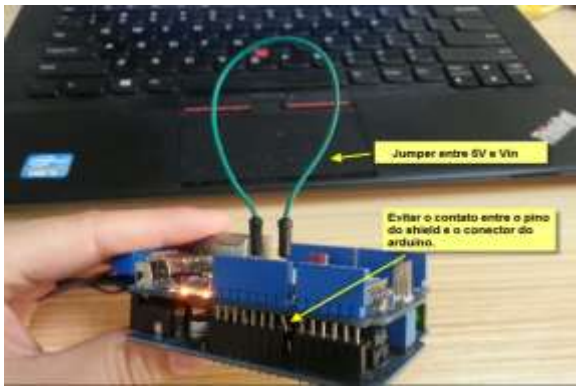


Fonte: http://www.embarcados.com.br/shield-arduino-yun/

8.6.4 Requisitos de Energia

Caso o conjunto (arduino + shield) for alimentado pela porta USB é recomendável utilizar um jumper entre os pinos 5V e Vin do Shield e evitar que o pino Vin do Shield tenha contato com o conector Vin do Arduino. Para isso deve-se entortar o pino Vin do Shield conforme figura 291 a seguir. Isso porque o módulo requer 200mA de corrente operando em pleno processamento o que pode ocorrer superaquecimento no Arduino.

Figura 291 - Yun Shield



http://wiki.dragino.com/index.php?title=File:Yun_Shield_powered_by_USB.png

8.6.5 Conexão do Shield ao Arduino

Se a arduino utilizado for o modelo Leonardo, basta conectar o shield ao arduino sem nenhuma configuração adicional, pois o IDE já irá reconhecer como Arduino Yun.

Caso seja o modelo Arduino Uno deverá ser aplicado um jumper conforme mostrado na figura 292 a seguir. Esse procedimento desabilita o Arduino para trabalhar como dispositivo USB passando a operar via Rede, uma vez que este é o diferencial deste shield.

Figura 292 - Yun Shield – Jumper Serial / IP

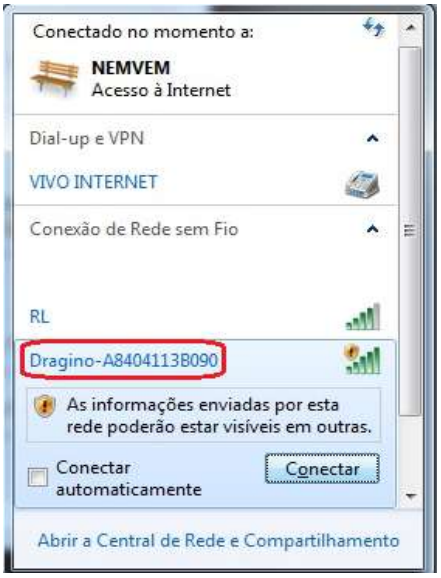
http://wiki.dragino.com/index.php?title=Main_Page

Para mais detalhes sobre utilização do Yun Shield com outros modelos consulte o anexo 10.

8.6.6 Preparando o ambiente para o reconhecimento

Uma vez que esteja encaixado o shield e o arduino, conecte a fonte de energia do arduino e ligue na tomada. Em seguida pesquise pela rede Wireless disponível cujo nome aparecerá Dragino mais uma sequência de números conforme figura a seguir.

Figura 293 - Rede Wireless Dragino

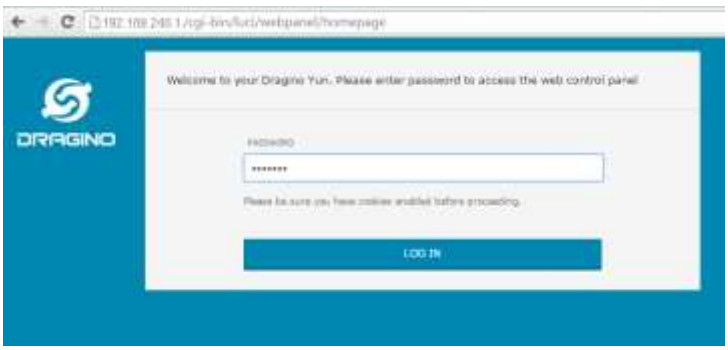


Fonte: Acervo Pessoal

A senha padrão exigida neste momento é "***dragino***".

Depois de conectado as configurações poderão via web poderão ser exibidas utilizando-se um navegador e digitando o seguinte endereço na barra de endereços: 192.168.240.1 e confirmando a senha "dragino" exigida na aplicação web.

Figura 294 - Senha para Web Acesso



Fonte: Acervo Pessoal

8.6.7 Adaptando a IDE

Para que aparece os modelos Arduino Uno – Dragino Yun, Arduino Leonardo - Dragino Yun ou Arduino Mega 2560 – Dragino Yun devemos alimentar a IDE com essas informações sobre o hardware. Para isso deve-se acessar o endereço a seguir e baixar os arquivos necessários.

https://github.com/dragino/modules/tree/f91562ec33e8e4662ec23e7c88ab051f3d3bbe15

Clique no botão Download Zip conforme figura a seguir

Figura 295 - Download do Hardware

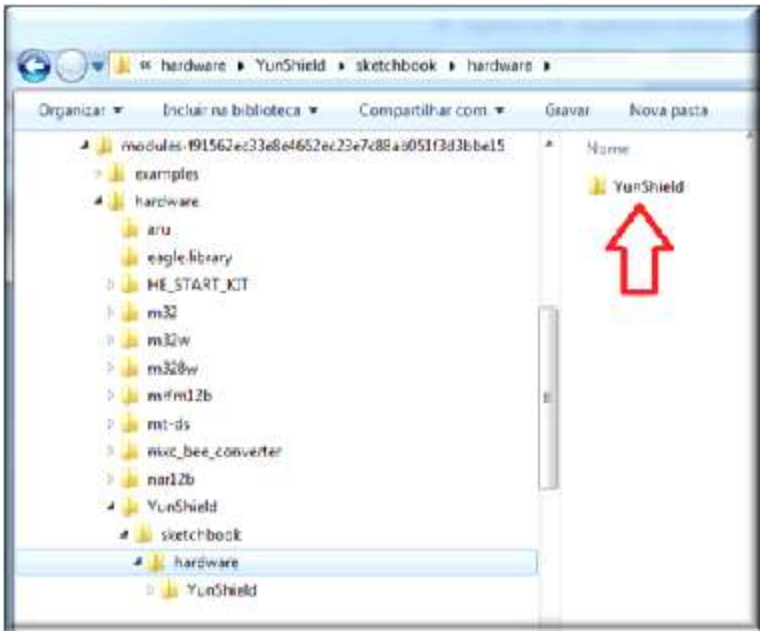
Fonte: https://github.com/dragino/modules/tree/f91562ec33e8e4662ec23e7c88ab051f3d3bbe15

Com isso será baixado um arquivo compactado com vários arquivos inclusos.

Descompacte e será criado uma pasta com o mesmo nome do arquivo zip o qual geralmente é “modulesxxxxxxxx” onde x representa números.

Abra a pasta descompactada e navegue até a seguinte pasta hardware\YunShield\sketchbook\hardware conforme figura a seguir e copie a pasta YunShield.

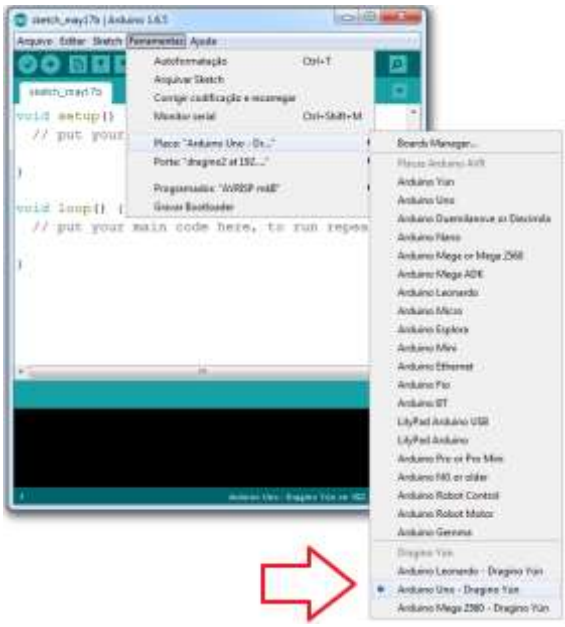
Figura 296 - Pasta do Hardware



Fonte: Acervo Pessoal

Abra a pasta do IDE Arduino em seguida abra a pasta Hardware. Cole a pasta YunShield dentro da pasta Hardware. Esse procedimento fará aparecer as opções Arduino Uno – Dragino Yun, Arduino Leonardo - Dragino Yun ou Arduino Mega 2560 – Dragino Yun na lista de modelos do IDE.

Figura 297 - Modelo do conjunto

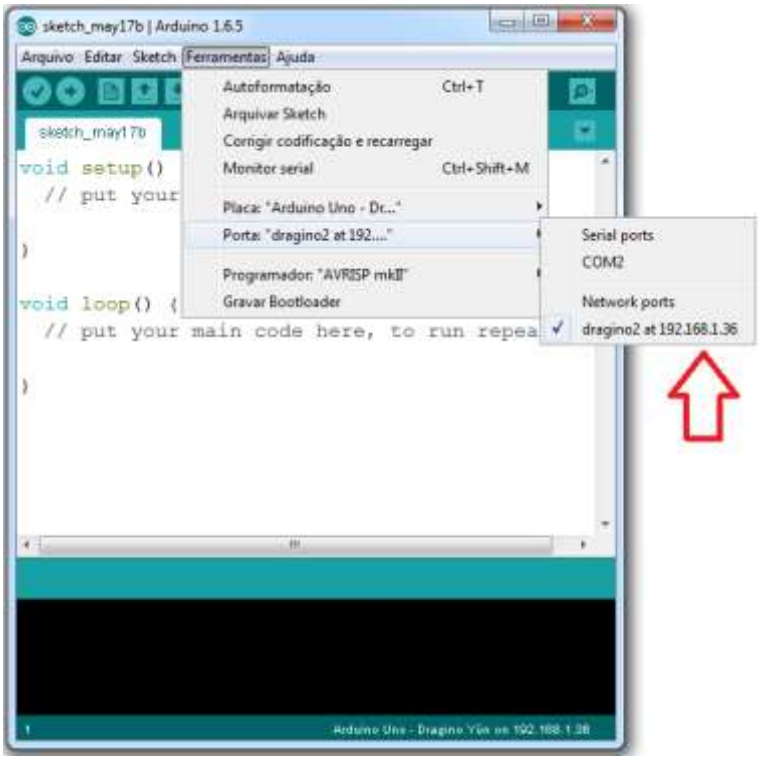


Fonte: Acervo Pessoal

Estando o computador e o Arduino+YunShield na mesma rede selecione no menu Ferramentas(Tools)→Placa(Board) o modelo desejado. Em nossa experiência utilizamos o modelo Arduino Uno – Dragino Yun.

Em seguida vá ao menu Ferramentas(Portas) e veja se associou o endereço IP correspondente ao conjunto conforme figura a seguir.

Figura 298 - Porta do Conjunto



Fonte: Acervo Pessoal

Obs.:

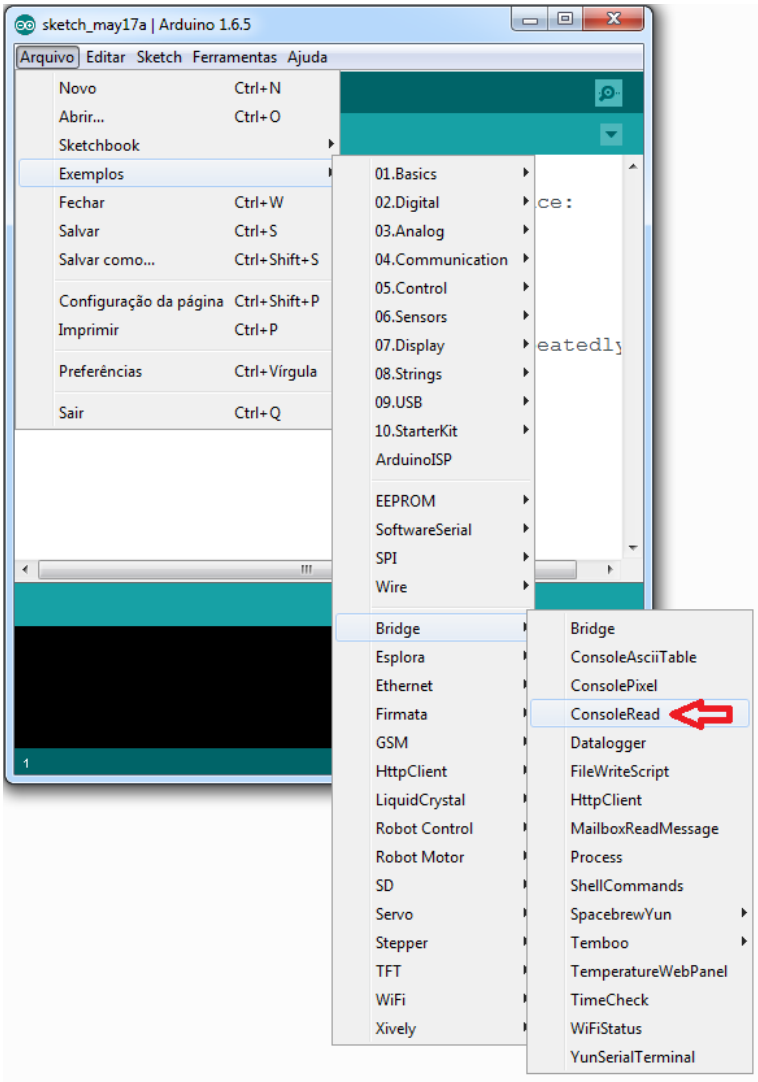
- Se a conexão for feita via wireless esse endereço IP deverá ser 192.168.240.1
- Se a conexão for feita via cabo esse endereço assumirá o range da rede como por exemplo 192.168.1.x onde x pode ser qualquer número entre 1 e 255.

8.6.8 Carregando um exemplo

Abra a IDE e selecione menu

Arquivo(File)→Exemplos(Exemples)→Bridge→ConsoleRead

Figura 299- Exemplo



Fonte: Acervo Pessoal

Isso irá carregar o programa de exemplo para uma comunicação entre o computador e o Shield via caboa ou wireless.

8.6.9 Programação

A programação pode ser vista a seguir:

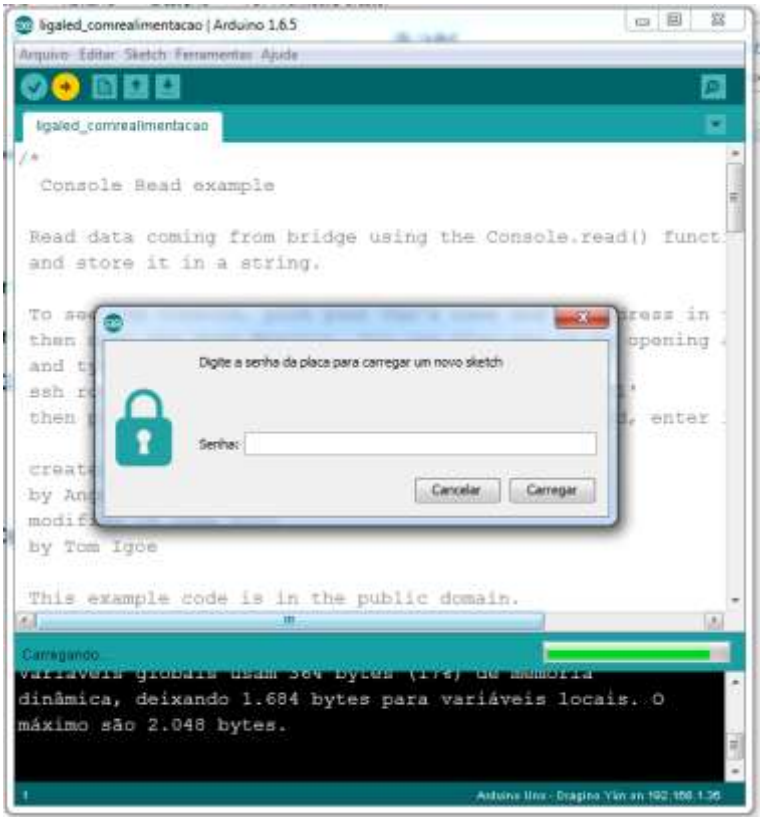
```
#include <Console.h>
String name;
void setup() {
  Bridge.begin(); // Inicializa o console e aguarda a abertura da porta:
  Console.begin();
    while (!Console); // Aguardando a conexão com a porta
  Console.println("Hi, what's your name?");
}
void loop() {
  if (Console.available() > 0) {
    char c = Console.read(); // lê o próximo caractere recebido
    // Verifica se o caractere da nova linha é o último caractere recebido
    if (c == '\n') {      //Mostra o texto com o nome recebido
      Console.print("Hi ");
      Console.print(name);
      Console.println("! Nice to meet you!");
      Console.println();      // pede novamente o novo nome e apaga o nome antigo
      Console.println("Hi, what's your name?");
      name = "";  // limpa o nome
    }
    else {              // se o buffer do Cosole.read() estiver vazio retorna -1
      name += c; // anexa o caractere lido para a nova sequência de nome
    }
  } else {
    delay(100);
  }}
```

Este exemplo serve para mostrar o envio e recebimento de mensagens entre computador e shield através do serial monitor da IDE.

8.6.10 Testando

Faça o upload do programa. Deverá aparecer a seguinte mensagem pedindo a senha para a conexão ao Shield.

Figura 300 - Acesso Serial Monitor



Fonte: Acervo Pessoal

Se correr tudo bem aparecerá a seguinte mensagem na parte inferior do IDE.

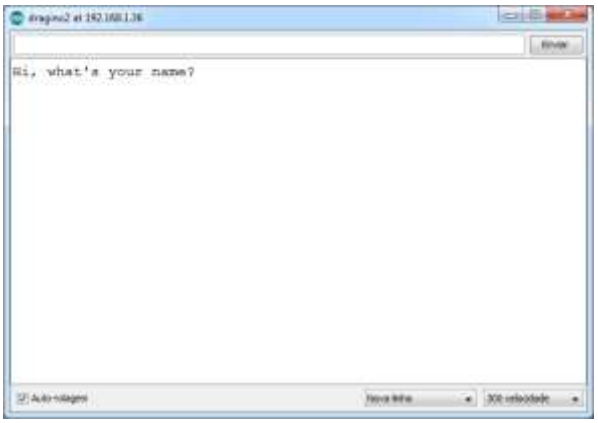
O sketch usa 7.600 bytes (23%) de espaço de armazenamento para programas. O máximo são 32.256 bytes. Variáveis globais usam 364 bytes (17%) de memória dinâmica, deixando 1.684 bytes para variáveis locais. O máximo são 2.048 bytes. avrdude: AVR device initialized and ready to accept instructions Reading |	| 100% 0.00s avrdude: 1 bytes of hfuse written avrdude: verifying hfuse memory against 0xDE: Reading | ## | 100% 0.00s avrdude: verifying ... avrdude: 1 bytes of hfuse verified

```
################################################## |
100% 0.01s

avrdude: Device signature = 0x1e950f

avrdude: NOTE: "flash" memory has been specified, an erase
cycle will be performed

        To disable this feature, specify the -D option.

avrdude: erasing chip

avrdude: reading input file "0xFF"

avrdude: writing lfuse (1 bytes):

Writing |
################################################## |
100% 0.00s

avrdude: 1 bytes of lfuse written

avrdude: verifying lfuse memory against 0xFF:

avrdude: load data lfuse data from input file 0xFF:

avrdude: input file 0xFF contains 1 bytes

avrdude: reading on-chip lfuse data:

Reading |
################################################## |
100% 0.00s

avrdude: verifying ...

avrdude: 1 bytes of lfuse verified

avrdude: reading input file "0xDE"

avrdude: writing hfuse (1 bytes):

Writing |
##################################################

avrdude: load data hfuse data from input file 0xDE:

avrdude: input file 0xDE contains 1 bytes

avrdude: reading on-chip hfuse data:
```

```
avrdude: reading input file "0x05"

avrdude: writing efuse (1 bytes):

Writing |
################################################## |
100% 0.00s

avrdude: 1 bytes of efuse written

avrdude: verifying efuse memory against 0x05:

avrdude: load data efuse data from input file 0x05:

avrdude: input file 0x05 contains 1 bytes

avrdude: reading on-chip efuse data:

Reading |
################################################## |
100% 0.00s

avrdude: verifying ...

avrdude: 1 bytes of efuse verified

avrdude: reading input file "/tmp/sketch.hex"

avrdude: writing flash (32768 bytes):

Writing |
################################################## |
100% 4.39s

avrdude: 32768 bytes of flash written

avrdude: verifying flash memory against /tmp/sketch.hex:

avrdude: load data flash data from input file /tmp/sketch.hex:

avrdude: input file /tmp/sketch.hex contains 32768 bytes

avrdude: reading on-chip flash data:

Reading |
################################################## |
100% 4.05s

avrdude: verifying ...

avrdude: 32768 bytes of flash verified

avrdude done.  Thank you.
```

Vá ao menu Ferramentas (Tools) e clique na opção Monitor Serial (Serial Monitor). Abrirá uma janela conforme figura a seguir.

Figura 301 – Recebimento



Fonte: Acervo Pessoal

Aparecerão os dizeres “Oi, qual é o seu nome”. Isso já significa que a informação veio do conjunto Arduino+YunShield para o seu computador.

Agora verifique se o envio também está funcionando. Para isso digite seu nome no campo superior desta janela e clique em Enviar (Send) conforme figura a seguir.

Figura 302 - Envio

Fonte: Acervo Pessoal

Então o conjunto Arduino+YunShield recebe a informação e se comunica novamente com os dizeres “Oi Fulano! Prazer em conhece-lo!” e em seguida repete a frase de apresentação.

8.7 Controlar e monitorar um processo utilizando Yun Shield

Nessa experiência iremos utilizar um recurso importante desse Shield. Trata-se da função Servidor Web.

Servidor web pode ser um programa de computador responsável por aceitar pedidos de clientes, geralmente os navegadores, e servi-los com respostas, incluindo opcionalmente dados, que geralmente são páginas web, tais como documentos com objetos embutidos (imagens, etc.) ou um computador que executa um programa que provê a funcionalidade descrita anteriormente. Esses documentos são criados no formato HTML. Esse formato **HTML** (abreviação para a expressão inglesa ***HyperText Markup Language***, que significa ***Linguagem de Marcação de Hipertexto***) é uma linguagem de marcação utilizada na construção de páginas na Web. Documentos HTML podem ser interpretados por navegadores.

Como já foi visto o Shield Yun possui um sistema operacional que gerencia todo o hardware. Pois bem, utilizaremos a porta de comunicação USB do shield para hospedar (armazenar) a pagina web.

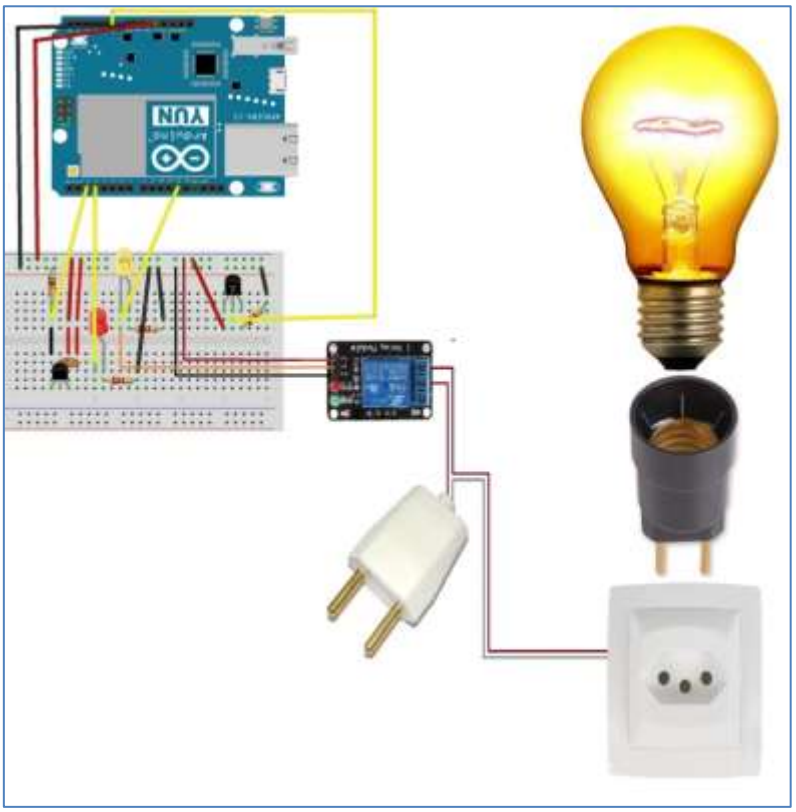
8.7.1 O Hardware

A experiência contará com:

- Acionamento de uma lâmpada e um led;
- Acionamento de um led;
- Monitoramento da lâmpada;
- Monitoramento de temperatura do processo.

Monte o circuito conforme o diagrama a seguir

Figura 303 - Diagrama ligação Yun Shield



Fonte: Acervo Pessoal

Para monitorar a temperatura usaremos um sensor de temperatura LM35. Esse sensor fornece sinais analógicos variantes de acordo com a temperatura do processo.

Para monitorar o estado da lâmpada usaremos um LDR.

Tanto o monitoramento do estado da lâmpada como da temperatura oferecem resposta em tempo real.

Lista de Materiais

01	Arduino Uno + Cabo USB 2.0
01	Fonte 12V Chaveada 100-240VAC – P4
20	Jumpers Macho-Macho
20	Jumpers Macho-Fêmea
02	Módulo Rele 5V – 1 Canal
01	Ýun Shield c/ Wifi, Ethernet, USB

02	Resistor 10 KOhm x ¼ W
01	Resistor 220Rohm x ¼ W
01	Sensor de Temperatura LM35
01	Chave Tactil 12x12x4,3mm 4 Terminais
01	Transistor NPN - BC547
01	Led Difuso 10mm
01	Protoboard 170 Furos
02	Tomada Externa 10A Retangular Perplex
01	Plugue Macho Tomada 2P/10 Amperes
01	Bocal Termoplástico Adaptável com Plugue Macho
01	Lâmpada Incandescente 60W /220V
02	Metros Fio 1,5 mm Flexivel Branco

8.7.2 A página web

A programação da página web não será explicada neste material. Utilizaremos um exemplo disponível para download cujo endereço é informado no item 8.7.5 deste material. Sendo assim não iremos cria-la, mantendo integra a metodologia aplicada neste material. Ela poderá ser vista no Anexo 11

Para mais detalhes sobre programação da página web consulte o anexo 10.

8.7.3 Programação Arduino

```
#include < Bridge .h>
#include < YunServer .h>
#include < YunClient .h>
int dac = 0;
int DigitalPin[] = {2, 12, 13};
int DacPin = 3;
YunServer server;
void setup() {
pinMode (2,INPUT);
pinMode (4,INPUT);
pinMode (12,OUTPUT );
pinMode (13,OUTPUT );
digitalWrite (13, LOW);
```

```
Bridge .begin();
digitalWrite (13, HIGH);
server.listenOnLocalhost();
server. begin();
}
void loop() {
YunClient client = server.accept ();
if (client) {
process(client);
client. stop();
}
delay(50);
}
void process(YunClient client) {
String command = client.readStringUntil ('/');
if (command == "digital" ) {
digitalCommand(client);
}
if (command == "dac") {
dacCommand(client);
}
if (command == "status" ) {
statusCommand(client);
} }
void digitalCommand(YunClient client) {
int pin, value;
pin = client.parseInt ();
if (client.read() == '/') {
value = client.parseInt ();
digitalWrite (pin, value);
}
else {
value = digitalRead (pin);
}
client. print(F("digital," ));
client. print(pin);
client. print(F(","));
client. println (value);
}
void dacCommand(YunClient client) {
int pin, value;
pin = client.parseInt ();
if (client.read() == '/') {
value = client.parseInt ();
dac = value;
analogWrite (pin, value);
}
else {
value = dac;
}
client. print(F("dac," ));
client. print(pin);
client. print(F(","));
client. println (value);
}
```

```
void statusCommand(YunClient client) {
int pin, value;
client. print(F("status" ));
for (int thisPin = 0; thisPin < 3; thisPin++) {
pin = DigitalPin[thisPin];
value = digitalRead (pin);
client. print(F("#"));
client. print(pin);
client. print(F("="));
client. print(value);
} {
pin = DacPin;
value = dac;
client. print(F("#"));
client. print(pin);
client. print(F("="));
client. print(value);
} {
value = analogRead (0);
// convert the reading to millivolts:
float voltage = value * (5000/ 1024);
// convert the millivolts to temperature celsius:
float temperature = (voltage - 500)/10;
client. print(F("#A0"));
client. print(F("="));
client. print(temperature);
}
client. println ("");
}
```

8.7.4 Programação App Inventor

O software do App Inventor será bem simples pois trata-se apenas de botões que se comportarão como links para abrir a página web. Como o acesso ao hardware poderá ser feito de duas formas (diretamente na rede do wifi do Shield Yun ou através da rede LAN local por intermédio de um roteador) o aplicativo terá dos botões para direcionar o tipo de acesso desejado.

A programação dos botões é mostrada a seguir.

Figura 304 - Programação App Inventor

Fonte: http://appinventor.mit.edu

Veja que pelo acesso direto o endereço aponta para 192.168.240.1 pois é o endereço disponível pelo shield através de sua distribuição WiFi. Pelo acesso local (roteador) o shield estará ligado ao roteador através de um cabo de rede utilizando sua porta de comunicação ethernet e neste caso o endereço 192.168.1.99 aponta para o hardware. Esse endereço pode ser alterado de acordo com a necessidade. Para isso faça o procedimento descrito no item 7.6.5 deste material e altere o endereço IP do shield.

8.7.5 Testando

1º Baixe os arquivos da página web no seguinte endereço:

→ https://dl.dropboxusercontent.com/u/92299797/INOand HTML for AJAX.zip

2º Extraia o conteudo baixado

3º - Copie o conteúdo extraído

Figura 305 - Arquivos extraídos

Fonte: Acervo Pessoal

2º - Cole na raiz de um Pendrive.

3º - Abra o programa YUN.ino dentro da pasta “programação Arduino”

4º - Faça upload para o Arduino.

5º - Desconecte o cabo USB do arduino e em seguida insira o pendrive

6º - Ligue o arduino.

7º - Instale no dispositivo móvel o software YunDragino.apk localizado na pasta “Instalador APK”.

8º - Cerifique-se que seu dispositivo esteja com Wi-Fi ativado.

9º - Abra o software YUN no dispositivo móvel e escolha uma das opções:

- Acesso Direto: Quando seu dispositivo móvel estiver conectado diretamente à rede “Dragino-A8404113B090” cuja senha é dragino

- **Roteador**: Quando seu dispositivo móvel e o Arduino+YunShield estiverem na mesma rede.

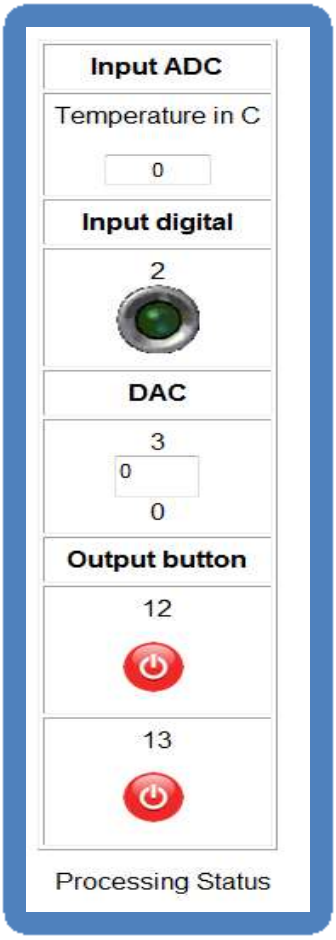
Figura 306 – Aplicativo

Fonte: Acervo Pessoal

10º - Selecione um navegador de sua preferência e acesse a página da aplicação. É importante que o navegador esteja com o histórico vazio.

11º - A página será apresentada com as características mostradas na figura a seguir:

Figura 307 - Layout da pagina web



Fonte: Acervo Pessoal

Pode ser visto a variação da temperatura em “Input ADC”

Em “Input Digital” verifica-se o monitoramento acionamento do pino 12 ou seja, ao clicar para ligar, se a luz tornar-se verde claro significa que o acionamento obteve êxito.

Em “Output Button” opções 12 e 13 faz-se o acionamento. Vale lembrar que apena o botão referente ao 12 oferece resposta, o botão do pino 13 não oferece resposta pois aciona apenas o led. Dessa forma se a lâmpada em algum momento se apagar a informação será mostrada na página.

9 Conclusões

A satisfação em poder entrar na faculdade e já no primeiro perído desenvolver produtos úteis ao mercado, superou as expectativas dos alunos, dispertando assim interesses profissionais uma vez que aprenderam a forma correta de desenvolver projetos com segurança e eficiência.

Verificou-se elevado interesse dos alunos com a utilização desse método de ensino. O laboratório foi disponibilizado para estudos e práticas de alunos voluntários a trabalhar com as tecnologias fora do horário das aulas. O que se obteve com isso foi um laboratório bastante frequentado e produtivo de onde saíram projetos e planos de aula para replicação do conhecimento.

O fato de que o aluno não necessite abandonar seu dispositivo móvel para estudar além de ser um atrativo é também um vínculo entre o aluno e o curso, pois o aparelho torna-se uma ferramenta. Neste cenário pode-se observar a evolução no processo construtivo ao invés de força-lo a deixar de usar o dispositivo, caracterizando assim uma reconstrução destrutiva.

A escolha da plataforma de desenvolvimento de aplicativos mobiles **appInventor 2** associada ao microcontrolador Arduíno e respectivos Shields wi-fi mostraram satisfazer e efetivar os objetivos almejados com a pesquisa e consequente implementação, trazendo resultados positivos, além dos esperados, já na primeira turma piloto. Um grupo formado por três alunos da primeira turma piloto, com dois períodos concretizados e mais um do primeiro período, competiram na Startup Weekend 2016 promovida pelo grupo Algar (figura 308), considerado um dos maiores eventos de startups do mundo, segundo a Exame (http://exame.abril.com.br/negocios/dino/noticias/uberlandia-mg-vai-receber-o-maior-evento-de-startups-do-mundo.shtml) concorrendo com profissionais já formados e diversos participantes com graus de formação mais elevado, obtiveram, por mérito próprio, sem a presença do coordenador e orientador do projeto, o segundo lugar com votação dos jurados e primeiro dos monitores do evento, tendo sido cobiçados por empresas que participaram do evento, abrindo oportunidades que, antes, demorariam alguns anos para terem.

O uso destes recursos como fonte de motivação e introdução do aluno a uma visão mais próxima do mercado de trabalho, quanto às habilidades e competências exigidas atualmente, foram ministradas em aulas complementares na disciplina já existente de Introdução à Engenharia de Computação, e, pelo sucesso obtido, uma nova ementa e conteúdo programático para a mesma foi elaborada e aprovada para início oficial no segundo semestre de 2016 na Faculdade de Engenharia Elétrica da Universidade Federal de Uberlândia e que pode ser comprovado através do Anexo XII.

Figura 308 - Alunos UFU no 2º Startup Weekend Uberlândia

Fonte: Acervo Pessoal

10 Referências

Ribeiro, L.R.C. (2008). Aprendizagem Baseada Em Problemas: PBL Uma Experiência No Ensino Superior. Ed. Edufscar, 1ª. Edição, 151p. https://doi.org/10.7476/9788576002970

Camargo Jr, H.; Lima, L. V.; Lima, Sandra Fernades de Oliveira; Pereira, Adriano Alves,; Pinheiro, Alan Petrônio. (2006). A method for preparing experts in computer engineering subjects. World Congress on Computer Science, Engineering and Technology Education. Santos: COPEC.

Ferreira, Daniela Carvalho Monteiro; Lima, Luciano Vieira; Camargo, Hélcio Júnior; Schiovato, Nayara da Silva Costa. (2014). Mapas de conhecimento estruturado: proposta de uma nova abordagem metodológica de ensino e aprendizagem. Cascavel: Educare.

Zanetti, H.A.P.;Oliveira, C.L.V. (2015). Arduíno Descomplicado. Ed. Erica|Saraiva.

McRoberts, M. (2015). Arduino Básico. Ed. Novatec. 2ª. Edição. P 512.

Costa, Nayara da Silva; Dias, Daniel Cardoso; Lima, L. V. (2009). Projeto e implementação de recursos didáticos multimidia interativos para melhoria do ensino de disciplinas de engenharia de computação. Buenos Aires: COPEC.

Lima, L.V; Costa, N. S.; Lima. T.O (2013). Leitores de Livro Digital: Passivos e Ativos, Uso e Aplicação. Congresso Nacional dos Bibliotecários UFU -2013

Reino, L.R.A.C.; Domínguez, A.H.;Freias Jr, O.G.F.; Carvalho, V.; Barros, P. A.M. e Braga (2015). Análise das Causas da Evasão na Educação a Distância em uma Instituição Federal de Ensino Superior. SBIE 2015.
https://doi.org/10.5753/cbie.sbie.2015.91

Braga, M.; Takimoto, T.; Silva, D.; Freitas Jr.,O.G. e Barros,P.A.M. (2015). Estrategias para Incentivar la Participación de Alumnos en Educación a Distancia. SBIE 2015.
https://doi.org/10.5753/cbie.sbie.2015.101

Resistores:
Disponível em: <http://fisicacomabud.com.br/aulas%20de%20f%C3%ADsica/aula-eletricidade/Resistencia%20eletrica%20e%20Resistores.pdf>.
Acessado em: 19 de set. 2016.

O que é resistência elétrica?
Disponivel em:<https://www.mundodaeletrica.com.br/o-que-e-resistencia-eletrica/>
Acessado em: 19 de set. 2016.

Definição do transistor.
Disponivel em: < https://webautomacaoindustrial.blogspot.com.br/2016/05/confira-aqui-funcao-e-evolucao-dos.html>
Acessado em: 19 de set. 2016.

Arduino.
Disponível em: < http://playground.arduino.cc/Portugues/HomePage>
Acessado em: 19 de set. 2016.

O que é simulação de processos/sistemas?
Disponivel: <http://www.erlang.com.br/simulacao.asp>
Acessado em: 19 de set. 2016.

Simulador de Arduino: Virtual Breadboard.
Disponível em: <http://www.embarcados.com.br/simulador-de-arduino-virtual-breadboard/>
Acessado em: 19 de set. 2016.

Proteus.
Disponivel em: <http://garagemdaeletronica.blogspot.com.br/2014/05/proteus-77-baixar.html>
Acessado em: 19 de set. 2016.

Google App Inventor
Disponível em: http://techtudo.com.br/tudo-sobre/google-app-inventor.html
Acessado em: 19 de set. 2016.

Sistemas Cliente/Servidor
Disponível em: http://penta2.ufrgs.br/redes296/cliente_ser/tutoria_.htm
Acessado em: 19 de set. 2016.

11 Anexos

11.1 ANEXO I - Modelos Comuns de Arduinos e Shields

<table>
<tr><th>Arduino</th><th>Especificações Técnicas</th></tr>
<tr><td>ARDUINO UNO
  </td><td>
<table>
<tr><td>Microcontroller</td><td>ATmega328</td></tr>
<tr><td>Operating Voltage</td><td>5V</td></tr>
<tr><td>Input Voltage (recommended)</td><td>7-12V</td></tr>
<tr><td>Input Voltage (limits)</td><td>6-20V</td></tr>
<tr><td>Digital I/O Pins</td><td>14 (of which 6 provide PWM output)</td></tr>
<tr><td>Analog Input Pins</td><td>6</td></tr>
<tr><td>DC Current per I/O Pin</td><td>40 mA</td></tr>
<tr><td>DC Current for 3.3V Pin</td><td>50 mA</td></tr>
<tr><td>Flash Memory</td><td>32 KB (ATmega328) of which 0.5 KB used by bootloader</td></tr>
<tr><td>SRAM</td><td>2 KB (ATmega328)</td></tr>
<tr><td>EEPROM</td><td>1 KB (ATmega328)</td></tr>
<tr><td>Clock Speed</td><td>16 MHz</td></tr>
</table>
</td></tr>
<tr><td>ARDUINO LEONARDO
  </td><td>
<table>
<tr><td>Microcontroller</td><td>ATmega32u4</td></tr>
<tr><td>Operating Voltage</td><td>5V</td></tr>
<tr><td>Input Voltage (recommended)</td><td>7-12V</td></tr>
<tr><td>Input Voltage (limits)</td><td>6-20V</td></tr>
<tr><td>Digital I/O Pins</td><td>20</td></tr>
<tr><td>PWM Channels</td><td>7</td></tr>
<tr><td>Analog Input Channels</td><td>12</td></tr>
<tr><td>DC Current per I/O Pin</td><td>40 mA</td></tr>
<tr><td>DC Current for 3.3V Pin</td><td>50 mA</td></tr>
<tr><td>Flash Memory</td><td>32 KB (ATmega32u4) of which 4 KB used by bootloader</td></tr>
<tr><td>SRAM</td><td>2.5 KB (ATmega32u4)</td></tr>
<tr><td>EEPROM</td><td>1 KB (ATmega32u4)</td></tr>
<tr><td>Clock Speed</td><td>16 MHz</td></tr>
</table>
</td></tr>
<tr><td>ARDUINO DUE</td><td>
<table>
<tr><td>Microcontroller</td><td>AT91SAM3X8E</td></tr>
<tr><td>Operating Voltage</td><td>3.3V</td></tr>
<tr><td>Input Voltage (recommended)</td><td>7-12V</td></tr>
<tr><td>Input Voltage (limits)</td><td>6-16V</td></tr>
<tr><td>Digital I/O Pins</td><td>54 (of which 12 provide PWM output)</td></tr>
<tr><td>Analog Input Pins</td><td>12</td></tr>
<tr><td>Analog Outputs Pins</td><td>2 (DAC)</td></tr>
<tr><td>Total DC Output Current</td><td>130 mA</td></tr>
</table>
</td></tr>
</table>

on all I/O lines	
DC Current for 3.3V Pin	800 mA
DC Current for 5V Pin	800 mA
Flash Memory	512 KB all available for the user applications
SRAM	96 KB (two banks: 64KB and 32KB)
Clock Speed	84 MHz
Length	101.52 mm
Width	53.3 mm
Weight	36 g

ARDUINO YUN

Microcontroller	ATmega32u4
Operating Voltage	5V
Input Voltage	5V
Digital I/O Pins	20
PWM Channels	7
Analog Input Channels	12
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (of which 4 KB used by bootloader)
SRAM	2.5 KB
EEPROM	1 KB
Clock Speed	16 MHz
Linux microprocessor	
Processor	Atheros AR9331
Architecture	MIPS @400MHz
Operating Voltage	3.3V
Ethernet	IEEE 802.3 10/100Mbit/s
WiFi	IEEE 802.11b/g/n
USB Type-A	2.0 Host
Card Reader	Micro-SD only
RAM	64 MB DDR2
Flash Memory	16 MB
PoE compatible 802.3af card support (see the note below)	
Length	73 mm
Width	53 mm
Weight	32 g

ARDUINO TRE

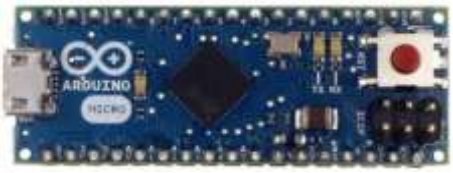
Microcontroller	Atmel ATmega32u4
Clock Speed	16 MHz
Flash Memory	32 KB (ATmega32u4)
SRAM	2.5 KB (ATmega32u4)
EEPROM	1 KB (ATmega32u4)
Digital I/O Pins (5V logic)	14
PWM Channels (5V logic)	7
Analog Input Channels	6 (plus 6 multiplexed on 6 digital pins)
Processor	Texas Instrument Sitara AM3359AZCZ100 (ARM Cortex-A8)
Clock Speed	1 GHz
SRAM	DDR3L 512 MB RAM

Networking	Ethernet 10/100
USB port	1 USB 2.0 device port, 4 USB 2.0 host ports
Video	HDMI (1920x1080)
Audio	HDMI, stereo analog audio input and output
Digital I/O Pins (3.3V logic)	23
PWM Channels (3.3V logic)	4
MicroSD card	
Support LCD expansion connector	

ARDUINO ZERO

Microcontroller	ATSAMD21G18, 48pins LQFP
Operating Voltage	3.3V
Digital I/O Pins	14, with 12 PWM and UART
Analog Input Pins	6, 12-bit ADC channels
Analog Output Pins	1, 10-bit DAC
DC Current per I/O Pin	7 mA
Flash Memory	256 KB
SRAM	32 KB
EEPROM	up to 16KB by emulation
Clock Speed	48 MHz

ARDUINO MICRO



Microcontroller	ATmega32u4
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	20
PWM Channels	7
Analog Input Channels	12
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega32u4) of which 4 KB used by bootloader
SRAM	2.5 KB (ATmega32u4)
EEPROM	1 KB (ATmega32u4)
Clock Speed	16 MHz
Length	48 mm
Width	18 mm
Weight	13 g

Arduino Esplora

Microcontroller	ATmega32u4
Operating Voltage	5V
Flash Memory	32 KB of which 4 KB used by

	bootloader
SRAM	2.5 KB
EEPROM	1 KB
Clock Speed	16 MHz

Arduino Mega ADK

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
USB Host Chip	MAX3421E

Arduino Ethernet

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage Plug (recommended)	7-12V
Input Voltage Plug (limits)	6-20V
Input Voltage PoE (limits)	36-57V
Digital I/O Pins	14 (of which 4 provide PWM output)
Arduino Pins reserved:	
	10 to 13 used for SPI
	4 used for SD card
	2 W5100 interrupt (when bridged)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz
W5100 TCP/IP Embedded Ethernet Controller	
Power Over Ethernet ready Magnetic Jack	
Micro SD card, with active voltage translators	

Arduino Mega 2560

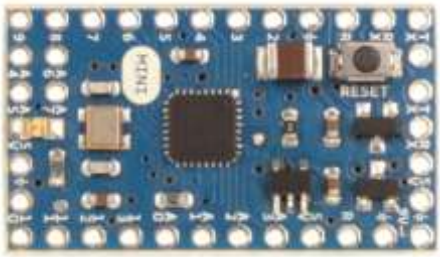
Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V

Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Arduino Robot

Microcontroller	Atmega32u4
Operating Voltage	5V
Input Voltage	5V through flat cable
Digital I/O Pins	5
PWM Channels	6
Analog Input Channels	4 (of the Digital I/O pins)
Analog Input Channels (multiplexed)	8
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (Atmega32u4) of which 4 KB used by bootloader
SRAM	2.5 KB (Atmega32u4)
EEPROM (internal)	1 KB (Atmega32u4)
EEPROM (external)	512 Kbit (I2C)
Clock Speed	16 MHz
Keypad	5 keys
Knob	potentiomenter attached to analog pin
Full color LCD	over SPI communication
SD card reader	for FAT16 formatted cards
Speaker	8 Ohm
Digital Compass	provides deviation from the geographical north in degrees
I2C soldering ports	3
Prototyping areas	4

Arduino Mini



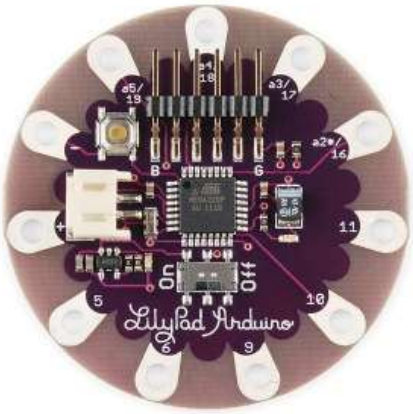
Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage	7-9 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8 (of which 4 are broken out onto pins)
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	16 MHz
Length	30 mm

Width	18 mm

Arduino Nano

Microcontroller	Atmel ATmega168 or ATmega328
Operating Voltage (logic level)	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32 KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	16 MHz
Dimensions	0.73" x 1.70"

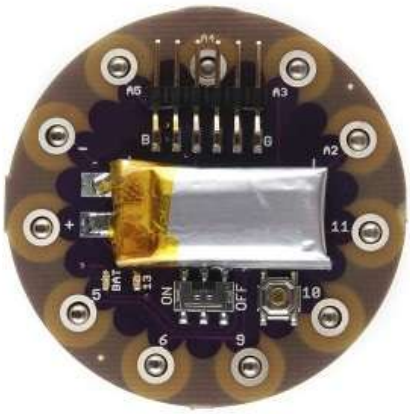
Arduino Lilypad Simple



Microcontroller	ATmega328
Operating Voltage	2.7-5.5 V
Input Voltage	2.7-5.5 V
Digital I/O Pins	9 (of which 5 provide PWM output)
Analog Input Pins	4
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	8 MHz

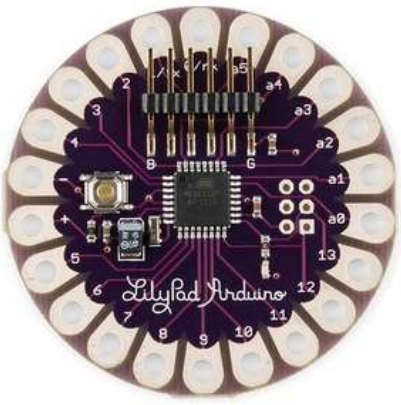
Arduino Lilypad Simple Snap

Microcontroller	ATmega328
Operating Voltage	2.7-5.5 V
Input Voltage	2.7-5.5 V
Digital I/O Pins	9 (of which 5 provide PWM output)
Analog Input Pins	4
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	8 MHz



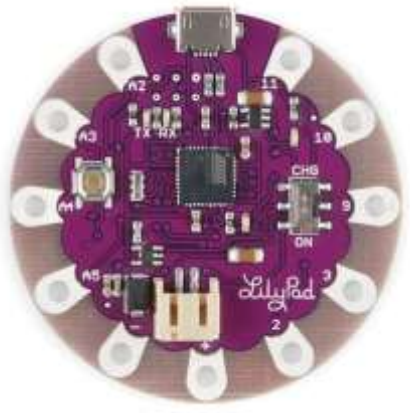
Radius	18 mm

Arduino Lilypad



Microcontroller	ATmega168V or ATmega328V
Operating Voltage	2.7-5.5 V
Input Voltage	2.7-5.5 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (of which 2 KB used by bootloader)
SRAM	1 KB
EEPROM	512 bytes
Clock Speed	8 MHz

Arduino Lilypad USB

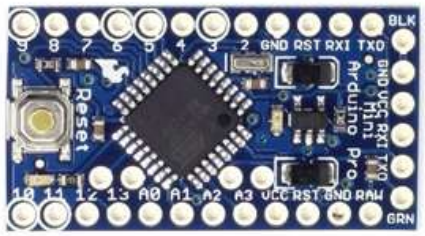


Microcontroller	ATmega32u4
Operating Voltage	3.3V
Input Voltage	3.8V to 5V
Digital I/O Pins	9
PWM Channels	4
Analog Input Channels	4
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (ATmega32u4) of which 4 KB used by bootloader
SRAM	2.5 KB (ATmega32u4)
EEPROM	1 KB (ATmega32u4)
Clock Speed	8 MHz

K

Arduino Pro Mini

Microcontroller	ATmega168
Operating Voltage	3.3V or 5V (depending on model)
Input Voltage	3.35 -12 V (3.3V model) or 5 - 12 V (5V model)
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8



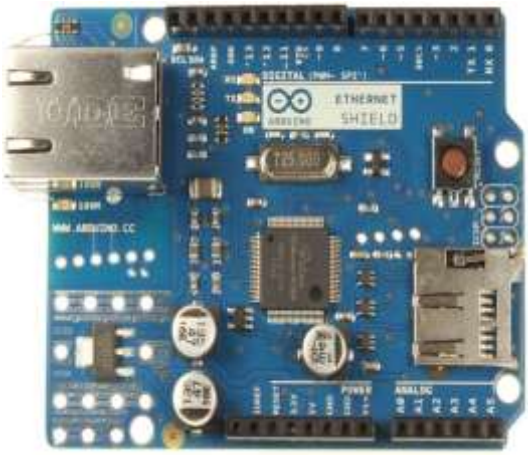
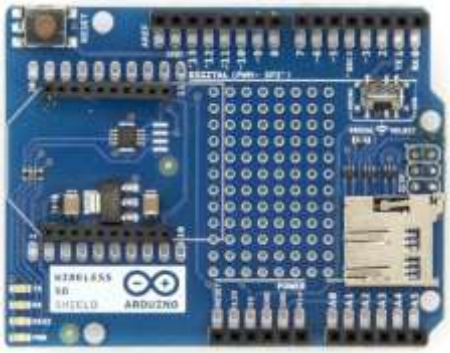
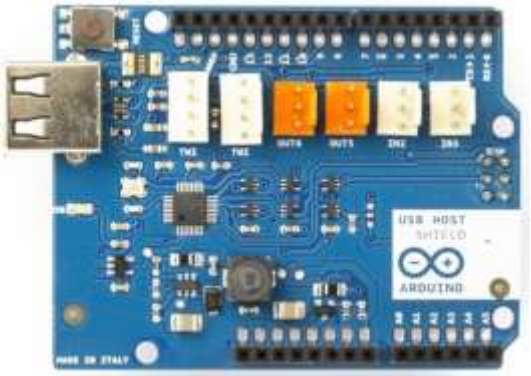
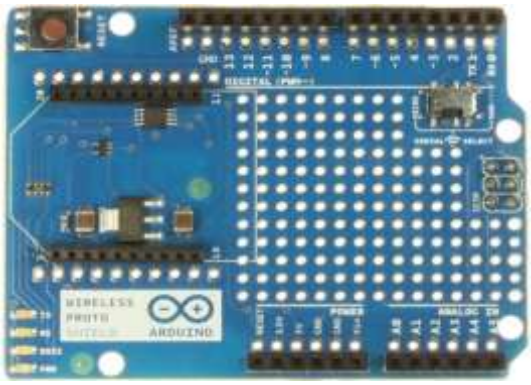
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (of which 2 KB used by bootloader)
SRAM	1 KB
EEPROM	512 bytes
Clock Speed	8 MHz (3.3V model) or 16 MHz (5V model)

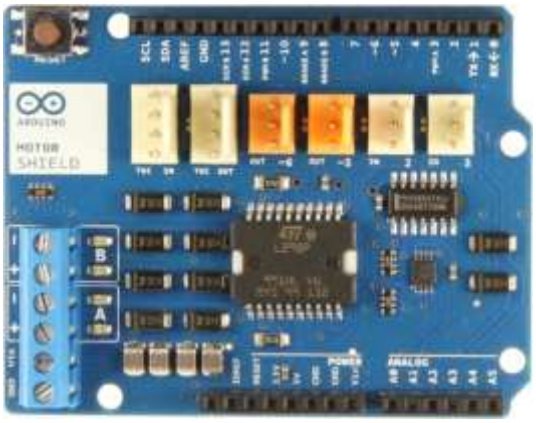
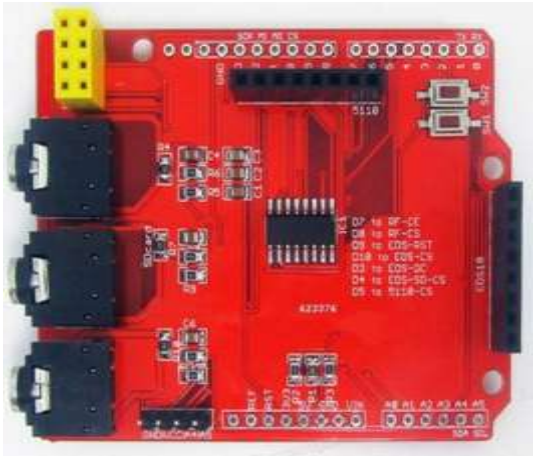
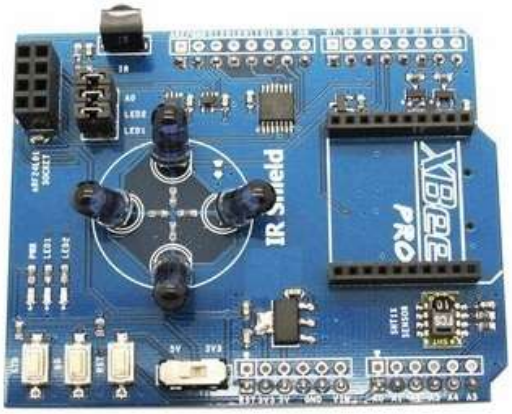
Arduino Fio

Microcontroller	ATmega328P
Operating Voltage	3.3V
Input Voltage	3.35 -12 V
Input Voltage for Charge	3.7 - 7 V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	8
DC Current per I/O Pin	40 mA
Flash Memory	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
Clock Speed	8 MHz

Arduino Pro

Microcontroller	ATmega168 or ATmega328
Operating Voltage	3.3V or 5V
Input Voltage	3.35 -12 V (3.3V versions) or 5 - 12 V (5V versions)
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
Flash Memory	16 KB (ATmega168) or 32KB (ATmega328) of which 2 KB used by bootloader
SRAM	1 KB (ATmega168) or 2 KB (ATmega328)
EEPROM	512 bytes (ATmega168) or 1 KB (ATmega328)
Clock Speed	8 MHz (3.3V versions) or 16 MHz (5V versions)

Arduino GSM Shield	Arduino Ethernet Shield
	
Arduino WiFi Shield	**Arduino Wireless SD Shield**
	
Arduino USB Host Shield	**Arduino Wireless Proto Shield**
	

Arduino Motor Shield	Arduino Proto Shield
	
ShieldSensorCorrenteTensãoPotencia	**ShieldEthernetWifiRs232**
	
ShieldJoystick	**Shield IR Controle Remoto**
	

11.2 Anexo II - Free Software Foundation e Open Source

O que é Free Software Foundation?

A **Free Software Foundation** (**FSF**, *Fundação para o Software Livre*) é uma organização sem fins lucrativos, fundada em 04 de Outubro de 1985 por Richard Stallman e que se dedica a eliminação de restrições sobre a cópia, redistribuição, estudo e modificação de programas de computadores – bandeiras do movimento do software livre, em essência. Faz isso promovendo o desenvolvimento e o uso de software livre em todas as áreas da computação, mas particularmente, ajudando a desenvolver o sistema operacional GNU e suas ferramentas.

Até meados da década de 1990 a fundação dedicava-se mais à escrita do software. Como hoje existem muitos projetos independentes de software livre, a FSF dedica-se mais aos aspectos legais e estruturais da comunidade do software livre. Entre suas atribuições atuais, encarrega-se de aperfeiçoar licenças de software e de documentação, de desenvolver um aparato legal acerca dos direitos autorais dos programas criados sob essas licenças, de catalogar e disponibilizar um serviço com os softwares livres desenvolvidos (o Free Software Directory), e de discutir e aperfeiçoar a própria definição de software livre. A FSF mantém artigos históricos que abrange a filosofia do software livre e mantém a definição do termo Software Livre, deixando claro quais são as características necessárias para que seja considerado livre.

A FSF patrocina o Projeto GNU (um esforço contínuo para fornecer um sistema operacional completo e licenciado como software livre) e detém direitos autorais sobre uma grande proporção do sistema operacional GNU e outros softwares livres. Esta atitude visa defender o software livre dos esforços em torná-lo proprietário.

Todos os anos a FSF coleta milhares de atribuições de direitos autorais de desenvolvedores e corporações que trabalham com software livre. A FSF registra os direitos autorais em um escritório de direitos autorais dos EUA e faz cumprir a licença sob a qual o software foi distribuído, normalmente a GNU General Public License. Isto é feito para garantir que os distribuidores de software livre respeitem a obrigação de garantir a liberdade a todos os usuários em compartilhar, estudar e modificar o código-

fonte do software. A FSF realiza este trabalho por meio da FSF Free Software Licensing and Compliance Lab, que foi formalizada em dezembro de 2001.

O que é Open Source ?

É comum ver Software Livre e Código Aberto (*Open Source*) sendo tratados como se fossem a mesma coisa. De igual maneira, não é difícil encontrar a expressão "código aberto" como mero sinônimo de "código-fonte aberto". Não há, necessariamente, erros aqui, mas há diferenças.

O Open Source é um movimento que surgiu em 1998 por iniciativa principal de **Bruce Perens**, mas com o apoio de várias outras pessoas que não estavam totalmente de acordo com os ideais filosóficos ou com outros aspectos do Software Livre, resultando na criação da **Open Source Initiative** (OSI). A Open Source Initiative não ignora as liberdades da Free Software Foundation, por outro lado, tenta ser mais flexível. Para isso, a organização definiu dez quesitos para que um software possa ser considerado Open Source:

A definição do Open Source foi criada pela Open Source Iniciative (OSI) a partir do texto original da Debian Free Software Guidelines (DFSG) e determina que um programa de código aberto deve garantir:

1. Distribuição livre

- A licença não deve restringir de nenhuma maneira a venda ou distribuição do programa gratuitamente, como componente de outro programa ou não.

2. Código fonte

- O programa deve incluir seu código fonte e deve permitir a sua distribuição também na forma compilada. Se o programa não for distribuído com seu código fonte, deve haver algum meio de se obter o mesmo seja via rede ou com custo apenas de reprodução. O código deve ser legível e inteligível para qualquer programador.

3. Trabalhos Derivados

- A licença deve permitir modificações e trabalhos derivados, e deve permitir que eles sejam distribuídos sobre os mesmos termos da licença original.

4. Integridade do autor do código fonte

- A licença pode restringir o código fonte de ser distribuído em uma forma modificada apenas se a licença permitir a distribuição de arquivos patch(de atualização) com o código fonte para o propósito de modificar o programa no momento de sua construção. A licença deve explicitamente permitir a distribuição do programa construído a partir do código fonte modificado. Contudo, a licença pode ainda requerer que programas derivados tenham um nome ou número de versão diferentes do programa original.

5. Não discriminação contra pessoas ou grupos

- A licença não pode ser discriminatória contra qualquer pessoa ou grupo de pessoas.

6. Não discriminação contra áreas de atuação

- A licença não deve restringir qualquer pessoa de usar o programa em um ramo específico de atuação. Por exemplo, ela não deve proibir que o programa seja usado em um empresa, ou de ser usado para pesquisa genética.

7. Distribuição da Licença

- Os direitos associados ao programa devem ser aplicáveis para todos aqueles cujo o programa é redistribuído, sem a necessidade da execução de uma licença adicional para estas partes.

8. Licença não específica a um produto

- Os direitos associados ao programa não devem depender que o programa seja parte de uma distribuição específica de programas. Se o programa é extraído desta distribuição e usado ou distribuído dentro dos termos da licença do

programa, todas as partes para quem o programa é redistribuído devem ter os mesmos direitos que aqueles que são garantidos em conjunção com a distribuição de programas original.

9. Licença não restrinja outros programas

- A licença não pode colocar restrições em outros programas que são distribuídos juntos com o programa licenciado. Isto é, a licença não pode especificar que todos os programas distribuídos na mesma mídia de armazenamento sejam programas de código aberto.

10. Licença neutra em relação a tecnologia

- Nenhuma cláusula da licença pode estabelecer uma tecnologia individual, estilo ou interface a ser aplicada no programa.

Entendendo as diferenças entre Free Software Foundation e Open Source

Analisando as características da Free Software Foundation e da Open Source Initiative, percebemos que, em muitos casos, um software livre pode também ser considerado código aberto e vice-versa.

A diferença está, essencialmente, no fato de a OSI ter receptividade maior em relação às iniciativas de software do mercado. Dessa forma, empresas como Microsoft e Oracle, duas gigantes do software proprietário, podem desenvolver soluções de código aberto utilizando suas próprias licenças, desde que estas respeitem os critérios da OSI. No Software Livre, empresas como estas provavelmente enfrentariam algum tipo de resistência, uma vez que suas atividades principais ou mesmo os programas oferecidos podem entrar em conflito com os ideais morais da Free Software Foundation.

Apesar disso, Free Software Foundation e Open Source Initiative não são inimigas. Embora não concordem em alguns pontos, ambas as entidades se respeitam e reconhecem a importância da atuação de cada uma. De certa forma, pode-se dizer inclusive que o trabalho das duas organizações se complementa: a FSF atuando mais pelo lado social; a OSI, pelos contextos técnicos e de mercado.

11.3 Anexo III - O que é IDE

IDE, do inglês ***Integrated Development Environment*** ou **Ambiente Integrado de Desenvolvimento**, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de *software* com o objetivo de agilizar este processo.

Geralmente os IDEs facilitam a técnica de RAD (de Rapid Application Development, ou "Desenvolvimento Rápido de Aplicativos"), que visa a maior produtividade dos desenvolvedores.

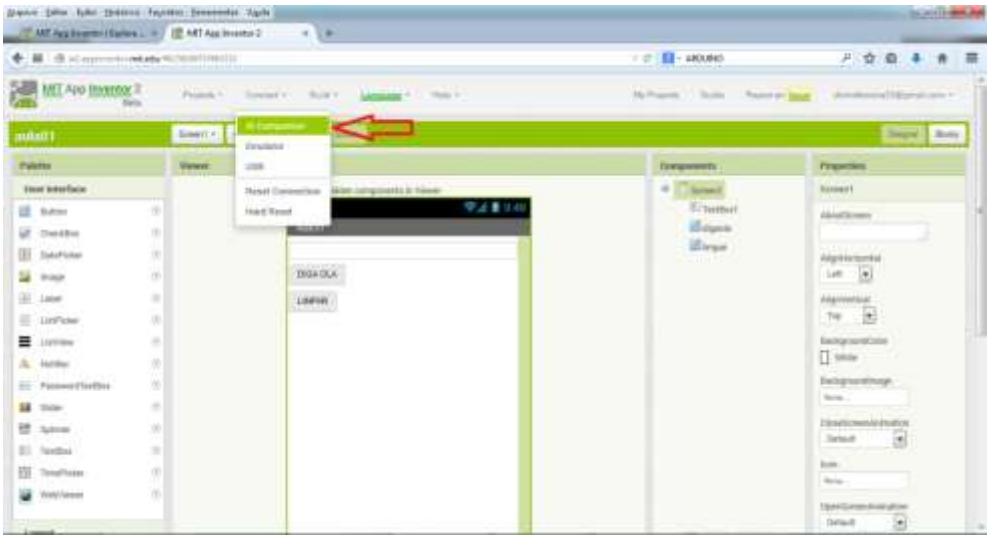
As características e ferramentas mais comuns encontradas nos IDEs são:

- **Editor** - edita o código-fonte do programa escrito na(s) linguagem(ns) suportada(s) pela IDE;
- **Compilador (*compiler*)** - compila o código-fonte do programa, editado em uma linguagem específica e a transforma em linguagem de máquina;
- ***Linker*** - liga (*linka*) os vários "pedaços" de código-fonte, compilados em linguagem de máquina, em um programa executável que pode ser executado em um computador ou outro dispositivo computacional.
- **Depurador (*debugger*)** - auxilia no processo de encontrar e corrigir defeitos no código-fonte do programa, na tentativa de aprimorar a qualidade de software;
- **Modelagem (*modeling*)** - criação do modelo de classes, objetos, interfaces, associações e interações dos artefatos envolvidos no software com o objetivo de solucionar as necessidades-alvo do software final.
- **Geração de código** - característica mais explorada em Ferramentas CASE, a geração de código também é encontrada em IDEs, contudo com um escopo mais direcionado a templates de código comumente utilizados para solucionar problemas rotineiros. Todavia, em conjunto com ferramentas de modelagem, a geração pode gerar praticamente todo o código-fonte do programa com base no modelo proposto, tornando muito mais rápido o processo de desenvolvimento e distribuição do software;
- **Distribuição (*deploy*)** - auxilia no processo de criação do instalador do software, ou outra forma de distribuição, seja discos ou via internet.
- **Testes Automatizados (*automated tests*)** - realiza testes no software de forma automatizada, com base em scripts ou programas de testes previamente especificados, gerando um relatório, assim auxiliando na análise do impacto das alterações no código-fonte. Ferramentas deste tipo mais comuns no mercado são chamadas robôs de testes.
- **Refatoração (*refactoring*)** - consiste na melhoria constante do código-fonte do software, seja na construção de código mais otimizado, mais limpo e/ou com melhor

entendimento pelos envolvidos no desenvolvimento do software. A refatoração, em conjunto com os testes automatizados, é uma poderosa ferramenta no processo de erradicação de "bugs", tendo em vista que os testes "garantem" o mesmo comportamento externo do software ou da característica sendo reconstruída.

11.4 Anexo IV – Testando Via WiFi

Uma vez que o PC/Notebook e o dispositivo móvel estejam na mesma rede WiFi, clique no menu CONECT e na opção AL COMPANION conforme figura a seguir.



Será mostrada uma janela com um código e um querycod. pode-se utiliizar qualquer um dos dois conforme figura a seguir

Abra o software mit ai2 companion no dispositivo móvel. Será mostrado a tela da figura a seguir.

Digite o código no campo Six Digit Code

Depois em connect with code

Ou se preferir;

Clique em scan QR code e aproxime a Camera do dispositivo móvel para que a captura seja efetuada. Aguarde um tempo para a conexão automática.

Caso tenha ocorrido tudo certo até o momento deverá aparecer a imagem da aplicação no dispositivo móvel.

11.5 Anexo V – Testando Via Emulador

11.5.1 Instalando e executando o emulador

Se você não tem um telefone ou tablet com Android, você ainda pode criar aplicativos com o App Inventor. App Inventor oferece um emulador Android, que funciona como um Android na tela do computador. Então você pode testar seus aplicativos em um emulador e ainda distribuir o aplicativo para os outros, até mesmo através da Play Store.

Para usar o emulador, primeiro você precisa instalar um software no computador (isso não é necessário para a solução Wi-Fi). Siga as instruções abaixo de acordo com seu sistema operacional:

11.5.1.1 Para Windows

Instalando o App Inventor Setup 2 no Windows

Instalando o software Windows para App Inventor Setup tem duas partes:

Instalando o pacote de software de configuração App Inventor. Este passo é o mesmo para o Windows XP, Vista e 7.

Se você optar por usar o cabo USB para conectar a um dispositivo, então você precisará instalar os drivers do Windows para o seu telefone Android.

NOTA: App Inventor 2 não funciona com o Internet Explorer. Para usuários do Windows, recomendamos o uso ou Chrome ou Firefox como seu navegador para uso com o App Inventor.

Instalando o pacote de software de configuração App Inventor

Você deve executar a instalação a partir de uma conta que tenha privilégios de administrador. Instalações sem privilégios de administrador não são suportadas atualmente.

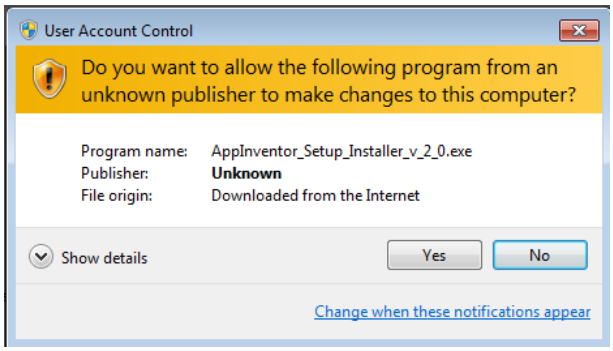
1 - Faça o download do instalador.
→ http://appinv.us/aisetup_windows

2 - Localize o arquivo AppInventor_Setup_Installer_v_2_1.exe (~ 101 MB) em seu arquivo de Downloads ou seu Desktop. O local do download em seu computador depende de como o seu navegador está configurado.

3 - Abra o arquivo.

4 - Clique através dos passos do instalador. Não altere o local de instalação, mas gravar o diretório de instalação, porque você pode precisar dele para verificar os drivers mais tarde. O diretório irá variar de acordo com a sua versão do Windows e se está ou não está logado como administrador.

5 - Você pode ser solicitado se você quiser permitir que um programa de uma fonte desconhecida de fazer alterações a este computador. Clique em Sim (Yes).



Localizando o software de configuração

Na maioria dos casos, o App Inventor deve ser capaz de localizar o software de configuração por conta própria. Mas, se ele pedir a localização do software, o caminho para entrar é C: \ Program Files \ AppInventor \ commands-for-AppInventor. Se você estiver usando uma máquina de 64 bits, você deve digitar Program Files (x86), em vez de Arquivos de Programas.

Continue com a configuração - Escolha a opção emulador clicando no botão [Setu Emulator]

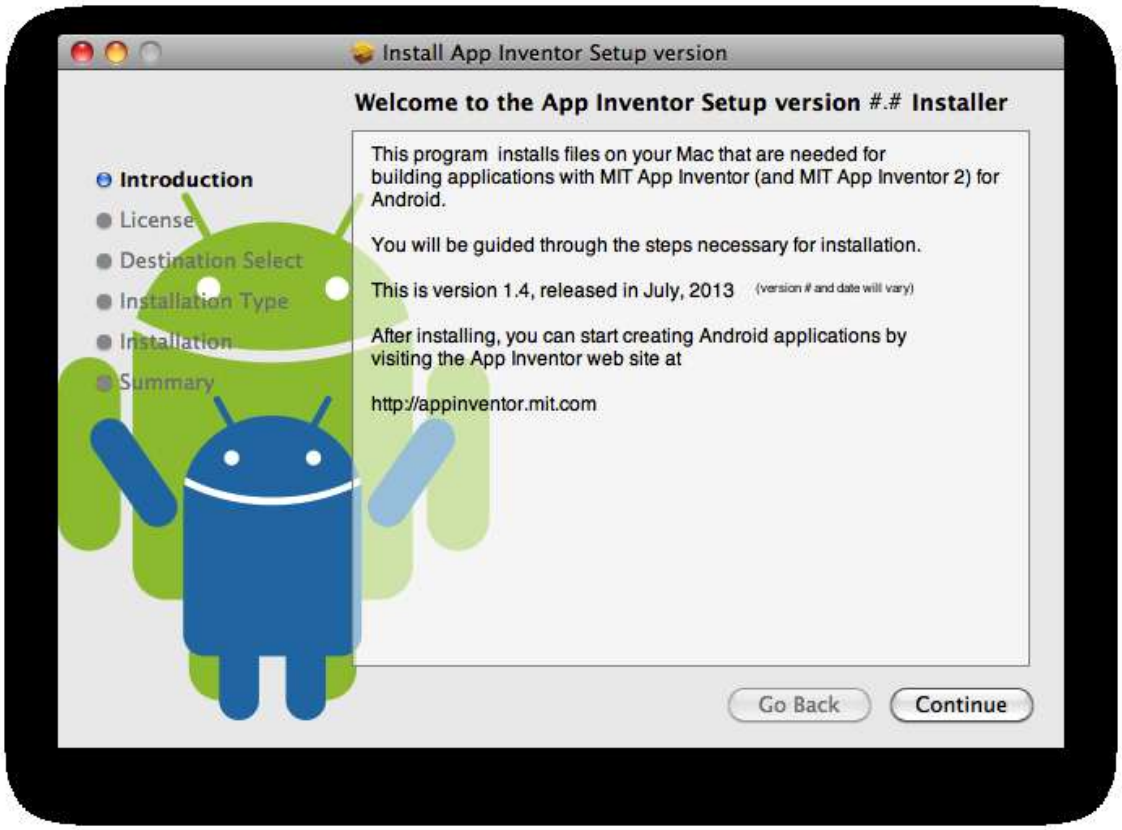
11.5.1.2 Para Mac

1 - Para obter o emulador Android para o seu Mac, faça o download e instale o pacote de instalação através do link a seguir:

➔ http://appinv.us/aisetup_mac

2 - Dê um duplo clique no arquivo baixado para iniciar o instalador. (Você pode precisar olhar na pasta de downloads do seu browser. O arquivo é nomeado como AppInventor_Setup_v_X.X.dmg (onde XX é o número da versão)

3 - Clique em Continuar (Continue).

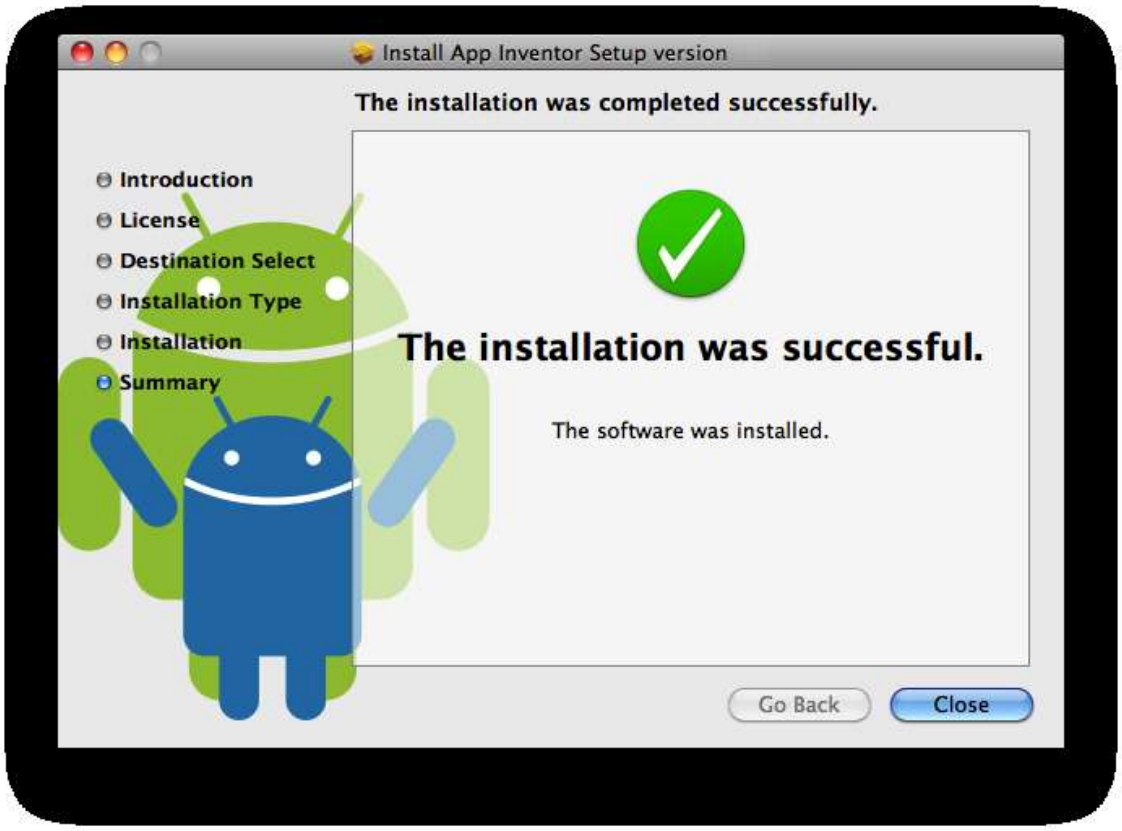


4 - Leia e aceite o contrato de licença de software.

5 - Por padrão Não altere o local de instalação, clique em Instalar.

6 - Se solicitado, digite sua senha para confirmar que você realmente deseja instalar o software. Clique em OK.

7 - O instalador confirma que o pacote de instalação App Inventor foi instalado.



Continue com a configuração - Escolha a opção emulador clicando no botão [Setu Emulator]

11.5.1.3 Para Linux

Instalando o App Inventor 2 Setup em GNU / Linux

Você vai precisar de privilégios sudo para fazer a instalação.

Nota: Os programas de configuração são software de 32 bits. Se você tem um sistema de 64 bits pode ser necessário instalar o bibliotecas para deixar sua máquina executar 32-bit.

Uma maneira de fazer isso é executar o comando sudo apt-get install lib32z1, mas isso pode não funcionar em todas as distribuições GNU / Linux, e você pode precisar de fazer algum estudo para o seu sistema particular.

sudo rm -rf / usr / google / AppInventor

sudo rm -rf ~ / .appinventor

As instruções para os sistemas que podem instalar pacotes Debian

Utilize estas instruções para os sistemas que podem instalar pacotes Debian (por exemplo, Debian ou Ubuntu):

Nota: Se você já instalou o pacote de instalação para o App Inventor Classic, você deve removê-lo, uma vez que pode interferir com a nova instalação. Remova o pacote com sudo apt-get remove AppInventor-setup.

1 - Baixe o pacote Debian installer Setup AppInventor. Este é um arquivo chamado appinventor2-setup_1.1_all.deb. É um arquivo instalador de pacotes Debian. O local onde será baixado no computador depende de como o seu navegador está configurado. Normalmente, ele vai entrar em sua pasta de Downloads.

2 - Se o seu sistema pode instalar pacotes simplesmente clicando no arquivo do pacote, então faça isso.

3 - Se o seu sistema não suporta instaladores de pacotes clicáveis, em seguida, navegue até o diretório onde o arquivo está localizado e execute o comando

sudo dpkg --install appinventor2-setup_1.1_all.deb

Com qualquer método, talvez seja necessário para garantir que o arquivo **deb**, bem como o diretório no qual ele está são executáveis e legível. Em alguns sistemas, sudo não tem os privilégios padrão de ler e executar todos os arquivos.

4 - O software será instalado em / usr / google / AppInventor.

5 - Você também pode precisar configurar o seu sistema para detectar o dispositivo. Veja as instruções para desenvolvedores Android a criação de um dispositivo para o desenvolvimento. Siga as instruções na etapa "configurar o seu sistema para detectar o dispositivo" na opção "Se você está desenvolvendo no Ubuntu Linux".

➔ http://developer.android.com/guide/developing/device.html#setting-up

Instruções para outros sistemas GNU / Linux

1 - Faça o download do arquivo instalador tar do Setup AppInventor. Este é um arquivo chamado appinventor2-setup_1.1.tar.gz. É um arquivo tar Gzip comprimido.

2 - Instale os arquivos usando um método adequado ao seu sistema operacional. Você precisa verificar se o diretório de comandos para-AppInventor acaba em / usr / google / AppInventor.

Iniciando o software aiStarter

O programa aiStarter gerencia a comunicação entre o browser e o dispositivo Android. Ele deve estar em execução sempre que as pessoas usar o emulador ou o cabo USB; ele não precisa estar em execução quando as pessoas estão usando a configuração sem fio. Sempre que alguém faz login para usar o App Inventor com o emulador ou USB, eles terão de começar aiStarter. Isto pode ser feito com o comando

/ usr / Google / AppInventor / comandos-para-AppInventor / aiStarter &

Para maior comodidade, você pode querer mandar este comando para ser executado automaticamente sempre que alguém faz login, ou quando o sistema é iniciado. A forma exata de fazer isso depende de qual distribuição GNU / Linux você está usando. Consulte a documentação para a sua distribuição.

Localizando o diretório Setup

Na maioria dos casos, o App Inventor deve ser capaz de localizar o software de configuração por conta própria. Mas, se ele pedir a localização do software, o caminho para entrar é / usr / Google / AppInventor / comandos para-AppInventor

Continue com a configuração - Escolha a opção emulador clicando no botão [Setu Emulator]

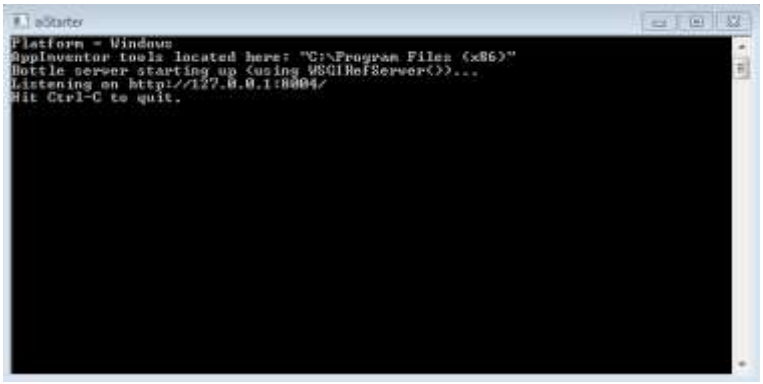
11.5.2 Iniciando o aiStarter (Windows e GNU / Linux apenas)

Usando o emulador ou o cabo USB requer o uso de um programa chamado aiStarter. Este programa é o auxiliar que permite que o navegador se comunique com o cabo USB ou emulador. O programa aiStarter foi instalado quando você instalou o pacote de instalação App Inventor. Você não precisa do aiStarter se você estiver usando apenas a configuração sem fio.

Em um Mac, aiStarter vai começar automaticamente quando você entrar em sua conta e ele será executado de forma invisível no fundo.

No Windows, não haverá atalhos para aiStarter partir do seu desktop, a partir do menu Iniciar, a partir de todos os programas e de pasta de inicialização. Se você quiser usar o emulador com o App Inventor, você terá de executar manualmente aiStarter em seu computador quando você faz login. Você pode começar aiStarter isso clicando no ícone na área de trabalho ou usando a entrada em seu menu iniciar.

Para iniciar aiStarter no Windows, clique duas vezes no ícone (mostrado acima). Você vai saber que você lançou com sucesso aiStarter quando você verá uma janela como a seguinte:



No GNU / Linux, aiStarter estará na pasta / usr / google / AppInventor / commands-for-AppInventor. Você precisa iniciá-lo manualmente. Você pode iniciá-lo a partir da linha de comando com

/ usr / google / AppInventor / comandos-para-AppInventor / aiStarter &

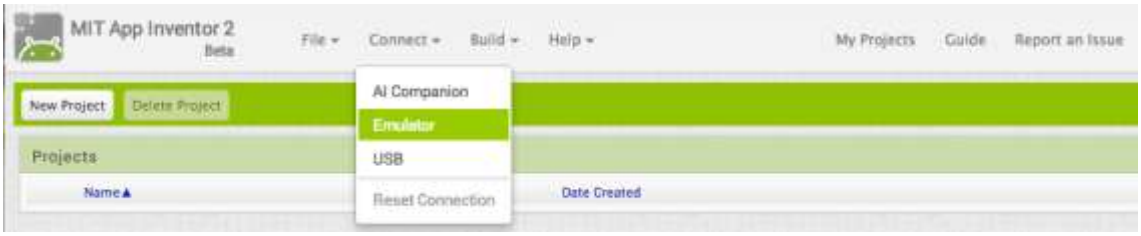
Para obter ajuda com aiStarter, consulte Connection Help.

➔ http://appinventor.mit.edu/explore/ai2/aistarter-help.html

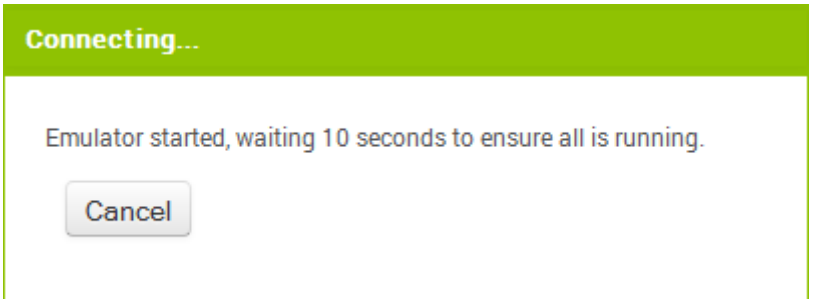
Passo 3. Abra um projeto App Inventor e conectá-lo para o emulador

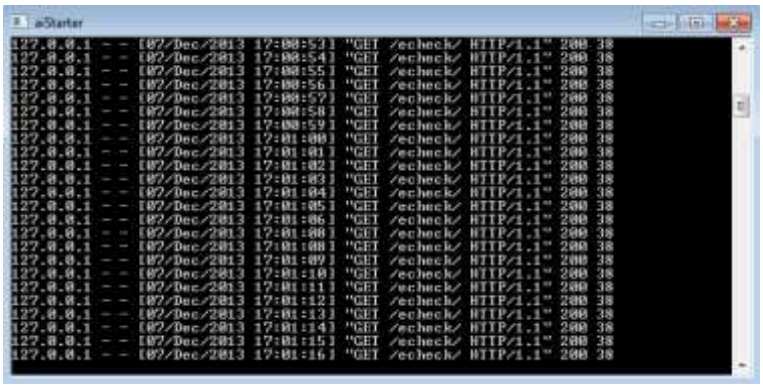
Primeiro, vá para o App Inventor e abrir um projeto (ou crie um novo - uso Project> Comece New Project e dê um nome ao projeto).

Então, a partir do menu do App Inventor (no software baseado em nuvem App Inventor no ai2.appinventor.mit.edu), vá até o menu Conectar e clique na opção Emulator.



Você vai receber um aviso dizendo que o emulador está se conectando. Iniciando o emulador pode levar alguns minutos. Você pode ver as telas de atualização, como o seguinte, como o emulador inicia-se:





O emulador aparecerá inicialmente com uma tela preta vazia (# 1). Aguarde até que o emulador está pronto, com um fundo de tela colorida (# 2). Mesmo depois que aparecer

no fundo você deve esperar até que o telefone emulado termine de preparar o seu cartão SD: haverá um aviso no topo da tela do telefone enquanto o cartão está sendo preparado. Quando ligado, o emulador irá lançar e mostrar o app que você tem aberto na App Inventor.

Se esta é a primeira vez que você estiver usando o emulador após a instalação do software de configuração App Inventor, você verá uma mensagem pedindo para atualizar o emulador. Siga as instruções na tela para realizar a atualização e voltar a ligar o emulador. Você vai precisar de fazer este tipo de atualização sempre que houver uma nova versão do software App Inventor.

Configuração completa! Agora você está pronto para construir seu primeiro app!

11.6 Anexo VI – Testando Via Cabo USB

11.6.1 Instalando e executando via cabo USB

Há alguns ambientes onde as conexões sem fio podem não funcionar ou até mesmo não estiver disponível. Estes incluem alguns hotéis, centros de conferência e escolas, que configuram suas redes sem fio para proibir dois dispositivos na rede se comunicarem uns com os outros. Neste caso a utilização e testes com App Inventor deverá ser feita através de um cabo USB.

Configuração de uma conexão USB pode ser difícil, especialmente em máquinas Windows, que precisam de driver de software especial para se conectar a dispositivos Android (este não é o caso com Mac ou Linux, que não precisam de drivers especiais.) Infelizmente, diferentes dispositivos podem exigir diferentes drivers. Com isso poderá ser necessário que o usuário pesquise na internet ou no site do fabricante por drivers para seu modelo de dispositivo. App Inventor oferece um programa de teste que verifica se o seu dispositivo conectado via USB pode se comunicar com o computador. Você deve executar este teste e resolver quaisquer problemas de conexão antes de tentar usar o App Inventor com USB nesse dispositivo.

Aqui estão os passos para começar a usar o App Inventor com o cabo USB:

Para usar o cabo USB, primeiro você precisa instalar um software no computador (isso não é necessário para a solução Wi-Fi). Siga as instruções abaixo de acordo com seu sistema operacional:

11.6.1.1 Para Windows

Instalando o App Inventor Setup 2 no Windows

Instalando o software Windows para App Inventor Setup tem duas partes:

Instalando o pacote de software de configuração App Inventor. Este passo é o mesmo para o Windows XP, Vista e 7.

Se você optar por usar o cabo USB para conectar a um dispositivo, então você precisará instalar os drivers do Windows para o seu telefone Android.

NOTA: App Inventor 2 não funciona com o Internet Explorer. Para usuários do Windows, recomendamos o uso ou Chrome ou Firefox como seu navegador para uso com o App Inventor.

Instalando o pacote de software de configuração App Inventor

Você deve executar a instalação a partir de uma conta que tenha privilégios de administrador. Instalações sem privilégios de administrador não são suportadas atualmente.

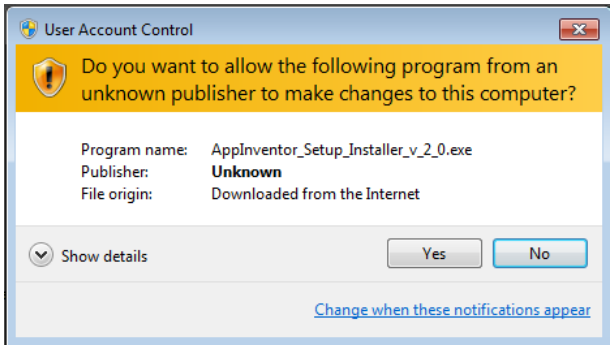
1 - Faça o download do instalador.
→ http://appinv.us/aisetup_windows

2 - Localize o arquivo AppInventor_Setup_Installer_v_2_1.exe (~ 101 MB) em seu arquivo de Downloads ou seu Desktop. O local do download em seu computador depende de como o seu navegador está configurado.

3 - Abra o arquivo.

4 - Clique através dos passos do instalador. Não altere o local de instalação, mas gravar o diretório de instalação, porque você pode precisar dele para verificar os drivers mais tarde. O diretório irá variar de acordo com a sua versão do Windows e se está ou não está logado como administrador.

5 - Você pode ser solicitado se você quiser permitir que um programa de uma fonte desconhecida de fazer alterações a este computador. Clique em Sim (Yes).



Localizando o software de configuração

Na maioria dos casos, o App Inventor deve ser capaz de localizar o software de configuração por conta própria. Mas, se ele pedir a localização do software, o caminho para entrar é C: \ Program Files \ AppInventor \ commands-for-AppInventor. Se você estiver usando uma máquina de 64 bits, você deve digitar Program Files (x86), em vez de Arquivos de Programas.

Continue com a configuração - Escolha a opção emulador clicando no botão [Setu Emulator]

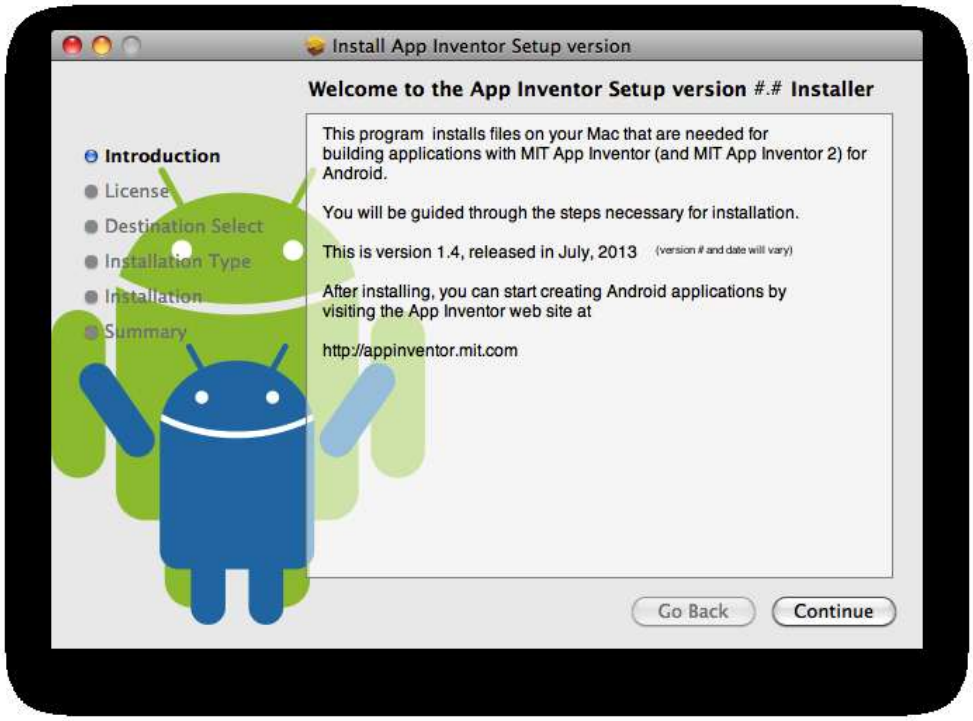
11.6.1.2 Para Mac

1 - Para obter o emulador Android para o seu Mac, faça o download e instale o pacote de instalação através do link a seguir:

➔ http://appinv.us/aisetup_mac

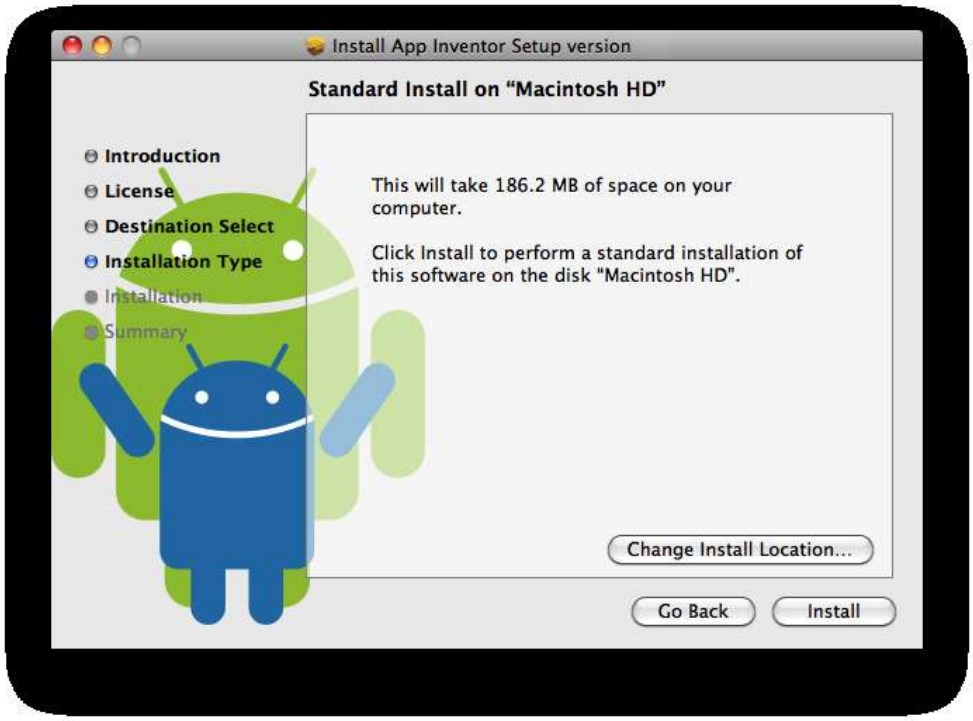
2 - Dê um duplo clique no arquivo baixado para iniciar o instalador. (Você pode precisar olhar na pasta de downloads do seu browser. O arquivo é nomeado como AppInventor_Setup_v_X.X.dmg (onde XX é o número da versão)

3 - Clique em Continuar (Continue).



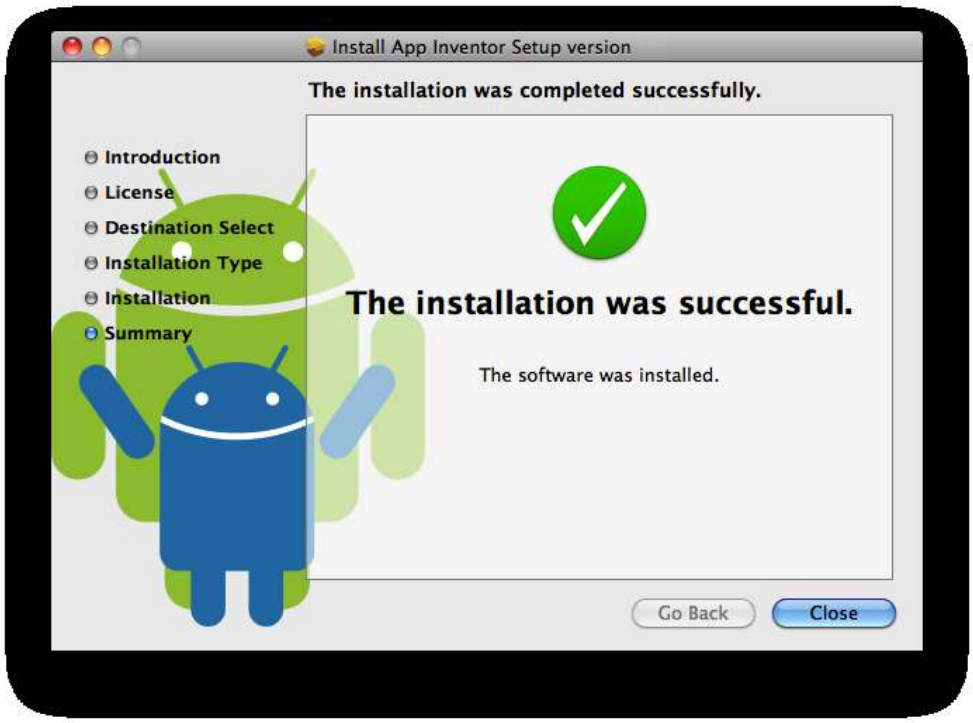
4 - Leia e aceite o contrato de licença de software.

5 - Por padrão Não altere o local de instalação, clique em Instalar.



6 - Se solicitado, digite sua senha para confirmar que você realmente deseja instalar o software. Clique em OK.

7 - O instalador confirma que o pacote de instalação App Inventor foi instalado.



Continue com a configuração - Escolha a opção emulador clicando no botão [Setup Emulator]

11.6.1.3 Para Linux

Instalando o App Inventor 2 Setup em GNU / Linux

Você vai precisar de privilégios sudo para fazer a instalação.

Nota: Os programas de configuração são software de 32 bits. Se você tem um sistema de 64 bits pode ser necessário instalar o bibliotecas para deixar sua máquina executar 32-bit.

Uma maneira de fazer isso é executar o comando sudo apt-get install lib32z1, mas isso pode não funcionar em todas as distribuições GNU / Linux, e você pode precisar de fazer algum estudo para o seu sistema particular.

sudo rm -rf / usr / google / AppInventor

sudo rm -rf ~ / .appinventor

As instruções para os sistemas que podem instalar pacotes Debian

Utilize estas instruções para os sistemas que podem instalar pacotes Debian (por exemplo, Debian ou Ubuntu):

Nota: Se você já instalou o pacote de instalação para o App Inventor Classic, você deve removê-lo, uma vez que pode interferir com a nova instalação. Remova o pacote com sudo apt-get remove AppInventor-setup.

1 - Baixe o pacote Debian installer Setup AppInventor. Este é um arquivo chamado appinventor2-setup_1.1_all.deb. É um arquivo instalador de pacotes Debian. O local

onde será baixado no computador depende de como o seu navegador está configurado. Normalmente, ele vai entrar em sua pasta de Downloads.

2 - Se o seu sistema pode instalar pacotes simplesmente clicando no arquivo do pacote, então faça isso.

3 - Se o seu sistema não suporta instaladores de pacotes clicáveis, em seguida, navegue até o diretório onde o arquivo está localizado e execute o comando

sudo dpkg --install appinventor2-setup_1.1_all.deb

Com qualquer método, talvez seja necessário para garantir que o arquivo **deb**, bem como o diretório no qual ele está são executáveis e legível. Em alguns sistemas, sudo não tem os privilégios padrão de ler e executar todos os arquivos.

4 - O software será instalado em / usr / google / AppInventor.

5 - Você também pode precisar configurar o seu sistema para detectar o dispositivo. Veja as instruções para desenvolvedores Android a criação de um dispositivo para o desenvolvimento. Siga as instruções na etapa "configurar o seu sistema para detectar o dispositivo" na opção "Se você está desenvolvendo no Ubuntu Linux".

➔ http://developer.android.com/guide/developing/device.html#setting-up

Instruções para outros sistemas GNU / Linux

1 - Faça o download do arquivo instalador tar do Setup AppInventor. Este é um arquivo chamado appinventor2-setup_1.1.tar.gz. É um arquivo tar Gzip comprimido.

2 - Instale os arquivos usando um método adequado ao seu sistema operacional. Você precisa verificar se o diretório de comandos para-AppInventor acaba em / usr / google / AppInventor.

Iniciando o software aiStarter

O programa aiStarter gerencia a comunicação entre o browser e o dispositivo Android. Ele deve estar em execução sempre que as pessoas usar o emulador ou o cabo USB; ele não precisa estar em execução quando as pessoas estão usando a configuração sem fio. Sempre que alguém faz

login para usar o App Inventor com o emulador ou USB, eles terão de começar aiStarter. Isto pode ser feito com o comando

/ usr / Google / AppInventor / comandos-para-AppInventor / aiStarter &

Para maior comodidade, você pode querer mandar este comando para ser executado automaticamente sempre que alguém faz login, ou quando o sistema é iniciado. A forma exata de fazer isso depende de qual distribuição GNU / Linux você está usando. Consulte a documentação para a sua distribuição.

Localizando o diretório Setup

Na maioria dos casos, o App Inventor deve ser capaz de localizar o software de configuração por conta própria. Mas, se ele pedir a localização do software, o caminho para entrar é / usr / Google / AppInventor / comandos para-AppInventor

Continue com a configuração - Escolha a opção emulador clicando no botão [Setup Emulator]

11.6.2 Iniciando o aiStarter (Windows e GNU / Linux apenas)

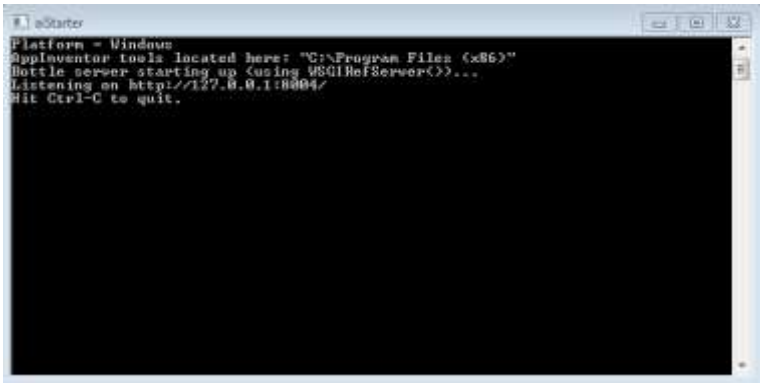
Usando o emulador ou o cabo USB requer o uso de um programa chamado aiStarter. Este programa é o auxiliar que permite que o navegador se comunique com o cabo USB ou emulador. O programa aiStarter foi instalado quando você instalou o pacote de instalação App Inventor. Você não precisa do aiStarter se você estiver usando apenas a configuração sem fio.

Em um Mac, aiStarter vai começar automaticamente quando você entrar em sua conta e ele será executado de forma invisível no fundo.

No Windows, não haverá atalhos para aiStarter partir do seu desktop, a partir do menu Iniciar, a partir de todos os programas e de pasta de inicialização. Se você quiser usar o emulador com o App Inventor, você terá de executar manualmente aiStarter em seu

computador quando você faz login. Você pode começar aiStarter isso clicando no ícone na área de trabalho ou usando a entrada em seu menu iniciar.

Para iniciar aiStarter no Windows, clique duas vezes no ícone (mostrado acima). Você vai saber que você lançou com sucesso aiStarter quando você verá uma janela como a seguinte:



No GNU / Linux, aiStarter estará na pasta / usr / google / AppInventor / commands-for-AppInventor. Você precisa iniciá-lo manualmente. Você pode iniciá-lo a partir da linha de comando com

/ usr / google / AppInventor / comandos-para-AppInventor / aiStarter &

Para obter ajuda com aiStarter, consulte Connection Help.

- ➔ http://appinventor.mit.edu/explore/ai2/aistarter-help.html

Configuração completa! Agora você está pronto para construir seu primeiro app!

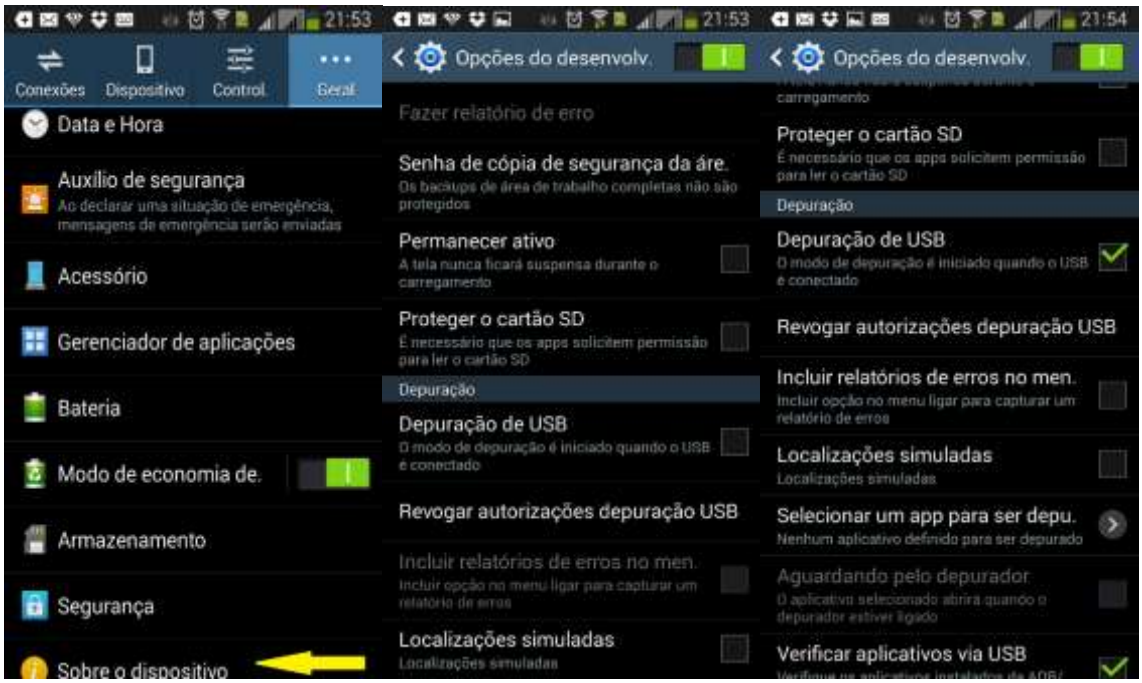
11.6.3 Configurando o dispositivo para USB

Em seu dispositivo Android, vá para Configurações do Sistema, Opções do desenvolvedor e veja se "Depuração USB" está ativado.

Na maioria dos dispositivos que executam o Android 3.2 ou mais, você pode encontrar esta opção em Configurações> Aplicativos> Desenvolvimento.

No Android 4.0 e mais recentes, está em Configurações> Opções do desenvolvedor.

No Android 4.2 e mais recentes, as opções de desenvolvedor está oculta por padrão. Para torná-lo disponível, vá para Configurações> Geral>Sobre o dispositivo e toque em cima sete vezes para tornar visível “Opções do Desenvolvedor”. Retorne à tela anterior para encontrar Opções de Desenvolvedor. Em seguida selecione Depuração USB.



11.6.4 Conectando o dispositivo ao PC

Conecte o dispositivo Android ao computador usando o cabo USB - certifique-se de que o dispositivo se conecta como um "dispositivo de armazenamento em massa" (e não "dispositivo de mídia") e que não é montado como uma unidade no computador. Isso pode significar que você tem que ir para o Finder (em um Mac) ou Meu Computador (no Windows) e desconectar qualquer unidade (s) que foram montadas quando conectou seu dispositivo Android.

No Android 4.2.2 e mais recente, o dispositivo irá aparecer uma tela com a mensagem de permitir a depuração USB? a primeira vez que o ligar ao computador novo. Pressione "OK". Este autentica o computador para o dispositivo, permitindo que o computador para comunicar com ele. Você precisa fazer isso para cada computador que você deseja conectar ao dispositivo, mas apenas uma vez por computador.

11.6.5 Testando a conexão

Vá para a pagina de teste de conexão no seguinte endereço:
http://appinventor.mit.edu/test

Veja se você recebeu uma confirmação de que seu computador detectou o dispositivo. Se o teste falhar, verifique as informações de ajuda para conexão USB no seguinte endereço:

http://appinventor.mit.edu/explore/ai2/connection-help.html

Retorne à página quando o teste tiver sucesso.

11.7 Anexo VII – Redes de Computadores

Montar uma rede já foi complicado e caro. Hoje em dia, praticamente todas as placas-mãe trazem placas de rede onboard (incluídas), e os cabos e switchs são extremamente baratos, o que fez com que as redes se tornassem extremamente comuns, permitindo compartilhar a conexão com a internet, transferir arquivos, compartilhar impressoras e assim por diante. Como não falo sobre a configuração de redes em outros tópicos do livro, vou aproveitar para fazer um apanhado geral sobre o assunto.

O uso mais corriqueiro é compartilhar a conexão com a internet. Você tem apenas uma linha ADSL ou apenas uma assinatura do serviço de acesso via cabo e pode acessar, ao mesmo tempo, a partir de todos os micros que tiver em sua casa ou empresa. Neste caso um dos micros atua como um ponto de encontro, enviando os pedidos de todos para a internet e devolvendo as respostas. Além de compartilhar a conexão, este servidor pode compartilhar arquivos, servir como firewall (protegendo a rede de acessos externos), rodar um proxy (que permite criar um cache de arquivos e páginas acessados, melhorando a velocidade da conexão), além de outros serviços.

Outra necessidade comum é compartilhar arquivos. Antigamente (naquela época em que os micros tinham 512 KB de memória e os homens eram homens e escreviam seus próprios sistemas operacionais) era usado o protocolo DPL/DPC (disquete pra lá, disquete pra cá), mas ele não era muito eficiente, principalmente quando o amigo que estava esperando os arquivos estava em outra cidade.

Hoje em dia, você pode compartilhar arquivos entre micros Windows simplesmente ativando o "Compartilhamento de arquivos para redes Microsoft" e o "Cliente para redes Microsoft" nas propriedades da rede e compartilhando as pastas desejadas (que passam a aparecer no ambiente de rede para os outros micros). No Linux, você pode compartilhar arquivos usando o Samba (que permite que os compartilhamentos sejam acessados também por máquinas Windows), NFS ou mesmo via SFTP (o módulo de transferência de arquivos do SSH).

Os componentes básicos da rede são uma placa de rede para cada micro, os cabos e o hub ou switch que serve como um ponto de encontro, permitindo que todos os micros se enxerguem e conversem entre si. As placas de rede já foram componentes caros, mas como elas são dispositivos relativamente simples e o funcionamento é baseado em padrões abertos, qualquer um pode abrir uma fábrica de placas de rede, o que faz com que exista uma concorrência acirrada que obriga os fabricantes a produzirem placas cada vez mais baratas e trabalhem com margens de lucro cada vez mais estreitas. As placas de rede mais baratas chegam a ser vendidas no atacado por menos de três dólares. O preço final é um pouco mais alto naturalmente, mas não é difícil achar placas por 20 reais ou até menos.

Fig. 01 - Placa de rede PCI

Temos três padrões de redes Ethernet: de 10 megabits, 100 megabits e 1 gigabit. As placas são compatíveis, mas, ao usar placas de velocidades diferentes, as duas vão conversar na velocidade da placa mais lenta.

As redes de 10 megabits são obsoletas, mas ainda é possível encontrar muitas instalações antigas por aí. Caso a rede já use cabos de categoria 5 (o número vem decalcado no cabo), é possível fazer um upgrade direto para 100 megabits, trocando apenas o hub e as placas.

Fig. 02 - Cabo de rede categoria 5e

Lembre-se de que a velocidade das placas é calculada em bits e não em bytes. Uma rede de 100 megabits permite uma taxa de transmissão (teórica) de 12.5 MB/s. Como além dos dados são transmitidas outras informações (a estrutura dos pacotes, retransmissões, códigos de correção de erros, etc.), a velocidade na prática fica sempre um pouco abaixo disso. Normalmente é possível transferir arquivos a no máximo 10.5 MB/s, com a taxa máxima variando sutilmente de acordo com a placa e o sistema operacional usado.

A opção para quem precisa de mais velocidade são as redes Gigabit Ethernet, que transmitem a até 1000 megabits (125 megabytes) por segundo. As placas gigabit atuais são compatíveis com os mesmos cabos de par trançado categoria 5, usados pelas placas de 100 megabits, por isso a diferença de custo fica por conta apenas das placas e do switch. Como hoje em dia a maioria das placas-mãe incluem chipsets de rede gigabit onboard e os switchs também estão caindo de preço, elas estão se tornando cada vez mais comuns.

Os cabos de rede também são um artigo relativamente barato. Os cabos de categoria 5, que usamos em redes de 100 ou 1000 megabits geralmente custam em torno de 80 centavos o metro, com mais alguns centavos por conector. Os cabos de categoria 5e são construídos dentro de normas um pouco mais estritas e normalmente custam o mesmo preço, por isso são sempre preferíveis.

Você pode comprar quantos metros de cabo quiser, junto com o número necessário de conectores, e crimpar os cabos você mesmo, ou pode comprá-los já prontos. É no caso dos cabos já crimpados que o preço começa a variar de forma mais expressiva. Algumas lojas chegam a crimpar os cabos na hora, cobrando apenas o valor do material, enquanto outras vendem os cabos por preços exorbitantes.

Fig. 03 - Cabos de rede de diferentes cores

Para crimpar os cabos de rede, o primeiro passo é descascar os cabos, tomando cuidado para não ferir os fios internos, que são frágeis. Normalmente, o alicate inclui uma saliência no canto da guilhotina, que serve bem para isso. Existem também descascadores de cabos específicos para cabos de rede.

Fig. 04- Descascando o cabo de rede usando a saliência no próprio alicate

É possível comprar alicates de crimpagem razoáveis por pouco mais de 50 reais, mas existem alicates de crimpagem para uso profissional que custam bem mais. Existem ainda "alicates" mais baratos, com o corpo feito de plástico, que são

mais baratos, mas não valem o papelão da embalagem. Alicates de crimpagem precisam ser fortes e precisos, por isso evite produtos muito baratos.

Os quatro pares do cabo são diferenciados por cores. Um par é laranja, outro é azul, outro é verde e o último é marrom. Um dos cabos de cada par tem uma cor sólida e o outro é mais claro ou malhado, misturando a cor e pontos de branco. É pelas cores que diferenciamos os 8 fios.

O segundo passo é destrançar os cabos, deixando-os soltos. Eu prefiro descascar um pedaço grande do cabo, uns 6 centímetros, para poder organizar os cabos com mais facilidade e depois cortar o excesso, deixando apenas a meia polegada de cabo que entrará dentro do conector. O próprio alicate de crimpagem inclui uma guilhotina para cortar os cabos, mas você pode usar uma tesoura se preferir.

Existem dois padrões para a ordem dos fios dentro do conector, o EIA 568B (o mais comum) e o EIA 568A. A diferença entre os dois é que a posição dos pares de cabos laranja e verde são invertidos dentro do conector.

Existe muita discussão em relação com qual dos dois é "melhor", mas na prática não existe diferença de conectividade entre os dois padrões. A única observação é que você deve cabear toda a rede utilizando o mesmo padrão. Como o EIA 568B é de longe o mais comum, recomendo-o que você utilize-o ao crimpar seus próprios cabos. Muitos cabos são certificados para apenas um dos dois padrões; caso encontre instruções referentes a isso nas especificações, ou decalcadas no próprio cabo, crimpe os cabos usando o padrão indicado.

No padrão EIA 568B, a ordem dos fios dentro do conector (em ambos os lados do cabo) é a seguinte:

1- Branco com Laranja

2- Laranja

3- Branco com Verde

4- Azul

5- Branco com Azul

6- Verde

7- Branco com Marrom

8- Marrom

Os cabos são encaixados nesta ordem, com a trava do conector virada para baixo, como neste diagrama:

Fig. 05 – Caracteristica conector RJ45

S você olhar o conector "de cima", vendo a trava, o par de fios laranja estará à direita e, se olhar o conector "de baixo", vendo os contatos, eles estarão à esquerda.

No caso de um cabo "reto" (straight), que vai ser usado para ligar o micro ao hub, você usa esta mesma disposição nas duas pontas do cabo. Existe ainda um outro tipo de cabo, chamado de "cross-over", que permite ligar diretamente dois micros, sem precisar do hub. Ele é uma opção mais barata quando você tem apenas dois micros. Neste tipo de cabo a posição dos fios é diferente nos dois conectores, de um dos lados a pinagem é a mesma de um cabo de rede normal, enquanto no outro a posição dos pares verde e laranja são trocados. Daí vem o nome cross-over, que significa, literalmente, "cruzado na ponta".

Para fazer um cabo cross-over, você crimpa uma das pontas seguindo o padrão EIA 568B que vimos acima e a outra utilizando o padrão EIA 568A, onde são trocadas as posições dos pares verde e laranja:

1- Branco com Verde

2- Verde

3- Branco com Laranja

4- Azul

5- Branco com Azul

6- Laranja

7- Branco com Marrom

8- Marrom

Esta mudança faz com que os fios usados para transmitir dados em um dos micros sejam conectados aos pinos receptores do outro, permitindo que eles conversem diretamente. A maioria dos hub/switchs atuais é capaz de "descruzar" os cabos automaticamente quando necessário, permitindo que você misture cabos normais e cabos cross-over dentro do cabeamento da rede. Graças a isso, a rede vai funcionar mesmo que você use um cabo cross-over para conectar um dos micros ao hub por engano.

Na hora de crimpar é preciso fazer um pouco de força para que o conector fique firme. A função do alicate é fornecer pressão suficiente para que os pinos do conector RJ-45 (que internamente possuem a forma de lâminas) esmaguem os fios do cabo, alcançando o fio de cobre e criando o contato. Você deve retirar apenas a capa externa do cabo e não descascar individualmente os fios, pois isso, ao invés de ajudar, serviria apenas para causar mau contato, deixando frouxo o encaixe com os pinos do conector.

Fig. 06 - Crimpando o cabo

É preciso um pouco de atenção ao cortar e encaixar os fios dentro do conector, pois eles precisam ficar perfeitamente retos. Isso demanda um pouco de prática. No começo, você vai sempre errar algumas vezes antes de conseguir.

Veja que o que protege os cabos contra as interferências externas são justamente as tranças. A parte destrançada que entra no conector é o ponto fraco do cabo, onde ele é mais vulnerável a todo tipo de interferência. Por isso, é recomendável deixar um espaço menor possível sem as tranças. Para crimpar cabos dentro do padrão, você precisa deixar menos de meia polegada de cabo (1.27 cm) destrançado. Você só vai conseguir isso cortando o excesso de cabo solto antes de encaixar o conector, como na foto:

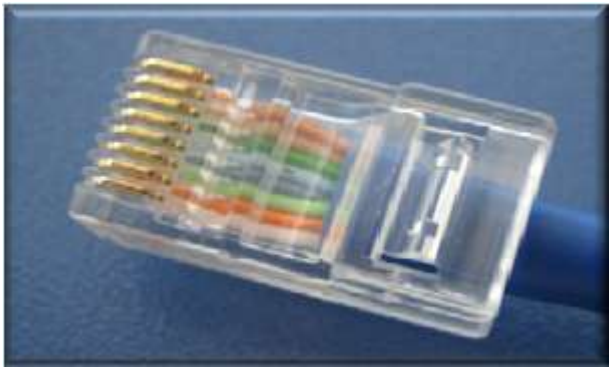


Fig. 07 – RJ 45 Crimpado

O primeiro teste para ver se os cabos foram crimpados corretamente é conectar um dos micros (ligado) ao hub e ver se os LEDs da placa de rede e do hub acendem. Isso mostra que os sinais elétricos enviados estão chegando até o hub e que ele foi capaz de abrir um canal de comunicação com a placa. Se os LEDs nem acenderem, então não existe o que fazer. Corte os conectores e tente de novo. Infelizmente, os conectores são descartáveis: depois de crimpar errado uma vez, você precisa usar outro novo, aproveitando apenas o cabo. Mais um motivo para prestar atenção. ;)

Os cabos de rede devem ter um mínimo de 30 centímetros e um máximo de 100 metros, distância máxima que o sinal elétrico percorre antes que comece a haver uma degradação que comprometa a comunicação.

Todas as placas são ligadas ao hub, ou ao switch, que serve como uma central, de onde os sinais de um micro são retransmitidos para os demais. É possível também ligar vários hubs ou switchs entre si (até um máximo de 7), formando redes maiores.

Fig. 10 - Um exemplo de hub/switch barato

A diferença entre um hub e um switch é que o hub apenas retransmite tudo o que recebe para todos os micros conectados a ele, é um tagarela. Isso faz com que apenas um micro consiga transmitir dados de cada vez e que todas as placas precisem operar na mesma velocidade (sempre nivelada por baixo, caso você coloque um micro com uma placa de 10 megabits na rede, a rede toda passará a trabalhar a 10 megabits).

Os switchs, por sua vez, são aparelhos mais inteligentes. Eles fecham canais exclusivos de comunicação entre o micro que está enviando dados e o que está recebendo, permitindo que vários pares de micros troquem dados entre si ao mesmo tempo. Isso melhora bastante a velocidade em redes congestionadas, com muitos micros.

Antigamente, existia uma grande diferença de preço entre os hubs burros e os switchs, mas os componentes caíram tanto de preço que a partir de um certo ponto a diferença se tornou insignificante, e os fabricantes passaram a fabricar apenas switchs, que por sua vez dividem-se em duas categorias: os switchs "de verdade", aparelhos caros, capazes de gerenciar o tráfego de uma quantidade maior de micros e que possuem várias ferramentas de gerenciamento e os "hub-switchs", os modelos mais simples e baratos, que usamos no dia-a-dia.

Configuração da rede

Assim como quase tudo na informática, as redes funcionam graças a uma mistura de hardware e software. A parte "física" da rede, que inclui as placas, cabos e switchs é responsável por transportar os sinais elétricos de um micro ao outro. Para que eles possam efetivamente se comunicar, é necessário utilizar um conjunto de normas e protocolos, que especificam como enviar informações e arquivos. Chegamos então ao TCP/IP, o protocolo comum que permite que computadores rodando diferentes programas e sistemas operacionais falem a mesma língua.

Pense nas placas, hubs e cabos como o sistema telefônico e no TCP/IP como a língua falada que você usa para realmente se comunicar. Não adianta nada ligar para alguém na China que não saiba falar Português. Sua voz vai chegar até lá, mas a pessoa do outro lado não vai entender nada. Além da língua em si, existe um conjunto de padrões, como por exemplo dizer "alô" ao atender o telefone, dizer quem é, se despedir antes de desligar, etc.

Ligar os cabos e ver se os leds do hub e das placas estão acesos é o primeiro passo. O segundo é configurar os endereços da rede para que os micros possam conversar entre si, e o terceiro é finalmente compartilhar a internet, arquivos, impressoras e o que mais você quer que os outros micros da rede tenham acesso.

Graças ao TCP/IP, tanto o Windows quanto o Linux e outros sistemas operacionais em uso são intercompatíveis dentro da rede. Não existe problema para as máquinas com o Windows acessarem a internet através da conexão compartilhada no Linux, por exemplo.

Independentemente do sistema operacional usado, as informações básicas para que ele possa acessar a internet através da rede são:

- Endereço IP: Os endereços IP identificam cada micro na rede. A regra básica é que cada micro deve ter um endereço IP diferente, e todos devem usar endereços dentro da mesma faixa.

O endereço IP é dividido em duas partes. A primeira identifica a rede à qual o computador está conectado (necessário, pois numa rede TCP/IP podemos ter várias redes conectadas entre si, veja o caso da internet), e a segunda identifica o computador (chamado de host) dentro da rede. É como se o mesmo endereço contivesse o número do CEP (que indica a cidade e a rua) e o número da casa.

A parte inicial do endereço identifica a rede, enquanto a parte final identifica o computador dentro da rede. Quando temos um endereço "192.168.0.1", por exemplo, temos o micro "1" dentro da rede "192.168.0". Quando alguém diz "uso a faixa 192.168.0.x na minha rede", está querendo dizer justamente que apenas o último número muda de um micro para outro.

Na verdade, os endereços IP são números binários, de 32 bits. Para facilitar a configuração e a memorização dos endereços, eles são quebrados em 4 números de 8 bits cada um. Os 8 bits permitem 256 combinações diferentes, por isso usamos 4 números de 0 a 255 para representá-los.

Todos os endereços IP válidos na internet possuem dono. Seja alguma empresa ou alguma entidade certificadora que os fornece junto com novos links. Por isso não podemos utilizar nenhum deles a esmo.

Quando você conecta na internet, seu micro recebe um (e apenas um) endereço IP válido, emprestado pelo provedor de acesso, algo como por exemplo "200.220.231.34". É através desse número que outros computadores na Internet podem enviar informações e arquivos para o seu.

Quando quiser configurar uma rede local, você deve usar um dos endereços reservados, endereços que não existem na internet e que por isso podemos utilizar à vontade em nossas redes particulares. Algumas das faixas reservadas de endereços são: 10.x.x.x, 172.16.x.x até 172.31.x.x e 192.168.0.x até 192.168.255.x

Você pode usar qualquer uma dessas faixas de endereços na sua rede. Uma faixa de endereços das mais usadas é a 192.168.0.x, onde o "192.168.0." vai ser igual em todos os micros da rede e muda apenas o último número, que pode ser de 1 até 254 (o 0 e o 255 são reservados para o endereço da rede e para o sinal de broadcast). Se você tiver 4 micros na rede, os endereços deles podem ser, por exemplo, 192.168.0.1, 192.168.0.2, 192.168.0.3 e 192.168.0.4.

- Máscara de sub-rede: A máscara é um componente importante do endereço IP. É ela que explica para o sistema operacional como é feita a divisão do endereço, Quais dos 4 octetos compõem o endereço da rede e quais contêm o endereço do host, isto é, o endereço de cada micro dentro da rede.

Ao contrário do endereço IP, que é formado por valores entre 0 e 255, a máscara de sub-rede é formada por apenas dois valores: 0 e 255, como em 255.255.0.0 ou 255.0.0.0, onde um valor 255 indica a parte do endereço IP referente à rede, e um valor 0 indica a parte do endereço IP referente ao host dentro da rede.

Se você está usando a faixa 192.168.0.x, por exemplo, que é um endereço de classe C, então a máscara de sub-rede vai ser 255.255.255.0 para todos os micros. Você poderia usar uma máscara diferente: 255.255.0.0 ou mesmo 255.0.0.0, desde que a máscara seja a mesma em todos os micros.

Se você tiver dois micros, 192.168.0.1 e 192.168.0.2, mas um configurado com a máscara "255.255.255.0" e o outro com "255.255.0.0", você terá na verdade duas redes diferentes. Um dos micros será o "1" conectado na rede "192.168.0", e o outro será o "0.2", conectado na rede "192.168".

- **Default Gateway (gateway padrão):** Quando você se conecta à internet através de um provedor de acesso qualquer, você recebe apenas um endereço IP válido. A princípio, isso permitiria que apenas um micro acessasse a web, mas é possível compartilhar a conexão entre vários micros via NAT, opção disponível tanto no Windows quanto no Linux.

Quando você compartilha a conexão entre vários micros, apenas o servidor que está compartilhando a conexão possui um endereço IP válido, só ele "existe" na internet. Todos os demais acessam através dele. O default gateway ou gateway padrão é justamente o micro da rede que tem a conexão, é ele que os outros consultarão quando precisarem acessar qualquer coisa na internet.

Por exemplo, se você montar uma rede doméstica com 4 PCs, usando os endereços IP 192.168.0.1, 192.168.0.2, 192.168.0.3 e 192.168.0.4, e o PC 192.168.0.1 estiver compartilhando o acesso à internet, as outras três estações deverão ser configuradas para utilizar o endereço 192.168.0.1 como gateway padrão.

- Servidor DNS: Memorizar os 4 números de um endereço IP é muito mais simples do que memorizar o endereço binário. Mas, mesmo assim, fora os endereços usados na sua rede interna, é complicado sair decorando um monte de endereços diferentes.

O DNS (domain name system) permite usar nomes amigáveis em vez de endereços IP para acessar servidores. Quando você se conecta à internet e acessa o endereço http://www.hardware.com.br/, é um servidor DNS que converte o "nome fantasia" no endereço IP real do servidor, permitindo que seu micro possa acessar o site.

Para tanto, o servidor DNS mantém uma tabela com todos os nomes fantasia, relacionados com os respectivos endereços IP. A maior dificuldade em manter um servidor DNS é justamente manter esta tabela atualizada, pois o serviço tem

que ser feito manualmente. Dentro da internet, temos várias instituições que cuidam dessa tarefa. No Brasil, por exemplo, temos a FAPESP. Para registrar um domínio é preciso fornecer à FAPESP o endereço IP real do servidor onde a página ficará hospedada. A FAPESP cobra uma taxa de manutenção anual de R$ 30 por esse serviço. Servidores DNS também são muito usados em intranets, para tornar os endereços mais amigáveis e fáceis de guardar.

Faz parte da configuração da rede informar os endereços DNS do provedor (ou qualquer outro servidor que você tenha acesso), que é para quem seu micro irá perguntar sempre que você tentar acessar qualquer coisa usando um nome de domínio e não um endereço IP. O jeito mais fácil de conseguir os endereços do provedor é simplesmente ligar para o suporte e perguntar.

O ideal é informar dois endereços, assim se o primeiro estiver fora do ar, você continua acessando através do segundo. Também funciona com um endereço só, mas você perde a redundância. Exemplos de endereços de servidores DNS são: 200.204.0.10 e 200.204.0.138.

Um exemplo de configuração de rede completa para um dos micros da rede, que vai acessar a internet através do micro que está compartilhando a conexão seria:

IP: 192.168.0.2
Máscara: 255.255.255.0
Gateway: 192.168.0.1 (o endereço do micro compartilhando a conexão)
DNS: 200.204.0.10 200.204.0.138

O micro que está compartilhando a conexão, por sua vez, terá duas placas de rede, uma para a internet e outra para a rede local, por isso vai ter uma configuração separada para cada uma. A configuração da internet é feita da forma normal, de acordo com o tipo de conexão que você usa, e a configuração da rede interna segue o padrão que vimos até aqui.

É possível usar também um servidor DHCP, que faz com que os clientes possam obter a configuração da rede automaticamente, a partir do servidor. Hoje em dia, mesmo os modems ADSL mais simples oferecem a opção de ativar um servidor DHCP, onde você só precisa especificar a faixa de endereços que será fornecida

aos clientes. Também é possível ativar o DHCP ao compartilhar a conexão, tanto no Linux, quanto no Windows.

Aqui temos um exemplo de configuração do servidor DHCP, num modem ADSL Kayomi LP-AL2011P. Assim como outros modems atuais, ele possui uma interface de administração que pode ser acessada via navegador, através de outro micro da rede:

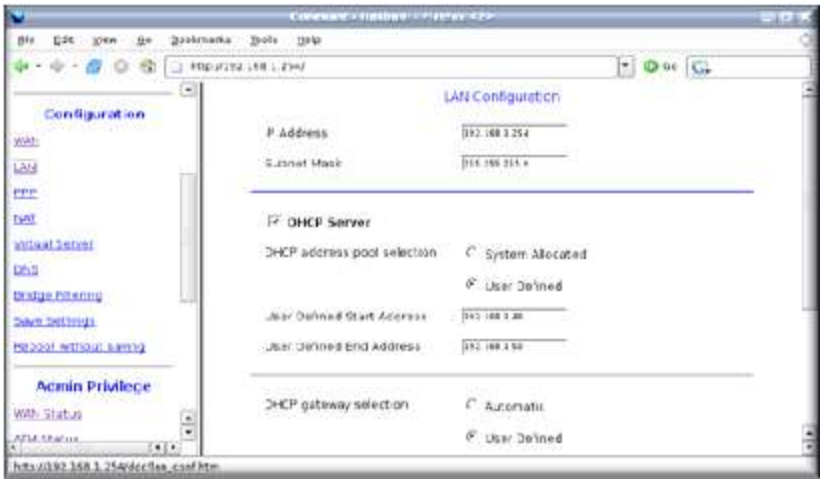


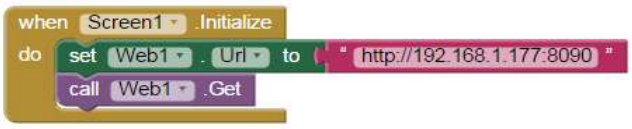
Fig. 11 - Configurando um servidor DHCP

Configurando o Shield Ethernet

Agora fica claro quando observamos o seguinte trecho da programação do arduino.

```
byte ip[] = { 192, 168, 1, 177 }; // Endereço de Ip do Shield Ethernet
EthernetServer server(8090); // Porta de serviço
```

Esse trecho configura o endereço IP que o shield ethernet irá assumir bem como a porta utilizada para a conexão. Quando inserimos a seguinte programação no App Inventor:



Estamos criando o link direto para a conexão do cliente com o adaptador de rede do projeto. O software no dispositivo móvel comunica com o arduino utilizando o shield de ethernet como meio de transmissão.

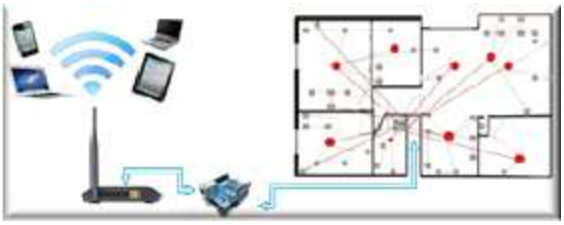


Fig. 12 - Aplicação

11.8 Anexo VIII - Redirecionamento de Portas

No acesso à Internet utilizamos protocolos e portas. Iremos estudar os dois protocolos (TCP e UDP) utilizados nessa comunicação.

O protocolo TCP

A comunicação pela internet é feita, basicamente, através de protocolos, sendo o TCP (**T**ransmission**C**ontrol **P**rotocol) um dos mais importantes deles. Isso porque o TCP está incluído no conjunto de protocolos que formam o TCP/IP, a base de comunicação via dados de toda a internet. De acordo com a definição dada por ***Júlio Battisti***, as principais características do TCP são:

- **Garantir a entrega de datagramas IP:** esta talvez seja a principal função do TCP, garantir que os pacotes sejam entregues sem alterações, sem terem sido corrompidos e na ordem correta. O TCP tem uma série de mecanismos para garantir esta entrega;
- **Executar a segmentação e o reagrupamento de grandes blocos de dados enviados pelos programas, garantir o seqüenciamento adequado e a entrega ordenada de dados segmentados:** esta característica refere-se ao recurso de dividir grandes arquivos em pacotes de dados menores e transmitir cada pacote separadamente. Os pacotes podem ser enviados por caminhos diferentes e chegar fora de ordem. O TCP tem mecanismos para garantir que, no destino, os

pacotes sejam ordenados corretamente, antes de serem entregues ao programa de destino.

- **Verificar a integridade dos dados transmitidos usando cálculos de soma de verificação:** o TCP faz verificações para garantir que os dados não foram alterados ou corrompidos durante o transporte entre a origem e o destino.
- **Enviar mensagens positivas dependendo do recebimento bem-sucedido dos dados. Ao usar confirmações seletivas, também são enviadas confirmações negativas para os dados que não foram recebidos:** no destino, o TCP recebe os pacotes de dados, verifica se estão ok e, em caso afirmativo, envia uma mensagem para a origem, confirmando cada pacote que foi recebido corretamente. Caso um pacote não tenha sido recebido ou tenha sido recebido com problemas, o TCP envia uma mensagem ao computador de origem, solicitando uma retransmissão do pacote. Com esse mecanismo, apenas pacotes com problemas terão que ser reenviados, o que reduz o tráfego na rede e agiliza o envio dos pacotes.
- **Oferecer um método preferencial de transporte de programas que devem usar transmissão confiável de dados baseada em sessões, como bancos de dados cliente/servidor e programas de correio eletrônico:** o TCP é muito mais confiável do que protocolos como o UDP (explicado adiante) e é indicado para programas e serviços que dependam de uma entrega confiável de dados.

O funcionamento do TCP é baseado em conexões. Assim, para um computador cliente iniciar uma "conversa" com um servidor, é necessário enviar um sinal denominado SYN para este último. O servidor então responde enviando um sinal SYN combinado com um sinal de nome ACK para confirmar a conexão. O cliente responde com outro sinal ACK, fazendo com que a conexão esteja estabelecida e pronta para a troca de dados. Por ser feita em três transmissões, esse processo é conhecido como three-way handshake (algo como triplo aperto de mãos).

O Protocolo UDP

O UDP (**U**ser **D**atagram **P**rotocol) é tido como um protocolo "irmão" do TCP, mas é mais simples e também menos confiável. Isso acontece porque o funcionamento do TCP é, como já dito, baseado em conexões, o que não ocorre com o UDP. Como

conseqüência, não há procedimentos de verificação no envio e recebimento de dados (todavia, pode haver checagem de integridade) e se algum pacote não for recebido, o computador de destino não faz uma nova solicitação, como acontece com o TCP. Tudo isso faz do UDP um pouco mais rápido, porém inutilizável em certas aplicações.

Por essas características, pode parecer que o UDP é inútil, mas não é. Há aplicações em que é preferível entregar os dados o mais rapidamente possível, mesmo que algumas informações se percam no caminho. É o caso, por exemplo, das transmissões de vídeo pela internet (streaming), onde a perda de um pacote de dados não interromperá a transmissão. Por outro lado, se os pacotes não chegarem ou demorarem a chegar, haverá congelamentos na imagem, causando irritação no usuário.

Portas TCP e portas UDP

Agora que você já conhece algumas características dos protocolos TCP e UDP, já está apto a entender o conceito de portas. Para uma compreensão mais fácil, usaremos o seguinte exemplo: suponha que, neste momento, você esteja usando um navegador de internet, um cliente de e-mail e um software de comunicação instantânea. Todas essas aplicações fazem uso da sua conexão à internet, mas como o computador faz para saber quais os dados que pertencem a cada programa? Simples, pelo número da porta que cada um utiliza. Por exemplo, se você está usando um programa de FTP (**F**ile **T**ransfer **P**rotocol), a conexão à internet é feita pela porta TCP 21, que é uma porta convencionada a este protocolo. Se estiver baixando arquivos pelo BitTorrent, uma das portas que vão de 6881 à 6889 estará sendo utilizada para tal atividade.

Compare seu computador a um prédio. Ao chegar uma correspondência, é necessário saber a qual apartamento entregá-la. Se no envelope estiver escrito que o destino é o apartamento número 123, onde reside Fulano, basta fazer a entrega. Em seu computador, o conceito é o mesmo: basta substituir a correspondência pelo pacote de dados, o apartamento pela porta e o Fulano pelo programa. No entanto, é importante frisar que um aplicativo pode utilizar mais de uma porta.

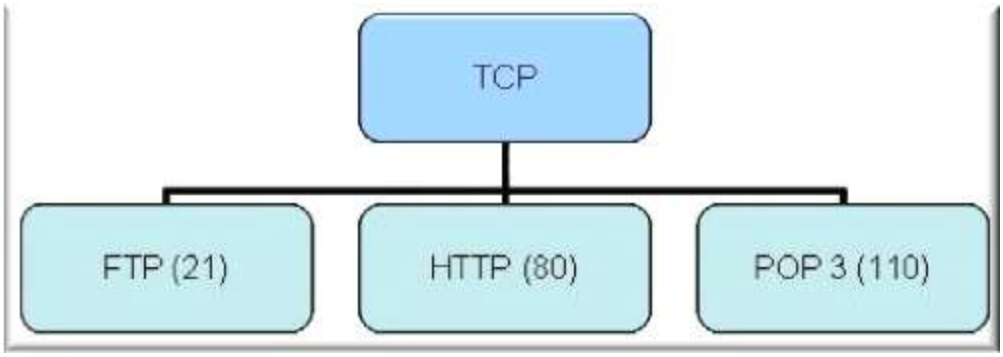


Fig.01 – Protocolo TCP e portas

Ao todo, é possível usar 65536 portas TCP e UDP, começando em 1. Tanto no protocolo TCP como no UDP, é comum o uso das portas de 1 a 1024, já que a aplicação destas é padronizada pela IANA (**I**nternet **A**ssigned **N**umbers **A**uthority). De acordo com essa entidade, eis algumas das portas TCP mais utilizadas:

21 - FTP; 23 - Telnet; 25 - SMTP;80 - HTTP; 110 - POP3; 143 - IMAP; 443 - HTTPS.

A IANA disponibiliza uma lista completa e atualizada da utilização das portas TCP e UDP no seguinte endereço: http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

Dependendo do caso, uma aplicação não precisa, necessariamente, estar restrita a um dado conjunto de portas. É possível utilizar outras, mas é necessário que isso seja especificado. É por isso, por exemplo, que há determinados endereços na internet que são disponibilizados assim: http://www.site.com:abcd, onde abcd é o número da porta. Neste caso, seu computador está sendo orientado a acessar o endereço pela porta abcd.

É graças ao conceito de portas que você consegue utilizar vários serviços ao mesmo tempo na internet. No entanto, isso também pode representar um perigo, razão pela qual é importante ter controle sob o tráfego de dados nas portas TCP e UDP. O uso de firewalls, por exemplo, ajuda a impedir que aplicações maliciosas utilizem portas abertas no computador para atividades prejudiciais. Além disso, um administrador de redes pode fazer configurações manuais para que determinadas portas fiquem bloqueadas, impedindo a conexão de aplicativos que fazem uso destas ou até mesmo que fiquem liberadas para a utilização de algum software ou recurso específico.

Configurando o redirecionamento de portas em um roteador

O redirecionamento de portas abre certas portas em sua rede doméstica ou empresarial, geralmente bloqueadas pelo roteador, para a Internet. Abrir portas específicas pode permitir que jogos, servidores, clientes de BitTorrent e outros aplicativos passem pela segurança do seu roteador que de outra forma não permite conexões a estas portas. Siga este guia para redirecionar as portas que você quer, não importa a sistema operacional.

No Sistema Operacional Windows abra o prompt de comandos através do menu iniciar/Todos os Programas/Acessórios/Prompt de Comandos ou através do teclado pressionando a tecla do Windows + a letra R e digitando CMD e clicando no botão [OK]. Feito isso aparecerá a seguinte janela:

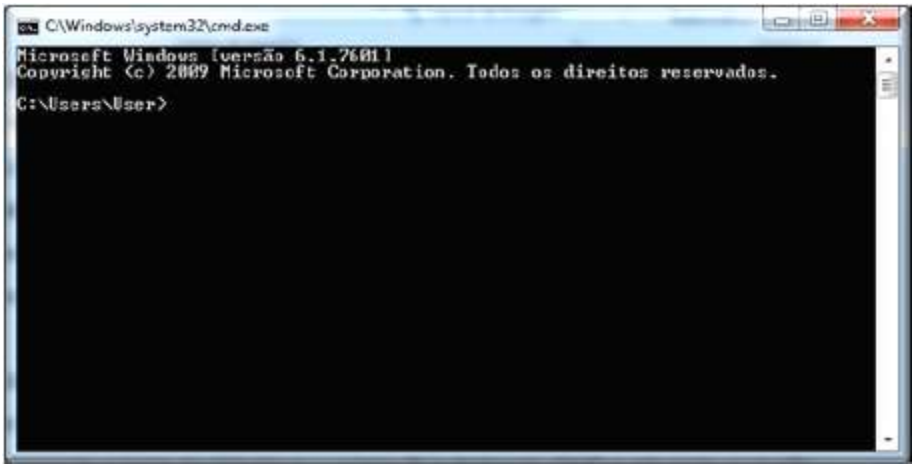


Fig. 02 – Prompt de Comandos

Digite ipconfig e pressione [ENTER].

Esse comando irá mostrar as configurações de endereço IP de sua rede conforme pode ser visto na figura 03 a seguir.

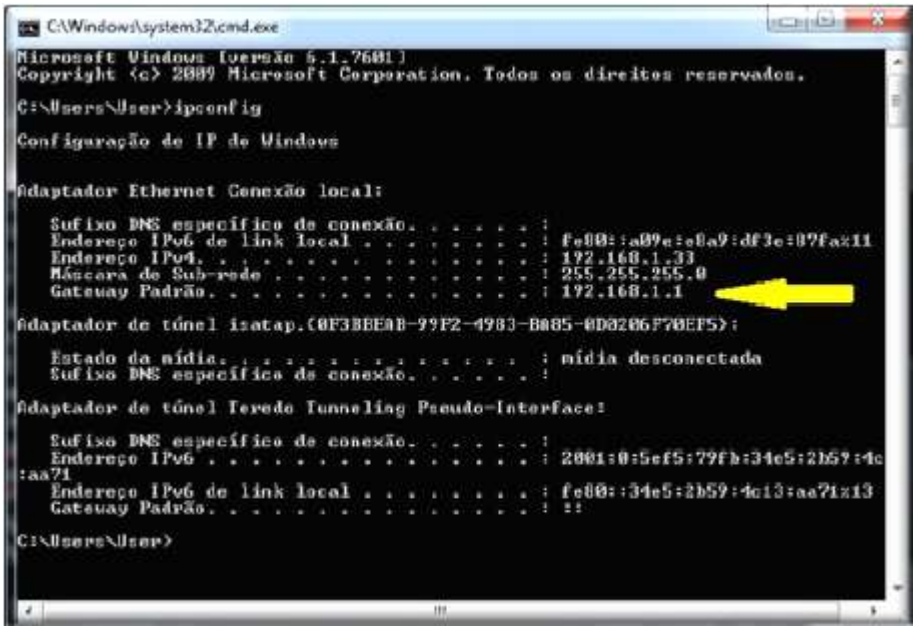


Figura 03 - ipconfig

Para o Mac: Abra o terminal e digite netstat -nr.
Para o Linux: Abra o terminal e digite route.

Observe os números que aparecem na linha de Gateway Padrão. No exemplo acima esses números são 192.168.1.1.

Tais números referem-se ao endereço IP do roteador da Internet ou seja, do equipamento que está fornecendo a conexão com a internet que pode ser um modem-roteador ou simplesmente um roteador.

Digite esse endereço em seu navegador. Será aberta uma página pedindo nome de usuário e senha. Caso já tenha configurado seu dispositivo anteriormente saberá o login a ser utilizado. Caso não saiba recorra ao manual do dispositivo ou procure no site do fabricante.

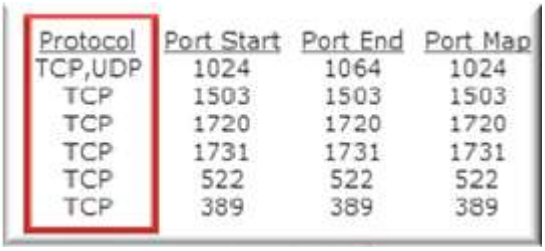
Encontre a seção de Redirecionamento de Portas (Port Forwarding). Cada roteador será levemente diferente. Os nomes mais comuns são Redirecionamento de Portas (Port Forwarding), Aplicativos (Applications), Jogos (Gaming), Servidores Virtuais (Virtual Servers). Se você não vir nenhuma destas opções ou algo similar, tente as

Configurações Avançadas (Advanced Settings) e procure por uma subseção Redirecionamento de Portas (Port Forwarding).

Encontre uma entrada pré-configurada. Muitos roteadores terão um menu dropdown com opções pré-configuradas para aplicativos conhecidos. Se você precisar abrir portas em um destes aplicativos, selecione ele na lista.

Se o programa que você quer adicionar não está na lista, você precisará criar uma entrada personalizada de redirecionamento de porta. Cada roteador tem uma forma levemente diferente de fazer isso, embora a informação necessária seja a mesma para qualquer roteador:

- Insira o nome do serviço. Insira um nome que seja relacionado ao programa para que você saiba para que aquela entrada serve.
- Escolha o tipo de serviço. Pode ser TCP, UDP, ou ambos. O tipo de serviço depende de qual programa você está desbloqueando. Se você não tiver certeza, escolha a opção TCP/UDP.



Protocol	Port Start	Port End	Port Map
TCP,UDP	1024	1064	1024
TCP	1503	1503	1503
TCP	1720	1720	1720
TCP	1731	1731	1731
TCP	522	522	522
TCP	389	389	389

Fig. 04 - Protocolos

Escolha as portas que você deseja usar. Se você quiser usar apenas uma porta, insira o mesmo número em Início (Start) e Fim (End). Se você deseja abrir um intervalo de portas (digamos, 5), você pode digitar 3784 em Início (Start) e 3788 em Fim (End).

Protocol	Port Start	Port End	Port Map
TCP,UDP	1024	1064	1024
TCP	1503	1503	1503
TCP	1720	1720	1720
TCP	1731	1731	1731
TCP	522	522	522
TCP	389	389	389

Fig. 05 - Portas

Escolha o endereço de IP interno para designar o redirecionamento da porta. Este é o endereço de IP do computador ou do hardware de rede que está executando o aplicativo que você está desbloqueando.

Em nosso exemplo do arduino com ethernet shield e App Inventor deverá ser utilizado o redirecionamento da porta 8090 para o endereço IP do shield de rede que é 192.168.1.177. Com isso é possível acessar a aplicação de fora da rede doméstica ou corporativa utilizando para isso a Internet.

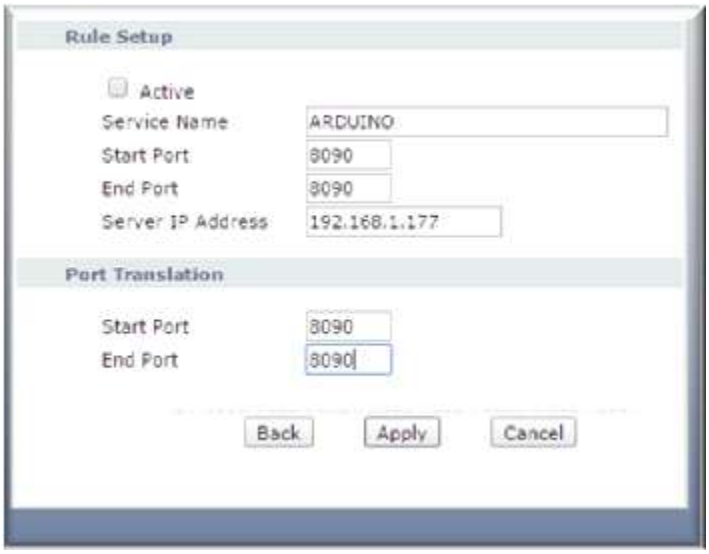


Fig. 06 – Redirecionando portas

Existe uma forma mais fácil de se redirecionar portas. Trata-se da opção DMZ. Uma **DMZ** ou **zona desmilitarizada** (do inglês **demilitarized zone** ou **DMZ**), também conhecida como **rede de perímetro**, é uma subrede física ou lógica que contém e expõe serviços de fronteira externa de uma organização a uma rede maior e não confiável, normalmente a Internet. Ao se ativar a opção DMZ e direcionar a qual IP ela será aplicada, todas as portas ou melhor todos os acessos externos passarão a ter acesso a tal IP sem nenhuma restrição ou bloqueio por firewall. Vale lembrar que o dispositivo fica mais vulnerável a ataques e neste caso se for um computador o dispositivo que se queira ter o acesso externo fica impraticável o uso deste recurso.

Fig. 07 - DMZ

11.9 Anexo IX – PLC - Power Line Communication

Power Line Communication – A informação que vem pela tomada

Quem já navegou pela internet com acesso de banda larga sabe o que é ter "o mundo em suas mãos". Os sistemas disponíveis no Brasil têm, em geral, um custo relativamente alto para implantação e velocidades razoáveis. Um dos fatores importantes na composição do preço é o custo da implantação da "última milha", a distância que separa o assinante do equipamento principal, que o conecta a um provedor de acesso. É apresentado, neste artigo, um sistema de transmissão em banda larga no qual o custo de implantação para o usuário final é muito mais baixo do que os sistemas tradicionais, já que aproveita uma infraestrutura existente em quase 100% dos lares brasileiros: a rede de energia elétrica para vencer a "última milha". Este sistema já vem sendo testado em várias partes do mundo, inclusive algumas empresas distribuidoras de energia elétrica no Brasil já fizeram testes (Cemig, Light, Eletropaulo).

• **PLC (Power Line Communication) –** Comunicação pela rede elétrica: sistema que modula a informação a ser transmitida sobre a energia elétrica, utilizando, assim, a rede elétrica existente como meio de transmissão. As taxas típicas dos sistemas em teste são da ordem de 2 Mbps, mas já existem fabricantes testando equipamentos com taxas de transmissão de dados até 200 Mbps.

Conceito de PLC

Na última década, vários grupos de pesquisa conseguiram desenvolver equipamentos com tecnologia para transmitir dados sem perdas e sem que as interferências externas inviabilizassem a instalação. A chave para resolver esta questão é a capacidade de processamento necessária, o que foi resolvido com o avanço dos processadores DSP (Digital Signal Processor), que permitem construir filtros, recuperar informações e modular dados praticamente em tempo real.

Em cima de uma senóide da rede elétrica, insere-se um sinal com a informação a ser transmitida, modulando a senóide. Este sinal composto (energia elétrica + dados) é enviado pela rede elétrica. Na recepção, filtros e processadores de sinais são utilizados para separar o que é energia elétrica do que é informação.

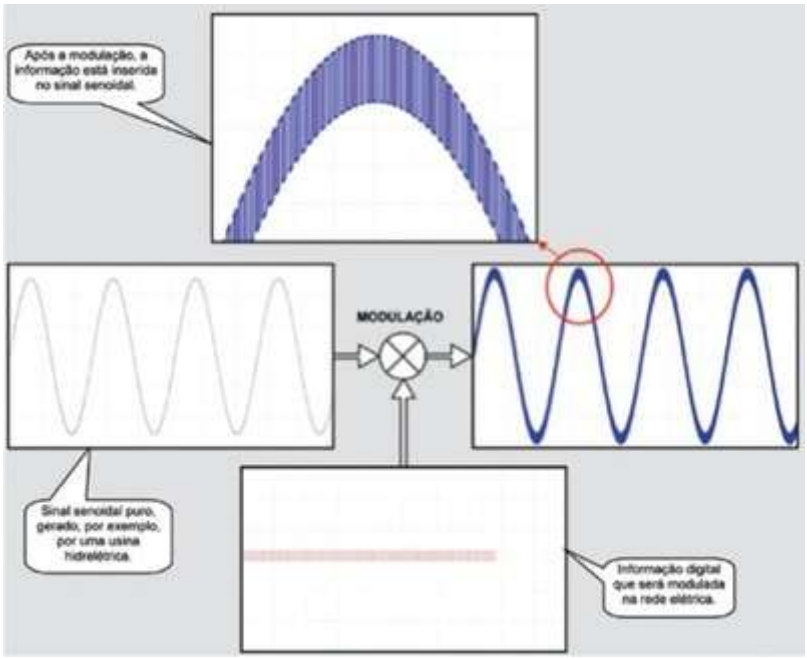


Figura 1 – Modulação da senóide

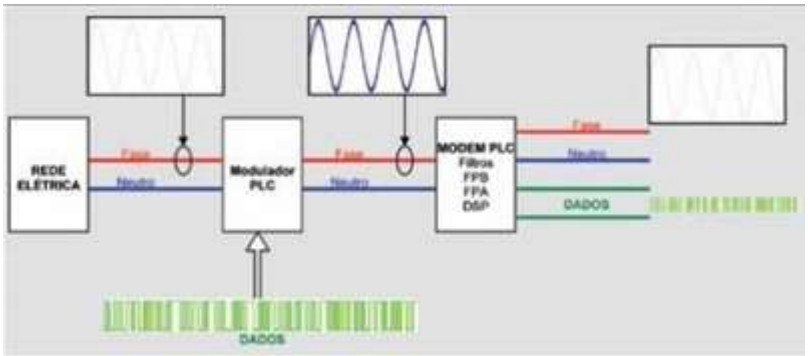


Figura 2 – Filtros e processadores separam a informação da energia elétrica

No entanto, interferências externas podem prejudicar a informação. Por utilizar a rede elétrica, a informação está sujeita a sofrer atenuações provenientes dos condutores elétricos e a receber diversos ruídos externos.

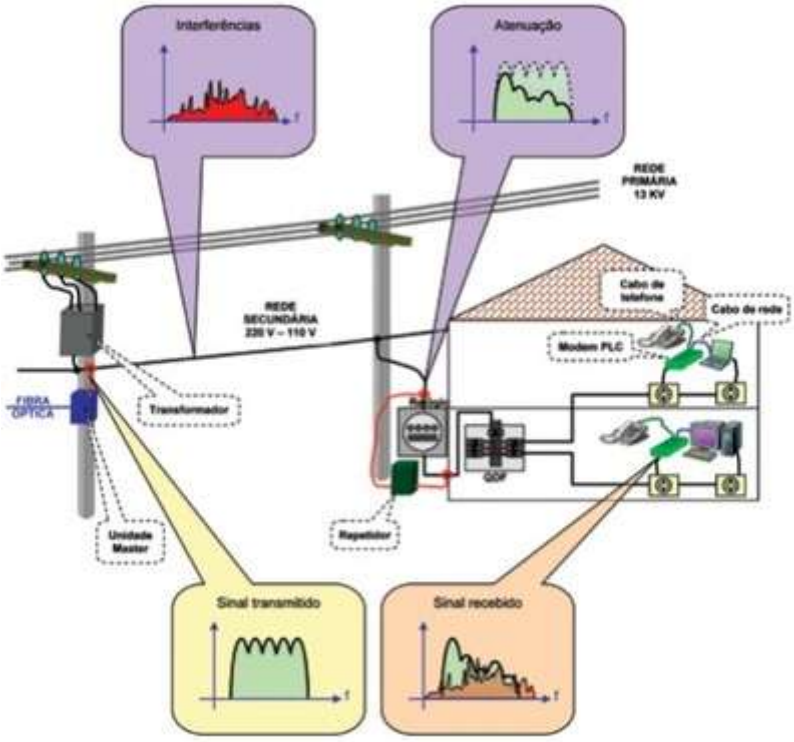


Figura 3 – Recepção de interferências externas

Os principais fatores de interferência para o sistema PLC são:

• **Linhas de transmissão desbalanceadas:** como são concebidas para transmitir energia elétrica e não dados, as linhas de transmissão não têm balanceamento e não são simétricas. Isto potencializa as características de antenas que as linhas de transmissão têm. Para transmitir dados em alta frequência, serão emitidas ondas eletromagnéticas e, com o efeito de antena, geram interferências em outros sistemas. Além disso, como em toda antena, os fios receberão interferências vindas de outros sistemas. Em resumo: sistemas com cabos aéreos geram e recebem interferências eletromagnéticas. No Japão, as redes PLC ainda não são permitidas em ambientes externos, por causa da

interferência constatada em sistemas de rádio amador. A parte externa é conectada por meio de fibra ótica e levada até a parte interna, onde, finalmente, é distribuída por meio do sistema PLC.

• **Atenuação do sinal de alta frequência:** na parte encapada da rede elétrica, a utilização de plásticos cria efeitos capacitivos. Estes atenuam sinais de alta frequência, o que é um grande problema para transmissões de longa distância. Por isso, as aplicações típicas do PLC referem-se à última milha, que corresponde, do transformador da rua até a casa do assinante, sempre em distâncias curtas (inferiores a 1,0 km). Para a transmissão em longas distâncias, costuma-se utilizar a conexão por meio de fibras óticas.

• **Eletrodomésticos:** quem nunca ligou o liquidificador em casa e notou que a imagem da TV recebeu uma série de chuviscos indesejados? E o secador de cabelos? Todos os eletrodomésticos que utilizam motores ou que fazem chaveamentos "sujam" a rede com interferências. O simples fato de ligar e desligar uma lâmpada também gera interferências. Isto sem falar nas lâmpadas compactas fluorescentes (lâmpadas PL ou econômicas), que usam circuitos eletrônicos para chavear a energia, sujando a rede. Estas distorcem a parte de dados que está sendo modulado na rede, aumentando a necessidade de processamento de dados para corrigir as perdas causadas. A consequência é a perda da eficiência de transmissão, o que acarretará em taxas mais baixas.

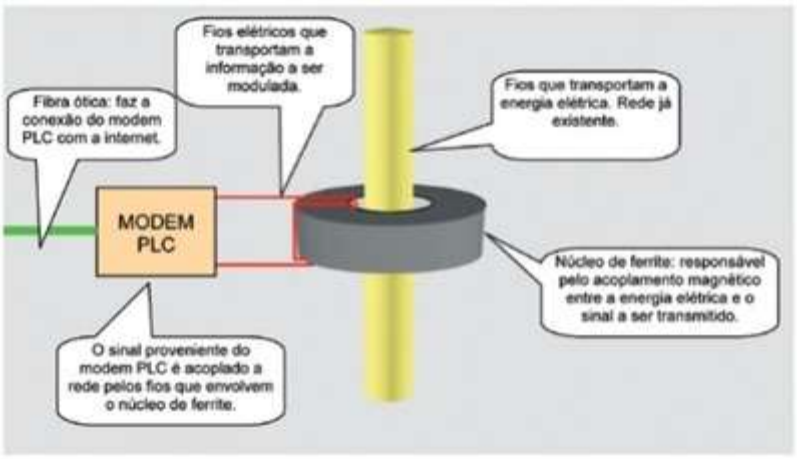


Figura 4 – Sinal, com a informação, sendo induzido no ferrite

• **Grande atenuação nos circuitos elétricos:** A queda de tensão existente nos circuitos elétricos para as altas frequências, decorrente do uso de condutores metálicos e suas características, provoca atenuações na tensão modulada pelo sistema PLC, degradando a relação sinal ruído. A atenuação típica é da ordem de 50 dB, como mostra a Figura 5.

• **Elementos da rede elétrica:** transformadores, emendas de cabos, relógios medidores, disjuntores, etc. Todos antigos e sem manutenção são mais elementos que contribuem como fonte de interferência para a transmissão de dados, além de bloquearem a passagem de dados em altas frequências, o que diminui a taxa de transmissão efetiva. Para superar estes desafios e conseguir as taxas de transmissões atuais, foram desenvolvidas técnicas de modulação, proteção e tratamento de erros, mostradas a seguir.

Como funciona

O primeiro passo é conseguir modular a informação sobre a energia elétrica. O método mais utilizado e que obtém resultados mais eficientes é com anéis de ferrite. Da mesma maneira que estes anéis podem ser utilizados como atenuadores de interferências externas, como no caso de monitores, eles também podem ser empregados na modulação de sinais.

O sinal contendo a informação é induzido no ferrite, como acontece no acoplamento magnético de um transformador. Este, por sua vez, envolve o cabo de energia, que também será induzido pelo sinal modulado.

Os pontos onde serão aplicados os núcleos de ferrite dependem da configuração de cada rede PLC. A rede elétrica tem sempre três níveis de tensão elétrica e, a cada um destes níveis, tem-se um tratamento diferenciado para as redes PLC:

Os pontos onde serão aplicados os núcleos de ferrite dependem da configuração de cada rede PLC. A rede elétrica tem sempre três níveis de tensão elétrica e, a cada um destes níveis, tem-se um tratamento diferenciado para as redes PLC:

Transmissão dos dados

• **Alta tensão:** em linhas de transmissão, em que valores superiores a 13 KV são comuns. A rede PLC ainda não é transmitida aqui, pois, entre outros problemas, os transformadores de alta tensão atenuam muito o sinal a ser transmitido, o que inviabiliza seu uso.

• **Média tensão:** é a tensão que sai das subestações e é transmitida até os bairros, passa por transformadores que abaixam a tensão para o uso residencial e industrial. São tensões da ordem de 13 KV até 220 V. Neste ponto, a rede PLC insere os seus dispositivos MV (Medium Voltage), que tem conexão com a internet, com a rede telefônica (no caso de se transmitir VoIP), ou qualquer outra fonte de dados que se queria transmitir. Os dispositivos MV fazem a conexão com o provedor de acesso via fibra ótica, ou qualquer outro meio de transmissão de alta capacidade. Veja que a fibra ótica deve ligar somente as subestações aos provedores de acesso, sem a necessidade de chegar à casa do assinante, diminuindo os custos de instalação. Nada impede que outro meio de transmissão seja utilizado para conectar o MV ao provedor, como o uso de um link de rádio, cabos coaxiais, etc.

• **Baixa tensão:** é a tensão final entregue ao usuário. Pelos transformadores localizados nos postes da rua, a tensão elétrica chega aos 110 V ou 220 V e são entregues aos assinantes. Se a distância entre os dispositivos MV e o assinante excede 500 m, são instalados repetidores, a fim de diminuir a atenuação que o sinal modulado recebe. A maioria dos testes feitos no Brasil foi feito em baixa tensão.

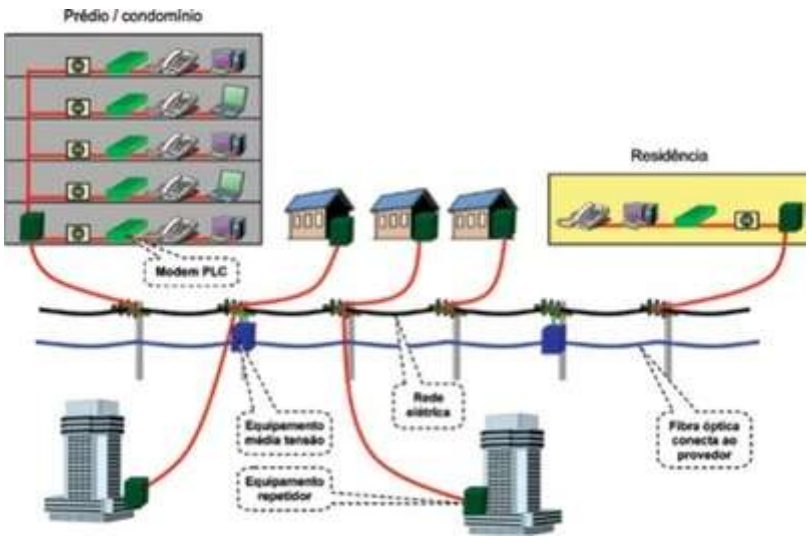


Figura 5 – Representação da transmissão de dados

Recepção dos dados

• **Casa/escritório do assinante:** aqui a tensão já tem o seu nível de uso pelos equipamentos domésticos e o sinal de banda larga já foi modulado na rede elétrica. São instalados os modens PLC, com a capacidade de separar o sinal de baixa frequência (energia elétrica) do sinal de alta frequência (dados, vídeo, voz, etc.). O modem irá decodificar o sinal recebido e fornecer uma saída no padrão da aplicação (rede, voz, controle, etc.).

Padrões testados mundialmente

O desenvolvimento desta tecnologia está ocorrendo simultaneamente em diversos grupos de pesquisas. Atualmente, países como Espanha, Alemanha, Suíça, Inglaterra e Japão já realizaram testes com redes PLC e contam com sistemas comerciais implantados, operando a todo vapor, fornecendo dados de alta velocidade pela rede elétrica.

Métodos de modulação

A fim de minimizar os efeitos das várias interferências que a rede elétrica recebe, várias técnicas de modulação foram testadas. O processamento digital dos sinais (DSP) é amplamente empregado. A técnica em que se conseguiu a maior eficiência de transmissão foi a OFDM (Orthogonal Frequency Division Multiplexing). A ideia básica aqui é que uma informação modulada em uma portadora (faixa de frequência bem definida) é muito sujeita a sofrer interferências e tem a recuperação de erros prejudicada. Para minimizar estes problemas, a primeira coisa que se tentou foi espalhar a informação, usando a técnica de Spread Spectrum (espalhamento espectral), a mesma utilizada nos telefones celulares de tecnologia CDMA. Isto, porém, não mostrou eficiência suficiente para viabilizar o uso do PLC para transmissão em distâncias superiores a 500 m.

Figura 6 – Técnica OFDM de modulação

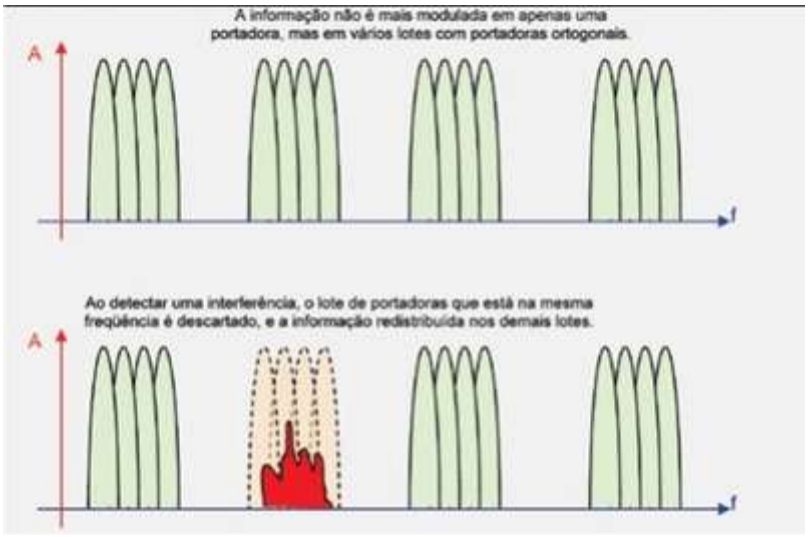


Figura 7 – Técnica CDMA de modulação

A solução foi não usar apenas uma portadora, mas sim dividir a informação em múltiplas portadoras, de modo que uma interferência corrompa apenas uma parte da informação. Isto também não mostrou a eficiência necessária para a transmissão de sinais em banda larga. Finalmente, fez-se o conjunto de portadoras ser distribuído ao longo de quase toda a banda de frequências disponíveis em lotes de portadoras ortogonais e criaram-se mecanismos para que as interferências fossem monitoradas. Ao se detectar uma interferência que ocupa determinada largura de banda, o lote de portadoras que está na mesma frequência é descartada e a informação é repetida nos demais lotes, fazendo a informação não ser perdida.

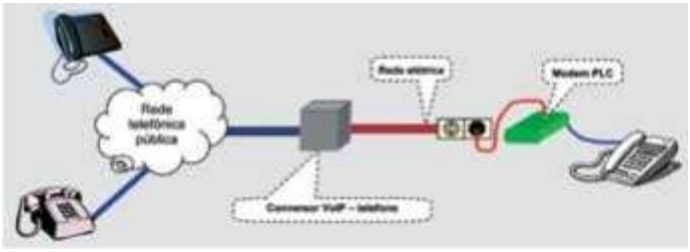


Figura 8 – Informação modulada em várias portadoras

VANTAGENS E DESVANTAGENS

Como toda tecnologia em desenvolvimento, o PLC apresenta suas vantagens e desvantagens. Dentre as vantagens podemos citar:

• Grande alcance e penetração, devido ao fato de aproveitar as conexões elétricas já existentes;

• Provavelmente é a alternativa com menor custo de instalação entre as existentes em banda larga;

• Uma excelente maneira para as distribuidoras de energia elétrica aumentar sua fonte de rendimentos;

• Muito fácil de instalar, ampliar e de realizar a manutenção.

Apresenta também algumas desvantagens, sendo que alguma delas compromete seu desempenho:

• A tecnologia ainda não está completamente padronizada, o que faz vários grupos de pesquisa desenvolva seus próprios padrões;

• É uma fonte potencial de interferência em sistemas de rádio, podendo causar problemas com bandas de frequências que já estão licenciadas para outros usos;

• É uma grande concorrente das redes DSL e CATV, o que pode gerar conflito de interesses entre grupos.

APLICAÇÕES

Domótica

O conceito de domótica, que é a automação das funções de uma residência resultando em casas inteligentes, está ligado ao uso de equipamentos que tenham características inteligentes, com sensores para medir as condições do ambiente e que possam interagir com uma central de comando, enviando e recebendo sinais. Para obter este nível de controle, era necessária a construção de uma rede de transporte dos dados dos sensores até a central de controle.

Com o PLC, esta rede não precisa ser instalada, diminuindo o custo do projeto, pois todos os equipamentos eletrônicos já são conectados à rede elétrica, o que facilita a interligação de todos os elementos.Basta então que os equipamentos da automação tenham módulos PLC embutidos, possibilitando que todos os elementos se comuniquem com uma central. Alguns fabricantes já desenvolvem estes módulos.

Segurança

Câmeras de segurança já são vendidas em versões que contém um modem PLC. Deste modo, todas as imagens são enviadas para qualquer ponto de energia no local em que a câmera está instalada, sem a necessidade de passar fios ou cabos. Na recepção, outro modem é utilizado e o sinal de vídeo é recuperado.

IP fone

O uso de VoIP (voz sobre IP) por meio da internet tem se popularizado com programas como o Skype ou Messenger. Já existem projetos para telefones VoIP utilizando

modens PLC. Para isso, basta ligar o telefone ao modem, este à tomada elétrica e a conexão entre o modem e o provedor de acesso VoIP será feita automaticamente. Com uma tarifa bem menor que a de uma ligação normal, você terá acesso a chamadas para o mundo todo. Caso uma ligação seja feita entre dois terminais PLC, o custo é zero.

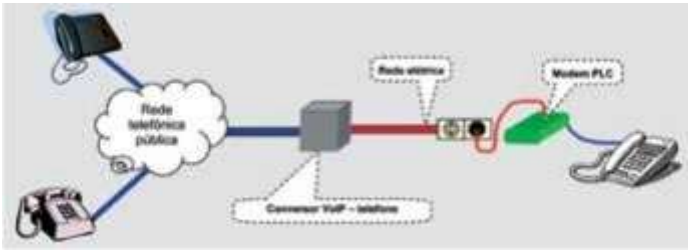


Figura 9 – Esquema de funcionamento da aplicação do PLC em IP fone

Música e entretenimento

Que tal criar um som ambiente a partir de um CD player em sua casa, sem a necessidade de passar toda aquela fiação necessária para a transmissão do áudio? A empresa tailandesa ST&T já fornece um sistema como este. Nele você liga à saída de áudio RCA de seu equipamento (micro system, CD player, reciever, home theater, etc.) a entrada de um transmissor PLC (veja figura 16 O sinal de áudio será modulado por toda a rede elétrica de sua residência

Onde existir uma tomada elétrica pode ser colocado dois tipos de receptores: um com um pequeno amplificador de potências, que modula o sinal de áudio e o fornece em uma saída em que pode ser ligado a caixas de som externas. Ou então um modem que já é uma caixa de som, com controle de volume, graves e agudos.

Figura 10 – Entrada de um transmissor PLC de equipamento tailandês

Assim, o som ambiente é obtido por um conjunto de equipamentos formado por um transmissor e quantos receptores você precisar, colocando ou não caixas externas e em quantos pontos existam tomadas elétricas.

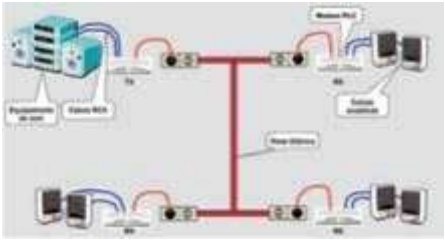


Figura 11 – Esquema para uso de PLC em sistemas de som

Redes de computadores

Além da função básica de transmitir dados de banda larga, com acesso à internet desde o provedor até a residência, já existem aplicações do PLC para a montagem de redes de computador, em que os cabos de rede são substituídos pelos cabos de energia elétrica já existentes. Para que isso? Imagine a situação: em sua casa, onde você já tem acesso à internet de banda larga, os computadores ficam no escritório, na parte superior de um sobrado.

Entretanto, você deseja colocar um ponto de rede em um quarto localizado na edícula ao fundo do terreno. Todo o cabeamento deve ser passado, o que tem um custo alto e demanda mão-de-obra especializada. Isso não é mais necessário usando pares de modens PLC, que conectam o computador à rede elétrica. O sistema permite o aumento do número de pontos de rede até certo limite (típico em 16 pontos), possibilitando a montagem de uma rede de pequeno porte e pequeno alcance (da ordem de 300 m), mas com a facilidade de não precisar passar cabos e ser de fácil instalação.

Medição automática de consumíveis

A empresa Senergy já implantou em conjuntos da Companhia de Desenvolvimento Habitacional e Urbano (CHDU) um sistema de leitura de água que utiliza a tecnologia PLC. Em parceria com a Sabesp e com o Governo do Estado de São Paulo, este sistema permite que o consumo seja acompanhado em tempo real em uma central de monitoramento, dispensando a leitura humana e detectando variações no padrão de consumo que pode indicar a existência de vazamentos e defeitos na rede.

Para isso, o hidrômetro é dotado de um modem PLC e conectado à entrada da rede elétrica do apartamento. Na caixa de entrada de energia elétrica do condomínio, fica uma unidade concentradora, que faz a leitura de todos os modens conectados e envia estas informações para a central de controle, que, por sua vez, faz o monitoramento do consumo e emite as contas de água. Diversos fabricantes de medidores já estão incorporando os modens PLC em seus leitores.

CONCLUSÃO

A tecnologia PLC tem um futuro promissor, já é usada em algumas partes do mundo, mas ainda tem algumas barreiras a vencer. Não deve ser encarada como uma tecnologia que vem para substituir as outras já existentes no acesso à banda larga, mas sim como mais uma que pode agregar valor a essa tecnologia e, principalmente, como uma alternativa de baixo custo para a grande massa da população.

Mesmo que não seja utilizada para trazer a informação desde o provedor até o usuário final, já estão disponíveis equipamentos em nível comercial para criar mini redes PLC em casa ou no escritório, solucionando, de forma barata, as construções de rede.

Referências

http://www.xeline.com/

http://www.echelon.com

http://www.powerlinecommunications.net/

http://www.polytrax.com/

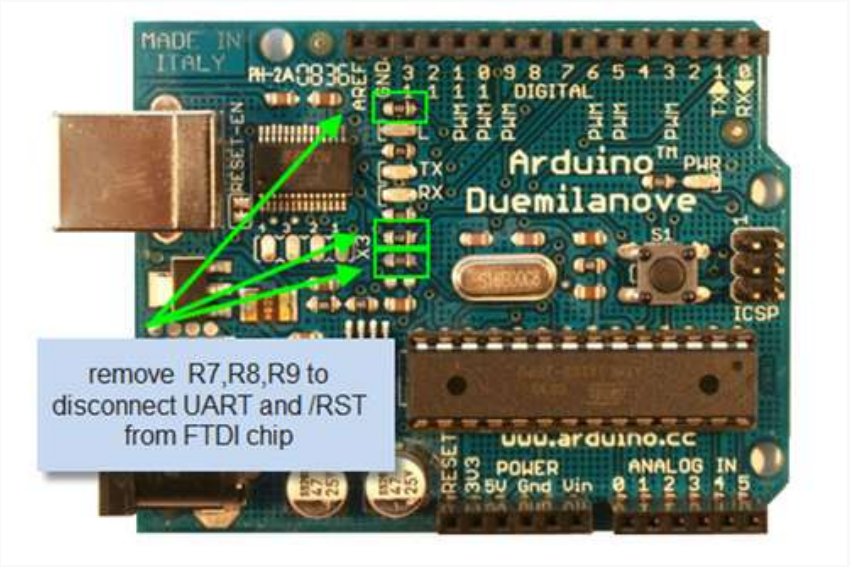
http://www.archnetco.com

http://www.upaplc.com

11.10 Anexo X – Utilização do Yun Shield com outros modelos

Conectar-se ao Arduino Duemilanove/Diecimila

1) Com Duemilanove / Diecimila, a interface mega avr uart é conectado ao chip FTDI, temos que desligá-los como mostrado na figura abaixo:



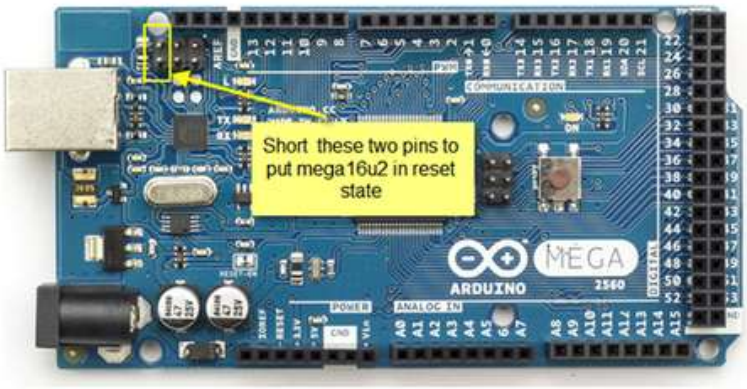
Use com Arduino Duemilanove / Diecimila

2) Use mesmo perfil Board como UNO no Arduino IDE

3) Coloque o Shield Yun no topo do Duemilanove e ligue-o via jack DC.

Conectar-se ao Arduino Mega2560

1) Em Mega2560, a conexão entre uart mega-2560 e atmega16u2 irá influenciar o recurso de ponte com Yun Shield. Então nós temos que desligá-lo pelo conjunto da mega16u2 em modo **de reposição. Como abaixo:**

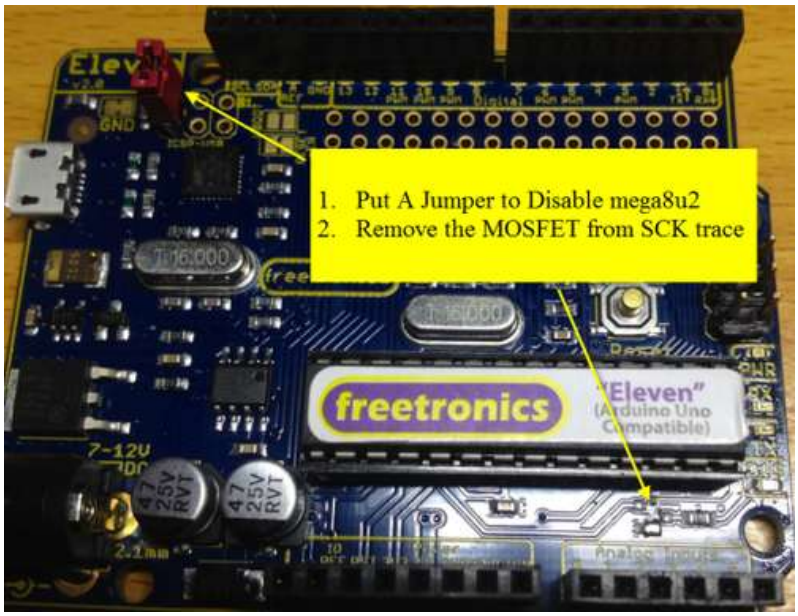


Use com Arduino mega 2560

2) Encaixe o Shield Yun na parte superior do mega 2560 e ligue-o via jack DC.

Conectar-se ao Freetronic "Eleven"

Enquanto conectar a Freetronic "Onze", o usuário pode utilizar o mesmo perfil para se conectar a UNO. mas precisa fazer a seguir modificação no hardware, que é um pouco diferente do Arduino Uno.



Conectar-se ao Dccduino UNO R3

Use com Dccduino UNO R3

11.11 Anexo XI – Programação da Pagina Web (capítulo 8.7)

<input type="text" style="text-align: center;" name="analogA0" id="analog0" value="0" size="6" readonly/> **Esta é a caixa de texto abaixo "Temperatura em C". O importante a notar aqui é o id e o valor. O script requer que tenhamos um id e um valor para que pudéssemos fazer referência a ela em um script e alterar o seu valor.**
** **Esta é a imagem abaixo da entrada digital 2. A imagem atual é led_off.jpg, mas assim que o script será executado o status de pino 2 mudará para 1 dependendo clique no botão, a imagem de origem será substituído por led_on. jpg. O script irá referenciar o ID de image2 para mudar a imagem.**
<input type="hidden" name="pin" value="3" id="pin3" /> *<input type="range" style="width: 50px; height: 30px;" id="dac3" min="0" max="255" value="0" step="1" onchange="sendDac(document.getElementById('pin3').value,* this.value*);"/>* *0* **Estes códigos estão representando o controle deslizante DAC3. O primeiro campo que não é mostrado significa que você não vai ver nada, mas é importante porque tem um valor e um id que será referenciado mais tarde. A segunda linha é o código do controle deslizante como representado pela entrada = range. O valor inicial é 0 e o valor final é de 255 e o passo é 1. 0 e 255 representa o valor que o PWM pode assumir. Além disso, movendo o controle deslizante fará com que um evento "sendDAC", que irá enviar o valor atual do controle deslizante com o valor de pin3 para o script. Vamos discutir os scripts mais tarde. valueDac3 (o número abaixo do controle deslizante) tem atualmente um valor de 0, mas será substituído por outro valor dependendo da saída do script.**
<input type="hidden" name="pin" value="12" id="pin12" /> *<input type="hidden" name="action" value="0" id="action12" />* ** **O código acima representa o pino de saída digital 12. A primeira linha representa o valor de pin12 a ser referenciada mais tarde. A segunda linha representa a ação que será tomada quando eu clicar a imagem. A ação vai ser 0 ou 1. Inicialmente, a imagem será verde escuro como o valor de action12 é 0. Mais tarde, a imagem de origem será alterado para on.jpg (verde sólido) carregando o evento "sendButton ", que envia o valor de pin12 e o valor de action12 para o script.**
* – *

Resposta do Arduino na descrição do ID de modo que podemos ver a comunicação.

Vamos dar uma olhada na maneira como a página irá se comportar quando ele recebe os eventos.

```
window.onload=Pinstatus;
function Pinstatus(){
morestatus();
}
```

A parte de cima é a parte do roteiro do código html. Quando a janela será carregado no usuário, o script "Pinstatus" será executado. Este script chama outro script "morestatus".

```
function morestatus(){
setTimeout(morestatus, 2000);
document.getElementById("description").innerHTML = "Processing Status";
server = "/arduino/status/99";
request = new XMLHttpRequest();
request.onreadystatechange = updateasyncstatus;
request.open("GET", server, true);
request.send(null);
}
```

O tempo limite no script faz com que ele seja executado a cada 2 segundos. Você se lembra do campo de descrição na seção anterior no final. Essa descrição será substituída por " *Processing Status* **" na página pelo atributo innerHTML. O "pedido = new XMLHttpRequest ()" cria um novo pedido e em uma mudança de estado, ele chama um outro script "updateasyncstatus". "Request.open (" GET ", servidor, true)" cria uma solicitação GET com um valor de servidor que em quantidades totais para "http: // endereço IP do Yun / arduino / status / 99". Além disso, o True indica que ele é um pedido assíncrono que significa que a página não será necessário para ser recarregado. "Request.send (null)" envia a solicitação criada para o Yun.**

O updateasyncstatus script atualiza o estado da página ou seja, todas as imagens que falávamos antes, as caixas de texto, os controles deslizantes, etc.

```
function updateasyncstatus(){
if ((request.readyState == 4) && (request.status == 200))
{
result = request.responseText;      //result will store the response received from Arduino
document.getElementById("description").innerHTML = result; //description will now show the response received
fullset = result.split("#");
```

A resposta que recebemos do Arduino será do status formato # 2 = 0 # 12 = 0 # 13 = 1 # 3 = 190 # A0 = 24. O que o result.split vai fazer é dividir a string em "#" em uma matriz de "pin e seu valor" pair. Assim, a matriz será semelhante 2 = 0,12 = 0,13 = 1, 3#190 # A0 = 24. fullset irá armazenar essa matriz e você pode chamar qualquer entrada com base em fullset [posição de entrada].

No seguinte "loop", vamos separar ainda mais os valores para torná-los utilizáveis. O loop será executado enquanto o comprimento da resposta recebido ou seja, 5.

```
for(i = 1; i
PinPair = fullset[i];                              //for the value of i=1, PinPair will be "2=0"
singleset = PinPair.split("=");       //this will cause PinPair to split at "=" into an array of [2,0]
PN = singleset[0];                                  //PN will hold the pin number,example 2
Pinstatus = singleset[1];                    //this will hold the status of pin 2, for example 0
if (PN > 11)                                  //  if pin number is greater than 11 i.e. pin 12 and 13
{
ActNum = "action" + PN;                                  // eg for pin 12, ActNum=action12
ImgNum = "image" + PN;                                                  // ImgNum=image12
if (Pinstatus == 0)
{
PinAct = "1";         //If the current status of pin 12 is 0, set image to off.jpg and PinAct to 1.
image = "off.jpg";
```

document.getElementById (ActNum) .value = PinAct "

Irá alterar o valor de action12 em HTML principal para 1. O que isto vai fazer é a próxima vez quando você clica no IMAGE12 aciona pin12, ele irá enviar um valor de 1, como atualmente o seu valor é 0. Isto o torna um interruptor.

document.getElementById (ImgNum) .src = image"

Vai mudar a imagem do botão 12 para ser vermelho (off.jpg) como o status pino é 0.

```
if (PN == 2)
{
ImgNum = "image" + PN;
if (Pinstatus == 1)
{
image = "led_on.jpg";
}
else
{
image = "led_off.jpg";
}
document.getElementById(ImgNum).src = image;
}
```

No caso de o pino 2, uma vez que não é um interruptor, que não requerem PinAct. Se o status de Pin é 1, image2 no HTML será substituído por led_on.jpg (verde sólido) e vice-versa.

```
if (PN == 3 )
{
PinVal = parseInt(singleset[1]);  //Since pin 3 is analog output, it has values between 0 and 255. This integer value will need to be extracted from the array using parseInt() in order to be usable.
DacNum = "dac" + PN;                                    //dac3
ValNum = "valueDac" + PN;                               //valueDAC3
document.getElementById(DacNum).value = PinVal;
```

Isto irá mostrar o valor de value DAC3 abaixo o controle deslizante

```
if (PN.substr(0,1) == "A")          //if the pin is an analog input pin starting with A
{
PinVal = parseFloat(singleset[1]);      //the value of A0 received from Arduino will not be actual value of A0 but calculated value of temperature, so we need to extract it as floating point value using parseFloat
AnalogNum = "analog" + PN.substr(1,2);                  //analog0
document.getElementById(AnalogNum).value = PinVal;

Isso vai mudar o valor na caixa de texto para analog0 o valor da temperatura recebeu do Arduino
```

```
function sendbutton(Pin,action){
document.getElementById("description").innerHTML = "Processing Button Click";
server = "/arduino/digital/" + Pin + "/" + action;          //arduino/digital/12/1
request = new XMLHttpRequest();
request.onreadystatechange = updateasyncbutton;
request.open("GET", server, true);
request.send(null);

A página atualiza o status de pinos de forma assíncrona a cada 2 segundos. O script a seguir irá alterar o estado do pino quando o botão (12 ou 13) é pressionado e que também irá comandar o arduino para mudar a saída digital (pino 12 ou 13). Quando pressionado pino 12, por exemplo, o script enviado 2 valores com o script. Os 2 valores do número de pinos e ação será usada aqui. (Nota: A explicação a seguir vai usar o pino 12 como um exemplo com um estado de pino 1.)
```

```
function updateasyncbutton(){
if ((request.readyState == 4) && (request.status == 200))
{
result = request.responseText;          //result will hold response of Arduino i.e. digital,12,1
document.getElementById("description").innerHTML = result;
singleset = result.split(",");          //the result will be split into array as {digital,12,1}
PinType = singleset[0];                                 //digital
PinNum = singleset[1];                                  //12
Pinstatus = singleset[2];                               //1
ActNum = "action" + PinNum;                             //action12
ImgNum = "image" + PinNum;                              //image12
```

A maneira como se comporta o seguinte código já foi explicado na seção mudança de status.

```
if (Pinstatus == 0)
{
PinAct = "1";
image = "off.jpg";
}
else
{
PinAct = "0";
image = "on.jpg";
}
document.getElementById(ActNum).value = PinAct;
document.getElementById(ImgNum).src = image;
document.getElementById("description").innerHTML = result;
```

O pedido AJAX será criado e enviado como "http: // endereço IP do Yun / arduino / Digital / 12/1 para Arduino (e processados pelo mesmo) e o roteiro updateasyncbutton será executado. roteiro updateasyncbutton irá processar a resposta ao voltar do Arduino, na forma de arte digital, 12,1.

O script a seguir entra em ação quando o valor do controle deslizante 3 mudanças. O evento irá enviar 2 valores, Pin = value = 3 e posição do controle deslizante

```
function sendDac(Pin,value){
ValNum = "valueDac" + Pin;                                  //valueDac3
document.getElementById(ValNum).innerHTML=value;
document.getElementById("description").innerHTML = "Processing Slider";
server = "/arduino/dac/" + Pin + "/" + value;               //arduino/dac/3/190
request = new XMLHttpRequest();
request.onreadystatechange = updateasyncDac;
request.open("GET", server, true);
request.send(null);
```

The above code creates and sends a request to Arduino as "http://ip **address of Yun/arduino/dac/3/190 and runs script updateasyncDac. What this function does is change the value of slider and the value below it.**

```
function updateasyncDac(){
if ((request.readyState == 4) && (request.status == 200))
{
result = request.responseText;                              //dac,3,190
singleset = result.split(",");                              //array of {dac,3,190}
PinType = singleset[0];                                     //dac
PinNum = singleset[1];                                      //3
PinVal = parseInt(singleset[2]);                            //190
DacNum = "dac" + PinNum;                                    //dac3
ValNum = "valueDac" + PinNum;                               //valueDac3
```

Os códigos a seguir já foi explicado na seção mudança de status

```
document.getElementById(DacNum).value = PinVal;
document.getElementById(ValNum).innerHTML = PinVal;
document.getElementById("description").innerHTML = result;
```

11.12 Anexo XII – Plano de Ensino Introdução a Engenharia da Computação

	UNIVERSIDADE FEDERAL DE UBERLÂNDIA **FACULDADE DE ENGENHARIA ELÉTRICA** **CURSO DE GRADUAÇÃO EM** **ENGENHARIA ELÉTRICA**	FEELT

PLANO DE ENSINO

COMPONENTE CURRICULAR: INTRODUÇÃO À ENGENHARIA DE COMPUTAÇÃO

CÓDIGO:		**PERÍODO/SÉRIE: 1º**		**TURMA:**
CH TEÓRICA: **30**	11.12.1.1 *11.12.1.2 CH PRÁTICA:* **00**	**CH TOTAL:** **30**	**OBRIGATÓRIA: (X)**	**OPTATIVA: ( )**
PROFESSOR (A): LUCIANO VIEIRA LIMA				**ANO/SEMESTRE: 2016/1**

EMENTA

Visão geral da atuação técnica, social e ambiental dos estudantes, dos engenheiros e da engenharia.

JUSTIFICATIVA

Preparar os estudantes de engenharia de computação, em suas diversas ênfases, a conhecer as modernas tecnologias em uso, as atribuições de um engenheiro de computação, os conceitos básicos físicos e de componentes e paradigmas de hardware e software. Os conhecimentos, habilidades e competência nesta disciplina são fundamentais que o profissional nas ênfases da FEELT deve conhecer e saber realizar tarefas nos domínios que vão utilizar no curso e após formar.

OBJETIVOS

Ao final do curso o estudante deverá ser capaz de:
1. Compreender a importância dos modelos abstratos, das simulações, das pesquisas e dos projetos na área da engenharia de Eletrônica e de Telecomunicações;
2. Compreender a importância das visões sistêmica e estratégica, da criatividade e inovação, do trabalho em equipe e da comunicação interpessoal na atuação dos engenheiros;
3. Desenvolver, por conta própria, um pequeno projeto de engenharia, ampliando sua autonomia intelectual.

DESCRIÇÃO DO PROGRAMA

1. A graduação em Engenharia de computação da UFU

   1.1. Princípios e objetivos

   1.2. Perfil do egresso

   1.3. Estrutura curricular

   1.4. Estrutura física

   1.5. Regulamento

2. Métodos e estratégias de estudo e aprendizagem

   2.1. Conceitos e definições

2.2. Seminários

2.3. Resumo

2.4. Resenha

2.5. Esquema

2.6. Sinopse

2.7. Técnica de sublinhar

2.8. Pesquisa bibliográfica

3. Comunicação profissional

3.1. Comunicação oral

3.1.1. Elementos observados na comunicação oral

3.1.2. Inibição e ansiedade

3.1.3. Presença

3.1.4. Voz

3.1.5. Contato com os olhos

3.1.6. Linguagem do corpo

3.1.7. Aparência

3.1.8. Utilização de recursos áudio-visuais

3.2. Redação técnica

3.2.1. Linguagem técnica

3.2.2. Auxiliares lingüísticos

3.2.3. Trabalhos escolares

3.2.4. Provas

3.2.5. Relatórios técnicos

3.2.6. Artigos

3.2.7. Monografias

4. Criatividade e inovação

4.1. O processo criativo

4.2. Barreiras que afetam a criatividade

4.3. Técnicas de estimulação da criatividade

4.4. Inovação

5. Pesquisas tecnológicas

5.1. Caracterização

5.2. Ética

5.3. Tipos

5.4. Métodos

5.5. Organização

6. Projetos de engenharia

6.1. Seleção do tema e formulação do problema

6.2. Coleta de informações

6.3. Concepção da solução

6.4. Avaliação do projeto

6.5. Especificação da solução final

6.6. Relatório final

7. Modelos e simulação

7.1. A importância dos modelos

7.2. Modelo icônico

7.3. Modelo diagramático

7.4. Modelo matemático

7.5. Modelo físico

7.6. Simulação icônica

7.7. Simulação analógica

7.8. Simulação matemática

8. Otimização

8.1. Modelos de otimização

8.2. Métodos de otimização

9. Projeto orientado

METODOLOGIA

Horário de Atendimento aos estudantes:

Terça feira: 17:00 – 18:30

Quinta feira: 14:00hs – 18:00 hs

Cronograma previsto para desenvolvimento do conteúdo:

Aula	Conteúdo
01-02	recepção dos ingressantes e participação do trote social pelo PET
03-04	Apresentação do plano de curso e de avaliação, PET e mobilidade estudantil
05-06	sistema de biblioteca e guia acadêmico do estudante
07-08	engenheiro e sociedade - CREA – palestra
09-10	oratória, apresentação multimídia, criatividade – palestra
11-12	relações étnicos raciais – palestra
13-16	conceitos básicos e fundamentais sobre corrente CC e CA, Tensão, resistência, impedância, potência, resistores para fins eletrônicos, diodos, transistor como chave, relés, LEDs, chaves (polos, posições, tipo de acionamento, contatos NA e NF - Questionário sobre fontes DC
17-18	prática, mostra de conceitos dos itens das duas aulas anteriores
19-22	paradigmas de programação - conceitos de sistemas digitais e de numeração
23-24	projeto com microcontroladores e aplicativos mobile, usando blue tooth - fonte DC
25-28	projeto com microcontroladores e aplicativos mobile, usando ethernet com fio e PLC -

	fonte DC
29-30	projeto com microcontroladores e aplicativos mobile usando ethernet sem fio - fonte DC
31-36	mostra de trabalhos com uma tecnologia com ou sem fio, e seminário sobre o questionário de fontes DC - avaliação

AVALIAÇÃO

Sistema de Avaliação

Conforme metodologias aplicadas, o foco principal é PBL, e, desta forma, serão feitas avaliações em apresentações de projetos, produtos e parecer(trabalhos de pesquisa acadêmica mas com visão de mercado).

1- Seminário sobre fontes DC, conceitos e componentes utilizados.
Valor 40 Pontos
2- Projeto com uma tecnologia de transmissão de dados com ou sem fio (blue tooth, ethernet com ou sem PLC)
Valor 60 pontos

BIBLIOGRAFIA

BIBLIOGRAFIA BÁSICA:

1. BAZZO, W. A.; PEREIRA, L.T.V. Introdução a Engenharia. Florianópolis: UFSC, 2000. ISBN: 85-328-0356-3

2. BARROS, A. P.; LEHFELD, N. A. S. Fundamentos de Metodologia: Um Guia para a Iniciação Científica. São Paulo: Makron Books, 1986. ISBN: 9780074500217

3. BASTOS, L. R. et al. Manual para a Elaboração de Projetos e Relatórios de Pesquisa, Teses, Dissertações. Rio de Janeiro: LTC, 2000. ISBN: 8521613563

BIBLIOGRAFIA COMPLEMENTAR:

1. SEVERINO, A. J. Metodologia do Trabalho Científico. São Paulo: Cortez, 2000. ASIN: B06VY69C7N

2. SILVA, J.C. Metodologia do Trabalho Escolar: Recomendações ao Aluno. COBENGE, 1983.

3. SILVA, E. L. e MENEZES, E. M. Metodologia da Pesquisa e Elaboração de Dissertação. 3a Edição. Laboratório de Ensino a Distância da UFSC. Florianópolis-SC, 2001.

4. BARRAS, R. Os Cientistas Precisam Escrever. São Paulo: T.A. Queiroz Editor, 3ª ed., 1991, 218 p.

5. FOOT, F., VICTOR, L. História da Indústria e do Trabalho no Brasil: das Origens aos anos Vinte. São Paulo, Global, 1982.

6. LINSINGEN, I.V., PEREIRA, L.T.V., CABRAL, C.G., BAZZO, W.A. Formação do Engenheiro. Ed. UFSC, Florianópolis, 1999.

7. Artigos Técnicos e Científicos na área de engenharia de computação.

1.

APROVAÇÃO

Aprovado em reunião do Colegiado do Curso de Graduação em Engenharia Biomédica

Em ___/____/_____

_____

Coordenador do Curso