

Implementando

Internet das Coisas (IoT)

Com PHP, MySQL, Android e Arduino



Vitor Amadeu Souza

Cerne
Conhecimento para o Desenvolvimento

www.cerne-tec.com.br

Vitor Amadeu Souza Implementando

Internet das Coisas (IoT)

Com PHP, MySQL, Android e Arduino

© 2015 by Cerne Tecnologia e Treinamento Ltda. © 2015 by Vitor Amadeu Souza

Nenhuma parte desta publicação poderá ser reproduzida sem autorização prévia e escrita de **Cerne Tecnologia e Treinamento Ltda.** Este livro publica nomes comerciais e marcas registradas de produtos pertencentes a diversas companhias. O editor utiliza as marcas somente para fins editoriais e em benefício dos proprietários das marcas, sem nenhuma intenção de atingir seus direitos.

Maio de 2015

Direitos reservados por: Cerne Tecnologia e Treinamento Ltda

Produção: Cerne Tecnologia e Treinamento **E-mail da Empresa:** cerne@cerne-tec.com.br

Home Page: www.cerne-tec.com.br **com.br Atendimento ao Consumidor:** sac@cerne-tec.com.br **Contato com o Autor:** vitor@cerne-tec.com.br

Dedicatória

Como nos meus outros livros, dedico este livro a minha querida esposa Renata Leal.

“Toda a dignidade do homem está no pensamento.”

Blaise Pascal Kits Didáticos e Gravadores da Cerne Tecnologia

A Cerne tecnologia têm uma linha completa de aprendizado para os microcontroladores da família PIC, 8051, Holtek, dsPIC, ARM, Arduino, etc. Veja os detalhes de um de nossos kits.



Kit Cerne Arduino

- Microcontrolador ATMEGA8;
- Comunicação serial RS232;
- Alimentação de 12V;
- Pinos de I/O;
- Gravação ICSP.

Uma linha completa de componentes para o desenvolvimento de seus projetos eletrônicos como displays, PICs, botões, leds, cristais e etc. Visite a nossa página na Internet, no

endereço www.cerne-tec.com.br e conheça melhor nossos serviços e produtos.



www.cerne-tec.com.br

Sumário

[**I. Metodologia de desenvolvimento 10**](#) 1. Introdução 10

[**II. HTML 17**](#)

- [1. Introdução 17](#)
- [2. Alterando o título 20](#)
- [3. Apresentando Textos 21](#)
- [4. Textos pré-formatados 25](#)
- [5. Alterando a cor e tamanho do texto 26](#)
- [6. Mostrando imagens ao fundo 30](#)
- [7. Criando Links 32](#)
- [8. Enviando e-mails 33](#)
- [9. Tabelas 35](#)
- [10. Caixas de Texto 38](#)
- [11. Caixas de Texto de Múltiplas Linhas 39](#)
- [12. ComboBox 40](#)
- [13. CheckBox 41](#)
- [14. Radio Button 42](#)
- [15. Listas Ordenadas 44](#)
- [16. Listas Não Ordenadas 45](#)

[**III. CSS 47**](#)

- [1. Introdução 47](#)
- [2. Alterando a cor de fundo 47](#)
- [3. Alterando a cor de texto 50](#)
- [4. Alterando o alinhamento do texto 51](#)
- [5. Alterando o posicionamento absoluto 52](#)
- [6. Alterando o fonte usado 54](#)
- [7. Modificando o tamanho de imagens 56](#)
- [8. Alterando o tamanho das letras 57](#)
- [9. Deixando o texto em negrito 58](#)
- [10. Deixando o texto em itálico 59](#)

[**IV. PHP 60**](#)

- [1. Introdução 60](#)
- [2. Apresentando Informações 60](#)
- [3. Comentários 62](#)
- [4. Variáveis 63](#)
- [5. Arrays 66](#)

- [6. Operadores Aritméticos](#) 67
- [7. Operadores Lógicos](#) 69
- [8. Operadores Relacionais](#) 70
- [9. Concatenação de Strings](#) 73
- [10. Controle de Repetição While](#) 75
- [11. Controle de Repetição For](#) 77
- [12. Tratamento Switch](#) 79
- [13. Criando Funções](#) 81
- [14. Funções do PHP](#) 84

V. MySQL 99

- [1. Introdução](#) 99
- [2. Iniciando o MySQL](#) 99

VI. Integrando PHP e MySQL 108

- [1. Introdução](#) 108
- [2. Função MySQL_CONNECT\(\)](#) 108
- [3. Função MySQL_SELECT_DB\(\)](#) 109
- [4. Função MySQL_QUERY\(\)](#) 110
- [5. Função MySQL_NUM_ROWS\(\)](#) 110
- [6. Função MySQL_FETCH_ARRAY\(\)](#) 111
- [7. Método GET](#) 114
- [8. Adicionando o banco de dados com o método GET](#) 116
- [9. Método GET](#) 119

VII. Conhecendo o MIT App Inventor 128

- 1. Criando um projeto** 128

VIII. Exemplos no App Inventor 134

- [1. Introdução](#) 134
- [2. Label](#) 134
- [3. Button](#) 137
- [4. CheckBox](#) 145
- [5. Image](#) 151
- [6. Slider I](#) 153
- [7. Notifier](#) 155
- [8. TextBox](#) 157

IX. Projetos no App Inventor 160

- [1. Introdução](#) 160
- [2. Calculadora](#) 161
- [3. Dado eletrônico](#) 163
- [4. Contador](#) 165
- [5. Conversor Pa <-> PSI](#) 168
- [6. Raízes de uma equação do 2º](#) 171
- [7. Enviando dados para WebServer](#) 173

X. Hardware e Software Arduino 176

- [1. Introdução](#) 176
- [2. Conhecendo o Software](#) 176
- [3. Conhecendo o Hardware](#) 178

XI. Ligando um Led no Arduino 180

- [1. Pinagem do Arduino 180](#)
2. Montando o Hardware 181
3. Programando o Arduino 182

XII. Transmissão Serial 189

1. Introdução 189
2. Montando o Hardware 189
3. Programando o Arduino 189
- [4. Programa para comunicação com Android 192](#)

XIII. Recepção Serial 194

- [1. Introdução 194](#)
- [2. Montando o Hardware 195](#)
- [3. Programando o Arduino 195](#)

XIV. Aplicativo com interface Bluetooth 198

- [1. Controlando o estado do LED 198](#)
- [2. Enviando o evento do botão pela Web 201](#)

Referências 205

Capítulo I Metodologia de desenvolvimento

1. Introdução

A proposta deste livro é explorar a Internet das Coisas ou *Internet of Things (IoT)* através de um exemplo prático que permita conectar um smartphone/tablet Android a um servidor PHP que possibilite a visualização de um evento através de um browser. Após esta experiência, um Arduino comunica com este mesmo smartphone/tablet através da rede Bluetooth para informar um evento a mesma página na Internet. Desta forma, será possível conectar o celular e Arduino a rede para monitoramento remoto, implantando assim o conceito IoT.

Os smartphone/tablets Android ganham a cada dia mais e mais aplicações e desta forma, faz-se necessário conectá-lo a um servidor Web de modo a transmitir alguma informação que possa ser observada e tratada em qualquer browser disponível no globo terreste.

O intuito deste livro é desenvolver uma aplicação Web baseada no PHP com banco de dados MySQL de modo a receber dados de um smartphone/tablet

Inventor, no qual Android programado através do MIT App Inventor, o gerador do evento é uma placa Arduino programada para enviar através da comunicação Bluetooth com o smartphone/tablet um comando, para que em seguida tais dados sejam enviados para o servidor e apresentados em formato de tabela

e salvos em um banco de dados no servidor Web.

Para se chegar a este fim diversos exemplos são construídos ao longo da obra usando as ferramentas comentadas, de modo a embasar o leitor com os tópicos principais para este fim. O exemplo a ser desenvolvido no smartphone/tablet foi feito no MIT App Inventor e consiste de enviar um evento de botão

pressionado gerado pela placa Arduino para um servidor Web programado em PHP, no qual recebe tal pacote e mostra em seguida na tela além de salvá-lo em banco de dados MySQL juntamente com a data e hora da ocorrência.

2. Conhecendo o WampServer

Para otimizar o tempo de desenvolvimento, será utilizada uma ferramenta chamada *WampServer*, que permitirá utilizar o mesmo computador, onde será feita a escrita do código para simulá-lo com o PHP e banco de dados MySQL. Este software pode ser baixado gratuitamente no seguinte endereço:

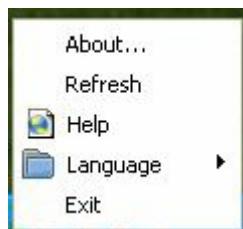
www.wampserver.com

A versão utilizada nesta obra foi a 2.1, no entanto, versões mais recentes podem ser utilizadas. Ao instalar o *WampServer*, este disponibilizará na sua máquina um servidor com os sistemas Apache, MySQL e PHP, evitando assim que após a escrita do código, seja necessário atualizar um servidor externo e utilizar um dispositivo para acessá-lo e verificar a aplicação funcionando, permitindo assim testar no próprio computador a aplicação através de um browser ou permitir ao acessá-lo como servidor via WiFi.

Após o download e instalação do *WampServer*, inicialize-o. Irá aparecer próximo ao relógio do Windows um ícone indicando que o *WampServer* está sendo executado, observe:



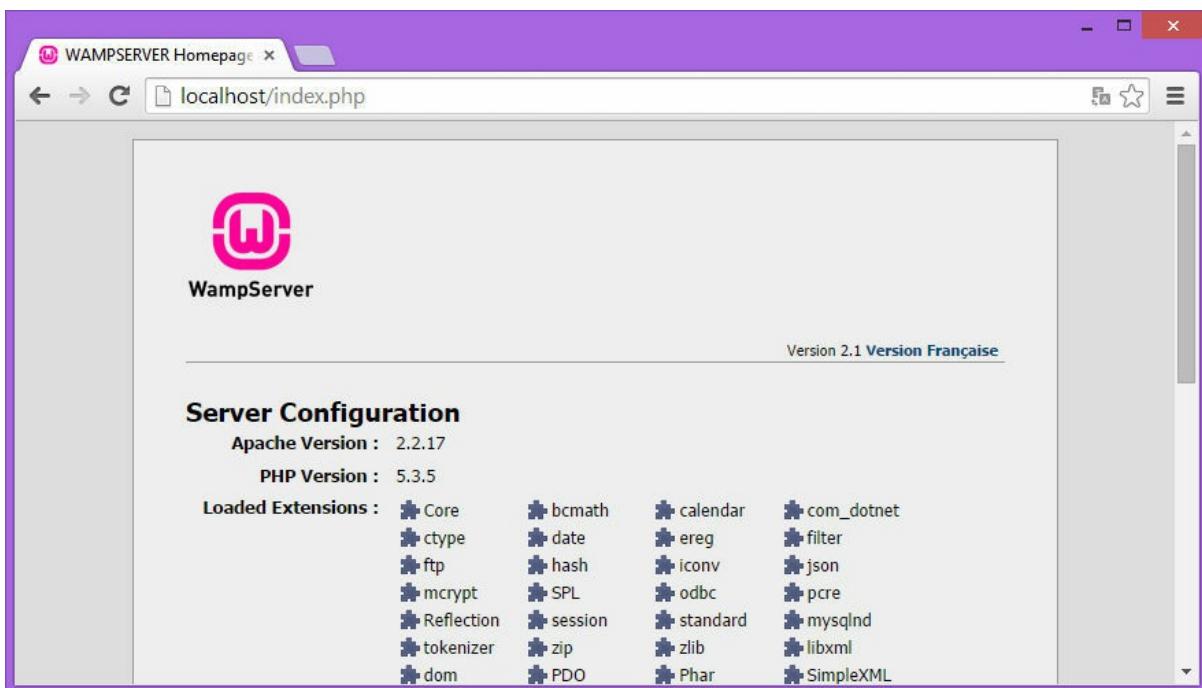
O primeiro passo para começar a usar o *WampServer* é clicar com o botão direito do mouse sobre seu ícone. Neste momento, irá aparecer uma janela com as seguintes opções.



Escolha a opção *Language* e em seguida a opção *Portuguese*. Para deixar o *WampServer* habilitado, dê um clique com o botão esquerdo, onde uma janela com as seguintes opções será apresentado.

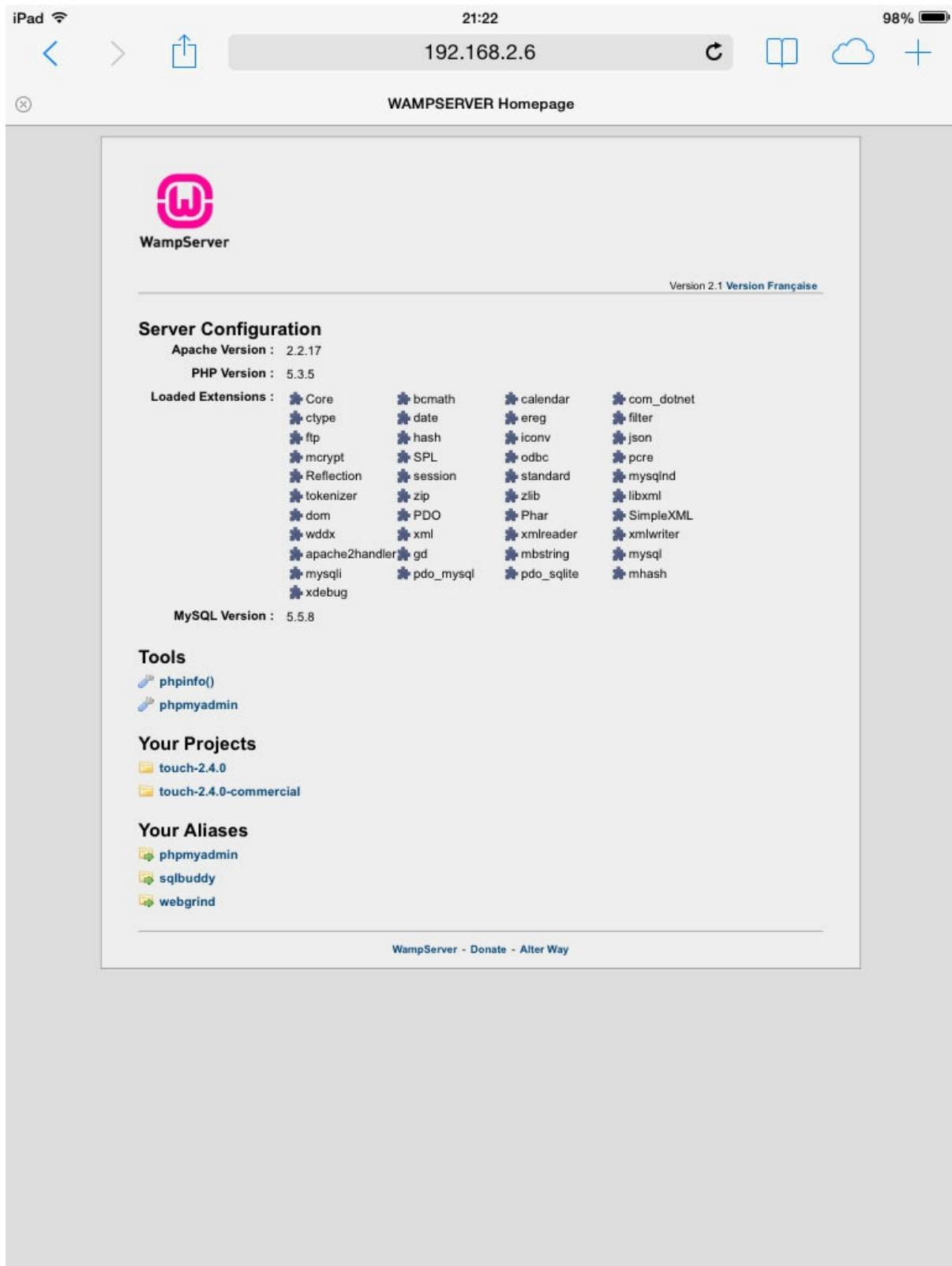


Escolha a opção *Iniciar todos os Serviços* para que o *Wamp Server* fique assim ativo. Para comprovar, inicialize o seu navegador de internet (Internet Explorer, Firefox, Google Chrome, Netscape, etc). Em seguida, digite <http://localhost/index.php>, se tudo estiver funcionando corretamente será observado o seguinte resultado:



Para que o dispositivo móvel ou outra máquina possa acessar o computador com o *WampServer* instalado, basta colocar o IP da sua máquina seguido da página a ser aberta. Por exemplo, supondo que o IP da máquina seja 192.168.2.6, deverá ser digitado no campo de endereço do browser <http://192.168.2.6/index.php>. A figura abaixo ilustra a tela acima aberta através do smartphone/tablet utilizando a rede local de acordo com o método recomendado.

Dica: Para saber o IP da sua máquina utilize o ipconfig do Windows.



Quando o *WampServer* é instalado no computador, o mesmo fica alogado tipicamente na pasta C:\wamp. Observe que dentro desta pasta, há outra chamada www. É nesta pasta que deverá ser colocado os projetos desenvolvidos ao longo desta literatura. Ou seja, se for criado o arquivo *exemplo1.php*, digite no campo de endereço do browser o valor <http://localhost/exemplo1.php>. Se o mesmo teste for feito através do smartphone/tablet e o IP do servidor for 192.168.2.6, digite <http://192.168.2.6/exemplo1.php>.

Obs: Caso o arquivo seja HTML puro, digite com a extensão .htm.

Capítulo II HTML

1. Introdução

Nesta literatura o enfoque será na linguagem PHP, porém é fundamental entender o funcionamento do HTML, já que o PHP funciona normalmente em conjunto com esta linguagem. Para escrever os exemplos propostos, pode-se usar o bloco de notas (notepad) que já vem instalado por default no Windows ou usar o Notepad ++, que é um editor de texto melhorado se comparado ao bloco de notas. Este software é gratuito e pode ser baixado pelo link abaixo:

<http://notepad-plus-plus.org/>

Todos os exemplos deverão ser salvos na pasta c:\wamp\www, pois é esta a pasta onde o WampServer irá acessar para executar os exemplos. Por exemplo, se for criado um exemplo chamado teste1.php e salvá-lo em seguida na pasta indicada, o acesso ao arquivo será feito da seguinte forma através do browser:

<http://localhost/teste1.php>

Obs: Se for feito através do smartphone/tablet e o IP da máquina for, por exemplo, 192.169.2.6, digite<http://192.168.2.6/teste1.php>

Sendo assim, não esqueça de salvar todos os exemplos na pasta c:\wamp\www. Quando um exemplo for feito apenas em HTML, pode-se salvar e executá-lo usando a extensão .htm, que é o padrão para HTML. Quando houver códigos que fazem uso de php e HTML, deve-se salvá-lo com extensão .php. Pode-se também desenvolver arquivos puramente HTML e salvá-lo com a extensão .php sem nenhum problema.

Neste tópico da literatura será verificado os pontos mais importantes do HTML, já que é um fundamento importante para estudar o PHP. A estrutura básica de um programa HTML é a seguinte.

```
<html>
<head> <title> </title>
</head>

<body> </body>
</html>
```

Observe que no HTML, tudo possui um início e fim, sendo o fim representado pela presença da /. Note como exemplo a declaração <html> que dá início ao código e o </html> ao fim. No identificador <head> encontra-se o cabeçalho do arquivo. Entre ele, há a opção <title>, onde pode-se definir o título que terá a página a ser apresentada no browser. Em seguida, encontra-se o corpo <body> da página, onde através dela pode-se colocar imagens, tabelas, textos, botões, etc. Nos próximos tópicos, será apresentado os conceitos fundamentais para manipulação do HTML.

Dica: O HTML não é case sensitive, ou seja, letras maiúsculas e minúsculas possuem o mesmo significado.

2. Alterando o título

Neste exemplo será alterado o título de uma página web. Para isso, crie um novo arquivo em um dos editores citados e digite o seguinte código.

```
<html>
<head> <title> Exemplo no HTML </title>
</head>
<body>
</body>
</html>
```

Salve o arquivo com a extensão .htm e chame-o de exemplo1.htm, na pasta informada anteriormente (c:\wamp\www). Através do smartphone/tablet, digite o IP do servidor seguido do nome do arquivo. O resultado será como verificado a seguir.



3. Apresentando Textos

Pode-se apresentar textos no HTML, observe o exemplo a seguir.

```
<html>
```

```
 <head>
```

```
   <title> Exemplo no HTML </title>
```

```
 </head>
```

```
<body>  
Apresentando Textos no HTML </body>  
</html>
```

Na figura abaixo o resultado obtido no smartphone/tablet.



Apresentando Textos no HTML

Obs: Esta imagem está com zoom.

Pode-se montar um texto, colocando-o em múltiplas linhas. Para isso usa-se o comando de quebra de linha, que é o
. No exemplo abaixo, pode-se visualizar melhor este conceito.

```
<title> Exemplo no HTML </title> </head>
<body>
```

Apresentando Textos no HTML
 Em diversas linhas
 E testando no browser


```
</body>
</html>
```



Apresentando Textos no HTML

Em diversas linhas

E testando no browser

Também há o comando `<p>` que é o parágrafo, colocando um espaço entre duas linhas. Abaixo o código modificado para usar o `<p>`.

```
<title> Exemplo no HTML </title> </head>  
<body> Apresentando Textos no HTML <p> Em diversas linhas <p> E testando no browser <p>
```

```
</body>
</html>
```



Apresentando Textos no HTML

Em diversas linhas

E testando no browser

4. Textos pré-formatados

Os textos pré-formatados têm a vantagem de não ser necessário usar o comando de quebra de linha para sair de uma linha para outra, ou seja, o texto é impresso da forma que o mesmo foi colocado na página HTML. Observe o exemplo a seguir.

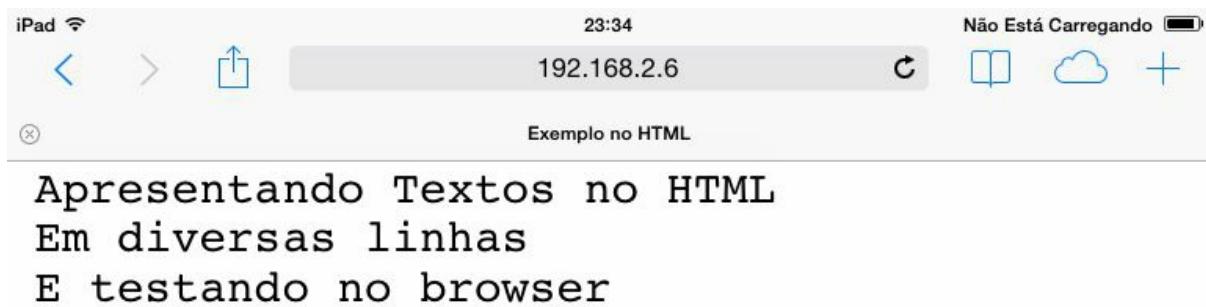
```
<html>
```

```
<head>
<title> Exemplo no HTML </title>
</head>
<body>

<pre> Apresentando Textos no HTML Em diversas linhas
E testando no browser

</pre>
</body>
</html>
```

Observe que neste exemplo, não foi necessário colocar o comando de quebra de linha entre o texto impresso. O resultado no browser será o que está apresentado abaixo.



5. Alterando a cor e tamanho do texto

Para alterar o tamanho da letra no HTML há as *tags* `<h1>` a `<h6>`, onde quanto maior o índice menor será a letra e vice-versa. Observe o exemplo abaixo que mostra um texto em diversos tamanhos.

```
<html>
```

```
<head>
<title> Exemplo no HTML </title>
</head>
<body>
<h4> Apresentando Textos no HTML </h4> <h5> Apresentando Textos no HTML </h5> <h6> Apresentando Textos no
HTML </h6>
<h1> Apresentando Textos no HTML </h1> <h2> Apresentando Textos no HTML </h2> <h3> Apresentando Textos no
HTML </h3>
</body>
</html>
```

A próxima figura mostra o resultado obtido no browser do smartphone/tablet.



Apresentando Textos no HTML

Apresentando Textos no HTML

Apresentando Textos no HTML

Apresentando Textos no HTML

Apresentando Textos no HTML

Apresentando Textos no HTML

Para alterar a cor do texto, na declaração <body> pode-se informar a opção *text*, como apresentado abaixo em que a cor da letra texto fica vermelha.

<html>

```
<head> <title> Exemplo no HTML </title>
</head>
```

```
<body text="red">
<h1> Apresentando Textos no HTML </h1> <h2> Apresentando Textos no HTML </h2> <h3> Apresentando Textos no
HTML </h3> <h4> Apresentando Textos no HTML </h4> <h5> Apresentando Textos no HTML </h5> <h6> Apresentando
Textos no HTML </h6>

</body>
</html>
```



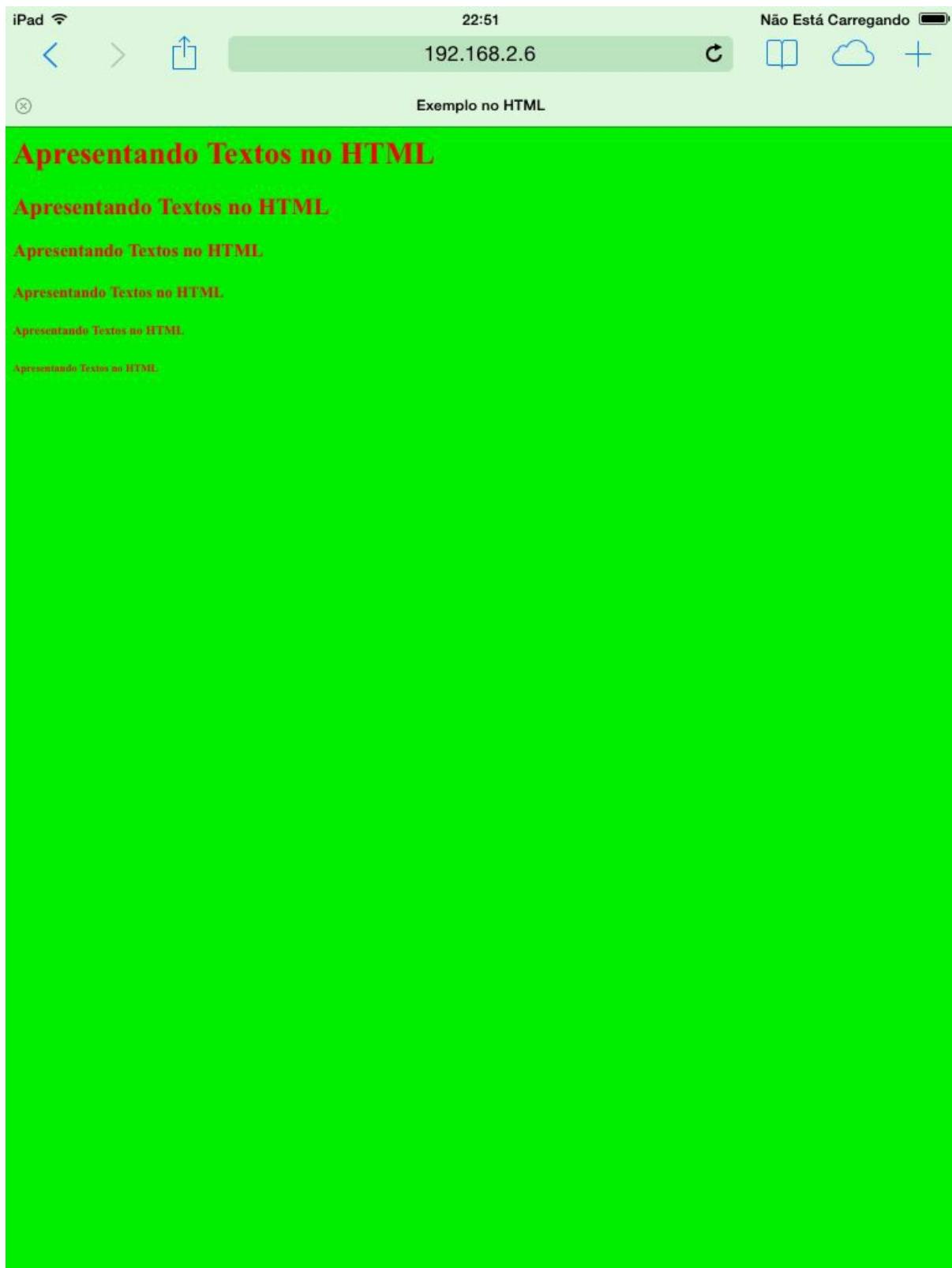
Note que todas as cores podem ser implementadas, como blue (azul), black(preto), white (branco) dentre outros. Pode-se também alterar a cor de fundo, fazendo uso neste caso do parâmetro bgcolor, conforme mostra o exemplo abaixo.

```
<html>
<head>
```

```
<title> Exemplo no HTML </title>
</head>
<body bgcolor="green" text="red">

<h1> Apresentando Textos no HTML </h1>
<h2> Apresentando Textos no HTML </h2>
<h3> Apresentando Textos no HTML </h3>
<h4> Apresentando Textos no HTML </h4>
<h5> Apresentando Textos no HTML </h5>
<h6> Apresentando Textos no HTML </h6>

</body>
</html>
```



6. Mostrando imagens ao fundo

Para mostrar imagens ao fundo da página HTML, utiliza-se o comando *background*. Observe no exemplo abaixo, que carrega o arquivo *imagem.jpg* (este arquivo deve estar na mesma pasta do arquivo exemplo1.htm).

```
<html>
```

```
<head>
<title> Exemplo no HTML </title>
</head>
<body background="imagem.jpg" text="green">

<h1> Apresentando Textos no HTML </h1> <h2> Apresentando Textos no HTML </h2> <h3> Apresentando Textos no
HTML </h3> <h4> Apresentando Textos no HTML </h4> <h5> Apresentando Textos no HTML </h5> <h6> Apresentando
Textos no HTML </h6>

</body>
</html>
```

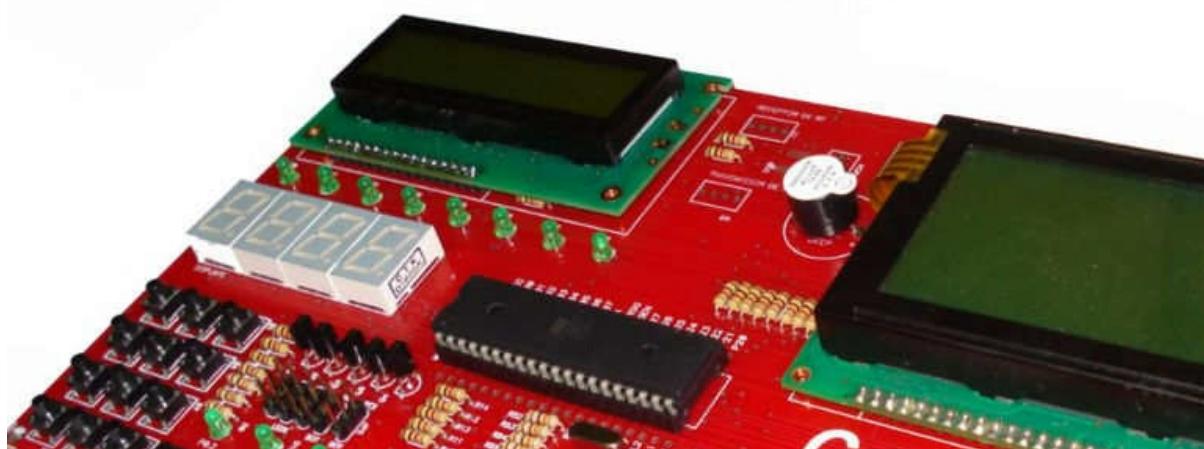
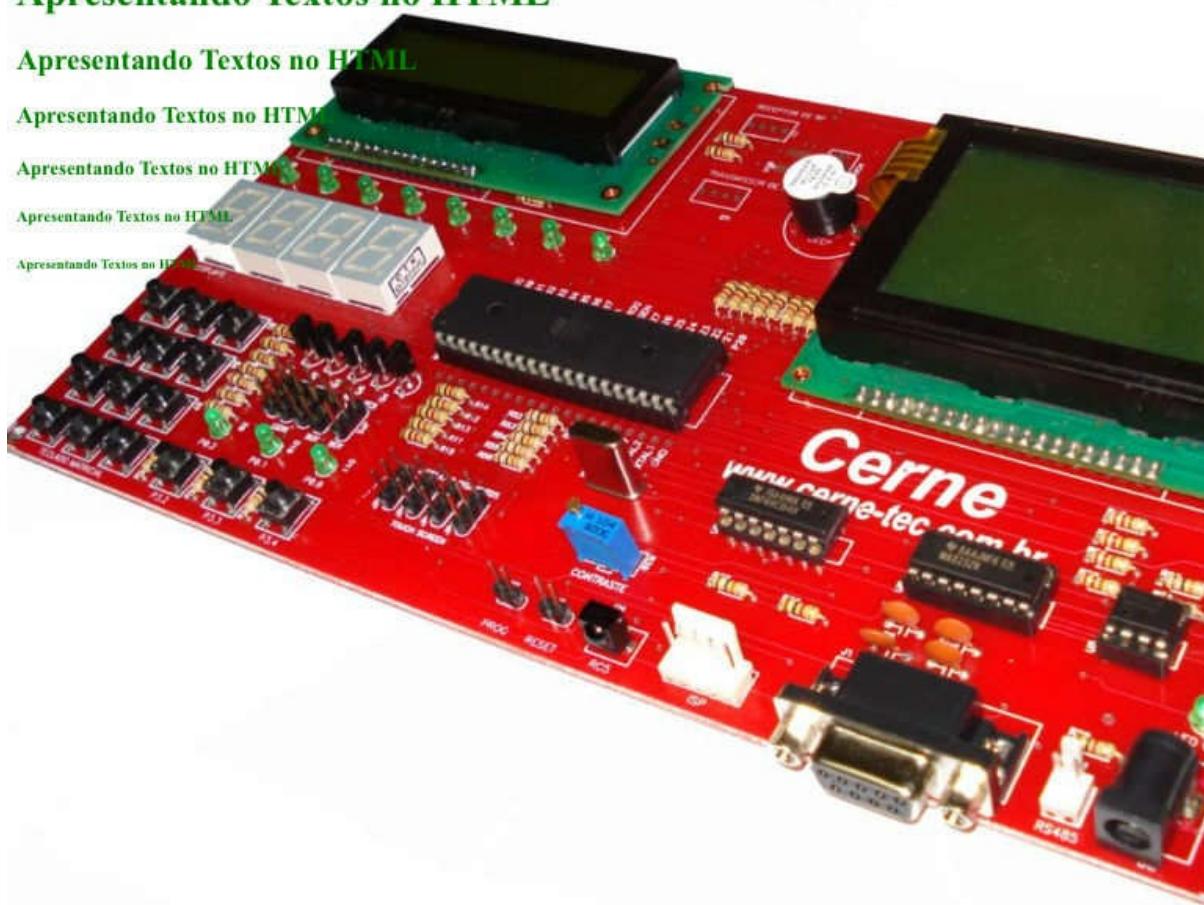
É importante frisar que o arquivo imagem.jpg deve estar presente na pasta c:\wamp\www. Na figura abaixo o resultado verificado no smartphone/tablet.



Apresentando Textos no HTML

Apresentando Textos no HTML

Apresentando Textos no HTML



7. Criando Links

É muito comum encontrar links nas páginas de internet, fazendo que uma página possa visitar outras com apenas um clique. Para habilitar os links, usa-se o comando `<a href>`. No exemplo abaixo, pode-se visualizar a chamada de uma página usando este conceito.

<html>

```
<head>
<title> Exemplo no HTML </title>
</head>
<body>

<a href="http://www.cerne-tec.com.br">Cerne Tec</a> </body>
</html>
```



Cerne Tec

8. Enviando e-mails

Há como chamar o programa de e-mail padrão do smartphone/tablet para enviar e-mails através de uma página HTML usando o mesmo recurso do exemplo anterior. Observe o exemplo a seguir.

```
<head>
```

```
<title>Exemplo no HTML</title>
</head>
<body>
<h1>
<a href="mailto:vitor@cerne-tec.com.br"> Clique aqui para nos enviar um e-mail.</a>
</h1> </body> </html>
```

A seguir o resultado no browser.



Ao clicar no link, será aberto o programa de e-mail padrão com o e-mail configurado a ser enviado já preenchido.



Enviada do meu iPad



9. Tabelas

As tabelas são um importante meio para apresentar informações de forma ordenada. Para isso, utiliza-se o comando `<table>` para criar a tabela, o comando `<tr>` para criar a linha e o comando `<td>` para a criação da coluna. Observe o exemplo abaixo seguido do resultado observado no tablet.

```
<head>
<title> Exemplo no HTML </title> </head> <body> <table>
<tr><td>Item <tr><td>Feijão <tr><td>Arroz </td><td>Preço </td><tr> </td><td>R$ 2,00</td><tr> </td><td>R$ 7,00</td>
<tr>
<tr><td>Manteiga </td><td>R$ 3,00</td><tr> <tr><td>Queijo </td><td>R$ 9,00</td><tr> </table>
</body>
</html>
```



Item	Preço
Feijão	R\$ 2,00
Arroz	R\$ 7,00
Manteiga	R\$ 3,00
Queijo	R\$ 9,00

Neste exemplo seria interessante contar com uma borda, de forma que a mesma ficasse mais organizada. Para isso, há o comando *border*, onde pode-se atribuir a largura da borda. Observe o exemplo a seguir com a borda habilitada.

```
<html>
<head>
<title> Exemplo no HTML </title>
```

```
</head>
<body>
<table border=3>

<tr><td>Item <tr><td>Feijão <tr><td>Arroz <tr><td>Manteiga </td><td>R$ 3,00</td><tr> <tr><td>Queijo </td><td>R$ 9,00</td><tr>

</table>
</body>
</html>
</td><td>Preço </td><tr> </td><td>R$ 2,00</td><tr> </td><td>R$ 7,00</td><tr>
```

The screenshot shows an iPad screen with a table of grocery items and their prices. The table has two columns: 'Item' and 'Preço'. The items listed are Feijão, Arroz, Manteiga, and Queijo, with their respective prices of R\$ 2,00, R\$ 7,00, R\$ 3,00, and R\$ 9,00. The table is styled with a dark border and light-colored cells.

Item	Preço
Feijão	R\$ 2,00
Arroz	R\$ 7,00
Manteiga	R\$ 3,00
Queijo	R\$ 9,00

10. Caixas de Texto

As caixas de texto são uma importante fonte de entrada para um *WebApp*. Para habilitar este recurso utiliza-se o comando *input*, onde através deste pode-se fornecer uma informação a uma página. Neste comando, define-se um nome a variável usada e também o tamanho máximo que esta poderá suportar através do comando *maxlength*. A seguir um exemplo com este recurso.

```
<html>
<head>

<title> Exemplo no HTML </title>
</head>
<body>

Digite seu nome:
<input type="text" name="nome" maxlenht=20> </body>
</html>
```

11. Caixas de Texto de Múltiplas Linhas

As caixas de texto de múltiplas linhas são usadas quando deseja-se que o usuário entre com mais informações em uma página do que suportaria em apenas uma caixa de texto. Para isso, usa-se o comando *textarea*, em que define-se o número de linhas (*rows*) e o número de colunas (*cols*). Observe o exemplo abaixo.

```
<html>
<head>
<title> Exemplo no HTML </title>
</head>
<body>
<textarea name="sugestao" rows="5" cols="40">
Digite sua sugestão aqui...
</textarea>
</body>
</html>
```

12. ComboBox

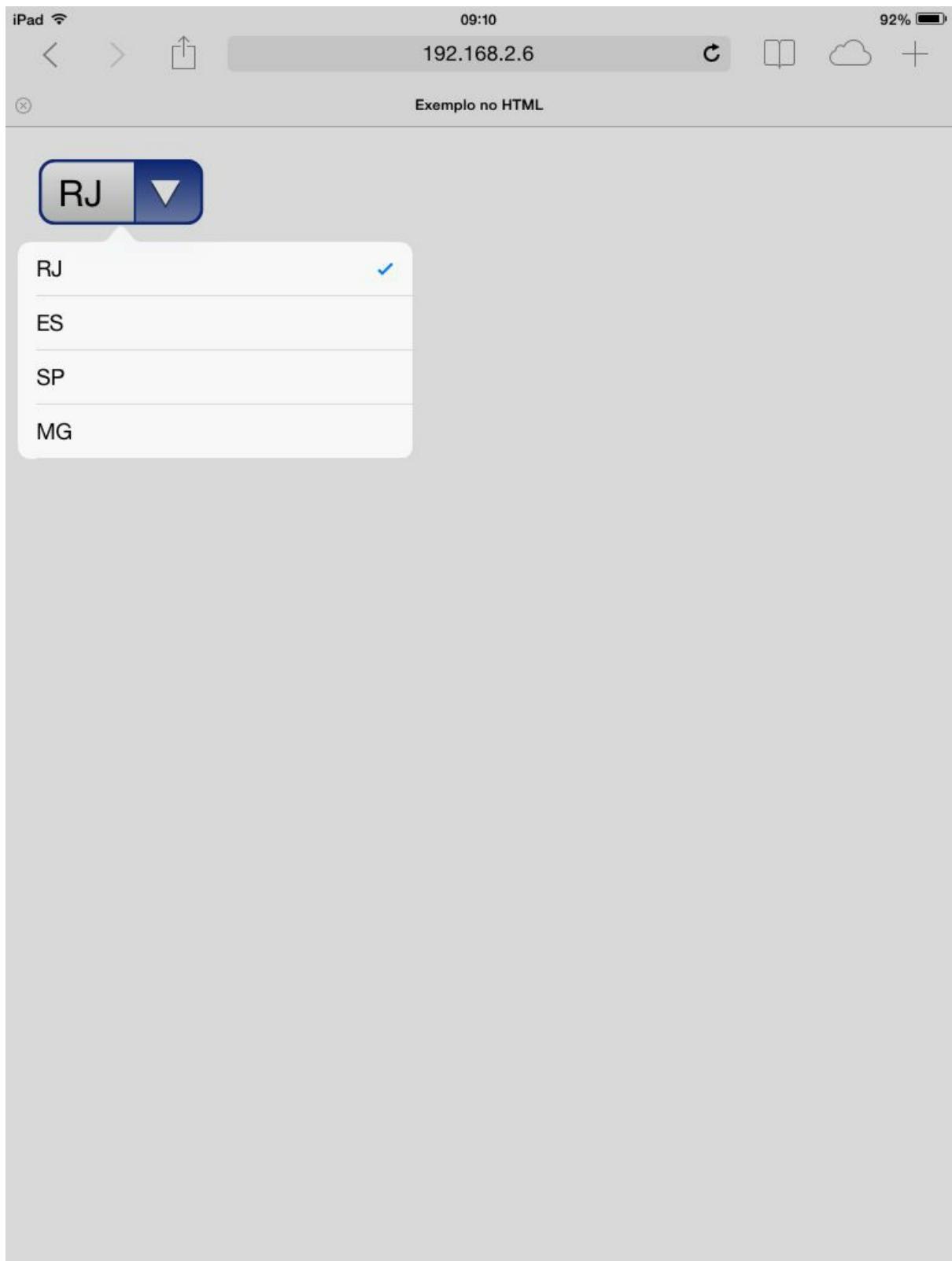
A ideia do *combobox* é oferecer uma lista de opções ao usuário, onde este possa escolher uma das opções listadas. Por exemplo, os estados da federação poderiam estar apresentados em um *combobox* e o usuário escolher o mesmo nesta lista. Para criar um *combobox*, usa-se o comando *<select>* e *<option>*. Acompanhe no exemplo abaixo.

```
<html>
<head> <title> Exemplo no HTML </title>
</head>

<body>

<select name="opcao">
<option value="RJ">RJ</option> <option value="ES">ES</option> <option value="SP">SP</option> <option
value="MG">MG</option>

</select>
</html>
```



13. CheckBox

Os *checkboxs* são muito usados para fornecer diversas opções em um *WebApp*, onde o usuário possa escolher uma ou todas que estão sendo listadas. Usa-se o comando *input* apresentado anteriormente, porém com a opção *checkbox* habilitada ao invés da opção *text*. Observe o exemplo a seguir.

```
<html>
<head>

<title> Exemplo no HTML </title>
</head>
<body>
<input type="checkbox" name="re[]" value="EF">E.Fundame. <input type="checkbox" name="re[]" value="EM">E.Médio
<input type="checkbox" name="re[]" value="ES">E.Superior </body>

</html>
```



E.Fundame. E.Médio E.Superior

14. Radio Button

O *radiobutton* diferente do *checkbox* permite escolher apenas uma opção da lista apresentada. Enquanto no *checkbox* o usuário pode escolher todas as opções listadas no *radiobutton* é possível escolher apenas uma. Observe o exemplo abaixo.

```
<html>
```

```
<head>
<title> Exemplo no HTML </title>
</head>

<body>
<input type="radio" name="re[]" value="EF">E.Fundame. <input type="radio" name="re[]" value="EM">E.Médio
<input type="radio" name="re[]" value="ES">E.Superior </body>

</html>
```



E.Fundame. E.Médio E.Superior

15. Listas Ordenadas

As listas são muito importantes para apresentação de dados ao

usuário, podendo estas serem ordenadas ou não. Inicialmente será visto as listas ordenadas, porém no próximo tópico as não ordenadas serão apresentadas. Observe a seguir um exemplo de lista ordenada.

1. Brasil
2. Argentina
3. Bolívia
4. Peru
5. Colômbia

Uma lista ordenada é criada através da tag `` e encerra-se com a tag ``. Cada elemento da lista é colocado através da tag ``. Observe a seguir o exemplo apresentado anteriormente no HTML.

```
<html>
<head>
<title>Exemplos no HTML</title> </head>
<body>
<ol>
<li>Brasil</li> <li>Argentina</li> <li>Bolívia</li> <li>Peru</li> <li>Colômbia</li>
</ol>
</body>
</html>
```



1. Brasil
2. Argentina
3. Bolívia
4. Peru
5. Colômbia

16. Listas Não Ordenadas

Um exemplo de listas não ordenadas são as apresentadas a seguir.

São Paulo
Rio de Janeiro Minas Gerais Espírito Santo Bahia

O procedimento para criar esta lista segue praticamente o conceito apresentado anteriormente, tendo como diferença que ao invés de usar a tag usa-se a tag. Observe o código HTML a seguir.

```
<html>
<head>
<title>Exemplos no HTML</title>
</head>
<body>

<ul> <li>São Paulo</li> <li>Rio de Janeiro</li> <li>Minas Gerais</li> <li>Espírito Santo</li> <li>Bahia</li>
</ul>
</body> </html>
```



- São Paulo
- Rio de Janeiro
- Minas Gerais
- Espírito Santo
- Bahia

Capítulo III CSS

1. Introdução

O CSS ou *Cascading Style Sheets* é uma linguagem de estilo utilizada para definir a

apresentação de documentos escritos em uma linguagem de marcação, como HTML ou XML. Seu principal benefício é prover a separação entre o formato e o conteúdo de um documento. Através dele, pode-se mudar características na página como cor, tipo de letra, tamanho, etc. O CSS *InLine* é chamado assim, pois no próprio código HTML coloca-se as modificações que deseja-se fazer usando este recurso. Além deste modo há os tipos incorporado e externo.

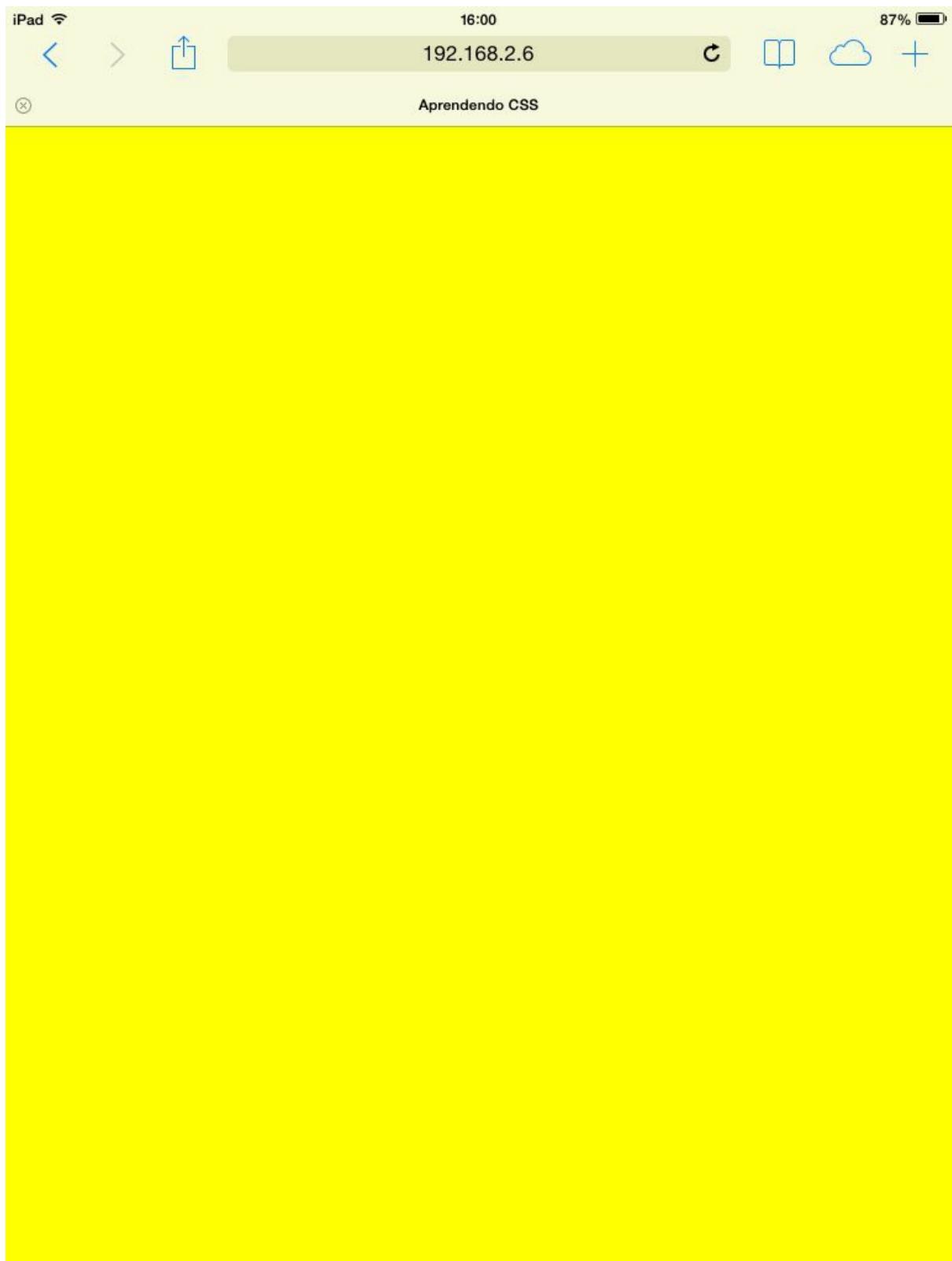
2. Alterando a cor de fundo

Sempre que utilizar o CSS, deve-se iniciar com o código com a tag *style* seguido da alteração que deseja-se fazer entre aspas. Cada tag do CSS finaliza com ponto e vírgula (;). Observe a seguir um código HTML usando o CSS que altera a cor de fundo da página HTML.

```
<html>
<head>
<title>Aprendendo CSS</title>
</head>
<body style="background-color:yellow;">
</body>
</html>
```

Note que dentro da tag *body* foi colocado o comando que permite alterar a cor de fundo, através do seguinte comando CSS:
style="background-color:yellow;"

Após iniciar com a tag *style*, verifica-se que todos os comandos ficam entre aspas e encerra-se com o ponto e vírgula. A seguir o resultado deste código verificado no browser do smartphone/tablet. Observe que a cor de fundo foi modificada.



Outras cores podem ser utilizadas, como as listadas na tabela a seguir:

Cor **Descrição** Blue Azul Red Vermelho

Yellow Amarelo White Branco Black Preto Green Verde Pode ser feito combinações de cores através do comando

, atribuindo-se um valor de 24 bits em formato background-color hexadecimal, como no próximo exemplo:

```
<html>
<head>
<title>Aprendendo CSS</title>
</head>
<body style="background-color:#FFCC00;">
</body>
</html>
```

O resultado neste caso foi à apresentação de um amarelo mostarda na tela do browser.

3. Alterando a cor de texto

Para alterar a cor do texto impresso no browser, utiliza-se o parâmetro color que pode assumir as mesmas condições do background-color abordado anteriormente. Observe o exemplo a seguir e faça o mesmo para comprovar o seu funcionamento. <html>

```
<head>
<title>Aprendendo CSS</title>
</head>
<body>
<h1 style="color:blue;">Texto Colorido em CSS</h1>
</body>
</html>
```



Texto Colorido em CSS

4. Alterando o alinhamento do texto

Pode-se ajustar o alinhamento dos textos impressos através do CSS, usando para

parâmetros `left` e `right` para alinhamento a esquerda, `right` para alinhamento a direita e `center` para alinhamento centralizado. Observe a seguir o exemplo dado anteriormente com centralização no centro da tela.

```
<html>
<head>
<title>Aprendendo CSS</title>
</head>
<body> <h1 style="color:blue;text-align:center;"> Texto Colorido e Centralizado em CSS </h1>
</body>
</html>
```



Texto Colorido e Centralizado em CSS

5. Alterando o posicionamento absoluto

Pode-se imprimir de forma absoluta em qualquer parte da tela do browser usando a tag chamada `position:absolute`, onde informa-se a distância ao topo da tela e do lado esquerdo através do parâmetro `top` e `left`. O exemplo a seguir imprime no centro da tela, lembrando que tal posicionamento pode ser alterado atribuindo novos valores as tags `top` e `left`.

```
<html>
<head>
<title>Aprendendo CSS</title>
</head>
<body>
<h1 style="color:blue;text-align:center;position:absolute;top:50%;left:50%;">
Texto Colorido, Posicionado e Centralizado em CSS
</h1>
</body>
</html>
```



Texto Colorido, Posicionado e Centralizado em CSS

6. Alterando o fonte usado

É possível trabalhar com diversas fontes, como arial, courier, verdana, calibri, etc. Para isso, usa-se a tag font-family no CSS. Observe o exemplo a seguir, em que a fonte utilizada é a courier.

```
<html>
```

```
<head>
<title>Aprendendo CSS</title>
</head>
<body>
<h1 style="font-family:courier"> Texto Courier
</h1>
</body>
</html>
```



Texto Courier

Quando a fonte utilizada for composta por duas palavras, usase o apóstrofo (‘) para tal marcação. No exemplo a seguir, a fonte courier new foi utilizada, observe como ficou a configuração da mesma no CSS.

```
<html>
<head>
```

```
<title>Aprendendo CSS</title>
</head>
<body>

<h1 style="font-family:'courier new'"> Texto Courier New
</h1>

</body>
</html>
```



Texto Courier New

7.Modificando o tamanho de imagens

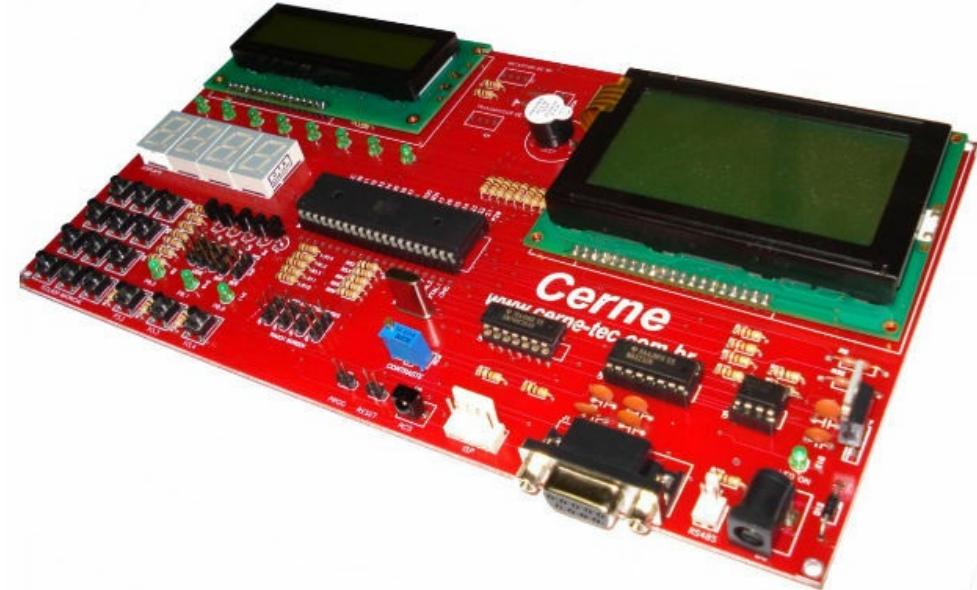
Para alterar o tamanho de imagens, utiliza-se as tags `width` e `height`, onde a primeira permite ajustar a largura da imagem e a segunda a altura. O exemplo a seguir permite realizar tal ajuste.

```
<html>
```

```
<head>
<title>Exemplos CCS</title>
</head>

<body>

</body>
</html>
```



8.Alterando o tamanho das letras

Para alterar o tamanho da letra usa-se a tag `font-size` do CSS.
Verifique o exemplo a seguir.

```
<html>
<head>
```

```
<title>Exemplos CCS</title>
</head>
<body>

<p style="font-size:12pt">Tamanho 12</p> <p style="font-size:20pt">Tamanho 20</p> <p style="font-
size:30pt">Tamanho 30</p> <p style="font-size:40pt">Tamanho 40</p>

</body>
</html>
```



Tamanho 12

Tamanho 20

Tamanho 30

Tamanho 40

9. Deixando o texto em negrito

O texto impresso no browser pode ficar em negrito, onde para isso precisa-se usar a tag `font-weight:bold`.

```
<html>
<head>
```

```
<title>Exemplos CCS</title>
</head>
<body>

<p style="font-weight:bold;">Texto em Negrito</p> </body>
</html>
```



Texto em Negrito

10. Deixando o texto em itálico

Utiliza-se a mesma tag apresentada anteriormente, tendo a diferença que o parâmetro será *italic*. Acompanhe o exemplo.

```
<html>
```

```
 <head>
  <title>Exemplos CCS</title>
 </head>
```

```
<body>
<p style="font-style:italic;">Texto em Itálico</p>
</body>
</html>
```



Texto em Itálico

Capítulo IV PHP

1. Introdução

Neste capítulo será apresentado exemplos práticos de forma a mostrar os recursos da

linguagem PHP. No próximo capítulo será estudado o banco de dados MySQL, para desenvolver uma aplicação integrando PHP e MySQL.

2. Apresentando Informações

Todos os códigos feitos em PHP devem ser salvos com a extensão .php. Além disso, a estrutura básica de um programa em php é a seguinte.

```
<html>
<body> <?php
?> </body>
</html>
```

O comando usado para apresentar informações é o echo, conforme o exemplo abaixo.

```
<html>
<body>
<?php echo 'Primeiro Exemplo';
?>
</body>
</html>
```

Verifique que os comandos em php terminam com ; (ponto e vírgula). Abaixo o resultado do primeiro exemplo em php sendo executado no Safari do smartphone/tablet.



Primeiro Exemplo

No exemplo acima, o mesmo foi salvo como exemplo.php. Logo, considerando que o IP do servidor seja 192.168.2.6 a chamada do endereço do browser ficará conforme link a seguir.

<http://192.168.2.6/exemplo.php>

3. Comentários

Os comentários são trechos de código usados para informar o que determinada parte de código realiza. São muito úteis, pois através deles pode-se visualizar novamente um determinado código e lembrar o que este faz. Os comentários são representados por//. Observe abaixo um exemplo de comentário:

```
<html>
<body>

<?php
echo 'Primeiro Exemplo'; //Mostra mensagem na tela
?>
</body>

</html>
```

No php há o comentário de linha, como apresentado anteriormente e o comentário de múltiplas linhas. A seguir um exemplo deste tipo de comentário.

```
<body>
<?php

echo 'Primeiro Exemplo'; /* Primeiro
Exemplo
No PHP */
?>
</body>

</html>
```

Neste caso o comentário é representado por/* para iniciar e para terminar o bloco comentado.*/

4. Variáveis

Variáveis são porções de memória volátil, onde pode-se guardar informações para posterior tratamento. Há diversos tipos de dados como inteiro, ponto flutuante e strings, sendo estes os principais. Não é necessário declarar variáveis no php e todas as variáveis devem iniciar com o caracter \$. Abaixo um exemplo que demonstra os três tipos principais de variáveis sendo atribuídos valores a estas, para em seguida apresentar seu conteúdo na tela do tablet.

```
<html>
<body>
<?php
$b=50.35; //Valor casa decimal $c='PHP'; //Valor string
$c='PHP'; //Valor string
$a=1000; //Valor Inteiro
echo $a; //Imprime valor de a echo $b; //Imprime valor de b echo $c; //Imprime valor de c
?>
</body>
</html>
```



100050.35PHP

Note que todas as variáveis foram impressas na mesma linha, dificultando assim a visualização dos valores presentes na mesma. Para sanar tal efeito utilize em conjunto o comando HTML
, como sugere o código a seguir.

```
<body> <?php $a=1000;  
$b=50.35;  
$c='PHP';
```

```
echo "$a <br>"; echo "$b <br>"; echo "$c <br>";
?>
</body>

</html> //Valor Inteiro
//Valor casa decimal //Valor string
//Imprime valor de a //Imprime valor de b //Imprime valor de c
```

O resultado será o da tela abaixo.



É importante frisar que o PHP é uma linguagem do tipo *case sensitive*, ou seja, diferencia os caracteres maiúsculos e minúsculos. Se, por exemplo, declarar uma variável e chamá-la de \$a e declarar outra variável e chamá-la de \$A, ambas ocuparão regiões diferentes da memória do computador.

5. Arrays

Através dos arrays, pode-se ter uma variável, porém com diversos índices, no qual acessa-se o seu conteúdo através deste valor. Verifique o exemplo a seguir.

```
<html> <body> <?php  
$media[0]=10; $media[1]=5; $media[2]=7; $media[3]=8;  
echo nl2br("$media[0] \n\r"); echo nl2br("$media[1] \n\r"); echo nl2br("$media[2] \n\r"); echo nl2br("$media[3] \n\r");  
?>  
</body>
```



10

5

7

8

6. Operadores Aritméticos

O php dispõe dos operadores padrão utilizados em linguagens de programação como o adição (+), subtração (-), multiplicação (*) e divisão (/). Observe o exemplo a seguir em que são realizadas as quatro operações aritméticas para posteriormente, atualizar o valor na tela do smartphone/tablet.

```
<html>
<body>
<?php
$a=10;
$b=5;
$c=$a + $b; //Soma
$d=$a - $b; //Subtração $e=$a * $b; //Multiplicação $f=$a / $b; //Divisão
?>
</body>
</html>
```

Pode-se melhorar a apresentação do texto, colocando cada um dos resultados em uma linha, usando para isso o comando de quebra de linha já visto em tópicos passados no HTML, que é o comando `
`. Além disso, é possível colocar um texto em formato de string que permita visualizar de forma mais clara os resultados. Observe o exemplo abaixo e teste o mesmo no browser do smartphone/tablet.

```
<html>
<body>
<?php
$a=10;
$b=5;
$c=$a + $b; //Soma
$d=$a - $b; //Subtração $e=$a * $b; //Multiplicação $f=$a / $b; //Divisão
echo "Soma: $c <br>"; //Apresenta $c echo "Subtração:$d <br>"; //Apresenta $d echo "Multiplicação: $e <br>";//Apresenta $e echo "Divisão: $f <br>";//Apresenta $f
?>
</body>
```



Soma: 15
Subtração: 5
Multiplicação: 50
Divisão: 2

7. Operadores Lógicos

O php apresenta os operadores lógicos convencionais, como *and*, *or* e *not*. No exemplo a seguir, o resultado está apresentado no Safari do smartphone/tablet.

```
<html>
<body>
<?php
```

```
echo "And: $c <br>"; //Apresenta $c echo "Or: $d <br>"; //Apresenta $d
$a=1;
$b=0;
$c=$a and $b;//And $d=$a or $b;//Or
?>
</body>
</html>
```



And: 1

Or: 1

8. Operadores Relacionais

O PHP disponibiliza os operadores relacionais, que são úteis ao comparar duas variáveis. Observe abaixo a lista de operadores relacionais da linguagem.

Operador Função == Testa a igualdade != Testa a diferença > Testa se é maior

\geq Testa se é maior ou igual \leq Testa se é menor
 \leq Testa se é menor ou igual

O próximo exemplo faz uso do operador igualdade. Neste exemplo, foi usada a declaração *if* (se) que testa o resultado e mostra a informação se são iguais ou diferentes.

```
<html>
<body>
<?php

$a=1; $b=0;
if($a == $b)

{
echo "A e B são iguais <br>";
}
else
{
echo "A e B são diferentes <br>";
}
?>

</body> </html>
```



A e B são diferentes

O próximo exemplo trata todos os operadores relacionais.

```
<html>
<body>
<?php
$a=1; $b=0;
if($a == $b)
```

```
{  
echo "A e B são iguais <br>";  
}  
if($a > $b)  
{  
echo "A é maior B <br>";  
}  
if($a >= $b)  
{  
{ echo "A é menor que B <br>";  
}  
if($a <= $b)  
{  
echo "A é menor ou igual a B <br>";  
}  
if($a != $b)  
{  
echo "A é diferente de B <br>";  
}  
?  
</body>  
</html>
```

```
echo "A é maior ou igual a B <br>"; }  
if($a < $b)
```



A é maior B

A é maior ou igual a B

A é diferente de B

9. Concatenação de Strings

Concatenação é o fato de juntar, ou seja, unir duas ou mais strings como pode ser visualizado no próximo exemplo.

```
<html> <body> <?php
```

```
$a='Cerne '; $b='Tecnologia'; echo $a.$b;
```

```
?>  
</body>  
</html>
```

No PHP, o operador de concatenação é o . (ponto). O resultado no browser está apresentado abaixo.



Cerne Tecnologia

10. Controle de Repetição While

O *while* (enquanto) é um tratamento que permite repetir um bloco de comandos determinado número de vezes enquanto a condição avaliada for verdadeira. No exemplo abaixo, será impresso de 1 a 10 usando o controle *while*.

```
<html> <body>
```

```
<?php $a=1;  
while($a<=10) {  
  
echo "$a <br>"; $a++;  
}  
?>  
</body>  
</html>
```

A variável de controle \$a faz com que o bloco de impressão seja executado 10 vezes, já que o while será tratado enquanto \$a for menor ou igual a 10. Dentro do bloco, a variável é incrementada de uma unidade através do \$a++, que faz com que a cada iteração \$a seja incrementado até que seja maior que 10 e o loop while deixe de ser executado. A seguir o resultado deste exemplo no browser.



1
2
3
4
5
6
7
8
9
10

Se o código fosse repetido 100 vezes a condição de teste do loop while ficaria da seguinte forma, no qual observa-se que apenas a condição de término foi alterada.

```
<html>
<body>
<?php $a=1;
```

```
while($a<=100) {  
    echo "$a <br>"; $a++;  
}  
?>  
</body>
```

11. Controle de Repetição For

O controle de repetição `for` apresenta um funcionamento parecido com o `while`, no qual é definido o valor inicial, final e condição de término para que determinado loop seja repetido.

```
<html> <body>  
<?php  
for($a=1;$a<=15;$a++) {  
    echo "$a <br>"; }  
?  
</body> </html>
```

Inicialmente é atribuído um valor inicial a variável `$a`, que neste caso é 1. Em seguida, o loop é executado enquanto esta variável for menor ou igual a 15. Na próxima etapa há o incremento de `$a`, que neste caso é unitário. Dentro do loop de repetição, é impresso o conteúdo da variável `$a`, onde observa-se o resultado a seguir no browser.



1
2
3
4
5
6
7
8
9
10
11
12
13
14
15

Para aumentar ou diminuir o número de repetições basta modificar a condição de incremento ou término do bloco for.o número de repetição. Abaixo, temos um exemplo que o bloco é repetido apenas 5 vezes.

```
<html>
<body>
```

```
<?php for($a=1;$a<=5;$a++) {  
    echo "$a <br>"; }  
?>  
</body>
```

12. Tratamento Switch

Este tratamento permite que uma variável seja comparada com várias opções chamadas `case` e caso alguma delas seja igual, um comando poderá ser executado. O próximo exemplo mostra à variável sendo inicializada e em seguida passando pelo tratamento `switch`, onde caso o valor presente na mesma seja igual a algum dos cases, será impresso na tela do smartphone/tablet uma informação.

```
<html>  
<body>  
  
<?php $a=3;  
switch($a) {  
  
    case 1:  
        echo 'O valor é 1';break;  
    case 2:  
        echo 'O valor é 2';break;  
    case 3:  
        echo 'O valor é 3';break;  
    case 4:  
        echo 'O valor é 4';break;  
    case 5:  
        echo 'O valor é 5';break;  
    case 6:  
        echo 'O valor é 6';break;  
    default:  
        echo 'Nenhum valor encontrado!'; break;  
}  
?>  
</body>
```

Como a variável possui o valor inicial 3, a seguinte mensagem será impressa.



O valor é 3

Ao atribuir a variável \$a o valor 7, não haverá tratamento para este valor, logo a condição *default* será executada. O *default* sempre é executado caso não seja encontrado nos cases anteriores algum caso que satisfaça a condição em teste. A tela a seguir mostra o resultado observado no Safari.



Nenhum valor encontrado!

13. Criando Funções

Além das funções disponibilizadas pelo PHP, é possível criar suas próprias funções neste ambiente, onde estas são identificadas como funções do usuário. A criação de funções é interessante, pois desta forma é possível estabelecer uma biblioteca, onde precisando de determinada tarefa não será necessário recriar aquela função, obtendo-se assim a reutilização de código. A criação de uma função no PHP segue a seguinte sintaxe.

```
function nome_da_função ($arg1, $arg2, ... ,$argn)
{
    return $retorno;
}
```

Em `nome_da_função` define-se o nome que a função terá no escopo do programa. Não pode haver espaços nesta declaração. Em é feita a declaração dos parâmetros de`$arg1, $arg2,...,$argn` entrada da função. Em `return $retorno` define-se o valor de retorno que a função fornecerá para quem a chamou.

No exemplo abaixo é apresentada uma função para cálculo de circunferência. Note que a entrada da função é o raio e a mesma retorna o valor calculado do perímetro.

```
function circunferencia ($raio) {
    $circun=2 * 3.14 * $raio; return $circun;
}
```

Abaixo um exemplo completo utilizando a função apresentada.

```
<html>
<body>

<?php function circunferencia ($raio) {

$circun=2 * 3.14 * $raio; return $circun;
}

echo circunferencia(2); ?> </body>
</html>
```

Ao chamar a função é passado o parâmetro 2, ou seja, o raio. A mesma calcula baseado neste valor o perímetro e retorna com o valor esperado, apresentado no resultado abaixo.



12.56

É possível definir quantas funções o seu código necessitar, tornando assim o trabalho mais produtivo e modular.

14. Funções do PHP

O PHP vem integrado com uma série de funções. São funções que englobam tratamento de strings e funções matemáticas que podem ser úteis na programação. Neste tópico, algumas destas funções serão verificadas.

14.1 Funções Matemáticas

ROUND

Esta função permite arredondar um valor. Por exemplo, se o

resultado com casa decimal for 1.51, ela irá arredondar para 2 e se for 1.49 irá arredondar para 1. Neste caso, se maior que a metade do valor, neste caso 1.50, o resultado será arredondado para cima e se menor para baixo. Em seguida um exemplo.

```
<html>
<body>

<?php
echo round(1.49); //Arredonda para 1 echo "<br>"; //Salta para próxima linha echo round(1.51); //Arredonda para 2
?>
</body>
```



1

2

LOG

Esta função calcula o logaritmo natural, usando a base neperiana e . A seguir um exemplo.

```
<html>
<body>
```

```
<?php
echo log(50);
?>
```

</body>



3.9120230054281

LOG10

Esta função calcula o logaritmo na base 10. A seguir um exemplo.

```
<html>
<body>

<?php
echo log10(100);
?>
```

</body>



2

SQRT

Esta função permite calcular a raiz quadrada de um número. Abaixo um exemplo.

```
<html>
<body>
```

```
<?php
echo sqrt(100);
?>
```

</body>



10

PI

Representa a constante π já declarada no php, de acordo com o exemplo a seguir.

```
<html>
<body>
```

```
<?php
echo pi();
?>
```

</body>



3.1415926535898

COS

Calcula o cosseno de um número. O parâmetro de entrada deve estar em radianos, conforme mostra o próximo exemplo.

```
<html>
<body>
```

```
<?php
echo cos(PI());
```

?>

</body>



- 1

SIN

Calcula o seno de um número. O parâmetro de entrada deve estar em radianos.

```
<html>
<body>

<?php
echo sin(PI());
?>
```

</body>



1.2246063538224E-16

TAN

Calcula a tangente de um número. O parâmetro de entrada deve estar em radianos.

```
<html>
<body>
```

```
<?php
echo tan(PI());
?>
```

</body>



-1.2246063538224E-16

ACOS, ASIN, ATAN

Calcula o arco cosseno, seno e tangente de um número.

```
<html>
<body>

<?php
echo acos(0);
?>
```

</body>



1.5707963267949

14.2 Funções de strings

STRTOLOWER

Esta função permite colocar todos os caracteres em letras minúsculas.

```
<html>
<body>
```

```
<?php
$a="ABCD";
```

```
echo strtolower($a);
```

```
?>  
</body>
```



abcd

STRTOUPPER

Esta função permite colocar todos os caracteres com letras maiúsculas.

```
<html>
<body>
```

```
<?php
$a="abcd";
echo strtoupper($a);
```

?>
</body>



ABCD

TRIM

Esta função retira os espaços da parte direita e esquerda de uma determinada string.

```
<html>
<body>
```

```
<?php
$a=" abcd ";
echo trim($a);
```

?>
</body>



abcd

LTRIM

Esta função retira os espaços da parte esquerda de uma determinada string.

```
<html>
<body>

<?php
$a=" abcd ";
Echo ltrim($a);
```

?>
</body>



abcd

RTRIM

Esta função retira os espaços da parte direita de uma determinada string.

```
<html>
```

```
<body>
```

```
<?php
```

```
$a=" abcd ";  
echo rtrim($a);
```

?>
</body>



abcd

Capítulo V MySQL

1. Introdução

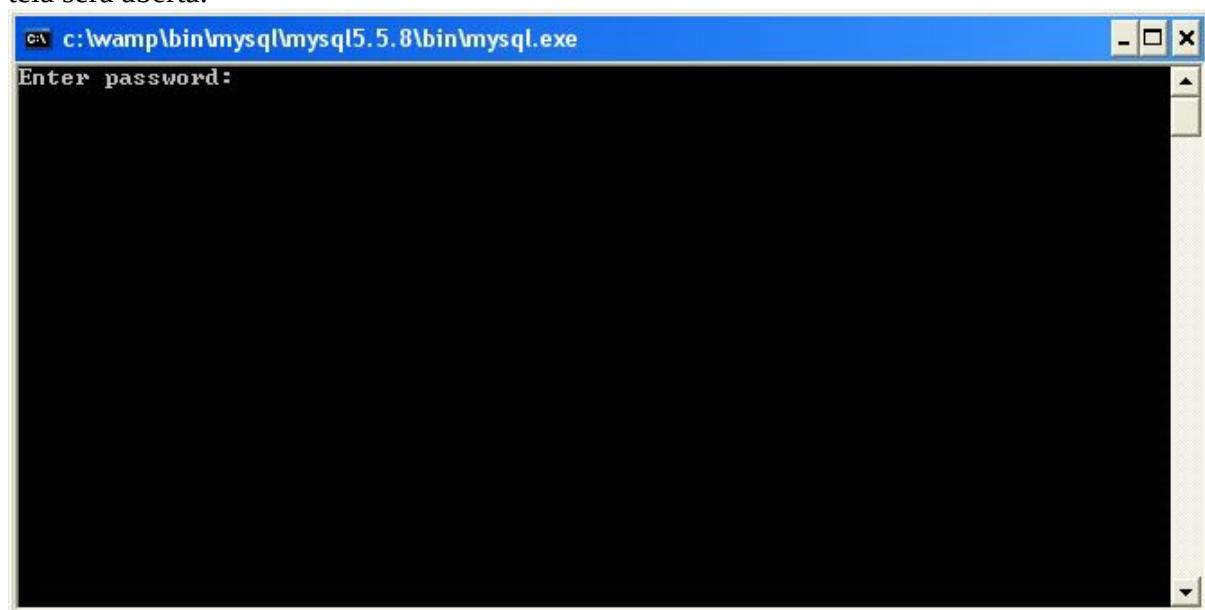
Neste capítulo será apresentado o banco de dados MySQL, mostrando como criar tabela assim

como inserir e deletar registros e fazer buscas em um BD (banco de dados).

2.Iniciando o MySQL

O primeiro passo para trabalhar com o MySQL é inicializando o mesmo. Para isso, dê um clique com o botão esquerdo do mouse sobre o ícone do *WampServer*, as seguintes opções irão aparecer.

Escolha a opção MySQL e em seguida a opção Console MySQL. Neste momento, a seguinte tela será aberta.



O MySQL vem inicialmente sem nenhuma senha, pressione então ENTER. A tela ficará da seguinte forma:

O primeiro comando a ser apresentado é o `show databases;` que mostra todos os BDs disponíveis naquele instante. Digite então e pressione enter em seguida.`show databases;`

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.5.8-log MySQL Community Server (GPL)

Copyright (c) 2000, 2010, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

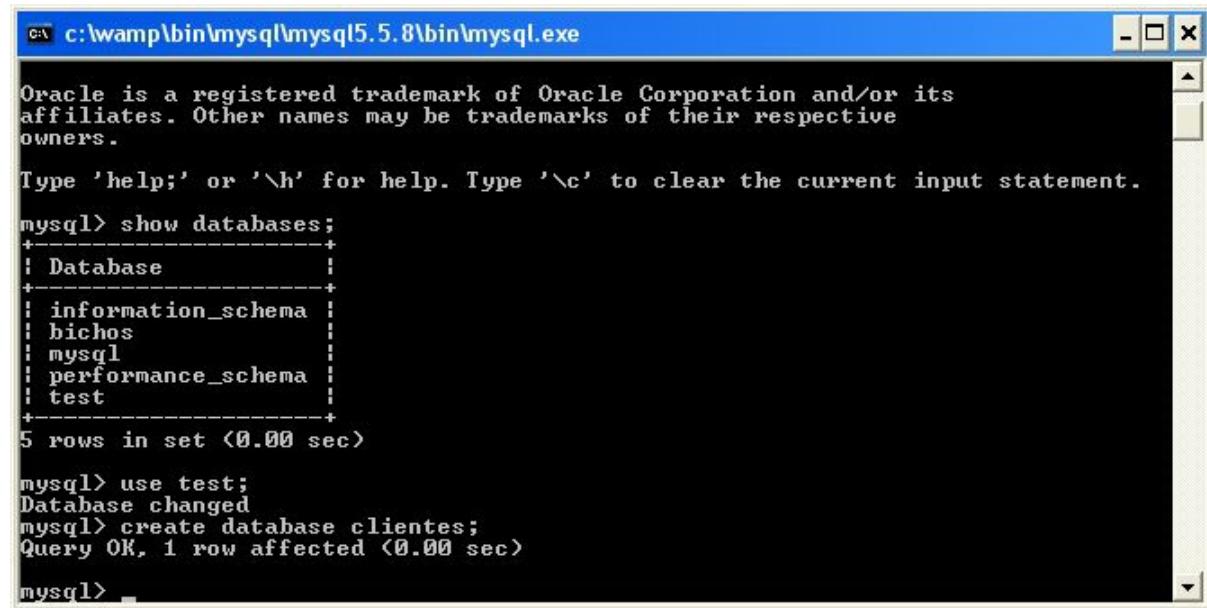
mysql> show databases;
+--------------------+
| Database          |
+--------------------+
| information_schema|
| bichos            |
| mysql              |
| performance_schema|
| test               |
+--------------------+
5 rows in set (0.05 sec)

mysql>
```

Note pela figura acima que foram detectados 5 bancos de dados. Para selecionar um dos

bancos de dados listados, usa-se o comando `use`. Por exemplo, digamos que seja selecionado o banco de dados `test`, o comando ficaria da seguinte forma.

Verifique que o MySQL informa que o banco de dados foi selecionado através do comando `Database changed`. Para criar um novo banco de dados, faz-se uso do comando `create database`. Observe abaixo, que foi criado um novo banco de nome `clientes` dados chamado `clientes`.



```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| bichos |
| mysql |
| performance_schema |
| test |
+-----+
5 rows in set (0.00 sec)

mysql> use test;
Database changed
mysql> create database clientes;
Query OK, 1 row affected (0.00 sec)

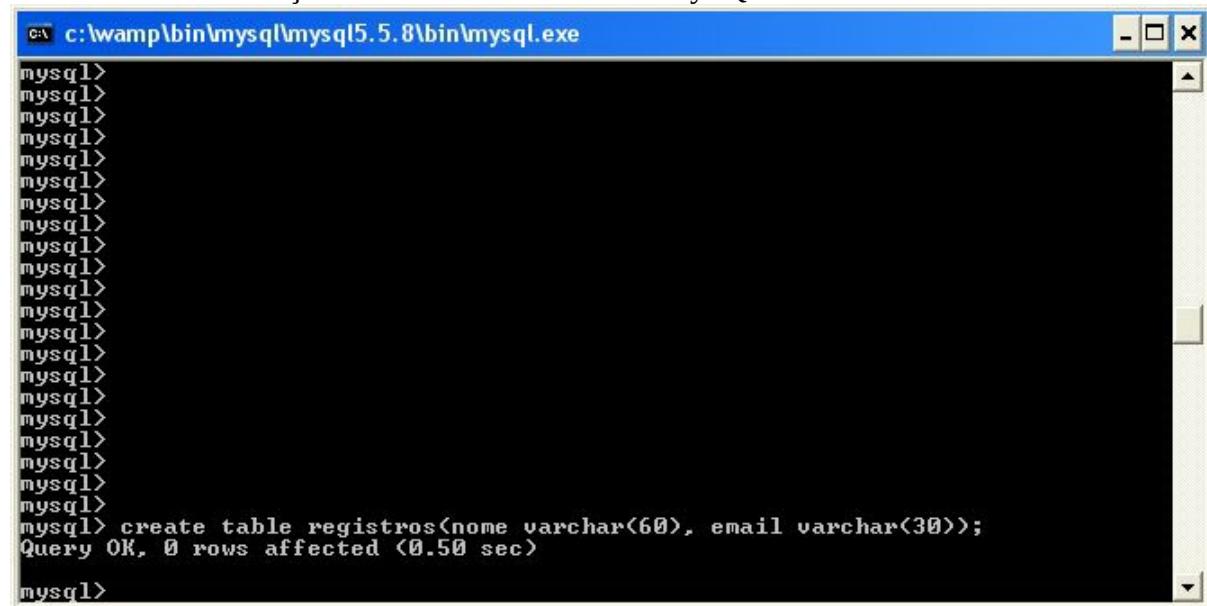
mysql>
```

Para selecionar este BD, faz-se uso novamente do comando `use`.

Um banco de dados é composto por tabelas. Para criar tabelas dentro deste banco de dados, usa-se o comando `create table`. Neste exemplo que está interessado em cadastrar clientes, pode-se ter o nome e e-mail dos mesmos. Suponha que o nome terá o máximo de 60 caracteres e o e-mail o máximo de 30 caracteres, sendo assim, a criação da tabela ficaria da seguinte forma.

```
create table registros(nome varchar(60),email varchar(30));
```

Observe abaixo a criação da tabela diretamente no MySQL:



```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
mysql>
mysql> create table registros(nome varchar(60), email varchar(30));
Query OK, 0 rows affected (0.50 sec)

mysql>
```

Sendo assim, foi criada uma tabela chamada registros que possui dois campos, um chamado nome com no máximo 60 caracteres e email com 30 caracteres por registro.

Para verificar quantos e quais registros há no BD chamado clientes, pode-se digitar:

Show tables;

Abaixo o resultado para este comando diretamente no mySQL:

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
mysql>
mysql> create table registros(nome varchar(60), email varchar(30));
Query OK, 0 rows affected (0.50 sec)

mysql> show tables;
+-----+
| Tables_in_clientes |
+-----+
| registros          |
+-----+
1 row in set (0.00 sec)

mysql>
```

Observe que há uma tabela no BD clientes chamado registros. Para observar quais os campos desta tabela, usa-se o comando describe nome_da_tabela que neste caso chama-se registros. Observe abaixo o resultado do MySQL.

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
mysql>
mysql>
mysql>
mysql>
mysql> create table registros(nome varchar(60), email varchar(30));
Query OK, 0 rows affected (0.50 sec)

mysql> show tables;
+-----+
| Tables_in_clientes |
+-----+
| registros          |
+-----+
1 row in set (0.00 sec)

mysql> describe registros;
+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+
| nome  | varchar(60) | YES |   | NULL    |       |
| email | varchar(30) | YES |   | NULL    |       |
+-----+-----+-----+-----+-----+
2 rows in set (0.09 sec)

mysql>
```

Observe que a tabela é formada por dois campos, como já previsto anteriormente. A partir deste ponto é possível povoar a tabela, em que povoar significa preencher com dados a mesma. Para isso, utiliza-se o comando insert, conforme o exemplo abaixo.

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
mysql> create table registros(nome varchar(60), email varchar(30));
Query OK, 0 rows affected (0.50 sec)

mysql> show tables;
+-----+
| Tables_in_clientes |
+-----+
| registros          |
+-----+
1 row in set (0.00 sec)

mysql> describe registros;
+-----+-----+-----+-----+-----+-----+
| Field | Type  | Null | Key  | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nome  | varchar(60) | YES |       | NULL    |       |
| email | varchar(30) | YES |       | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.09 sec)

mysql> insert into registros values ('Vitor','vitor@cerne-tec.com.br');
Query OK, 1 row affected (0.05 sec)

mysql> _
```

Através do comando `insert into registros values` é possível inserir dados na tabela. O exemplo abaixo insere mais dados na tabela `registros`.

```
c:\wamp\bin\mysql\mysql5.5.8\bin\mysql.exe
! Tables_in_clientes !
+-----+
| registros          |
+-----+
1 row in set (0.00 sec)

mysql> describe registros;
+-----+-----+-----+-----+-----+-----+
| Field | Type  | Null | Key  | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| nome  | varchar(60) | YES |       | NULL    |       |
| email | varchar(30) | YES |       | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.09 sec)

mysql> insert into registros values ('Vitor','vitor@cerne-tec.com.br');
Query OK, 1 row affected (0.05 sec)

mysql> insert into registros values ('Renata','renata@cerne-tec.com.br');
Query OK, 1 row affected (0.08 sec)

mysql> insert into registros values ('Joao','joao@cerne-tec.com.br');
Query OK, 1 row affected (0.06 sec)

mysql> _
```

Pelos comandos acima, foram inseridos 3 registros dentro da tabela. Para ver quais registros estão disponíveis, comando `select`, como apresentado a seguir.

```
select * from registros;
utiliza-se o
O comando acima permite selecionar todos os registros
```

provenientes da tabela `registros`. Abaixo pode-se visualizar este comando diretamente no MySQL, no qual nota-se que todos os registros inseridos foram apresentados.

```
c:\wamp\bin\mysql\mysql5.8\bin\mysql.exe
+-----+
| nome | varchar(60) | YES |      | NULL   |
| email | varchar(30) | YES |      | NULL   |
+-----+
2 rows in set <0.09 sec>

mysql> insert into registros values ('Vitor','vitor@cerne-tec.com.br');
Query OK, 1 row affected (0.05 sec)

mysql> insert into registros values ('Renata','renata@cerne-tec.com.br');
Query OK, 1 row affected (0.08 sec)

mysql> insert into registros values ('Joao','joao@cerne-tec.com.br');
Query OK, 1 row affected (0.06 sec)

mysql> select * from registros;
+-----+
| nome | email           |
+-----+
| Vitor | vitor@cerne-tec.com.br |
| Renata | renata@cerne-tec.com.br |
| Joao | joao@cerne-tec.com.br |
+-----+
3 rows in set (0.00 sec)

mysql>
```

Estes foram os conceitos fundamentais do MySQL, no qual será utilizado no próximo capítulo para integrar com o PHP. Tente refazer todos os exercícios, pois eles serão muito importantes em seguida.

Capítulo VI Integrando PHP e MySQL

1. Introdução

Para acessar conhecer algumas o banco de dados MySQL, será necessário funções que realizem isso. A seguir, será

apresentado uma série de funções no qual faz-se uso para conectar ao banco de dados.

2. Função MYSQL_CONNECT()

Esta função permite a conexão com o banco de dados. Ela deve ser a primeira chamada para estabelecer uma conexão com o BD. Deve-se neste caso informar o host, o usuário e login como parâmetros. Neste exemplo, o host será o localhost, o usuário default é root e não há senha definida. Deve-se guardar o resultado que esta função fornecerá, pois a mesma será usada nos comandos futuros. Abaixo pode-se visualizar um exemplo.

```
<html>
<body>

<?php
$conexao = mysql_connect('localhost', 'root', '');
?>

</body> </html>
```

3. Função MYSQL_SELECT_DB()

Esta função permite selecionar um banco de dados no MySQL.

Deve-se fornecer como parâmetro o nome do BD e o parâmetro que foi retornado pela função mysql_connect(). No exemplo a seguir pode-se visualizar um exemplo, baseando-se no BD criado no exemplo anterior chamado clientes.

```

<html> <body>

<?php
$conexao = mysql_connect('localhost', 'root', ""); mysql_select_db('clientes',$conexao);

?>
</body>

```

4. Função MYSQL_QUERY()

Esta função permite selecionar os dados presentes em um BD através do comando select visto no tópico passado. A seguir um exemplo.

```

<html> <body>

<?php
$conexao = mysql_connect('localhost', 'root', ""); mysql_select_db('clientes',$conexao);
$consulta = 'select * from registros'; $resultado = mysql_query($consulta, $conexao);

?>
</body> </html>

```

5. Função MYSQL_NUM_ROWS()

Esta função permite ler a linha atual do ponteiro da tabela. É importante quando necessita-se ler dados de uma tabela. O próximo código exemplifica o mesmo.

```

<html> <body>

<?php
$conexao = mysql_connect('localhost', 'root', ""); mysql_select_db('clientes',$conexao);
$consulta = 'select * from registros';
$resultado = mysql_query($consulta, $conexao); echo "<table border=1>\n";

echo "<tr><td>Nome</td><td>e-mail</tr>\n";
$linha=0;
while ($linha < mysql_num_rows($resultado)) {
$linha++; }

?>
</body>
</html>

```

6. Função MYSQL_FETCH_ARRAY()

Esta função permite ler o conteúdo da linha de uma tabela. É importante quando deseja-se ler os dados presentes em uma tabela. A seguir um exemplo completo que mostra todo o conteúdo do BD criado MySQL na tela do browser.

```

<html> <body> <?php $conexao = mysql_connect('localhost', 'root', "");

mysql_select_db('clientes',$conexao);
$consulta = 'select * from registros';
$resultado = mysql_query($consulta, $conexao); echo "<table border=1>\n";
echo "<tr><td>Nome</td><td>e-mail</tr>\n"; $linha=0;
while ($linha < mysql_num_rows($resultado)) {

$campo=mysql_fetch_array($resultado); echo("<tr><td>$campo[0]</td>"); echo("<td>$campo[1]</td></tr>"); $linha++;

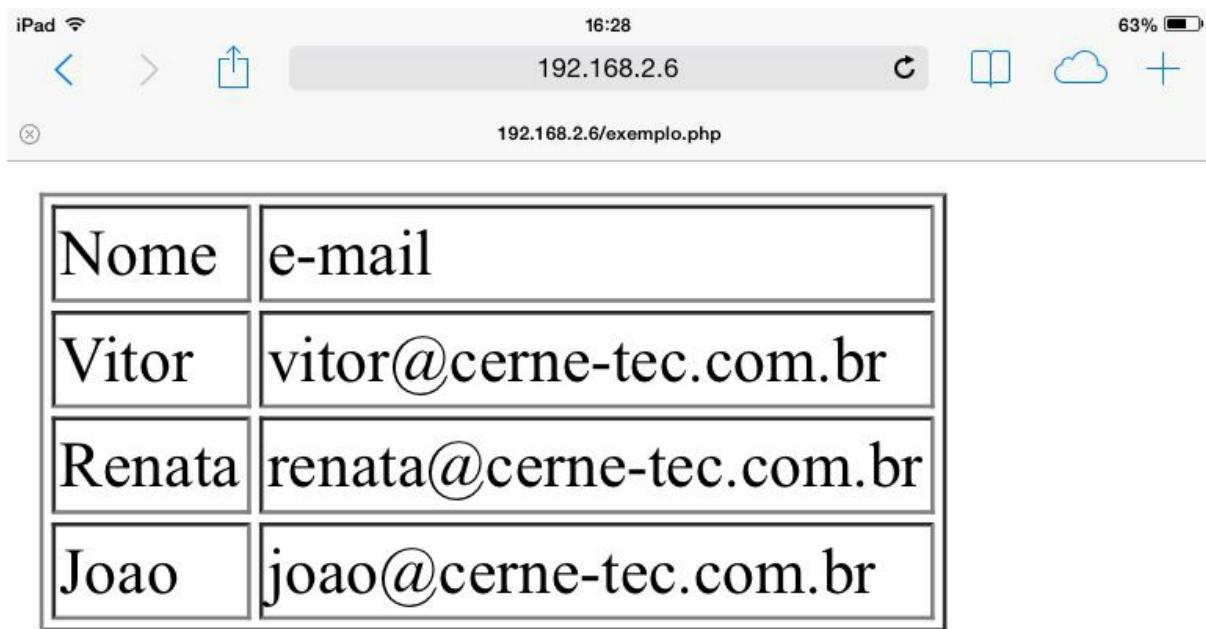
}

?>
</body>

```

O resultado no browser será como apresentado abaixo, de acordo com os dados gravados no

capítulo passado.



The screenshot shows an iPad browser interface with the following details:

- Top bar: iPad with signal, 16:28, 63% battery.
- Address bar: 192.168.2.6 and 192.168.2.6/exemplo.php.
- Toolbar: Back, Forward, Refresh, Home, Cloud, and Plus.

The main content is a table with the following data:

Nome	e-mail
Vitor	vitor@cerne-tec.com.br
Renata	renata@cerne-tec.com.br
Joao	joao@cerne-tec.com.br

Pode-se também inserir dados nesta tabela. O exemplo a seguir é uma complementação do anterior, em que primeiro são inseridos alguns dados e em seguida apresentados no browser.

```
<html>
<body>
<?php
```

```
$conexao = mysql_connect('localhost', 'root', '');
mysql_select_db('clientes',$conexao);
$consulta = 'select * from registros;';

$insert1="insert into registros values('Jony','jony@cerne-tec.com.br')";
$insert2="insert into registros values('José','jose@cerne-tec.com.br')";
$insert3="insert into registros values('Alan','alan@cerne-tec.com.br')";

mysql_query($insert1,$conexao); mysql_query($insert2,$conexao); mysql_query($insert3,$conexao);

$resultado = mysql_query($consulta, $conexao); echo "<table border=1>\n";
echo "<tr><td>Nome</td><td>e-mail</tr>\n"; $linha=0;

while ($linha < mysql_num_rows($resultado)) {
$campo=mysql_fetch_array($resultado); echo("<tr><td>$campo[0]</td>"); echo("<td>$campo[1]</td></tr>"); $linha++;
}

?>
</body> </html>
```

The screenshot shows an iPad displaying a web page at the URL 192.168.2.6/exemplo.php. The page contains a table with two columns: 'Nome' (Name) and 'e-mail' (Email). The table has seven rows, each containing a name and its corresponding email address. The names listed are Vitor, Renata, Joao, Jony, José, and Alan.

Nome	e-mail
Vitor	vitor@cerne-tec.com.br
Renata	renata@cerne-tec.com.br
Joao	joao@cerne-tec.com.br
Jony	jony@cerne-tec.com.br
José	jose@cerne-tec.com.br
Alan	alan@cerne-tec.com.br

7. Método GET

Esta função permite enviar através da própria URL um conteúdo a uma página Web. Por exemplo, digamos que o endereço da página na Web seja www.cerne-tec.com.br. Ao colocar seguido do endereço o campo ?x=valor estaremos atribuindo à variável *x* o *valor* atribuído que será recebido pelo servidor. Observe os exemplos a seguir.

Tabela: Exemplos do método GET www.cerne-tec.com.br?x=1 www.cerne-tec.com.br?y=Cerne www.cerne-tec.com.br?pais=Brasil www.cerne-tec.com.br?city=Rio www.cerne-tec.com.br?uni=USP

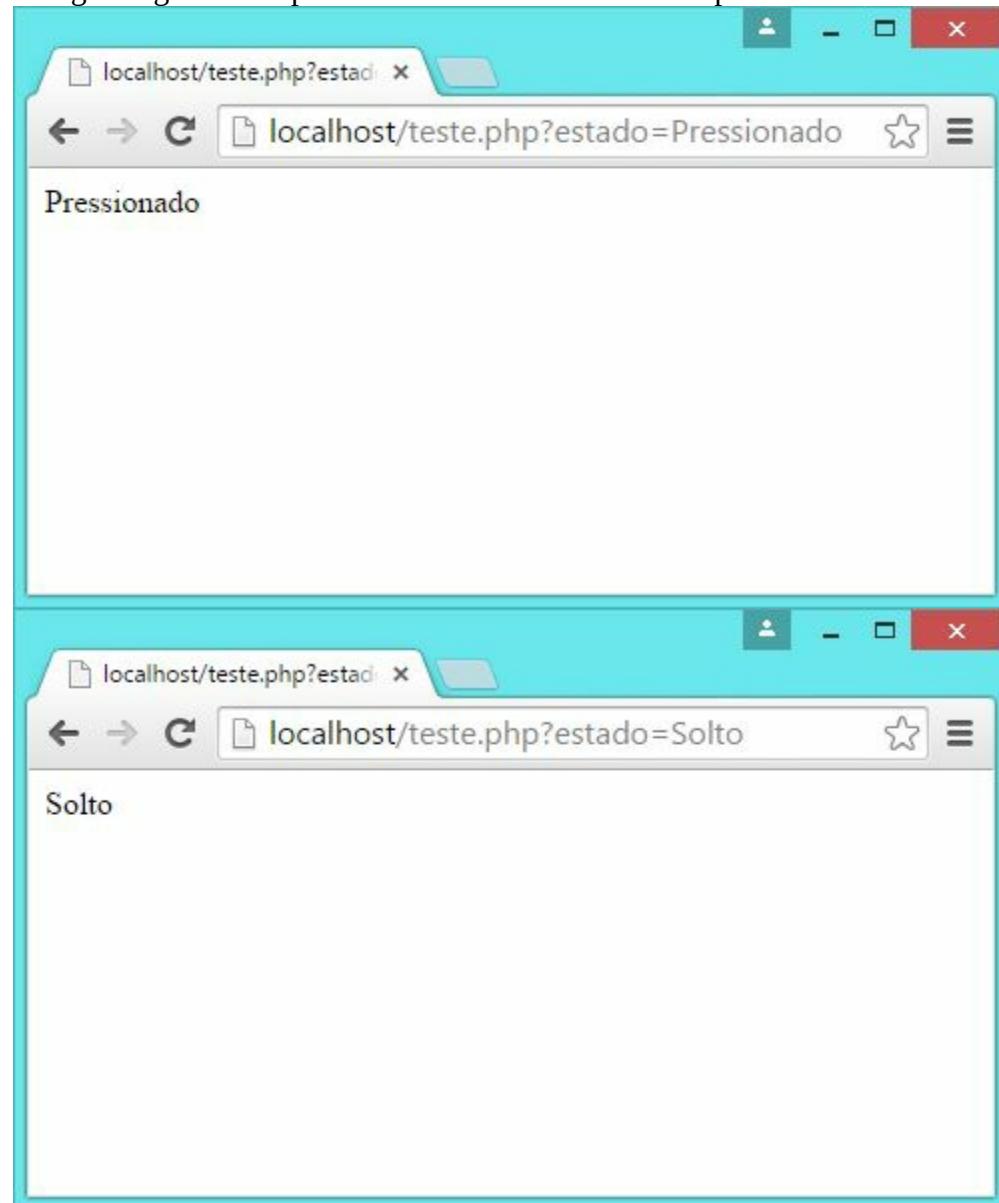
Atribui à variável x o valor 1 Atribui à variável y o valor Cerne Atribui à variável país o valor Brasil Atribui à variável city o valor Rio Atribui à variável uni o valor USP

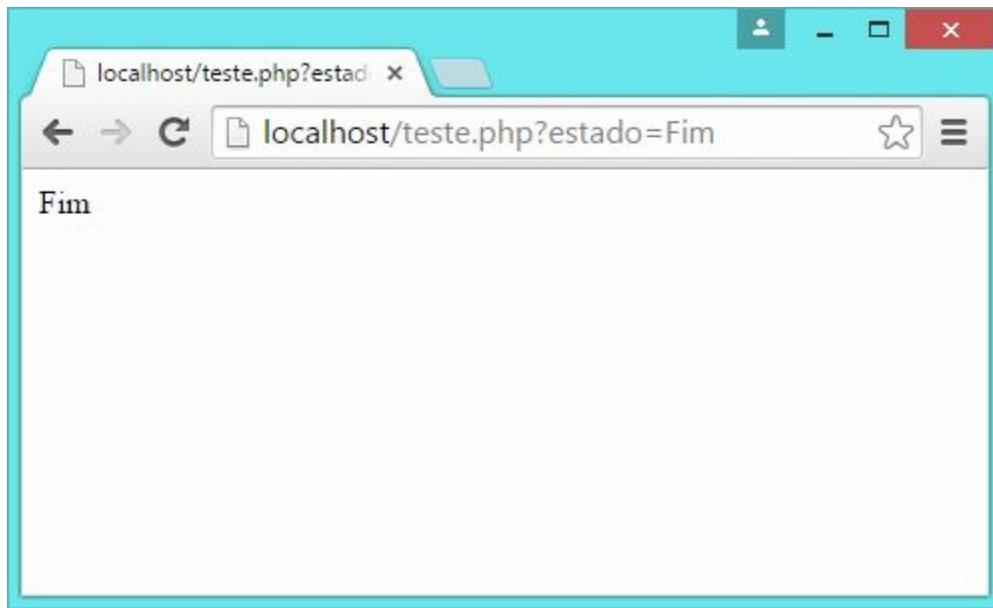
É importante frisar que deve haver no PHP a captura com o mesmo nome do parâmetro de modo a receber corretamente esta informação.

O próximo exemplo demonstra como receber um valor através do método GET e apresentar na tela o seu conteúdo.

```
<?php  
$estado=$_GET["estado"]; echo $estado;  
?>
```

A seguir alguns exemplos. Observe atentamente o campo informado na URL.





Este conceito será utilizado no MIT App Inventor, onde ao pressionar um botão será enviado o método GET para uma página de modo que esta receba este valor e possa salvar no banco de dados MySQL e apresentar em seguida em formato de tabela os registros salvos.

8. Adicionando o banco de dados com o método GET

O primeiro passo consiste em criar um novo banco de dados MySQL chamado *dados*. Em seguida, crie uma tabela chamada *registros* com um campo do tipo varchar (100) e identifique-o por *evento*.

O código PHP apresentado a seguir recebe um valor proveniente do método GET e caso o seu conteúdo seja diferente de vazio (Null), salva-o no banco de dados MySQL e apresenta em seguida todo o seu conteúdo.

```
<?php
$conexao = mysql_connect('localhost', 'root','');
mysql_select_db('dados',$conexao); $estado=$_GET["estado"];

if($estado!="") {
$estado=$estado." ".date(DATE_RFC822);
$insere1="insert into registros values ('$estado')"; mysql_query($insere1,$conexao);

}

$consulta = 'select * from registros'; $resultado = mysql_query($consulta,$conexao);
echo "<table border=0>\n";
echo "<tr><td><b>Registros</b></td></tr>\n";
$linha=0;
while ($linha < mysql_num_rows($resultado)) {

$campo=mysql_fetch_array($resultado);
echo("<tr><td>$linha</td><td>$campo[0]</td></tr>"); $linha++;

}
?>
```

As figuras a seguir mostram três testes feitos de maneira seguida. Observe que cada evento vai sendo salvo no banco de dados e em seguida todo o seu conteúdo é apresentado em tela.

The image displays three separate browser windows, each showing a list of log entries. The browser has a green header bar and a white content area. Each window has a title bar showing the URL `localhost/teste.php?estad`.

Top Window (Estado: Inicio):

	Registros
0	Inicio Fri, 10 Apr 15 13:19:02 +0000

Middle Window (Estado: Fim):

	Registros
0	Inicio Fri, 10 Apr 15 13:19:02 +0000
1	Fim Fri, 10 Apr 15 13:22:13 +0000

Bottom Window (Estado: Inicio):

	Registros
0	Inicio Fri, 10 Apr 15 13:19:02 +0000
1	Fim Fri, 10 Apr 15 13:22:13 +0000
2	Inicio Fri, 10 Apr 15 13:22:36 +0000

Além da string enviada através do método GET também é salvo de acordo com a hora do servidor a data no qual o evento é recebido. Isso é alcançado através do método `date()` do próprio PHP.

9. Publicando em um Website

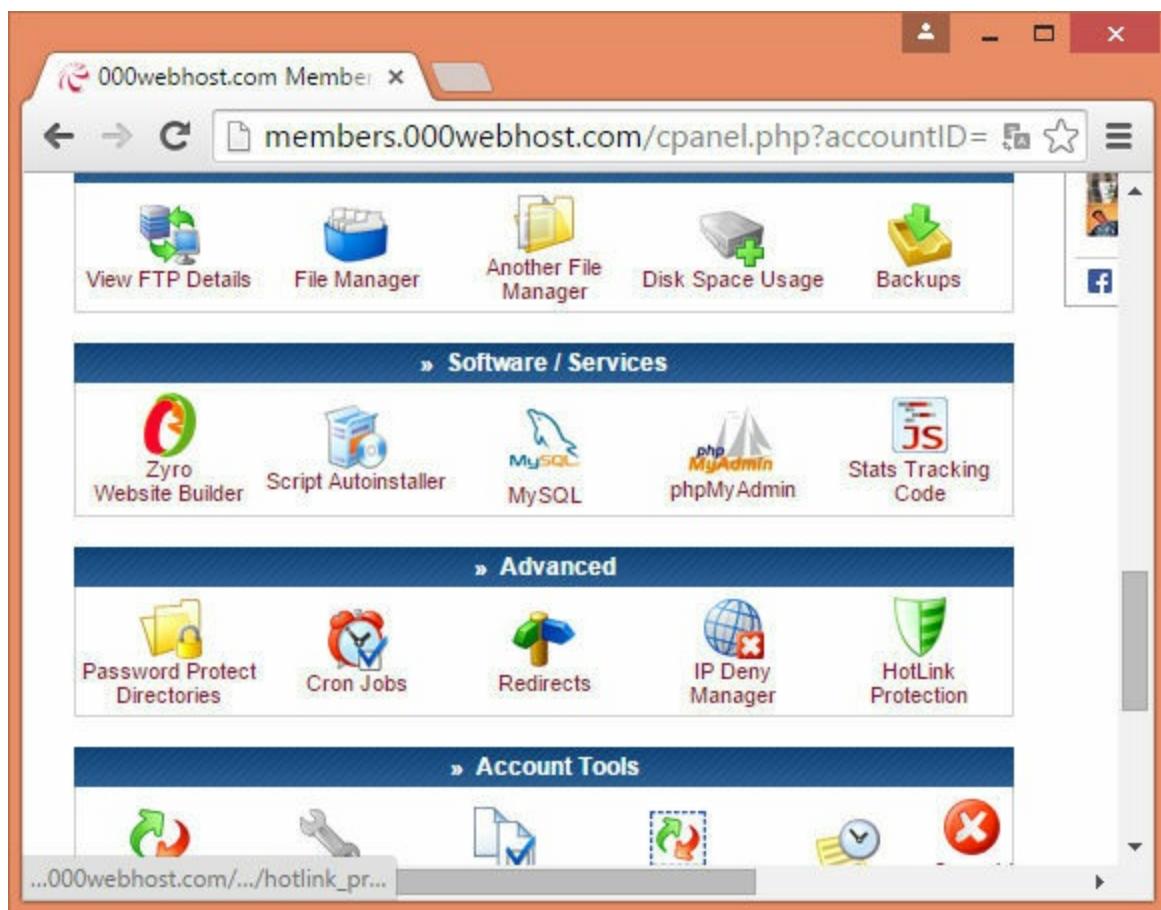
Após desenvolver esta aplicação chega à hora de publicar esta página na internet. Para isso, faz-se necessário um servidor de hospedagem que suporte o PHP e MySQL.

Obs: Nesta literatura foi gratuito, porém o leitor conveniente utilizou um servidor de hospedagem

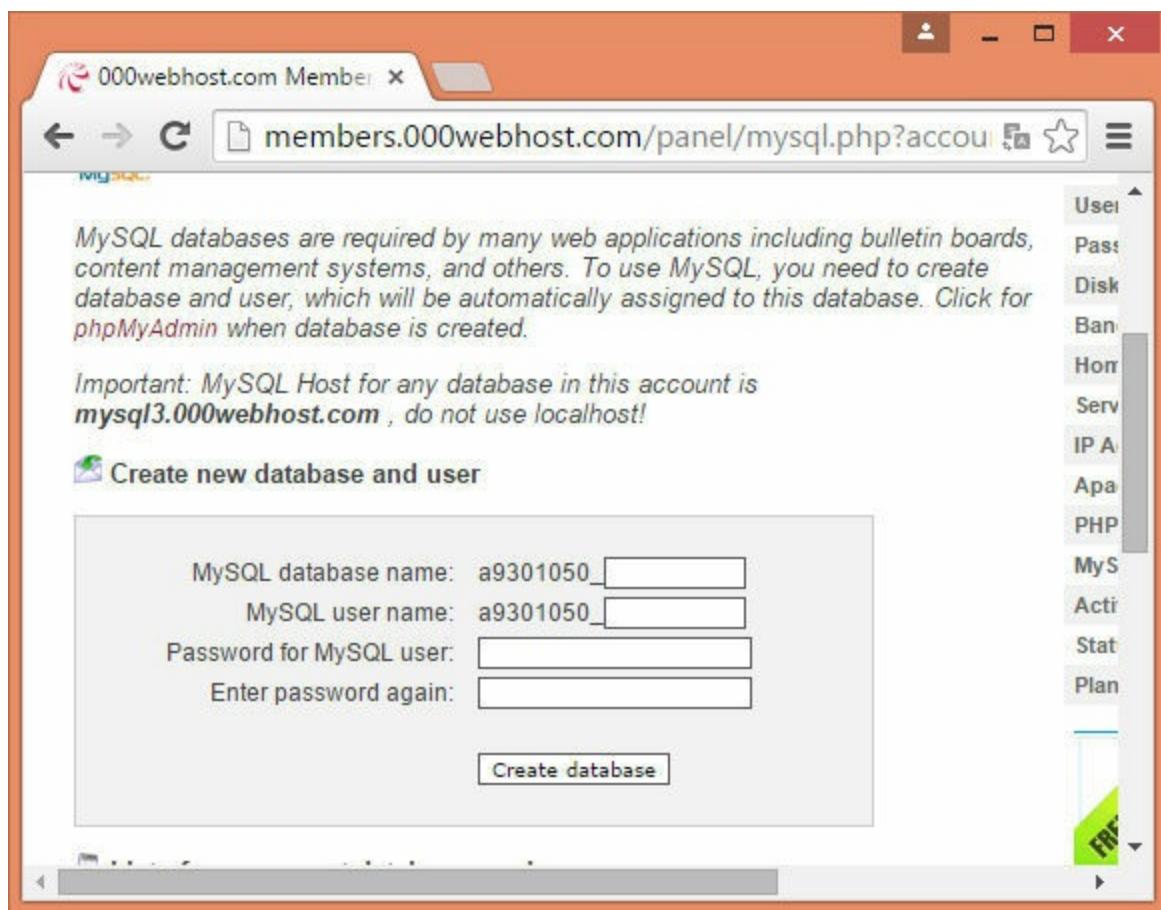
pode utilizar outro que julgar mais Acesse o endereço www.000webhost.com e faça o seu cadastro indo na opção *Sign Up* caso ainda não tenha cadastro.



Após o cadastro faça o *login* e acesse o painel de controle. O primeiro passo consiste em criar o banco de dados MySQL. Para isso, procure a opção MySQL como mostra a próxima figura.

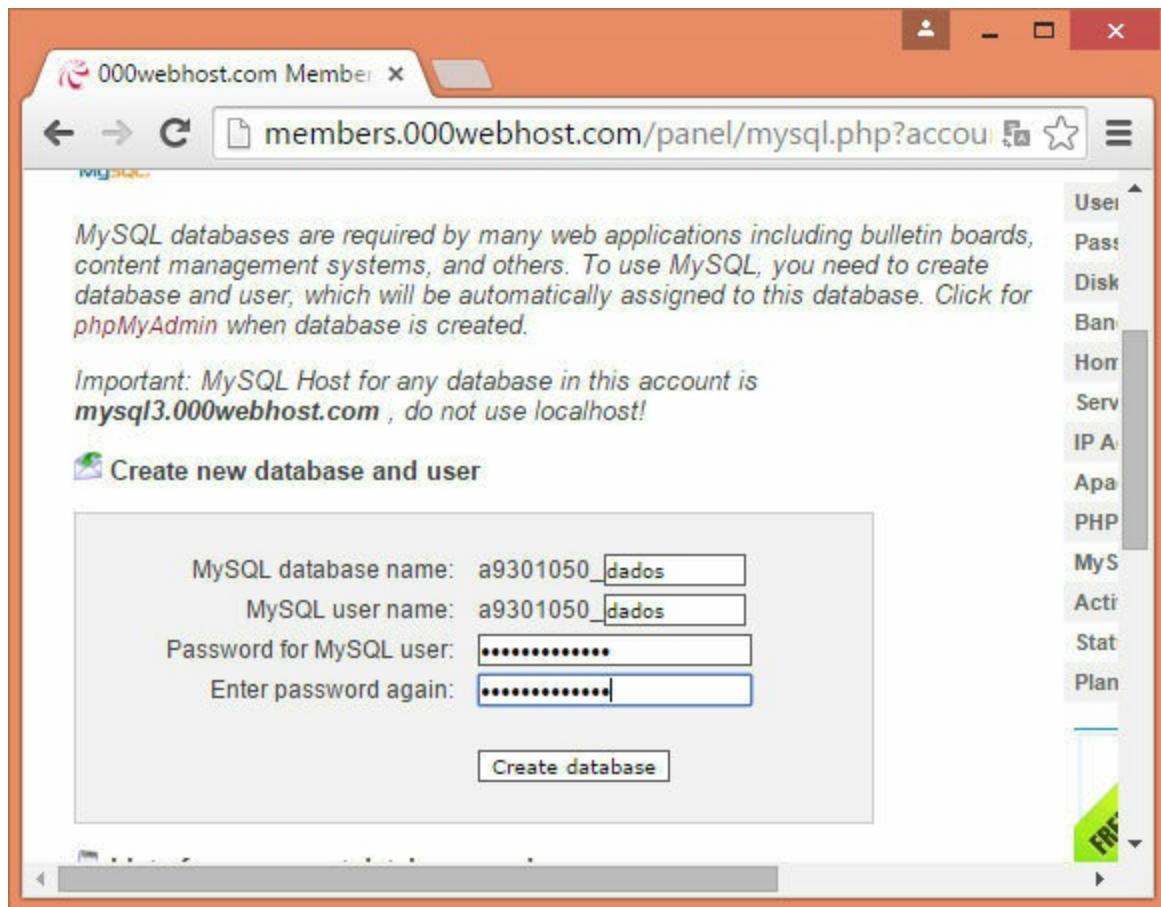


A janela abaixo é apresentada no qual permite a criação de um novo banco de dados.

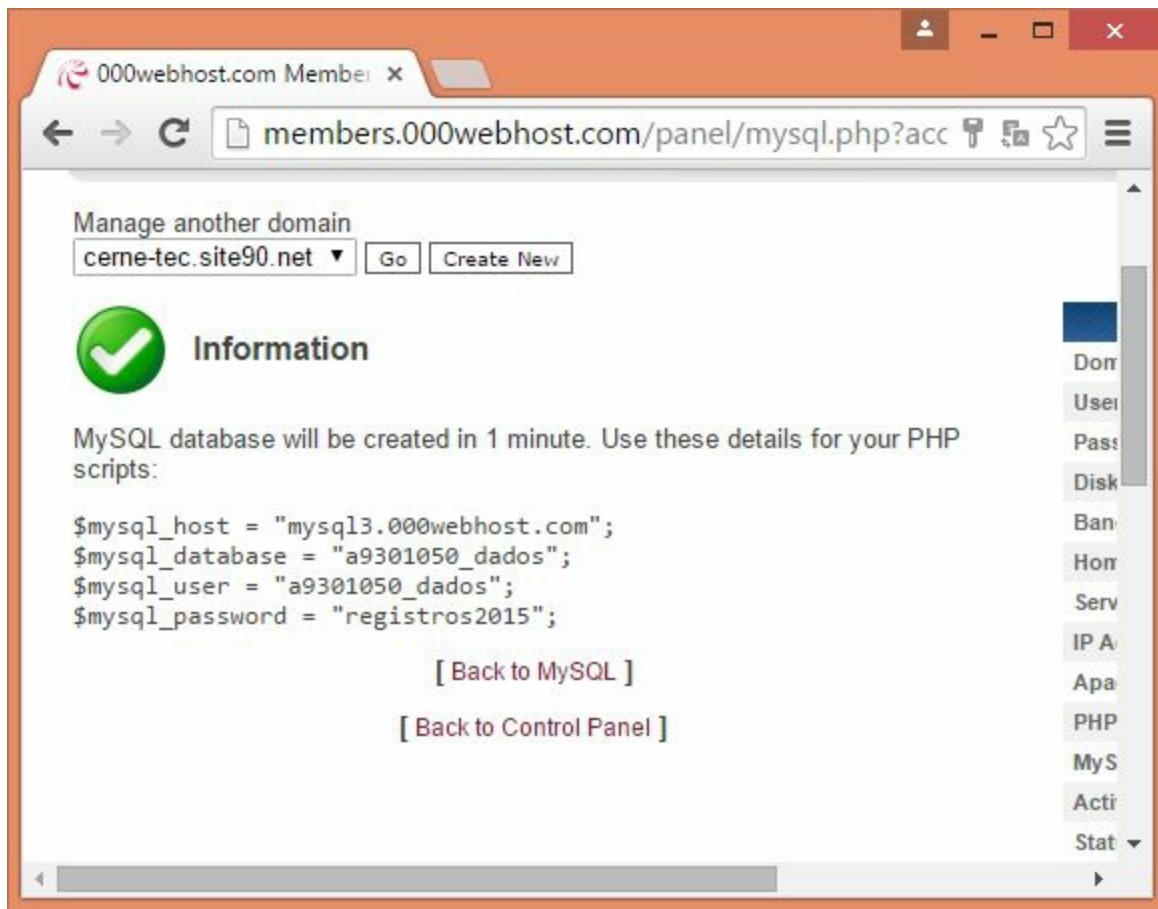


Obs: Os dois primeiros campos iniciam com a9301050_ no exemplo acima. Possivelmente o do leitor será diferente, pois este campo varia de acordo que um novo registro de usuário é criado no site.

Defina o banco de dados como mostra a próxima figura e no campo senha coloque registros2015.



Clique na opção *Create database*. É esperado que a mensagem a seguir seja apresentada, no qual informa que o BD foi criado com sucesso.

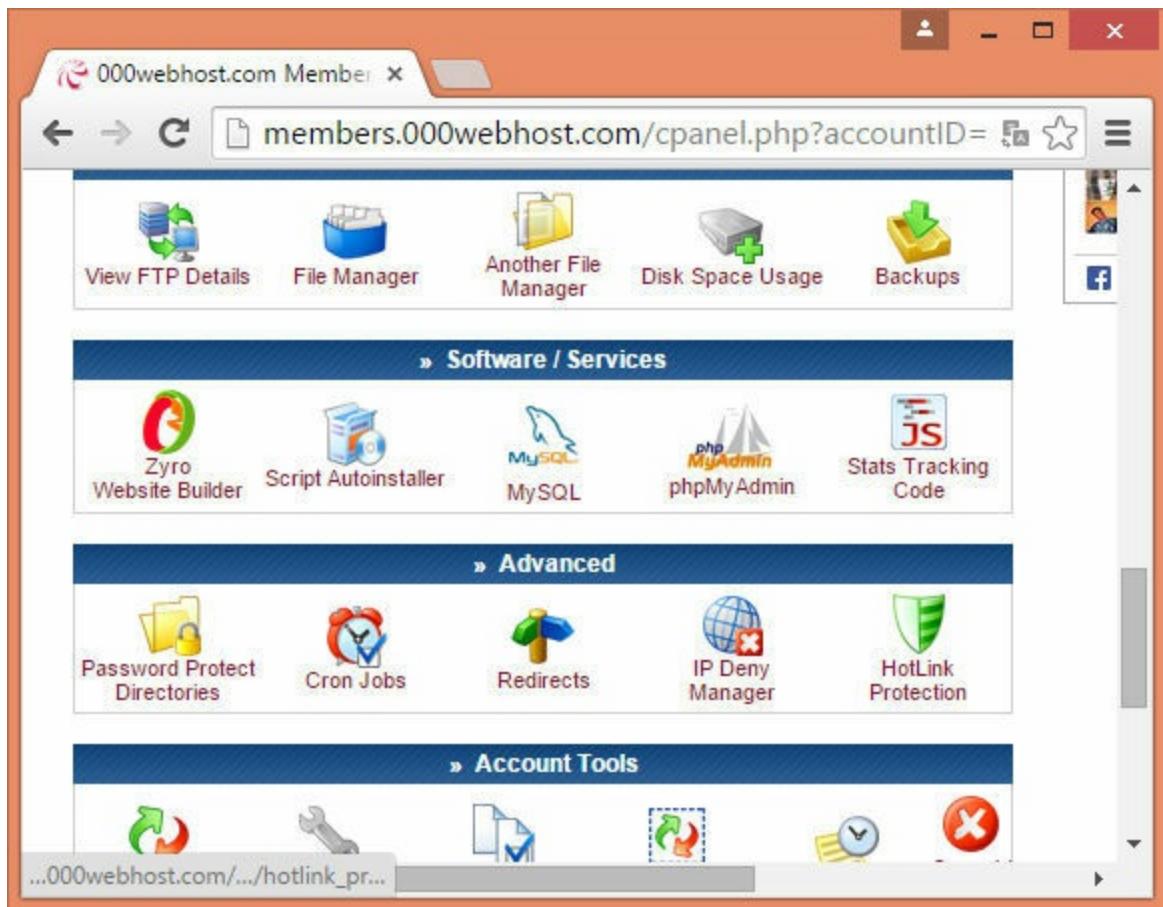


Volte ao arquivo PHP do tópico anterior e altere os campos de host, database, user e password definindo-o conforme as informações mostradas acima. Abaixo o código PHP atualizado.

```
<?php  
  
$conexao = mysql_connect('mysql3.000webhost.com', 'a9301050_dados','registros2015');  
mysql_select_db('a9301050_dados',$conexao);  
$estado=$_GET["estado"];  
  
if($estado!="") {  
$estado=$estado." ".date(DATE_RFC822);  
$insere1="insert into registros values ('$estado')"; mysql_query($insere1,$conexao);  
}  
echo "<table border=0>\n";  
echo "<tr><td><b>Registros</b></td></tr>\n";  
  
$consulta = 'select * from registros'; $resultado = mysql_query($consulta,$conexao);  
$linha=0;  
  
while ($linha < mysql_num_rows($resultado)) {  
$campo=mysql_fetch_array($resultado);  
echo("<tr><td>$linha</td><td>$campo[0]</td></tr>"); $linha++;  
}  
?>
```

Salve este arquivo como *server.php* e transfira-o para a página, clicando na opção File Manager no painel de controle.

O próximo passo consiste em criar a tabela no banco de dados recém criado. Para isso, volte ao painel de controle e clique na opção *phpMyAdmin* como mostra a próxima figura.



Ao entrar nesta janela, clique na opção *Enter phpMyAdmin*. Uma nova janela é aberta, como a apresentada na figura abaixo.

The screenshot shows the phpMyAdmin interface on a web browser. The URL in the address bar is `sql3.000webhost.com/phpMyAdmin/index.php?db=a9301050_dados`. The main title bar says "Server: localhost Database: a9301050_dados". Below the title bar, there are several tabs: Structure, SQL, Search, Query, Export, Import, and Operations. The Import tab is currently selected. A message below the tabs says "No tables found in database." On the left sidebar, there is a logo for "phpMyAdmin" and a link to "a9301050_dados (0)". The main content area has a large "Create new table on database a9301050_dados" button. Below it, there are two input fields: "Name:" and "Number of fields:". A "Go" button is located at the bottom of the creation form. At the bottom of the page, there is a link "Open new phpMyAdmin window". The background features a watermark of a sailboat.

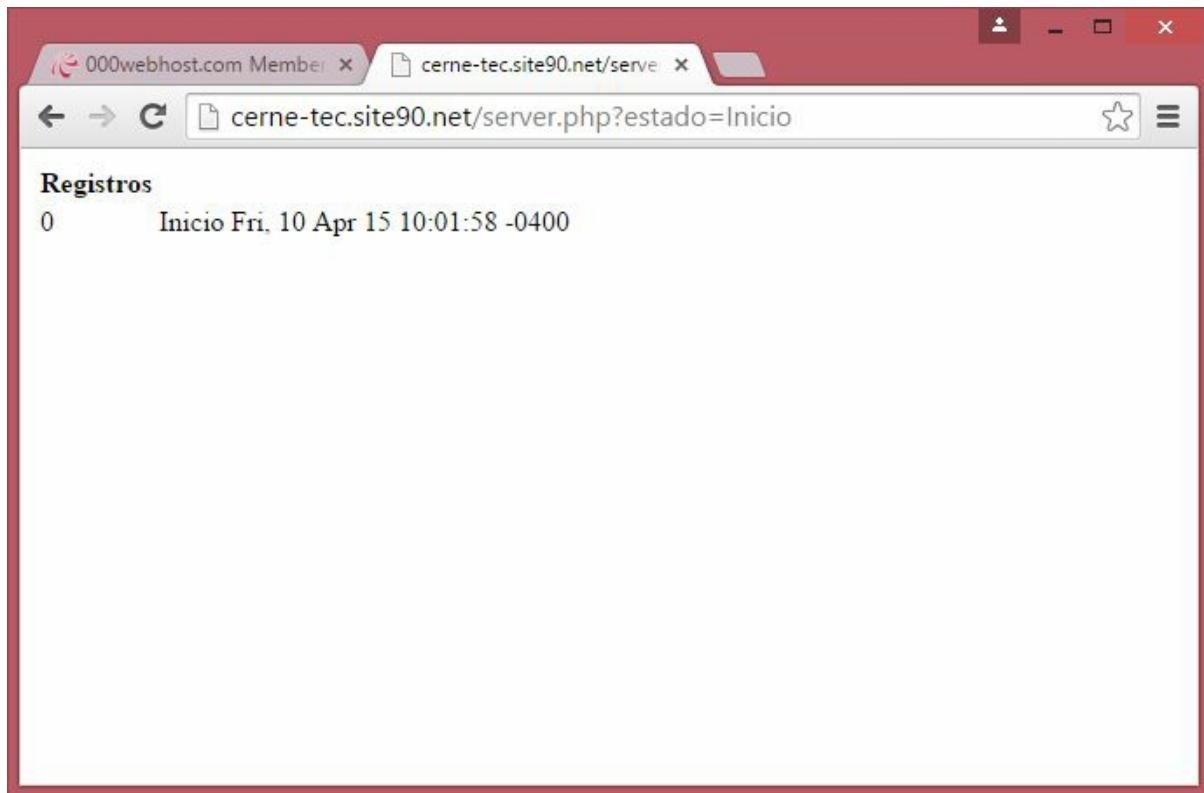
No campo hachurado acima informe no campo *Name* o valor registros e em *Number os fields* coloque 1. Em seguida, clique no botão *Go*. A janela fica como a figura abaixo.

The screenshot shows the phpMyAdmin interface on a web browser. The URL in the address bar is `sql3.000webhost.com/phpMyAdmin/index.php?db=a9301050_dados`. The main title bar says "Server: localhost Database: a9301050_dados Table: regi...". Below the title bar, there are several tabs: Structure, SQL, Search, Query, Export, Import, and Operations. The Structure tab is currently selected. The main content area shows a table structure with four columns: Field, Type, Length/Values¹, and Collation. The first row has empty fields for all columns. Below the table, there are sections for "Table comments:" and "Storage Engine:", both of which have empty input fields. At the bottom of the structure form, there are buttons for "Save", "Or Add 1 field(s)", and "Go". A yellow callout box contains a note: "1 If field type is "enum" or "set", please enter the values using this format: 'a','b','c'... If you ever need to put a backslash ("\\") or a single quote ('') amongst those values, precede it with a backslash (for example '\\xyz' or 'a\\b').". The left sidebar shows the "a9301050_dados (0)" database.

Preencha no campo *Field* a string *evento* e no campo *Length/Values* o valor 100. Em seguida, pressione *Save* para encerrar a criação da tabela.

A partir deste ponto já é possível testar a aplicação na Web, como mostra os exemplos abaixo.

Obs: Atualize o endereço da sua URL de acordo o que você definiu no servidor de hospedagem, pois no exemplo abaixo o endereço é cerne-tec.site90.net.



Capítulo VII Conhecendo o MIT App Inventor

1. Criando um projeto

O App Inventor for Android, também chamado de AIA foi desenvolvido inicialmente pelo Google, no entanto, no momento é mantido pelo MIT (Massachusetts Institute of Technology). Com esta ferramenta é possível criar aplicações para o Sistema Operacional Android, pois a mesma possui uma interface gráfica de programação que faz uso de blocos, facilitando assim o contato com a ferramenta. O AIA é uma WebApp em que a programação é feita diretamente em um Browser como o Google Chrome. Para ter acesso ao WebApp, visite o site do projeto como citado a seguir.

<http://appinventor.mit.edu/>

Em seguida, basta clicar no botão Create (no topo direito da tela). Neste momento, a plataforma pede para fazer login, no qual esta será feita através de uma conta do Google (como o Gmail). Em seguida, a tela ficará da forma apresentada abaixo, onde inicialmente é solicitado o nome inicial do novo projeto.

1
2 3 4 5

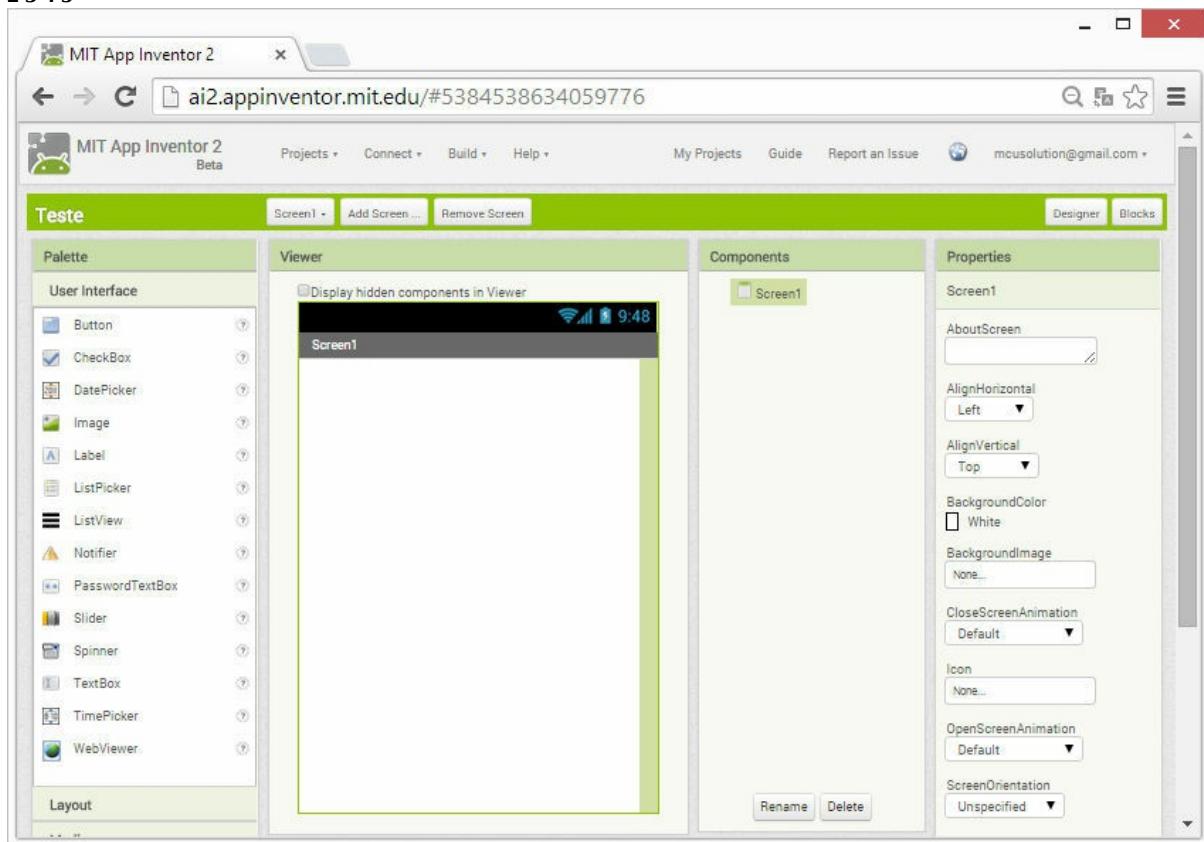


Figura 1.1 – Tela do MIT App Inventor

Na parte 1 está presente o menu, com as funções necessárias para o funcionamento do aplicativo, como a criação de novos projetos, compilação, ajuda etc. O campo 2 agrega os componentes disponíveis no App Inventor, como botões, labels, checkpoint, dentre outros. O campo 3 mostra a tela que o aplicativo apresenta no

decorrer do seu desenvolvimento. O campo 4 mostra todas as telas e os componentes associados a cada uma delas. Finalmente, o campo 5 é a janela de propriedades, no qual é possível alterar as propriedades inerentes de cada componente, como cor, texto, tamanho dentre outros.

Além destes campos principais há como escolher entre o modo *Designer*, onde o foco será a organização da tela e o modo *Block* para programar a lógica de funcionamento do aplicativo. Ambos os modos podem ser escolhidos através dos dois botões marcados a seguir.

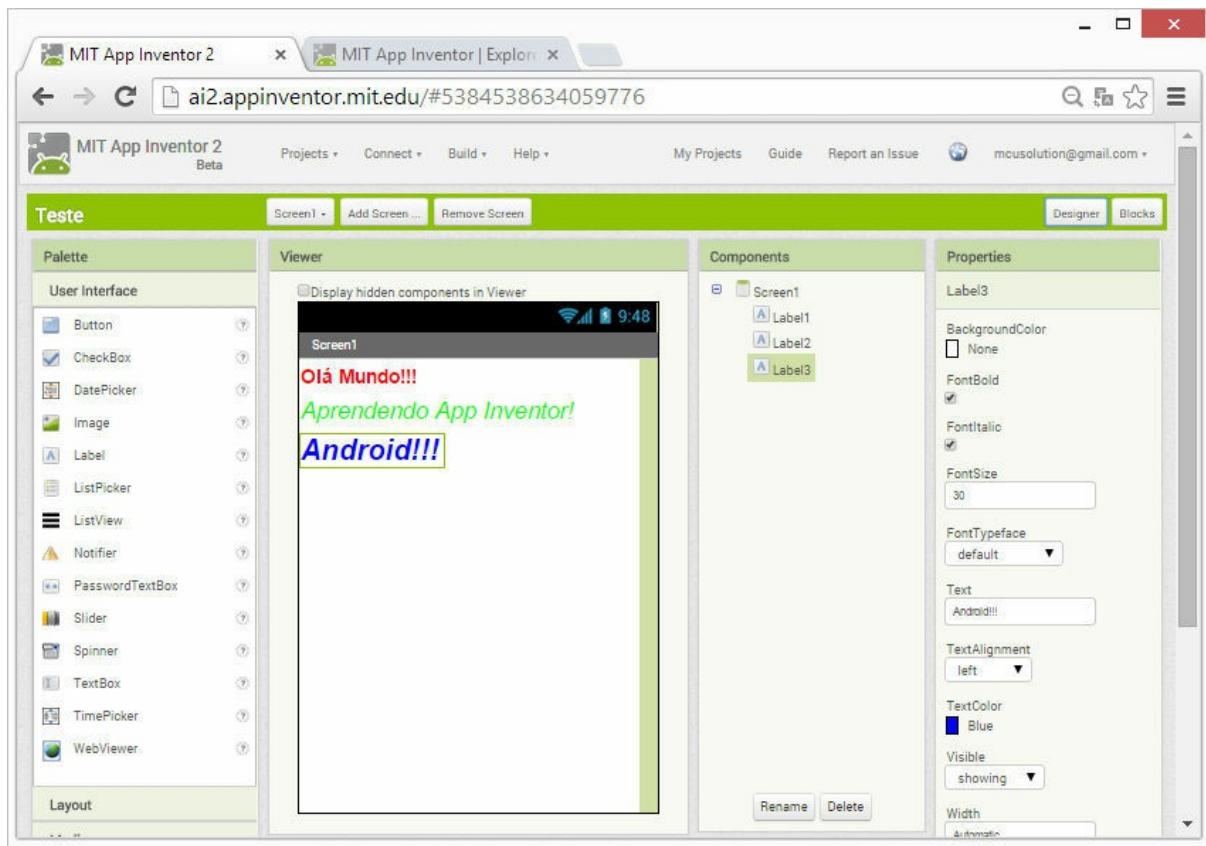


Figura 1.2 – Escolhendo o modo Designer ou Block

Para testar o programa desenvolvido há várias maneiras, sendo uma delas através de um emulador que vem disponível na ferramenta, como o apresentado na figura a seguir.

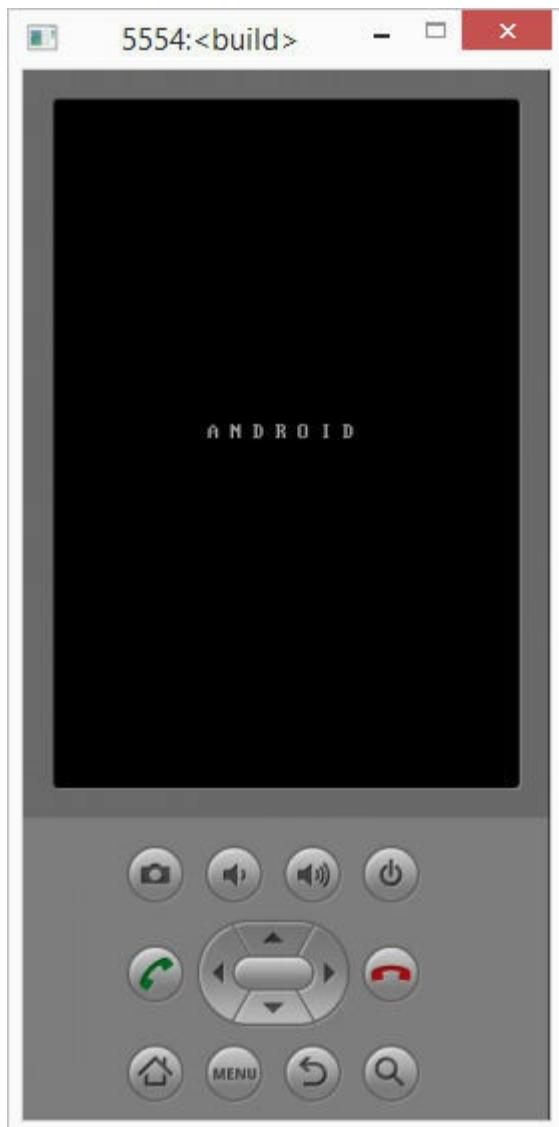


Figura 1.3 – Tela do emulador

É necessário instalar o programa aiStarter para habilitar o emulador tanto de software quanto de hardware. Tal software pode ser baixado através do link a seguir.

<http://appinventor.mit.edu/explore/ai2/setup-emulator.html> A emulação por software é acessível através do menu *Connect > Emulator*. Para emular através do smartphone/tablet na porta USB, vá até a opção menu *Connect -> USB*.

A opção adotada nesta literatura será a opção número 1, indicada pelo App Inventor que pode ser resumida segundo a figura a seguir.



Figura 1.4 – Modo de teste adotado

Fonte: <http://appinventor.mit.edu>

Neste modo de simulação o aplicativo será desenvolvido normalmente através da plataforma Web, no entanto, ao finalizar este passo deve-se ir ao menu *Build -> App (provide QR code for .apk)*, onde ao término do processo de compilação, será fornecido um QR code que deve ser lido pelo smartphone/tablet através do MIT AI2 Companion, para que o dispositivo possa acessar a URL fornecida e assim executar o aplicativo desenvolvido. A figura a seguir mostra o resultado com sucesso da compilação, onde ao término o QR code é apresentado para que possa ser feita a leitura através do aplicativo MIT AI2 Companion instalado no smartphone/tablet.

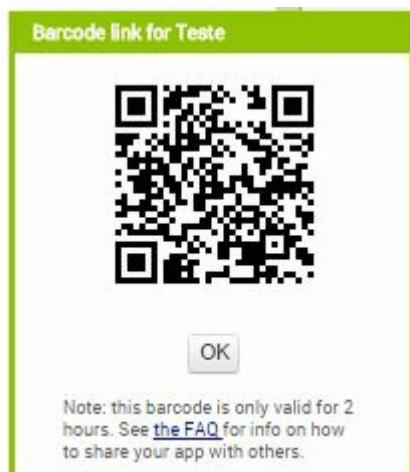


Figura 1.5 – QR code da aplicação

A URL informada pode ser compartilhada com outras pessoas, para que estas façam o acesso à aplicação e testem-a. Também é possível gerar o arquivo .apk e realizar o download para o computador, de modo a compartilhá-lo ou disponibilizá-lo no Google Play. Para isso, acesse o menu *Build -> App (save .apk to my computer)*.

O MIT AI2 Companion pode ser baixado através do link a seguir.

<http://appinventor.mit.edu/explore/ai2/setup-device-wifi.html>

Capítulo V Empréstimo de Aplicativos

1. Introdução

Baseado na metodologia apresentada, nesta parte da literatura o objetivo é apresentar diversos

exemplos que mostrem os recursos disponíveis no MIT App Inventor para programação em smartphones/tablets.

2. Label

Um *Label* é uma etiqueta, no qual pode-se visualizar informações sobre a aplicação, ou seja, ela indica o que determinada caixa de texto, seleção, dentre outras opções podem fazer. Para adicionar um *Label* a aplicação, vá ao campo de componentes (campo 2) *Palette* -> *User Interface* -> *Label* e mova este componente para a tela do dispositivo, como mostra a figura abaixo.

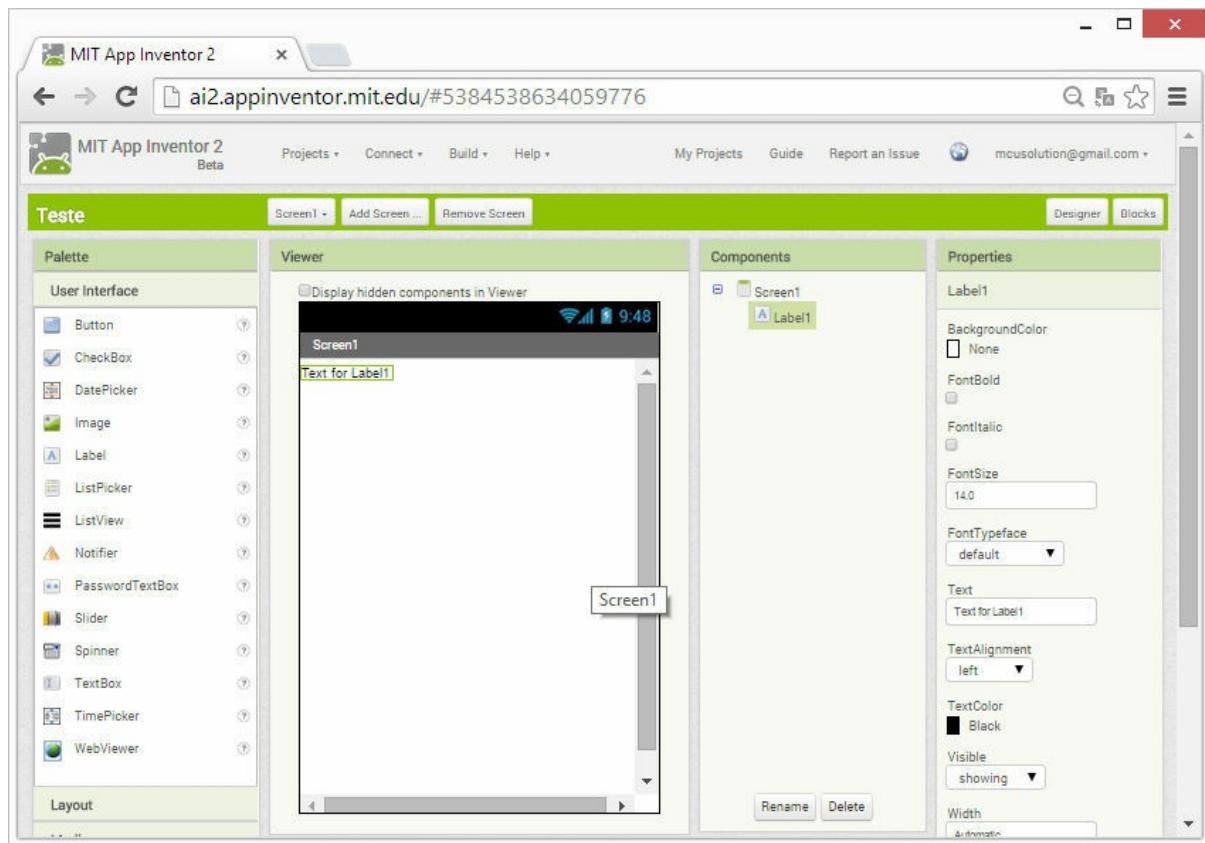


Figura 8.1 – Adicionando label à tela

Selecione o componente e em seguida posicione-o na tela do dispositivo, como mostra a próxima figura. Verifique que cada componente adicionado a tela será atualizado na árvore de componentes (campo 4).

Para alterar o texto (*string*) apresentado, selecione o Label adicionado. Em seguida, vá ao campo de propriedades (campo 5) e altere a propriedade *Text*, deixando-a com o valor “Olá Mundo!!!”. O resultado será o da tela a seguir.

Figura 8.2 – Resultado após modificação Adicione outros *Labels* à tela do dispositivo. A próxima figura apresenta um exemplo.



Figura 8.3 – Adicionando novas caixas de texto Para alterar as propriedades do componente como cor, tamanho, alinhamento, etc vá até a janela de propriedades. A próxima figura mostra a mesma aplicação acima com as propriedades modificadas.

Figura 8.4 – Alterando as propriedades do aplicativo Para compilar este projeto, vá ao menu *Build -> App (provide QR code for .apk)* e obtenha o código para acessar através da internet a aplicação do smartphone/tablet.

3. Button

Através de um botão (*Button*) é possível interagir com o dispositivo e neste exemplo, será visto como habilitar o seu funcionamento no App Inventor. Para isso, crie um novo projeto e deixe a tela da forma apresentada a seguir, onde o *Button* está localizado na mesma paleta de componentes do *Label*.

Figura 8.5 – Modelo do projeto

O objetivo deste exemplo é fornecer o nome da capital de acordo com o País selecionado. Inicialmente, somente há o Brasil. Para isso, é necessário alterar o nome (*name*) do componente Label assim como o seu campo *Text*, deixando ambos configurados para Resposta. Observe que na parte inferior do campo *Components* (campo 4) há um botão chamado *Rename*. Clique neste botão e coloque na opção *New Name* o valor resposta. Modifique as propriedades *Name* e *Text* do botão para Brasil.

Altere para o modo *Blocks*. A tela ficará conforme a figura a seguir.

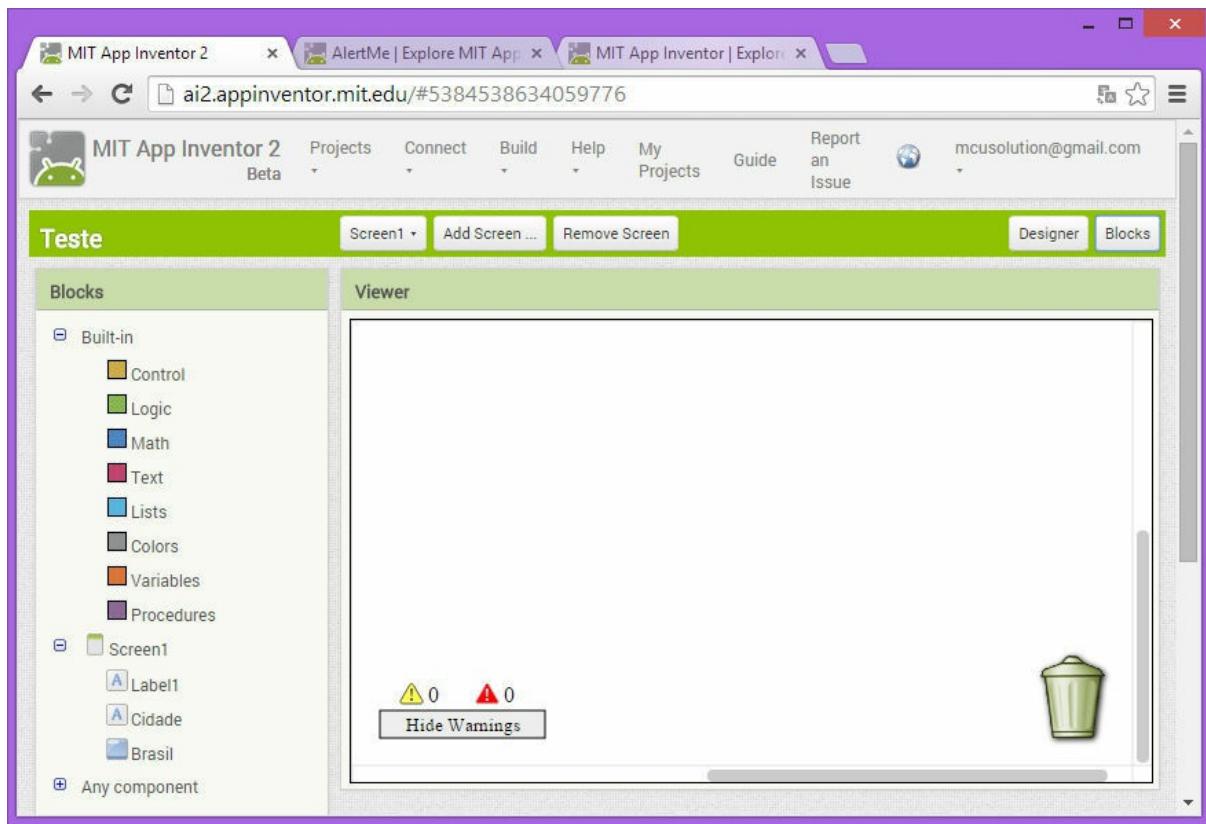


Figura 8.6 – Modo blocks

Observe na parte hachurada da figura anterior todos os componentes utilizados na aplicação, neste caso o botão e os dois labels. Clique no componente botão chamado Brasil, a tela ficará conforme a figura abaixo.

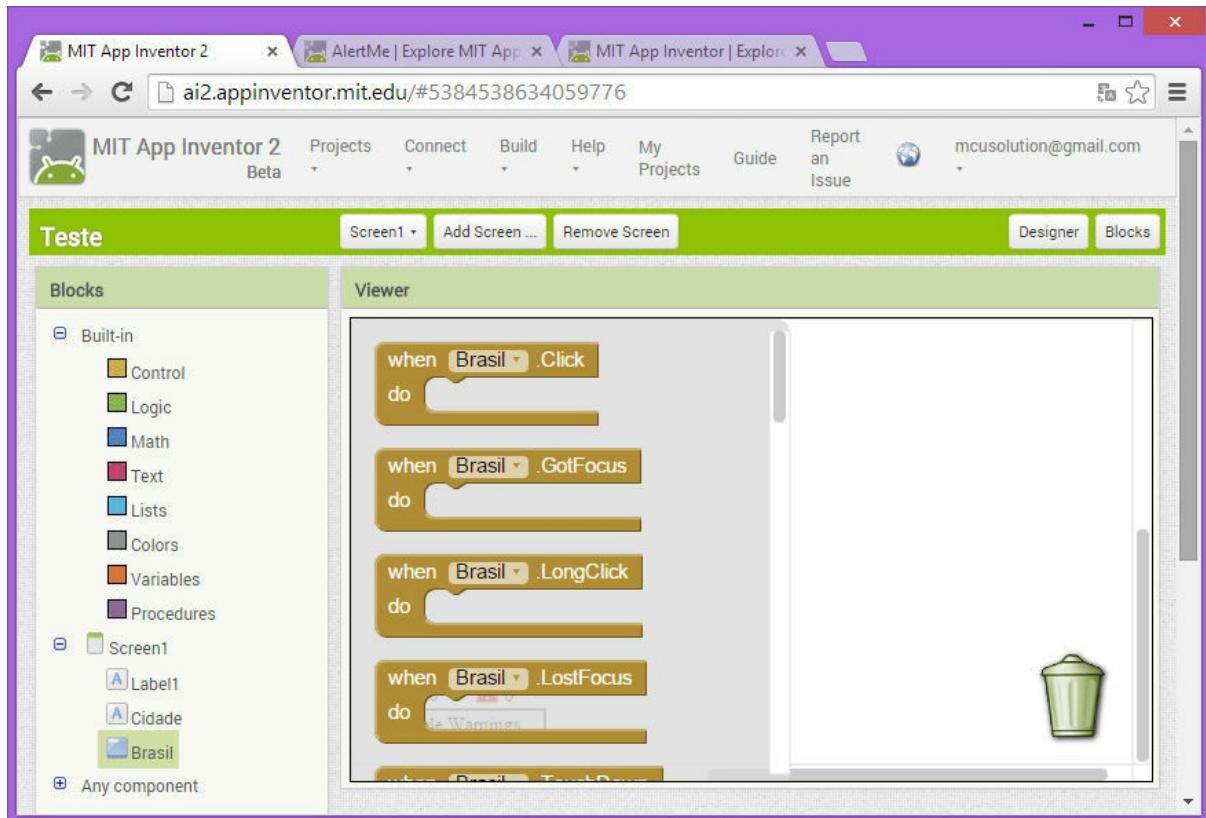


Figura 8.7 – Verificando os eventos do componente button

Verifique que há vários eventos associados ao botão. Por enquanto, o interesse está no evento *Click*, que será processado sempre que o botão for pressionado. Neste caso, o objetivo será mostrar na caixa de texto Resposta o valor “Brasília”. Sendo assim, move para a área de trabalho a primeira opção da listagem de eventos apresentada. O resultado será conforme a figura a seguir.

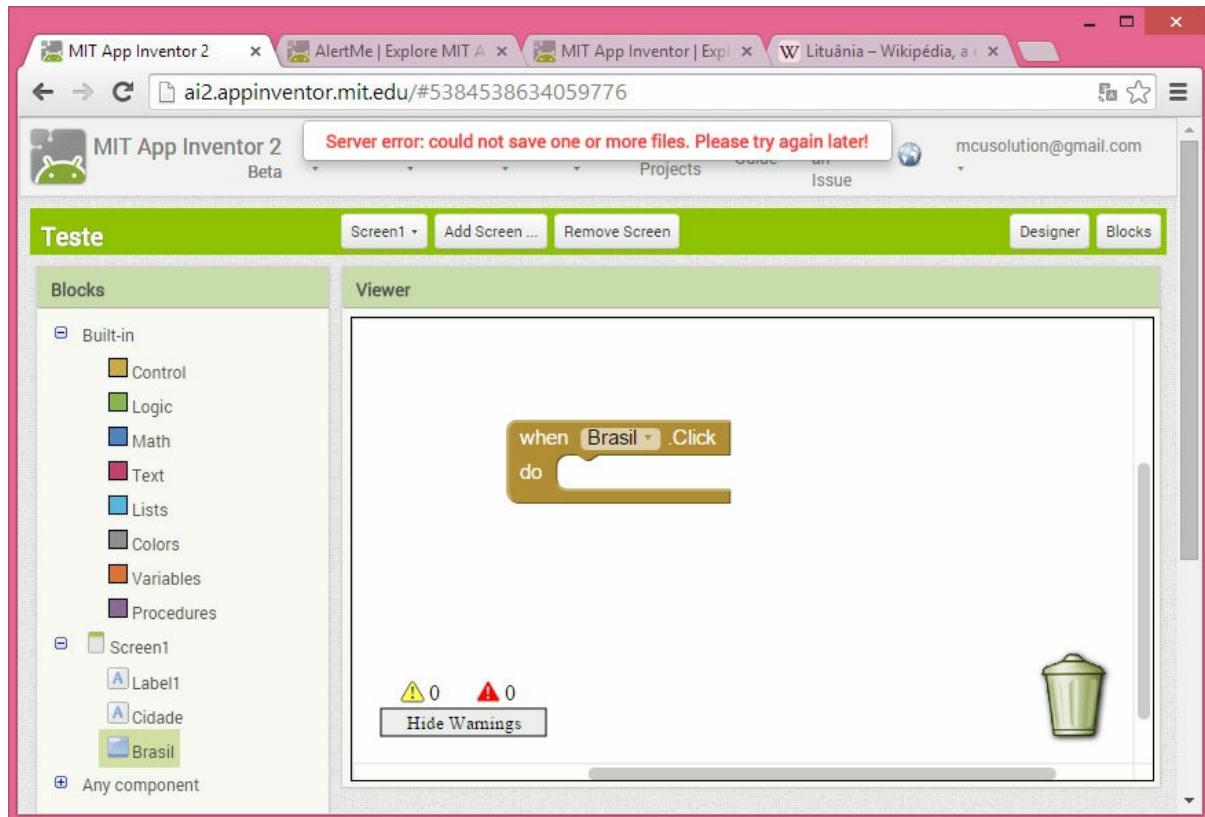


Figura 8.8 – Adicionando um evento ao programa Verifique que o evento é *when Brasil.Click do*, ou seja, quando o botão Brasil for pressionado deve executar uma ação, onde esta deve ser colocada dentro do bloco de ação.

Clique no componente *Label* Cidade e move para a área de trabalho o procedimento apresentado na figura a seguir.



Figura 8.9 – Movendo a ação do Label Cidade para a área de trabalho

Observe que há dois parâmetros configuráveis, sendo o primeiro o *Label* em si e o segundo a propriedade. Clique no segundo *ComboBox* e escolha a opção *Text*, que permitirá alterar o valor do texto do componente apresentado.

Vá até a aba *Built-in->Text*, localizada logo acima dos componentes e move o componente da imagem a seguir para a área de trabalho.



Figura 8.10 – Componente Text

Este componente permite definir o texto a ser apresentado no *Label Cidade*. Coloque entre as aspas o texto Brasília. O resultado final da lógica está apresentado na figura abaixo.



Figura 8.11 – Lógica implementada para o exemplo O bloco lógico implementado pode ser interpretado para carregar no campo de texto do componente Cidade a string “Brasília” quando o botão Brasil for pressionado (*Click*). Faça a compilação deste projeto e simule-o no smartphone/tablet.

A próxima janela mostra o resultado de uma simulação.



Figura 8.12 – Resultado da simulação

A fim de complementar o projeto, adicione mais botões com os países listados na tabela abaixo e renomeie os botões conforme os nomes sugeridos.

País Capital Name Uruguai Montevidéu Uruguai Angola Luanda Angola

China Pequim China EUA Washington EUA Portugal Lisboa Portugal

Tabela 8.1 – Listagem de países, capitais e names sugeridos

A tela do programa ficará conforme mostra a figura a seguir.



Figura 8.13 – Adicionando mais países

O bloco lógico terá a configuração final ilustrada na próxima figura.

```
when Brasil .Click
do set Cidade . Text to "Brasília"

when Uruguai .Click
do set Cidade . Text to "Montevidéu"

when Angola .Click
do set Cidade . Text to "Luanda"

when China .Click
do set Cidade . Text to "Pequim"

when EUA .Click
do set Cidade . Text to "Washington"

when Portugal .Click
do set Cidade . Text to "Lisboa"
```

Figura 8.14 – Resultado final do bloco

lógico 4. CheckBox

O *CheckBox* permite ao usuário do dispositivo escolher uma opção dentro de uma lista disponível na tela. Este componente é muito utilizado, principalmente quando uma dentre várias opções precisam ser apresentadas e selecionadas.

Neste tópico será desenvolvida uma aplicação voltada para um restaurante, no qual são apresentadas as opções de cardápio e após a seleção, o valor da refeição é apresentado. O componente *CheckBox* está na paleta de componentes, logo abaixo do *Button*.

Crie a aplicação e deixe-a com a interface sugerida a seguir.



Figura 8.15 – Tela do programa exemplo

Renomeie cada um dos componentes conforme sugestão a seguir.

Componente Lasanha

Panqueca Bacalhau

Empadão Sopa de ervilha Resposta

Valor Nome R\$ 15,00 lasanha R\$ 10,00 panqueca R\$ 20,00 bacalhau R\$ 12,00 empadão

R\$ 5,00 sopa

- resposta **Tabela 8.2 – Preços e nomes sugeridos aos componentes**

Desta forma, ao selecionar uma das opções o programa irá atualizar a caixa de resposta com o valor apresentado acima. Sendo assim vá ao bloco lógico do App Inventor e deixe-o como apresentado na ilustração a seguir.



Figura 8.16 – Bloco lógico do experimento

Observe que foi utilizado um evento do componente *CheckBox* chamado *Changed*, onde ao mudar o seu estado o bloco é executado, permitindo assim a alteração do valor na caixa de texto resposta. A figura a seguir mostra um resultado da simulação.

Outra aplicação que pode ser aplicação voltada para um parque desenvolvida é para uma de diversões, no qual são apresentadas as opções de entretenimento disponíveis e o custo unitário para quando o botão ok for pressionado, seja apresentado o valor total para o ingresso ao parque.

Crie uma nova aplicação e deixe-a com a interface e *names* sugeridos a seguir.



```
id=roda_gigante id=montanha_russa id=pula_pula id=trenzinho id=labamba id=barco_pirata id=carro_eletrico  
id=tiro_ao_alvo
```

Figura 8.17 – Tela do programa exemplo

Este exemplo faz o uso de variáveis, neste caso apenas uma para armazenar o valor total do ingresso. A variável neste exemplo é identificada como total e é inicializada em 0, como mostra o bloco lógico a seguir.

```
initialize global [total] to [0]
```

Figura 8.18 – Declarando e inicializando a variável total em 0

Em seguida, há um bloco para cada botão que ao ser acionado através do evento *Changed* verifica se o botão está marcado pela declaração *se (if)* e caso esteja, soma o conteúdo da variável total com o respectivo valor do ingresso, como mostra o bloco referente a roda gigante mostrado a seguir.

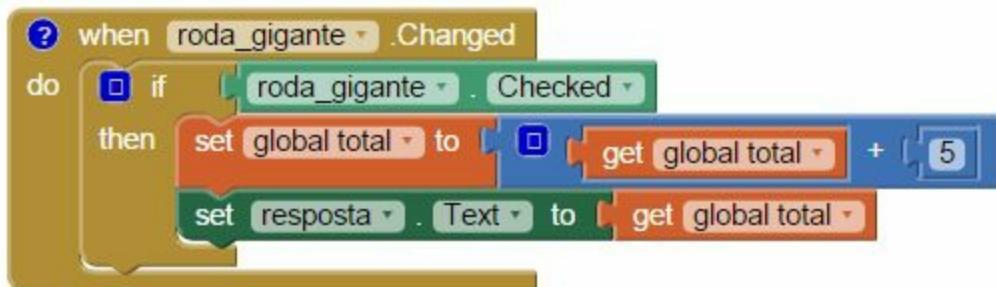


Figura 8.19 –

Bloco referente ao tratamento da opção roda gigante

A próxima figura ilustra todos os blocos lógicos referentes aos checkboxes utilizados no experimento.

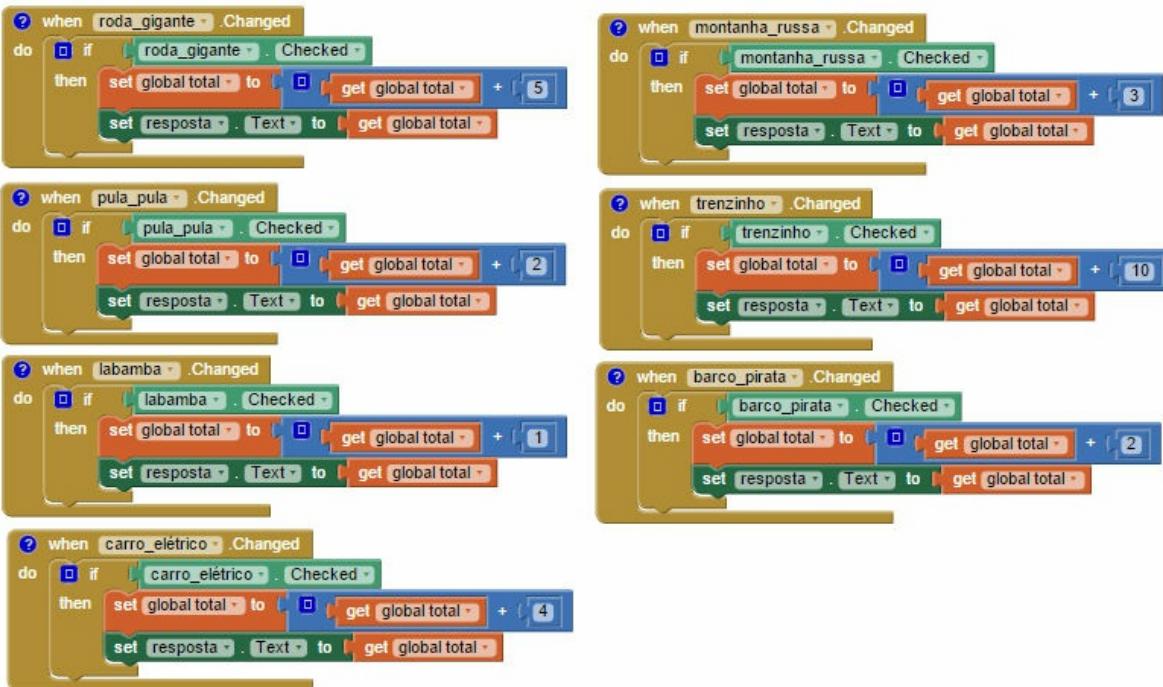


Figura 8.20 – Bloco lógico referente a todas as opções

A próxima figura mostra um resultado de simulação baseado neste experimento.



Parque de Diversões

Roda gigante R\$ 5,00

Montanha russa R\$ 3,00

Pula pula R\$ 2,00

Trenzinho R\$ 10,00

Labamba R\$ 1,00

Barco pirata R\$ 2,00

Carro elétrico R\$ 4,00

20

Figura 8.21 – Resultado de uma simulação feita no

programa

5. Image

Através do *Image* é possível mostrar imagens em uma aplicação no Android. Neste tópico será desenvolvida uma aplicação voltada para o campo turístico, onde será apresentada uma lista de três cidades e ao clicar sobre a mesma, será apresentado uma foto sobre a cidade avaliada. Primeiramente, baixe uma imagem referente às cidades de Florianópolis-SC, Domingos Martins – ES e Petrópolis-RJ e salve com a extensão .jpg com os nomes florianopolis.jpg, domingos_martins.jpg e petropolis.jpg (as imagens não podem ser acentuadas). Crie uma nova aplicação e deixe-a com a interface e *names* sugeridos a seguir.

id=cidade id=Florianópolis id=DomingosMartins id=Petrópolis

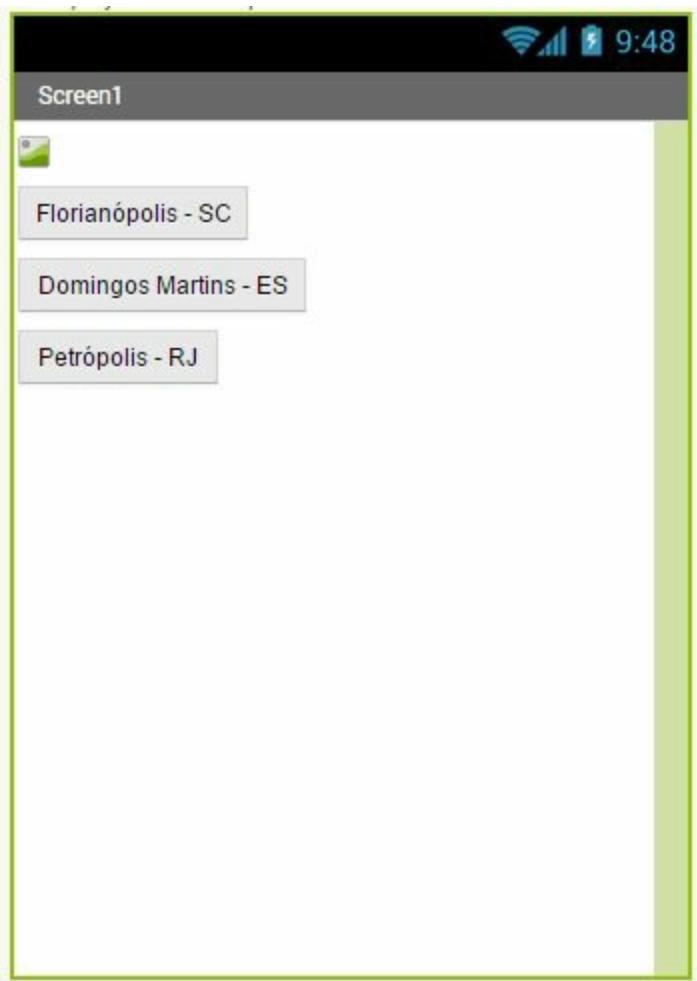


Figura 8.22 – Tela do exemplo proposto

Clique no componente *image* e vá à seção propriedades e na seção *pictures*, transfira as três imagens comentadas anteriormente. Vá ao bloco lógico e deixe-o conforme a figura abaixo.

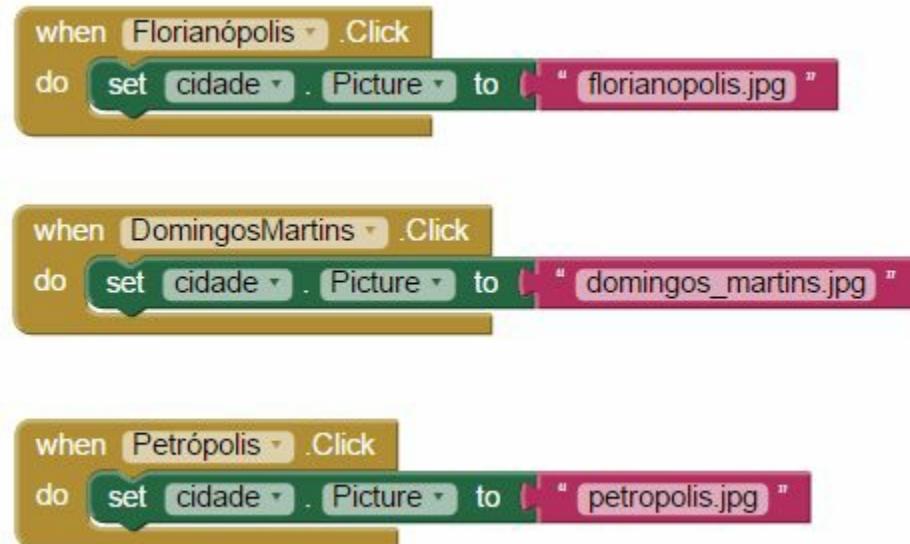


imagem de acordo com o botão pressionado

Figura 8.23 – Nova

O bloco lógico apresentado altera a imagem de acordo com o botão selecionado, carregando através do método *Picture* à mesma.

A próxima imagem mostra o resultado de uma simulação do programa apresentado.

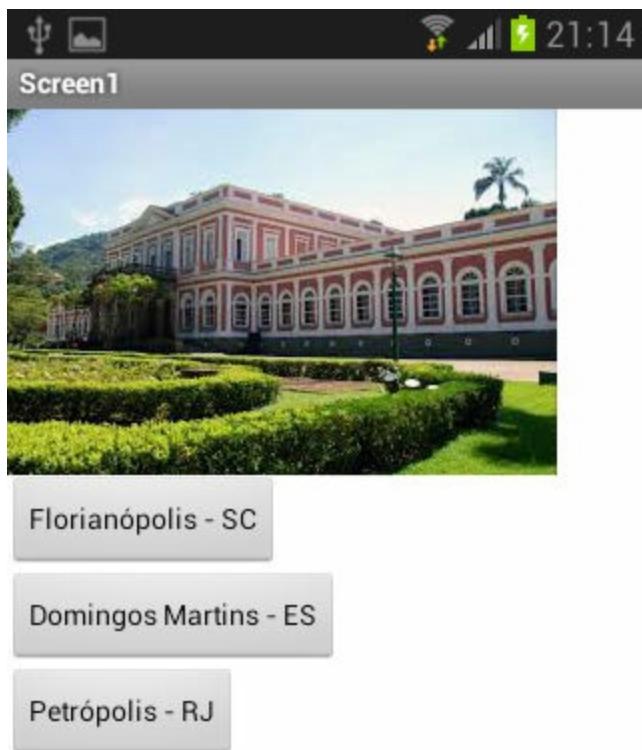


Figura 8.24 – Simulação do programa desenvolvido

6. Slider

O *slider* possibilita variar um valor de um mínimo a um máximo e este exemplo pretende demonstrar o seu uso, carregando em um *label* o valor presente no mesmo de acordo com a alteração de valor dentro da faixa de mínimo (*MinValue*) e máximo (*.MaxValue*). Para tal demonstração, deixe à tela similar a sugerida abaixo.

id= resposta

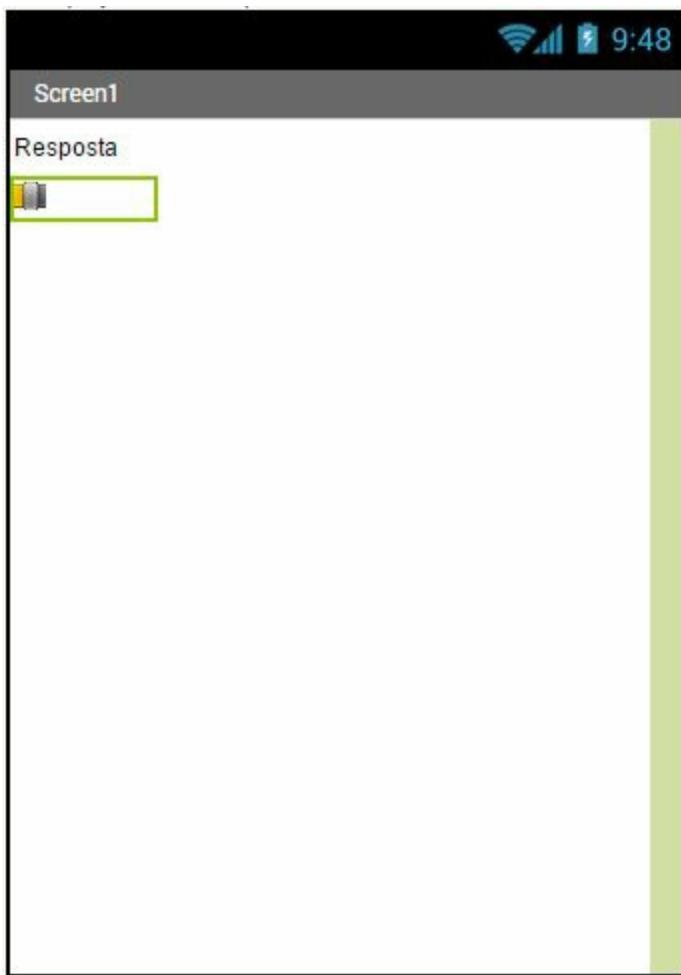


Figura 8.25 – Tela do programa para testes com o Slider O exemplo consistirá em carregar no *label* Resposta o valor ajustado no *slider* sempre que este for atualizado. Sendo assim, deixe o bloco lógico conforme a figura abaixo.

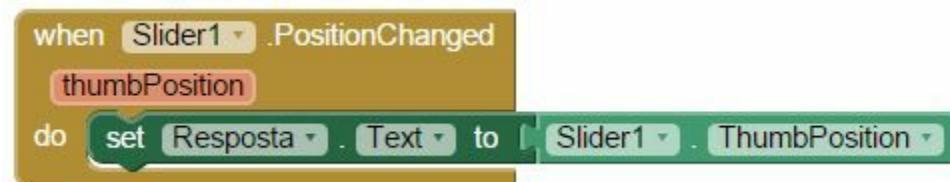


Figura 8.26 – Montagem do bloco lógico

Observe que o bloco lógico carrega na caixa de texto Resposta o valor atual do *slider* toda a vez que o conteúdo presente no mesmo for alterado.

7. Notifier

O *Notifier* é uma mensagem que pode ser apresentada como um alerta ao usuário do sistema Android. Para utilizá-lo, faz-se necessário do componente de mesmo nome disponível na paleta de componentes. Um exemplo para demonstrar o seu uso consistirá em dispor de 3 botões que mostram a faixa de horário de 0h as 12h, 12h as 18h e 18h as 0h. Quando um destes botões for pressionado, uma mensagem de saudação será impressa, neste caso “Bom dia”, “Boa tarde” e “Boa noite”, respectivamente. Para isso, monte um novo aplicativo e deixe-o conforme mostra a próxima figura.

id=BomDia id=BoaTarde id=BoaNoite



Figura 8.27 – Tela do experimento

Programe o bloco lógico conforme ilustra a próxima figura, onde a mensagem será apresentada sempre que um dos botões for pressionado.

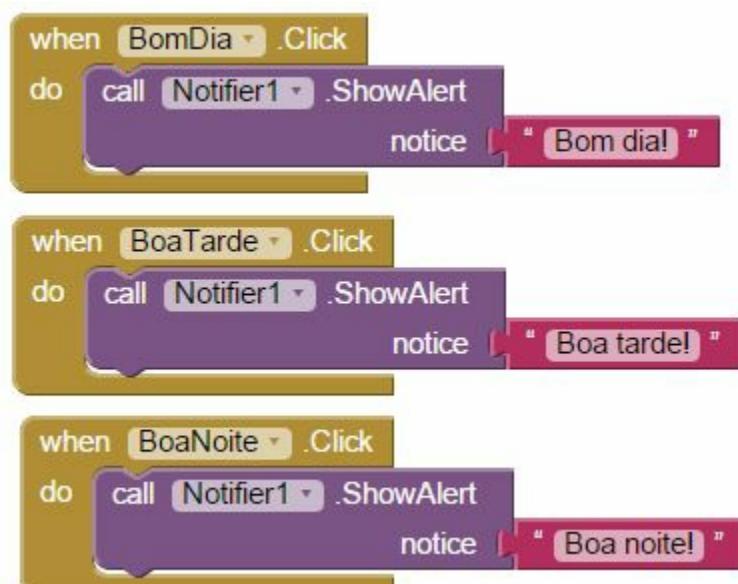
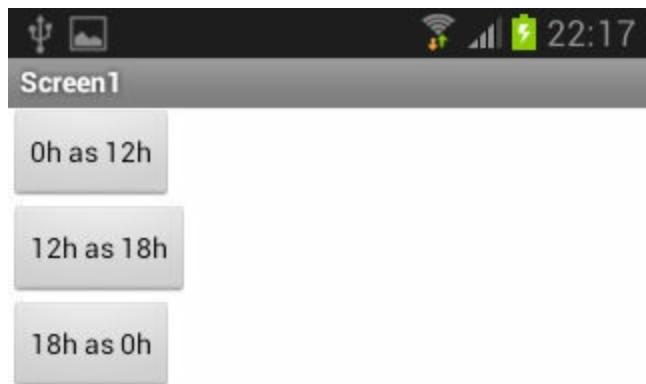


Figura 8.28 – Bloco lógico do experimento
A próxima janela mostra o resultado de uma simulação.



Boa tarde!

Figura 8.29 – Resultado de uma simulação

8. TextBox

O *TextBox* é a principal fonte de dados de entrada para a aplicação, onde o usuário do sistema poderá digitar e fornecer um conjunto de caracteres ou *string* assim como um valor numérico. Para demonstrar o uso deste recurso é proposta a criação de uma pequena calculadora que soma o conteúdo de duas caixas de texto e atribui a uma terceira caixa o resultado da operação. Sendo assim, deixe a tela e os *names* conforme sugestão abaixo.

```
id=A  
id=B  
id=Resposta
```

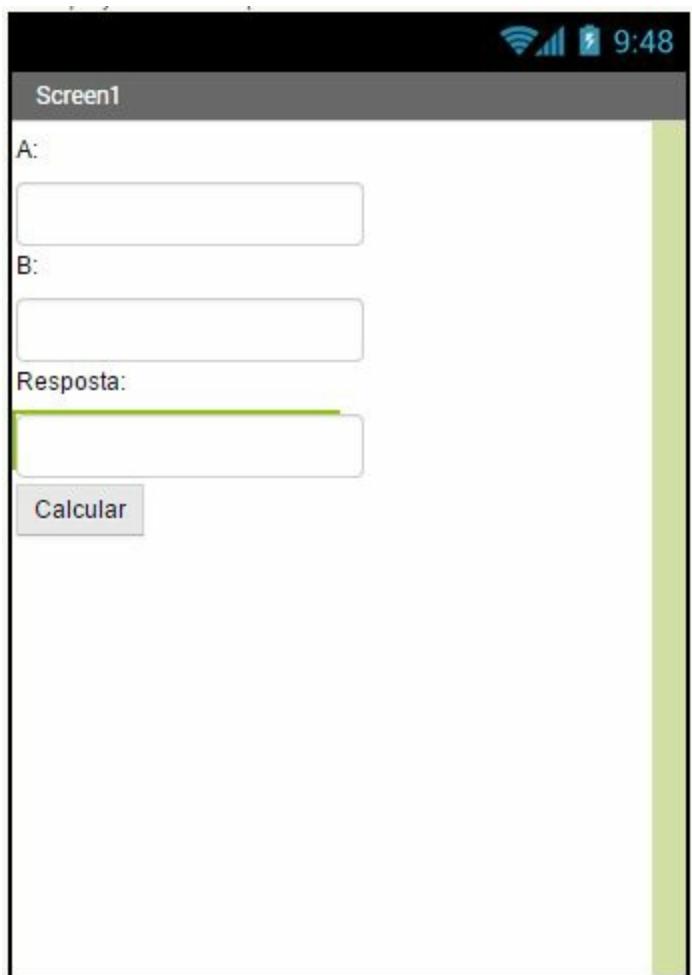


Figura 8.30 – Tela proposta

Habilite a propriedade *NumbersOnly* de cada *TextBox*. Em seguida, programe o bloco lógico como descrito a seguir.

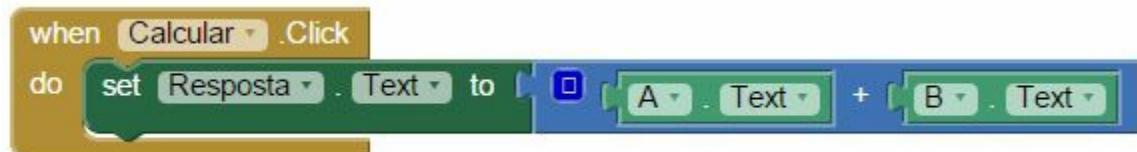


Figura 8.31 – Bloco lógico do experimento

Observe que uma operação matemática de soma é feita através do conteúdo da caixa de texto A e B, onde o resultado é atribuído a caixa de texto Resposta. As operações matemáticas ficam disponíveis na seção *Built-in -> Math*. A próxima figura mostra o resultado de uma simulação.



Figura 8.32 – Simulação da calculadora

Capítulo X Pode o noApp nven o

1. Introdução

Baseado nas ferramentas e exemplos explorados no capítulo passado, o foco deste passo será apresentar programas um pouco mais elaborados que forneçam um sentido mais amplo de aplicativo comercial feito no Android.

Uma das formas de disponibilizar de forma gratuita ou paga seus aplicativos é através do Google Play, cujo endereço está disponível no link abaixo.

https://play.google.com/store?hl=pt_BR

Após o cadastro no site é possível disponibilizar os aplicativos de forma gratuita ou paga. Neste caso, basta enviar o arquivo com a extensão .apk (*Android Package*) que seria o arquivo “executável” para o Android. Este arquivo pode ser gerado através do App Inventor através da opção *Build -> App (save .apk to my computer)*.

2. Calculadora

Este exemplo irá desenvolver uma calculadora, no qual implementará as 4 operações aritméticas. Há quatro botões, no qual permitem realizar a soma, subtração, produto e divisão entre os dois valores presentes em duas caixas de texto (*TextBox*) e apresentar o resultado.

Crie um novo projeto e deixe-o com a interface apresentada a seguir e com os *names* sugeridos.

A
B
Resposta

Soma
Subt
Mult
Divi

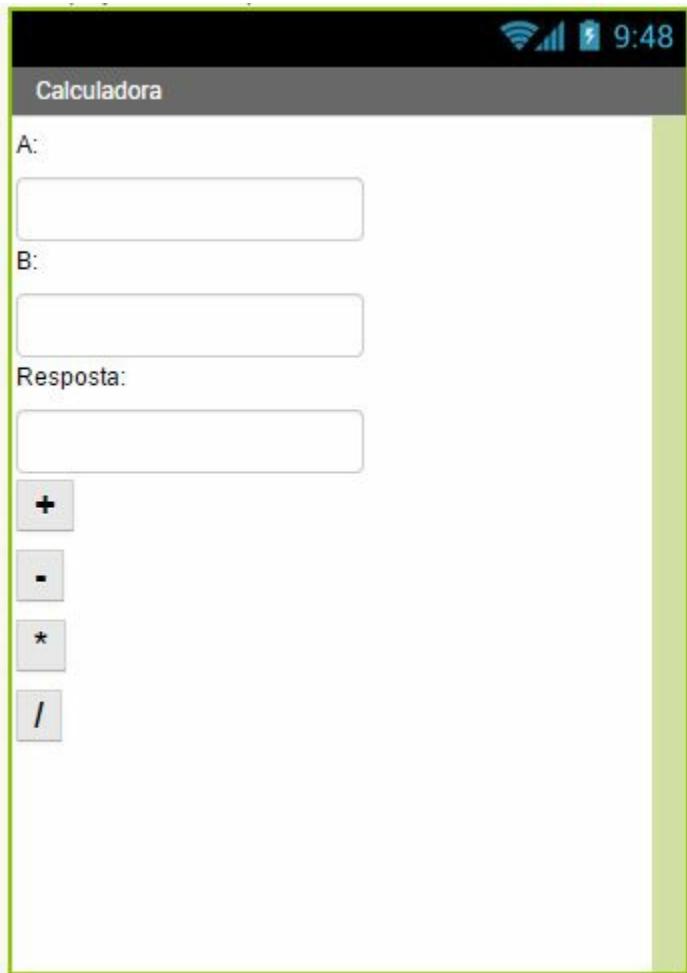


Figura 9.1 – Tela e nomes sugeridos

Programe o bloco lógico conforme sugestão abaixo, onde cada botão ao ser pressionado faz uma operação matemática distinta.

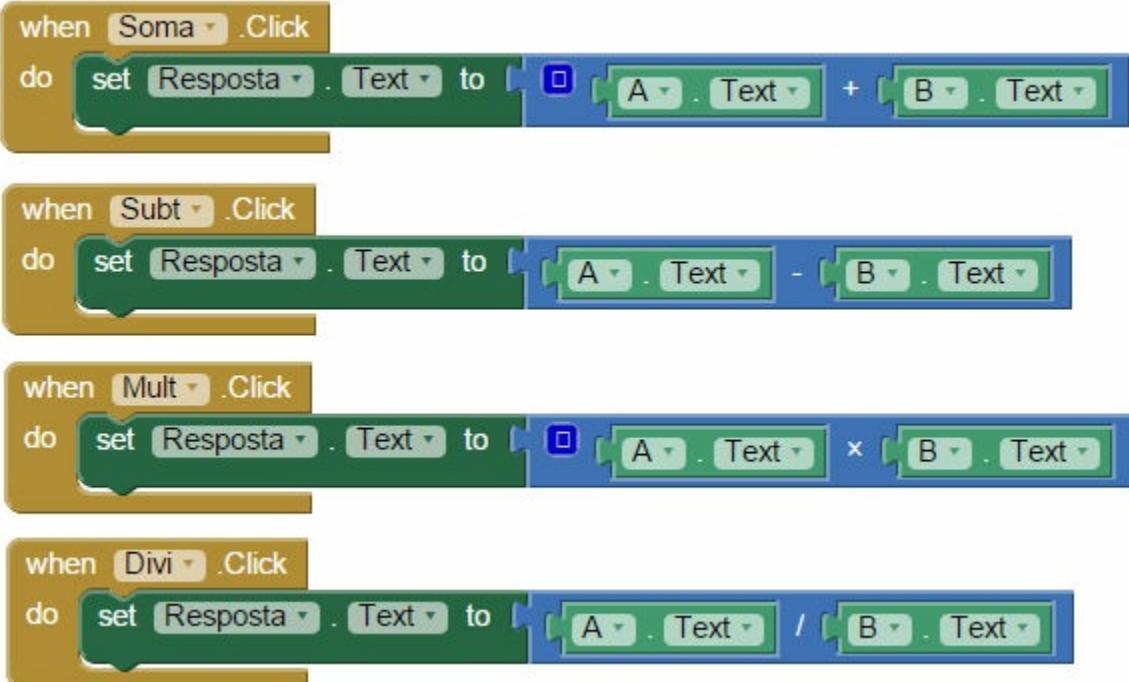


Figura 9.2 – Bloco lógico do experimento

A próxima tela mostra o resultado para uma soma feita através da calculadora.

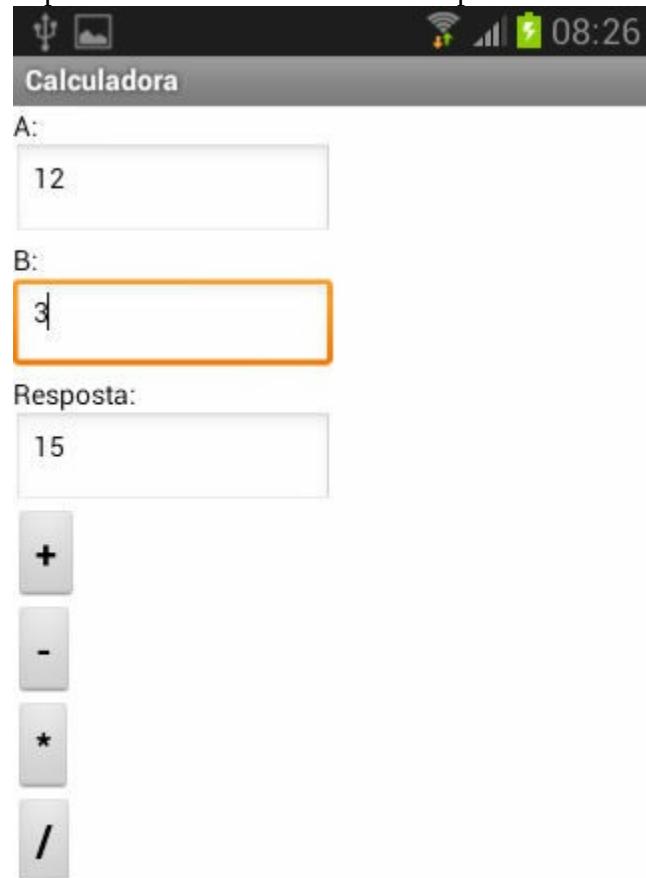


Figura 9.3 – Resultado de uma simulação

3. Dado eletrônico

Neste experimento ao pressionar o botão Ok, será apresentado um valor numérico aleatório

entre 0 e 10 utilizando para isso uma função randômica disponível no App Inventor. Deste modo, este projeto implementará um dado eletrônico. Acompanhe a seguir a interface que foi preparada.

id=dado



Figura 9.4 – Tela proposta para o experimento

Dica: Para aumentar ou diminuir o tamanho do TextBox, procure a propriedade *FontSize* na janela de propriedades do componente. A cor pode ser modificada através da propriedade *TextColor*.

A biblioteca *Math* disponível em *Built-in->Math*, fornece um meio eficaz de calcular um número randômico através do bloco apresentado a seguir.



Figura 9.5 – Cálculo de um valor randômico

Basta informar os valores mínimo e máximo para que este bloco retorne com um valor dentro da faixa informada. Neste experimento ao pressionar o botão Ok, será apresentado um valor numérico aleatório entre 1 e 6 utilizando para isso o bloco randômico apresentado. Desta forma, deixe o bloco lógico conforme apresentado na figura abaixo.



Figura 9.6 – Bloco lógico do exemplo

Observe que no momento que o botão Ok for pressionado, a caixa de texto dado será carregada com um valor aleatório (randômico) entre 1 e 6. A próxima figura mostra o resultado de uma simulação.



Figura 9.7 – Resultado da simulação 4. Contador

Um contador possui inúmeras utilidades seja na área industrial ou comercial, sendo uma delas contabilizar a quantidade de pessoas que estão em um ambiente ou a quantidade de produtos estocados. Neste experimento, será desenvolvido um programa que disponibilizará 3 botões, onde um deles incrementará o valor da contagem, o segundo decrementará e o terceiro fará o reset do contador. Desta forma, construa um aplicativo com a interface sugerida a seguir.

id=contador id=inc id=dec

id=limpar



Figura 9.8 – Tela do programa proposto O valor inicial do contador será zero e o botão limpar permite reinicializar o valor da contagem. O botão INC faz o incremento unitário da contagem enquanto que DEC faz o inverso. O primeiro passo para a construção deste aplicativo é declarar e inicializar a variável contagem em 0, como ilustra a figura a seguir.

initialize global cont to 0

Figura 9.9 – Declaração e inicialização da variável em 0

O bloco referente ao incremento, decremento e limpeza da variável está ilustrado a seguir, onde cada um dos blocos faz o tratamento pertinente a operação.

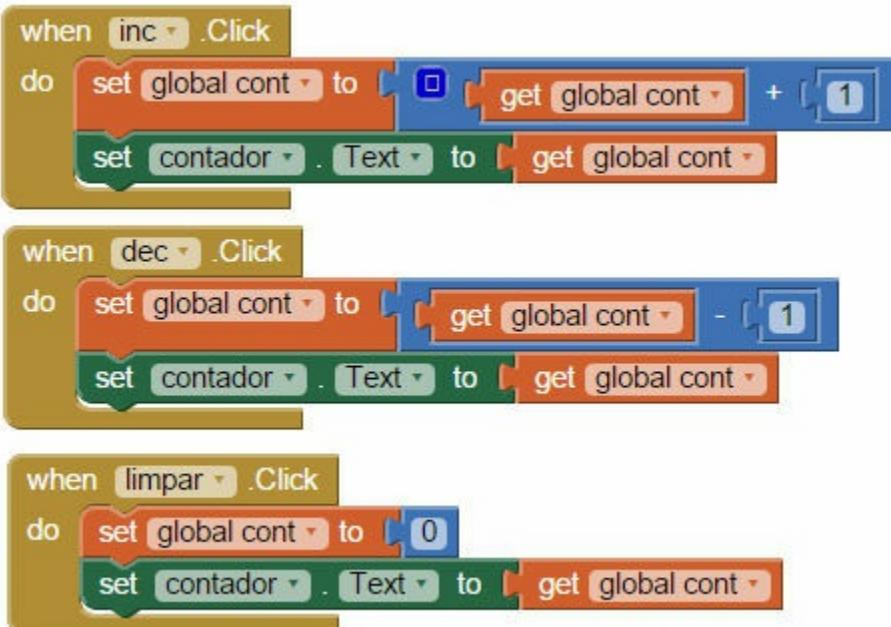


Figura 9.10 – Blocos

lógicos referentes ao incremento, decremento e limpeza

A próxima figura mostra o resultado de uma simulação, após o incremento da variável contagem.



Contador



Figura 9.11 – Resultado de uma simulação

5. Conversor Pa <-> PSI

Em engenharia ainda é muito comum trabalhar com unidades que não seguem o sistema

internacional (SI), sendo uma delas a unidade de pressão que no SI é dada pelo Pascal (Pa), no entanto no sistema Inglês que é utilizado também pelo Americano a unidade de pressão é o PSI. Este exemplo resolverá tal problema, pois permitirá converter a unidade Pa -> PSI assim como o inverso, sabendo que 1 Pascal é igual a 145×10^{-6} PSI. Sendo assim, construa um novo aplicativo e deixe-o com a interface mostrada a seguir assim como os *names* sugeridos.

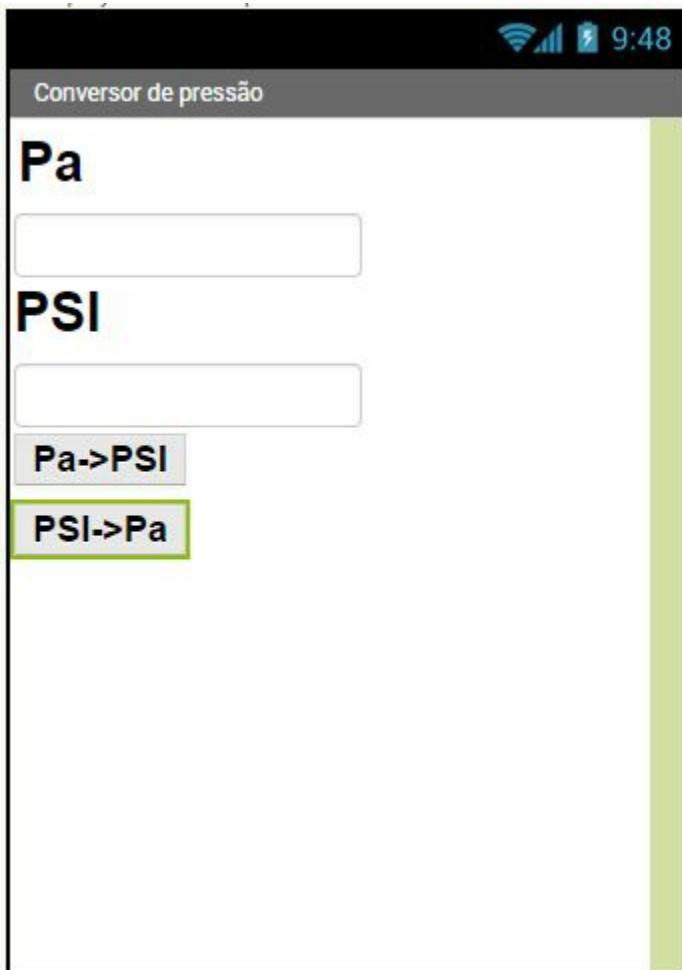


Figura 9.12 – Tela do programa

id=Pa

id=PSI

id=pa_to_psi id=psi_to_pa

Marque a opção *NumbersOnly* para as duas caixas de texto. Quando o botão Pa-> PSI for pressionado, o valor presente na caixa de texto Pa será multiplicado por 145×10^{-6} e o resultado encontrado será carregado na caixa de texto PSI. Ao pressionar o botão PSI -> Pa, o valor presente na caixa de texto PSI será dividido por 145×10^{-6} , encontrando assim o valor na unidade Pa e a caixa de texto deste será atualizada.

O bloco lógico apresentado a seguir permite converter ambas as unidades, tomando como base o valor de 0,000145 para realizar tal conversão através de produto e quociente.

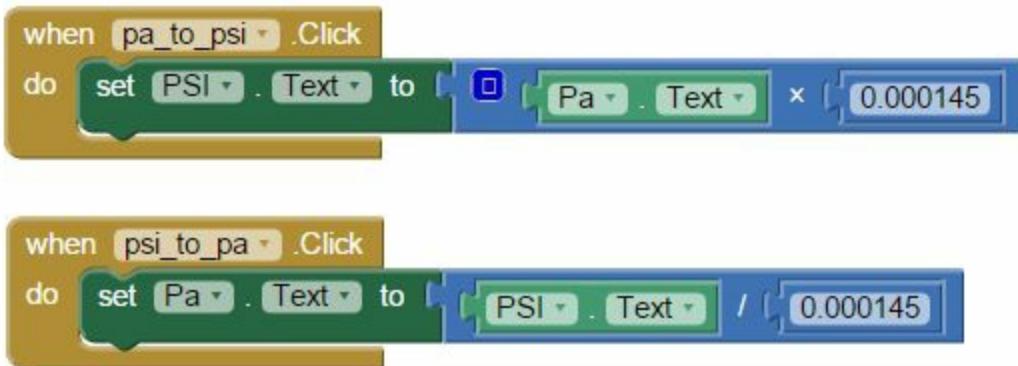


Figura 9.13 –

Bloco lógico que implementa a conversão Pa<->PSI Uma simulação desta experiência está ilustrado a seguir, no qual um valor em Pa é convertido para PSI.

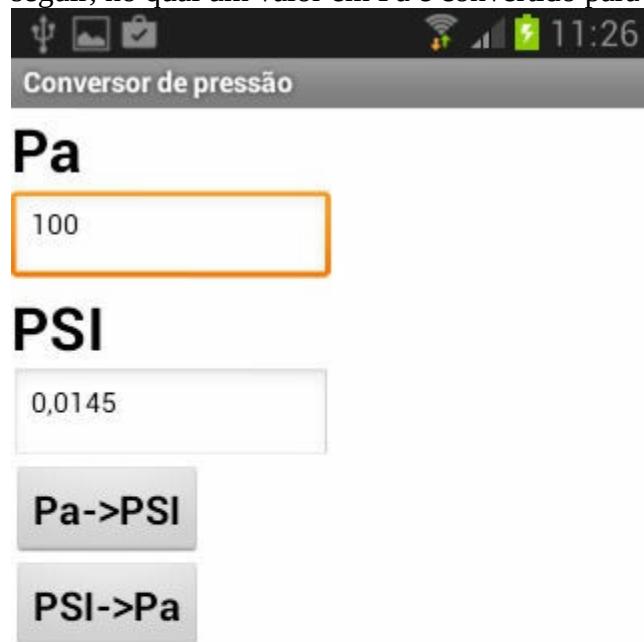


Figura 9.14 – Simulação do conversor Pa<->PSI

6. Raízes de uma equação do 2º

Obter o resultado das raízes de uma equação do 2º grau por vezes se faz necessária na vida acadêmica e este projeto propõe-se a encontrar x' e x'' de acordo com a função dada como parâmetro, neste caso informando os valores dos coeficientes a, b e c, já que uma equação do 2º é definida de acordo com a expressão abaixo.

$$ax^2 + bx + c = 0$$

Após informar os coeficientes a, b e c o aplicativo calcula o valor de Δ de acordo com o cálculo a seguir.

$$\Delta$$

$$= \\ b^2 - 4ac \\ -$$

Se Δ for menor que 0 significa que a raiz está no campo dos números complexos e desta forma o aplicativo informa pelas caixas de x' e x'' , já que o mesmo está operando apenas com números reais. Sendo Δ maior ou igual a 0, calcula-se x' e x'' da seguinte forma.

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a}$$

$$2a$$

$$\frac{-b - \sqrt{\Delta}}{2a}$$

Baseado neste modelo construa um novo aplicativo e deixe-o com a interface mostrada a seguir assim como os *ids* sugeridos.

```
id=a  
id=b  
id=c  
id=x1  
id=x2
```

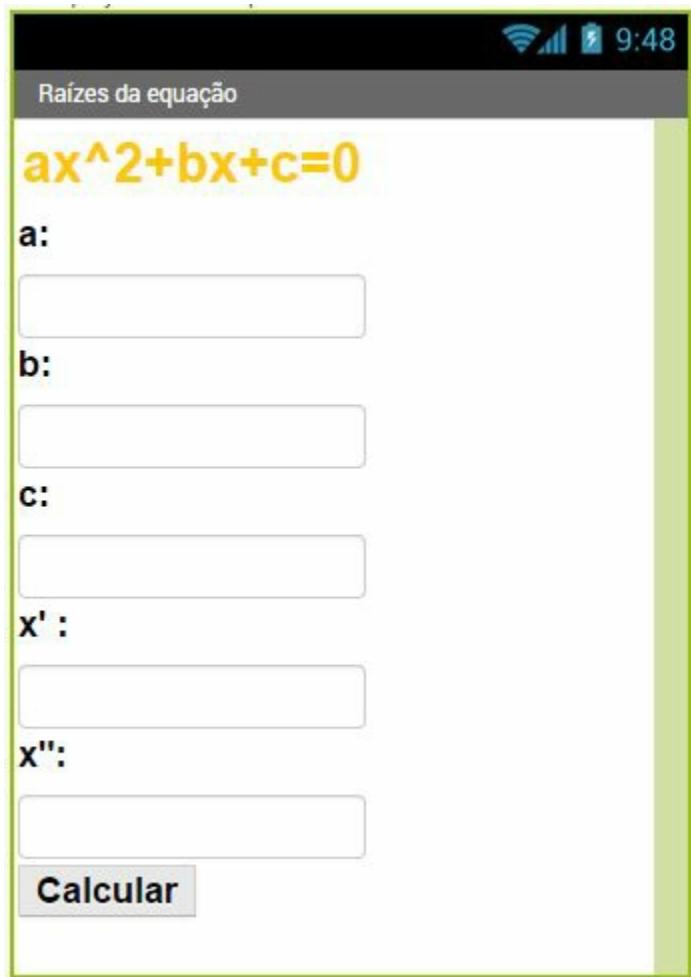


Figura 9.15 – Tela do programa de cálculo

O bloco lógico implantado para este exemplo está apresentado a seguir. Note que inicialmente, o valor de Δ é calculado, onde caso este seja maior ou igual a zero é feito o cálculo para se achar o valor de x_1 e x_2 .

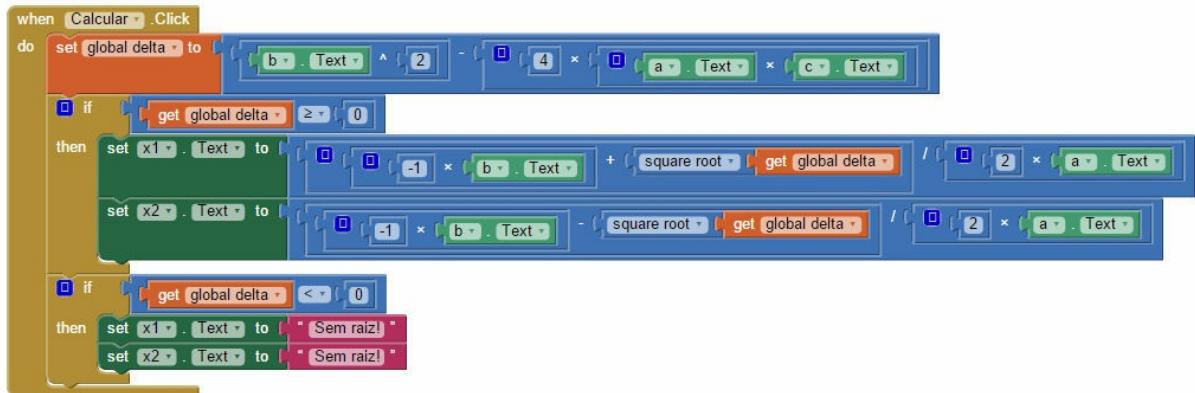
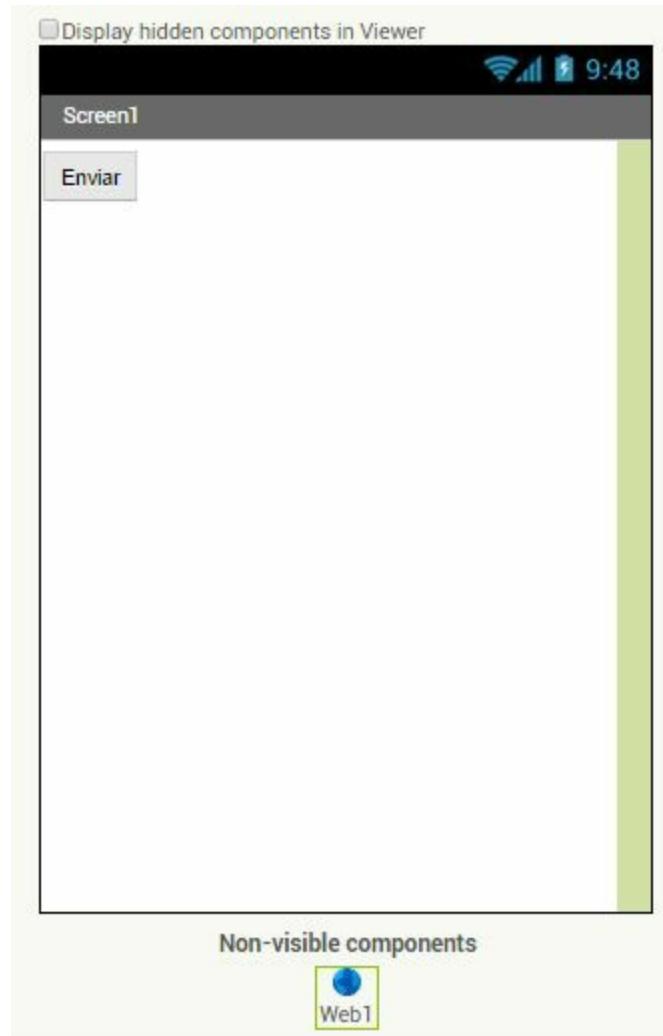


Figura 9.16 – Diagrama de blocos implementado

7. Enviando dados para WebServer

Crie um novo projeto e acrescente um botão (Button) ao mesmo.

Altere o campo *Text* para Enviar. Vá até a paleta de componentes e adicione o componente *Web*, disponível na opção *Connectivity->Web*. O resultado final fica como mostra a próxima figura.



Vá até o bloco de programação e deixe-o conforme a lógica abaixo.

Observe que assim que o botão for pressionado (*Button1.Click*) é setado em *Web1.Url* o endereço da página a ser visitada, neste caso cerne-tec.site90.net/server.php?estado=Evento:,

no qual já está configurado o parâmetro e o valor a ser carregado através do método *Get* (obs: Altere o endereço do site de acordo com a sua página Web). Em seguida é chamado o método *Get* do componente *Web1*, no qual direciona até a página e permite desta forma gerar um registro com a data e o horário da ocorrência. As próximas figuras mostram os testes realizados com este programa, observe que não há diferença do que fora feito diretamente através do Browser.

Capítulo X Hardware e Software Arduino

1. Introdução

O Arduino é uma ferramenta livre tanto em nível de hardware quanto de software, onde neste você encontra os recursos necessários para iniciar seus projetos, sem ser especialista nem em eletrônica quanto em software. Neste livro, são apresentados alguns exemplos de modo a chegar à proposta da obra que fornece uma base teórica de programação e hardware para você criar seus próprios projetos. Estes projetos podem ser úteis em disciplinas como robótica educacional quanto na criação de automatismos em geral, usados em projetos de escolas técnicas e faculdades.

2. Conhecendo o Software

O software usado neste livro foi o Arduino versão 17, que pode ser baixado gratuitamente no endereço abaixo:

<http://arduino.cc/en/Main/Software>

A grande vantagem deste software é o fato do mesmo funcionar em sistemas operacionais do tipo Windows, Linux e MacOS. A vantagem em usar o Linux é dispor de uma ferramenta gratuita que facilite o acesso aos alunos para realização de seus projetos.

Após realizar o download do software, basta descompactar a pasta e abri-la, onde você encontrará o arquivo arduino.exe, bastando sempre abrir este programa quando quiser “rodar” o software arduino. Na figura abaixo, é apresentado à tela deste programa, que voltará a ser comentado em breve.

3. Conhecendo o Hardware

Existem diversas placas que podem ser usadas para realizar os experimentos. Nas figuras abaixo, são apresentadas algumas destas.



Arduino Uno



Arduino Ethernet

Existem outros modelos e todos estes, estão disponíveis para visualização e compra no endereço abaixo:

<http://arduino.cc/en/Main/Hardware>

Este livro utiliza a placa didática Cerne Arduino, que está apresentada na figura a seguir:



Placa Cerne Arduino

Esta placa pode ser adquirida através do endereço abaixo: <http://www.cerne-tec.com.br/kitavr.htm>

Apesar do livro utilizar esta placa, nada impede que o leitor utilize outra similar.

Capítulo XI Ligando um Led no Arduino

1. Pinagem do Arduino

A placa Arduino é composta de um microcontrolador, que é uma espécie de computador em um só chip chamado ATMEGA8. Este chip está marcado na figura apresentada a seguir:



Observe na figura abaixo, que o ATMEGA8 é composto de 28 pinos, sendo alguns destes analógicos, onde é possível ligar sensores como resistores variáveis e outros do tipo digital, onde somente há como ler ou acionar dois estados, sendo este ligado ou desligado, o que induz que um botão pode estar aberto ou fechado assim como um led pode estar aceso ou apagado.

Arduino Pin Mapping

www.arduino.cc

(RESET) PC6	1	28	PC5 (ADC5/SCL)	analog input 5
digital pin 0 (RX)	(RXD) PD0	2	27	PC4 (ADC4/SDA)
digital pin 1 (TX)	(TXD) PD1	3	26	PC3 (ADC3)
digital pin 2	(INT0) PD2	4	25	PC2 (ADC2)
digital pin 3	(INT1) PD3	5	24	PC1 (ADC1)
digital pin 4	(XCK/T0) PD4	6	23	PC0 (ADC0)
	VCC	7	22	GND
	GND	8	21	AREF
	(XTAL1/TOSC1) PB6	9	20	AVCC
	(XTAL2/TOSC2) PB7	10	19	PB5 (SCK)
digital pin 5	(T1) PD5	11	18	PB4 (MISO)
digital pin 6	(AIN0) PD6	12	17	PB3 (MOSI/OC2)
digital pin 7	(AIN1) PD7	13	16	PB2 (SS/OC1B)
digital pin 8	(ICP1) PB0	14	15	PB1 (OC1A)
				digital pin 13 (LED)
				digital pin 12
				digital pin 11 (PWM)
				digital pin 10 (PWM)
				digital pin 9 (PWM)

ATmega8

Por exemplo, a identificação do pino número 14 será 8 no programa Arduino. O pino número 19, como pino 13 e assim sucessivamente. Observe que onde está escrito digital pin (pino digital), indica que é possível ler como citado anteriormente, dois estados, sendo estes aberto ou fechado assim como ligar ou desligar um led, por exemplo.

2. Montando o Hardware

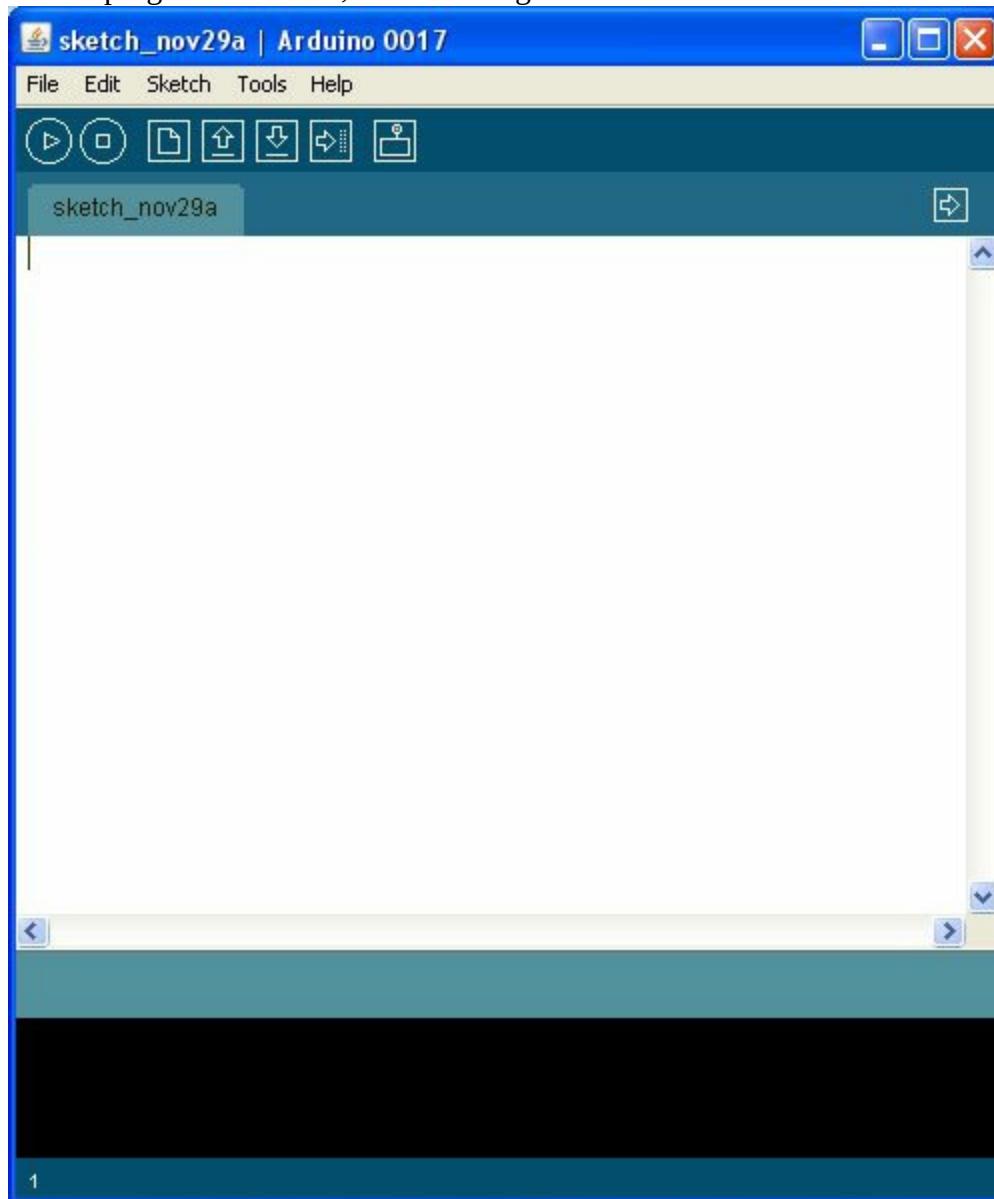
Neste primeiro exemplo o objetivo é ligar um led que está conectado à porta 13 do arduino. Na figura abaixo há uma figura de um led.



Existem vários tipos de leds, com diferentes cores, sendo que isto não interfere no exemplo. Note que o led é composto de dois terminais, sendo que o maior é o terminal positivo enquanto que o menor o negativo. Isso deve ser observado com atenção, pois se invertido, o led não funcionará. Além disso, é necessário ligar um resistor em série com o led, de forma a não queimá-lo. Um resistor típico pode ter sua resistência de $220\ \Omega$ a $470\ \Omega$. Ligue neste caso o terminal positivo do led ao pino 13 e o negativo ao GND da placa.

3. Programando o Arduino

Abra o programa Arduino, teremos a seguinte tela:



A estrutura

básica para montarmos um programa, é a seguinte:

```
void setup()
{
}
void loop()
{
}
```

Dentro das chaves do void setup(), será feita a configuração básica do Arduino. Entenda este bloco como a BIOS do computador, em que realiza-se todas as configurações iniciais para que o mesmo venha a funcionar em seguida. Dentro das chaves do bloco void loop(), é feita a lógica de programação que deseja-se que a placa execute.

Digite o pequeno trecho de código no software Arduino, o resultado será como o apresentado a seguir:



Observe que este bloco refere-se a estrutura básica do programa, ainda não foi implementado propriamente dito, ligar o led que está conectado ao pino 13.

Um pino digital pode ser configurado tanto entrada quanto saída, ou seja, se ligar um led ao pino, ele será uma saída para o chip e se ligar um botão, será uma entrada. Como o led fica conectado a porta 13, deve-se configurar esta porta como saída e para isso, utiliza-se a função `pinMode`, que permite configurar um pino como entrada ou saída. A seguir, é apresentada a configuração desta função.

```
pinMode(pino, modo);
```

Onde na opção `pino` informa-se qual pino deseja-se configurar e em modo, se ele será configurado como entrada ou saída, onde informa-se `INPUT` ou `OUTPUT` respectivamente. Sendo assim, o programa ficará da forma abaixo, em que o pino 13 está configurado como saída:

```
void setup()
{
pinMode(13,OUTPUT);
}

void loop()
```

} O pino está configurado como saída, mas ainda não foi ligado.
Para fazer isso, utiliza-se a função descrita abaixo:

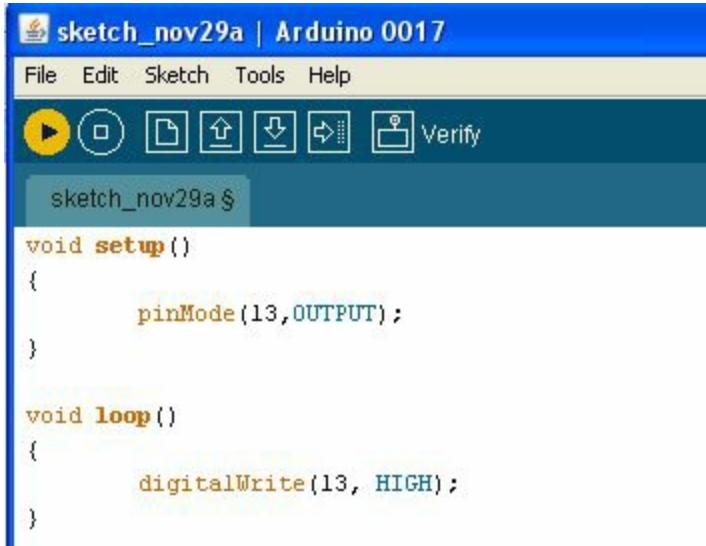
```
digitalWrite(pino, modo);
```

Onde na opção `pino`, informa-se qual pino está fazendo acesso e na opção `modo` o estado que assumirá, neste caso em nível alto, ou seja 5V ou nível baixo, neste caso 0V. Para nível alto deve-se escrever `HIGH` e para nível baixo `LOW`. Abaixo o código fonte completo.

```
void setup()
{
pinMode(13,OUTPUT);
}

void loop()
{
```

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



The screenshot shows the Arduino IDE interface. The title bar says "sketch_nov29a | Arduino 0017". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, upload, download, and verify. The code editor window contains the following code:

```
void setup ()  
{  
    pinMode(13,OUTPUT);  
}  
  
void loop ()  
{  
    digitalWrite(13, HIGH);  
}
```

Neste instante para verificar se o código escrito está utilizando-se a ferramenta de compilação do Arduino. Vá ao menu Sketch -> Verify/Compile ou pressione o botão marcado abaixo para iniciar a compilação.



Caso tudo esteja ok, a mensagem

marcada abaixo irá aparecer.

```
sketch_nov29a | Arduino 0017
File Edit Sketch Tools Help
sketch_nov29a §
void setup()
{
    pinMode(13,OUTPUT);
}

void loop()
{
    digitalWrite(13, HIGH);
}

Done compiling.

Binary sketch size: 592 bytes (of a 7168 byte maximum)
```

Caso tal mensagem não apareça, revise o seu código e observe se está conforme apresentado.

Neste momento, salve o exemplo indo ao menu File -> Save e salvando na pasta que melhor lhe convier. Em seguida, realize o upload para a placa de modo a carregar o programa na placa Arduino.

Capítulo XII Transmissão Serial

1. Introdução

A comunicação serial do tipo RS232 ainda é uma das mais usadas e neste capítulo é apresentado os passos para transmitir dados usando este tipo de interface. Será transmitida uma string, onde a placa Cerne Arduino dispõe dos recursos para que possa enviar tal informação.

2. Montando o Hardware

Este exemplo usa uma linha de transmissão e recepção que possibilitam a comunicação serial. Tais linhas já estão ligadas ao CI MAX232 que fica na placa, que converte os sinais do nível TTL para RS232 ligado a porta DB9. Neste caso, não será necessário montar um hardware de teste, já que ele está disponível na placa.

3. Programando o Arduino

Deve-se configurar a velocidade no qual é feita a comunicação, podendo esta ser de várias velocidades, como 1200 bps (bits por segundo), 2400 bps, 4800 bps, 9600 bps etc. Neste exemplo a velocidade usada é de 9600 bps. Abaixo está apresentado a função que configura a velocidade de comunicação:

```
Serial.begin(velocidade);
```

A função para enviar a string está apresentada a seguir:

```
Serial.println(texto);
```

O exemplo completo está apresentado logo abaixo enviando a string para o PC via RS232 a 9600 bps.

```
void setup() {  
  Serial.begin(9600);  
  Serial.println("Aprendendo Arduino");  
}  
  
void loop() {  
}
```

Para visualizar os dados será utilizado o Serial Monitor, que já vem com o próprio software Arduino. Clique na opção marcada abaixo para inicializar este software.

The screenshot shows the Arduino IDE interface. The title bar reads "Exemplo_TXSerial | Arduino 0017". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, upload, and others. The main code editor window contains the following sketch:

```
void setup()
{
    Serial.begin(9600);
    Serial.println("Aprendendo Arduino");
}

void loop()
{}
```

Below the code editor is a status bar with the number "10". The bottom half of the screen is occupied by a terminal window titled "Serial Monitor". It displays the following error message:

```
Problem uploading to board. See http://www.arduino.cc/en/Guide/Troubleshooting#upload
avrdude: stk500_getsync(): not in sync: resp=0x00
avrdude: stk500_disable(): protocol error, expect=0x14, resp=0x51
```

Neste momento, a seguinte tela surgirá, onde após a gravação da placa e remoção do jumper de gravação será possível visualizar o resultado apresentado abaixo a 9600 bps.



comunicação com Android

O próximo exemplo será utilizado em conjunto com o Android programado no App Inventor para envio do estado do botão assim que o mesmo for pressionado. Note que assim que o botão conectado ao pino 8 for pressionado, ou seja, levado a nível 0 é enviado um caractere “1”, onde a intenção é que quando o dispositivo Android receber tal byte possa gerar o evento no servidor Web da mesma forma do que foi visto no capítulo IX.

```
#define BOT 8

void setup() {
Serial.begin(9600); pinMode(BOT,INPUT); digitalWrite(BOT,1);
}

void loop() {
if(digitalRead(BOT)==0) {
Serial.write('1');

while(digitalRead(BOT)==0) delay(100);
}
}
```

Capítulo XIII Recepção Serial

1. Introdução

Neste capítulo é apresentado os passos necessários para

receber dados da RS232 e ligar ou desligar um LED de acordo com o caractere recebido. Caso o caractere recebido seja “1” o LED fica aceso e se for “0” apagado. Este mesmo exemplo será utilizado no exemplo a ser feito em conjunto com o App Inventor, pois um módulo Bluetooth como o apresentado na figura a seguir permite a placa Arduino comunicar por esta interface, já que ela faz a conversão Bluetooth <-> RS232.



Sendo assim, no momento do teste para automação a ser realizada entre o smartphone/tablet Android e o Arduino, deixe o programa apresentado neste capítulo carregado na placa juntamente com o módulo Bluetooth.

Desta forma, um LED poderá ser ligado ou desligado remotamente, através de um smartphone ou tablet com sistema Android. Todavia, este mesmo exemplo pode ser testado através da interface RS232.

2. Montando o Hardware

Este exemplo usa uma linha de transmissão e recepção que possibilitam a comunicação serial. Tais linhas já estão ligadas ao CI MAX232 que fica na placa, que converte os sinais do nível TTL para RS232 que fica ligado a porta DB9. Neste caso, não será necessário montar um hardware de teste, já que ele está disponível na placa. Basta neste caso conectar um LED ao pino 13 da placa Arduino conforme exemplo visto no acionamento de saídas.

3. Programando o Arduino

Para receber dados da porta serial é usada a função `Serial.read()`, que está apresentada abaixo:

Além desta função, utiliza-se a função `Serial.available()`, que serve para informar se há dados disponíveis a serem lidos. Abaixo é apresentada tal função:

```
Serial.available();
```

A seguir é apresentado o programa completo que permite receber os dados da serial e ligar ou desligar o LED de acordo com o caracter. Observe que para enviar dados, foi usado o programa Serial Monitor disponível no próprio programa, onde os caracteres digitados são enviados via serial.

```
void setup() {
  pinMode(13,OUTPUT);
  Serial.begin(9600);
}

void loop() {
  if (Serial.available() > 0) {

    char rx;
    rx=Serial.read();
    if(rx=='1')

      digitalWrite(13,HIGH); if(rx=='0')
      digitalWrite(13,LOW); }
}
```

Com este programa, sempre que a placa Arduino receber o caracter “1” o LED será aceso e quando receber “0” apagado. Para testar via Bluetooth, basta conectar o módulo adaptador a placa Arduino e realizar o mesmo procedimento a partir de um PC, com o uso de um

adaptador USB<->Bluetooth.

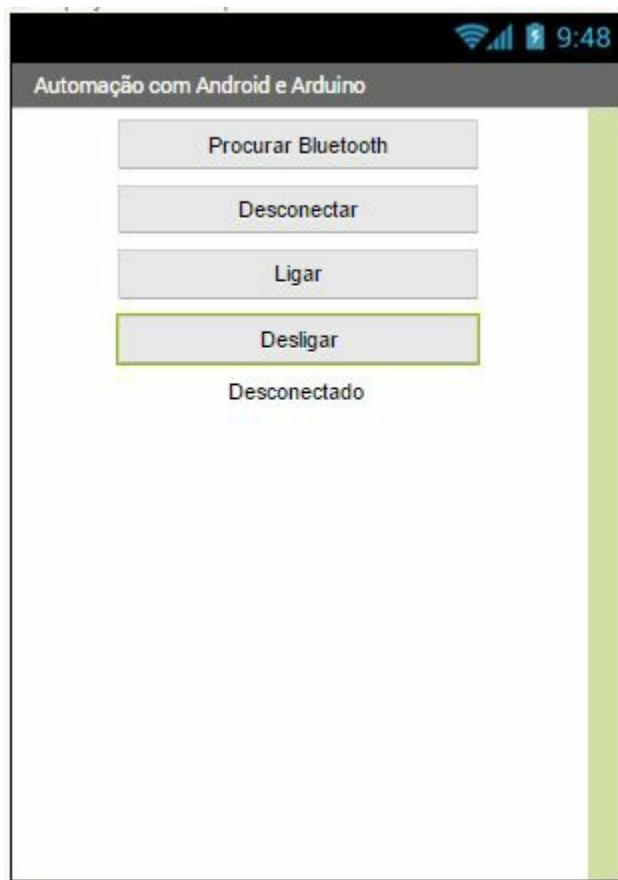
No próximo capítulo será apresentada a construção do programa no App Inventor que permite ligar ou desligar o LED usando a interface Bluetooth a partir de um dispositivo Android.

Capítulo XIV Aplicativo com interface Bluetooth

1. Controlando o estado do LED

Neste tópico um programa no App Inventor foi desenvolvido com o intuito de permitir ligar ou desligar um LED presente na placa Arduino. Este mesmo LED poderia ser um relé, que permitisse ligar ou desligar uma lâmpada assim como outras cargas. Deste modo, crie um novo projeto no App Inventor e deixe-o conforme sugerido na figura a seguir assim como os names sugeridos.

procurar desconectar ligar desligar estado



Vá à paleta *Connectivity* e adicione *BluetoothClient* ao projeto que apesar de não fundamental para o projeto.
o componente

estar visível é

Observe que o primeiro botão é um componente do tipo *ListPicker*, disposto na mesma paleta do botão. Este componente foi utilizado para permitir parear com o dispositivo Bluetooth instalado juntamente a placa Arduino.

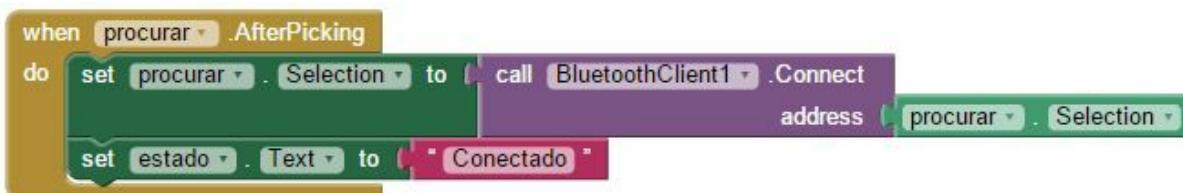
O botão desconectar irá interromper o link Bluetooth entre o Android e Arduino enquanto o botão Ligar e Desligar permite controlar o LED presente na placa. O label estado informa se naquele instante o Android está conectado ou não a placa Arduino via Bluetooth.

Vá ao bloco de programação e *procurar.BeforePicking* disponível no procure o bloco *When* componente *ListPicker* renomeado para procurar. Deixe sua configuração conforme o bloco a seguir.



Sendo assim o Android permite parear com o endereço disposto no módulo Bluetooth acoplado a placa Arduino, sendo esta a primeira tarefa a ser executada.

Adicione o bloco *When procurar.AfterPicking* disponível no componente *ListPicker* renomeado para procurar. Deixe sua configuração conforme o bloco a seguir.



Este bloco faz a conexão e altera o texto da caixa de texto estado para “Conectado”. Deixe os blocos lógicos dos botões ligar e desligar conforme a figura a seguir.



Cada um dos blocos envia pela Bluetooth o caractere “1” para ligar e “0” para desligar. Finalmente, coloque o bloco a seguir para desconectar da rede Bluetooth.



Este bloco chama a função para desconectar da rede Bluetooth e altera o estado do botão para “Desconectado”.

Após estes procedimentos, compile o programa e instale no smartphone/tablet Android. Deixe o programa do capítulo anterior carregado no Arduino juntamente com o módulo Bluetooth. Faça o pareamento entre o Android e Arduino pela Bluetooth e observe o acionamento para ligar e desligar o LED presente no kit através dos botões dispostos no programa Android.

2. Enviando o evento do botão pela Web

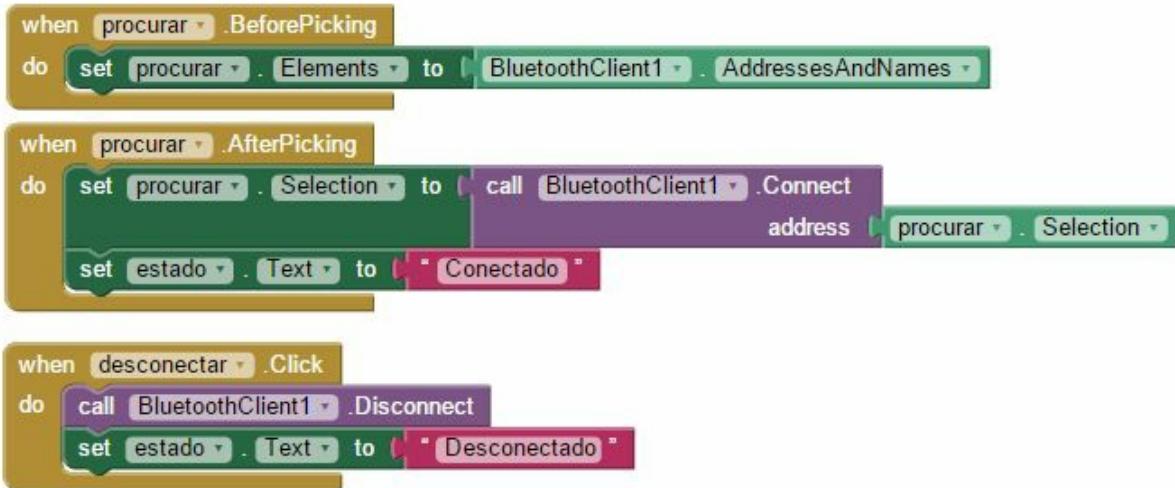
Este exemplo vai enviar através do Bluetooth o estado do botão da placa Arduino para o smartphone/tablet Android, onde ao receber tal evento o mesmo envia para o servidor Web o evento gerado pela placa Arduino. Para isso, crie um novo projeto e deixe o design como mostra a próxima figura.

procurar
desconectar
estado

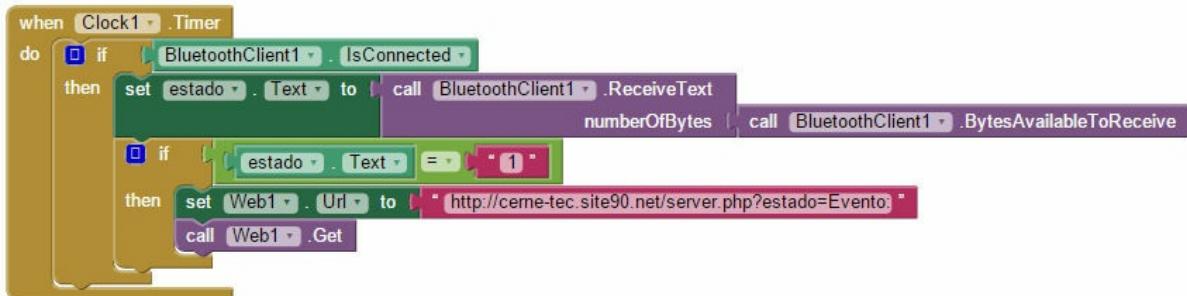


Adicione o componente *BluetoothClient* indo na paleta de componentes *Connectivity->BluetoothClient*, o componente *Clock* responsável pela verificação periódica de novos dados recebidos pelo Bluetooth indo na paleta de componentes *Sensors->Clock* e o componente *Web* que faz a conexão com o servidor Web disponível na seção *Connectivity->Web*.

O próximo bloco lógico não apresenta novidades em relação ao exemplo de comunicação Bluetooth, onde há o bloco de seleção, conexão e desconexão com o Bluetooth.



O próximo bloco é executado a cada 1000 ms, dado pelo tempo determinado pelo *Clock*, onde é verificado se há caracteres recebidos pelo Bluetooth e caso haja, este é informado através da caixa de texto *estado* e verificado se o caracter é o “1”, onde caso seja é enviado uma mensagem através do método GET para a mesma página utilizada no exemplo do capítulo IX para conexão com o servidor Web.



Faça o teste e comprove o funcionamento, onde ao pressionar o botão conectado ao pino digital 8 do Arduino esperase que ocorra um evento no WebServer como visto anteriormente.

Re e ên a

SOUZA, Vitor Amadeu. **Programação em PHP - HTML, MySQL e PHP**. São Paulo. Clube dos Autores, 2011.

SOUZA, Vitor Amadeu. **HTML Com CSS InLine, Incorporado e Externo**. São Paulo. Clube dos Autores, 2012.

SOUZA, Vitor Amadeu. **Programação para Google Android – Baseado no MIT App Inventor**. São Paulo. Clube dos Autores, 2014.