

AGENDA 1

PHP: INTRODUÇÃO A SERVER-SIDE



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

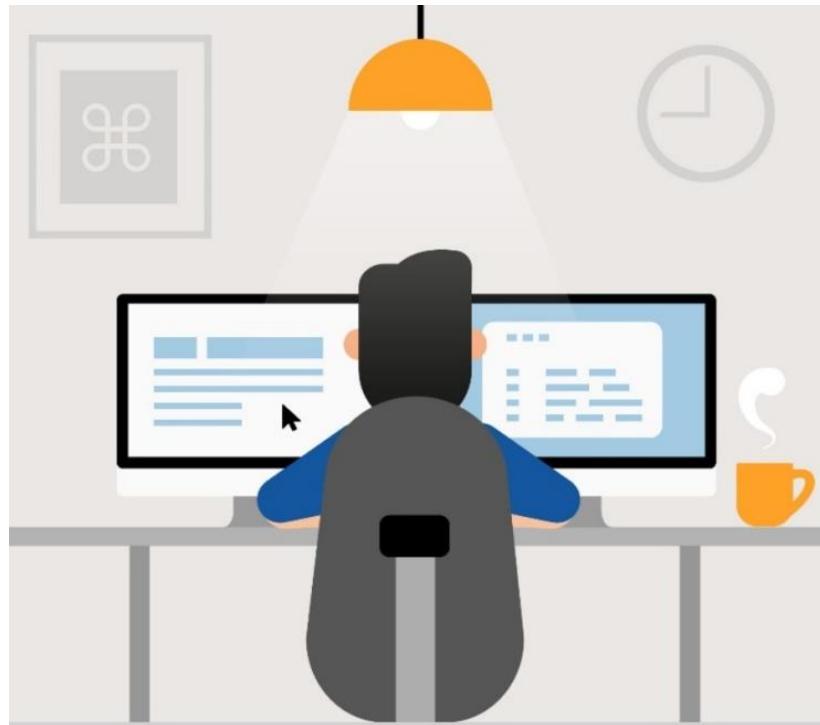
Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



Você já pensou em quantos sites você utiliza diariamente? Ou no quanto a internet facilita seu dia a dia?

Podemos destacar como principais exemplos: transações bancárias, estudos e pesquisas, utilização das redes sociais, compras em lojas virtuais dos mais diversos segmentos. Com tantos acessos e utilização de sites, você já pensou como eles funcionam? Como você consegue fazer uma compra e como a mercadoria chega até a sua casa?



Para exercer essas funções, esses tipos de sites precisam basicamente de dois componentes: um **cliente** e um **servidor web**. Imagine que você esteja consultando o seu extrato bancário! Muitos dos processos necessários para que as informações sejam apresentadas corretamente estão sendo executados no **servidor web** do banco. **O site do banco precisa de dados**, por exemplo: os dados da conta do cliente, os dados cadastrais do cliente, das transações financeiras realizadas, além de outras informações para garantir que ao final do processo, você obtenha as informações, conforme suas solicitações.

Vamos entender ou compreender como fazer aplicações web que permitem que os sites tenham a capacidade de realizar diversas **funcionalidades** e **serviços**? Nesta agenda, veremos um pouco de SERVER-SIDE (Lado do Servidor) e a introdução para **linguagem de programação PHP** (Hypertext Preprocessor – Pré processador de hipertexto) que nos fornecerá ferramentas e recursos para realizar as mais diversas funcionalidades de uma página WEB!



O desenvolvimento de um site pode ser muito complexo, do ponto de vista da programação. O mercado oferece diversas linguagens que seguem diferentes paradigmas. Vamos aprender a linguagem PHP (Hypertext Preprocessor – Pré processador de hipertexto), uma das mais conhecidas e utilizadas devido a sua tipagem dinâmica, utilizando o conceito Server-side como alicerce.



Sempre que você acessa um site, seja pelo computador, celular ou tablet, utiliza um navegador como o Internet Explorer, Chrome, Firefox, Safari etc. Esses navegadores conhecem basicamente três tecnologias: HTML, CSS e Javascript. O **HTML** tem a responsabilidade de **estruturar o conteúdo das páginas**, o **CSS** define a **formatação e a aparência do conteúdo** e o **Javascript** adiciona **interatividade a uma página web**.

Mas isso não é o suficiente para desenvolver sites mais complexos, como por exemplo: o site do banco que você utiliza para fazer suas transações ou um site de compras. Vamos pensar juntos: Onde ficarão armazenadas as informações da sua conta, como o seu extrato e os históricos de transferências ou últimas compras realizadas? Será que apenas o navegador consegue armazenar e exibir todas essas informações utilizando apenas essas três tecnologias apresentadas até agora?

Lembre-se de que o código que é executado no navegador está rodando na sua máquina, desta forma, todas essas informações teriam que estar no seu computador! Será que isso daria certo? Como fica, por exemplo, a questão da segurança dos seus dados?

O que acontece, nesse caso, é que quando você digita o endereço do site do banco no navegador do seu computador e pressiona a tecla “Enter”, ele faz uma solicitação, que chamamos de **requisição para o servidor do banco** (outro computador). Esse servidor, por sua vez, processa essa requisição, consulta o seu banco de dados e devolve uma resposta para o seu navegador. Esse código que é executado do lado do servidor utiliza linguagens, como: **Perl, Ruby, Python, PHP, Java e C#.**



Veja o caso do Zeca! Ele já desenvolveu alguns sites como *freelancer* e, como ele começou a ter cada mais clientes com maiores necessidades, os sites ficaram mais complexos, seus novos projetos começaram a precisar de novas tecnologias.

E foi assim que o Zeca começou a estudar novas tecnologias. Ele aprendeu a **a utilizar uma linguagem de programação server-side!**

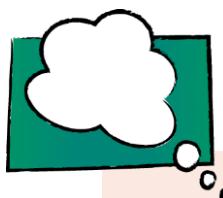


Você sabe como funciona essas linguagens? Mergulhe no tema desta agenda e descubra!



Todo site que você acessa pelo navegador de Internet (Chrome, Internet Explorer, Mozilla, Firefox etc) está hospedado em um computador servidor e então, ao acessar qualquer site na internet, por meio do seu navegador de internet, está fazendo solicitações, ou seja, **requisições a esse servidor**, e, não apenas você, mas outras pessoas também estão fazendo requisições pela internet a todo o momento. Caso muitas pessoas estejam acessando o site ao mesmo tempo, cabe ao **servidor gerenciar esses acessos** de maneira correta. Portanto, quando você acessa um site está de certa forma, acessando uma série de estruturas na internet (**protocolos** de comunicação e **estruturas físicas**) e por trás de tudo isso existe um computador do tipo **servidor** controlando todos esses processos.

No servidor, há linguagens que podem ser instaladas para fazerem alguns procedimentos, como se **conectarem** a um banco de dados ou fazer **rotinas** de programação. Nas primeiras agendas, aprendemos algumas linguagens que não são processadas do lado do servidor, mas sim, do **lado do usuário (cliente)**, como o **HTML, Javascript e CSS**. Apesar de que estes arquivos (.html , .css ou .js) estejam hospedados (alocados) nesse servidor, eles são baixados pelos navegadores de Internet e quem os processa são os computadores do usuário.



Assim, concluímos que essas linguagens não teriam **acesso ao banco de dados**, sendo processadas pelos computadores, pois seria um risco à **segurança e integridade dos dados**, então, a melhor escolha é que a linguagem seja processada do **lado do servidor**, concorda?

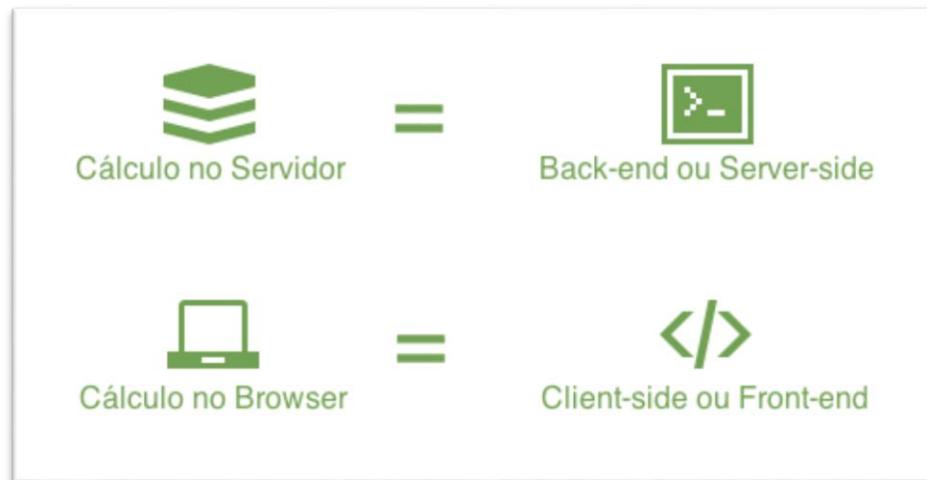


Imagen 03 - Adaptado de <http://tableless.github.io/iniciantes/assets/img/client-side-back-end.png>

A imagem 03 representa dois tipos de processamento: **lado do cliente** e **lado do servidor**. Com esses conceitos conseguiremos iniciar a programação em linguagem **server-side**, porém precisaremos de um **servidor web**.

Não há como trabalhar do lado do servidor sem que você tenha um servidor instalado. Você poderia acessar um servidor web externo, mas isso poderia gerar alguns custos de manutenção.

Servidores Web

Você pode instalar e configurar manualmente o servidor **Apache**, a linguagem **PHP** e o **banco de dados**, porém há soluções muito mais simples para começar o desenvolvimento de seus projetos. Trata-se de **pacotes de softwares** que já instalaram e configuraram o servidor web, a linguagem **PHP** e **banco de dados**, e há diversas opções no mercado, como:

Ferramenta	Site Official / Download
XAMP	https://www.apachefriends.org/index.html
WAMP	http://www.wampserver.com
EasyPHP	https://www.easyphp.org/
AppServ	https://www.appserv.org/
Zwamp	http://zwamp.sourceforge.net/

Todas essas ferramentas citadas anteriormente necessitam de instalação, porém, no mercado existem versões portáteis disponíveis e arquivos executáveis que não necessitam de instalação, sendo possível desenvolver e demonstrar seus sites **PHP** em qualquer lugar e a qualquer momento, utilizando apenas um pendrive ou similar.

Neste curso utilizaremos a ferramenta **USBWebServer**, ela é gratuita e é possível realizar seu download por meio do link: <https://usbwebserver.yura.mk.ua/>.

Perceba que o site oferece duas versões: a 8.6.1 (PHP 5) e a 8.6.2 (PHP7), conforme mostra a imagem 04.



Imagen 04–Links para download da ferramenta USBWebserver.

Neste curso vamos utilizar a versão **PHP 7**, então clicaremos no segundo link de download.
(Imagen 05)

[usbwebserver_v8.6.2.zip - PHP 7.1](https://usbwebserver.yura.mk.ua/usbwebserver_v8.6.2.zip - PHP 7.1)

Imagen 05 – Link para download do PHP 7.1.

Depois de realizar o download do arquivo, basta descompactá-lo em uma pasta como demonstrado pela imagem ao lado.

Execute o aplicativo **usbwebserver** (Imagen 07), clicando duas vezes sobre o mesmo .

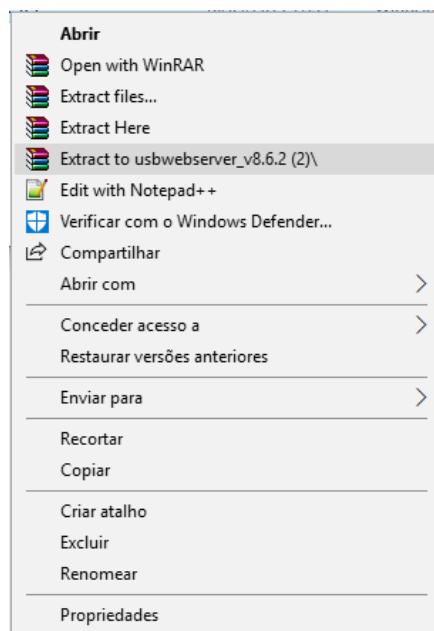


Imagen 06 – Descompactar arquivo baixado da internet

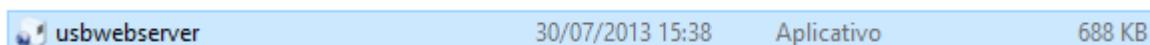


Imagen 07 – Aplicativo a ser executado conhecido como **usbwebserver**.

Se seu **firewall** estiver ativo, irão aparecer duas mensagens como na Imagem 08, pois trata-se de **permissões** para acesso externo ao servidor apache e ao banco de dados dessa ferramenta. Caso tenha outros computadores disponíveis em sua rede e você deseja que eles acessem o **apache** e **mysql**, clique em **permitir**, caso contrário clique em cancelar.

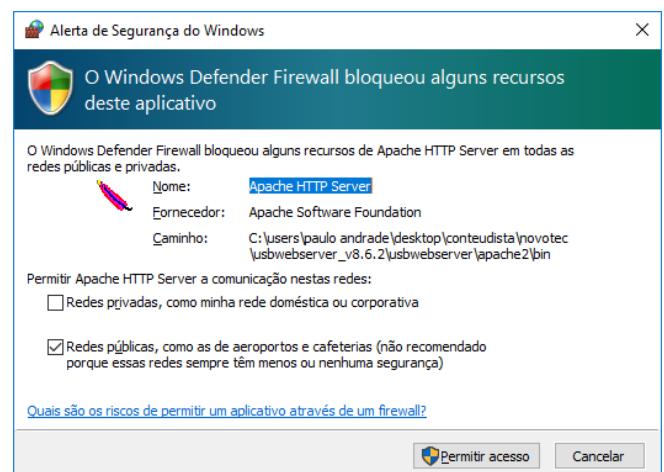
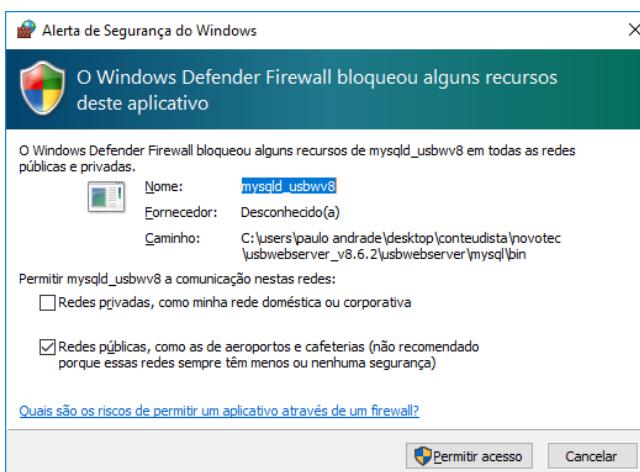


Imagen 08 – Permissão de Acesso externo com verificação do **firewall**.

Logo após, você deverá selecionar a linguagem que deseja utilizar (Imagen 09).

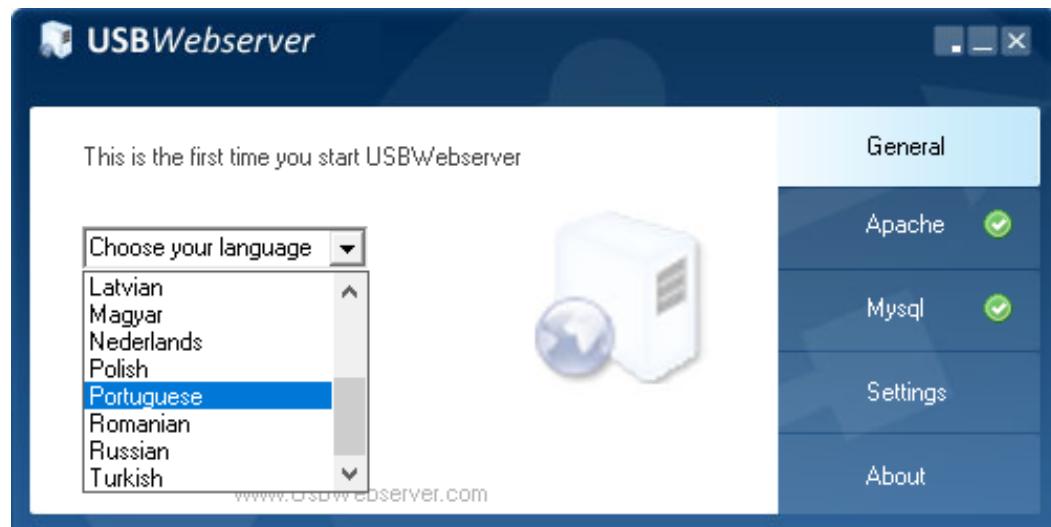


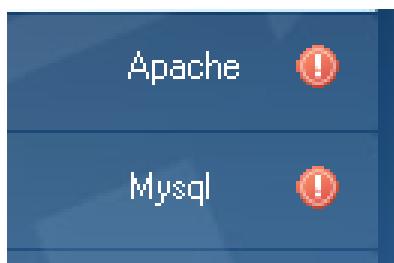
Imagen 09 – Escolha da linguagem, como a opção Portuguese.

Nas guias **apache** e **mysql** deverão aparecer duas bolinhas verdes, indicando que os serviços estão funcionando de forma adequada. (Imagen10)



Imagen 10 – Botões e serviços iniciados da forma adequada.

Obs.: Caso aconteça de uma ou ambas bolinhas ficarem vermelhas, com o símbolo de exclamação é porque algo impediu a inicialização do Apache e/ou MySql. (Imagen 11)



*Imagen 11 – Erro na inicialização dos serviços,
Apache e Mysql com símbolo de exclamação*

Alguns dos possíveis problemas e respectivas soluções.

1 – O local do arquivo: quando você descompactou o USB Webserver, este procedimento foi realizado dentro de alguma pasta, caso alguma das pastas da hierarquia tenha algum caracter especial como “º”, os serviços podem deixar de iniciar. **Como resolver:** troque **Usb WebServer** de pasta.

2 – Outro Servidor Web executando na máquina: pode acontecer de existir na máquina outro servidor web. **Como resolver:** clique na guia configurações e **troque a porta apache de 8080 para 8081**, ou outra próxima (**8082**). **Obs:** “Caso você realize esta troca na hora de acessar suas páginas deverá utilizar: “porta”. Exemplo: se trocou para a porta **8081**, deverá digitar na URL: **localhost:8081**.

3- Outro servidor de Banco de dados executando na máquina: pode acontecer de existir na máquina outro servidor de BD. **Como resolver:** clique na guia configurações e **troque a porta mysql de 3306 para 3307**, ou outra próxima (**3308**). **Obs.:** Caso você realize esta troca na hora de acessar, lembre-se de utilizar a porta escolhida.

Agora precisamos testar e verificar se está tudo correto! Para isso teremos dois caminhos:

- 1- Clicar sobre o botão **localhost** ou
- 2- Abri o navegador e digitar na URL: “**localhost**” e pressionar **ENTER**.

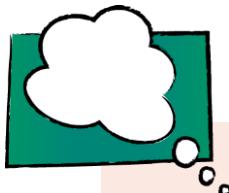
O resultado será a exibição do **PHP Info** e **webserver** em funcionamento:



Imagen 12 – Phpinfo, webserver funcionando.

Como discutido anteriormente, no **lado do servidor** estão **todos os arquivos do site**, portanto, ao utilizar um servidor web no computador, todos os arquivos do site ficarão na pasta/diretório do seu servidor, geralmente essa pasta tem o nome de “**www**”, “**public_html**”, “**htdocs**”, no caso do **USB Wserver** é denominada “**root**”. Para acessá-la, basta abrir a pasta na qual o **USB Webserver** foi descompactado e abrir a pasta **root** ou clicar no botão “**root dir**”, no guia geral do aplicativo **USB WebServer**.

Quando requisitado, tudo o que for **PHP** será processado pelo servidor e enviados apenas os resultados para o cliente.



Depois de tudo isso, vamos criar a primeira página em **PHP**?

Então vamos lá:

1. Dentro da pasta **root**, crie uma pasta denominada “**Agenda 1**”, nesta pasta vamos concentrar todas as atividades.
2. Abra o **Visual Studio Code**, crie um arquivo e salve dentro da pasta recém-criada com o nome “**ola**”, não esquecendo de escolher o tipo **PHP**, clique em salvar. (Imagen 13)

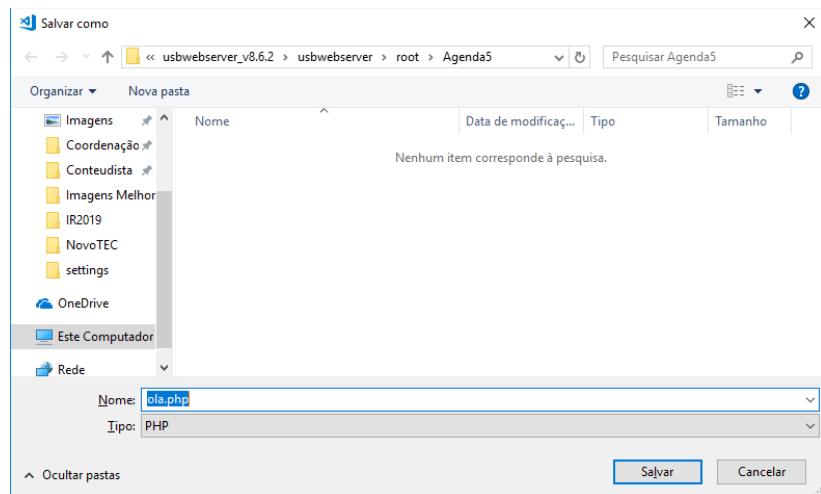


Imagen 13 – Salvando Arquivo PHP.

Obs: Nunca utilize caracteres especiais e espaços em branco para nomear arquivos.

Perceba que após o arquivo ter sido salvo, o **Visual Code** coloca o símbolo do **PHP** demonstrando que se trata de um arquivo **PHP** (Imagen 14).



Imagen 14 – Imagem da guia do Arquivo no Visual Studio Cod- arquivo salvo com o nome “ola” e extensão “.php”.

3. O próximo passo será a criação dos delimitadores PHP, como apresenta o código a seguir:

```
<?php
```

```
?>
```

Precisamos dos delimitadores pois, apesar do arquivo ser php, o mesmo pode possuir códigos HTML, logo os delimitadores identificam para o servidor que estes códigos são PHP.

4. Agora vamos enviar uma mensagem para a tela do navegador. Para isso, vamos utilizar o comando “echo”, este comando é basicamente um dos mais utilizados em sistemas PHP, por ele ser o responsável por enviar dados para o usuário, codifique então:

```
<?php
echo "Olá Mundo";
?>
```

Salve o arquivo. Abra seu navegador favorito e digite na url:
“localhost/Agenda5/ola.php”.

5. O resultado deverá ser como o apresentado na imagem a seguir:

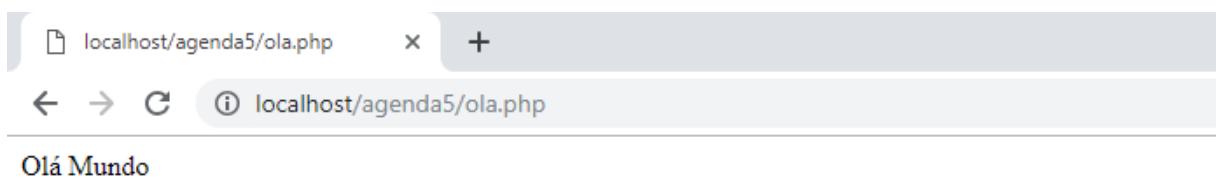


Imagen 15 –Resultado no navegador da página com a frase “Olá Mundo”.

Pronto, criamos a primeira página PHP! Neste momento, é possível verificar uma diferença entre o programado “Olá Mundo” em php e o em html, para verificar basta clicar com o botão direito e clicar em exibir código fonte da página.

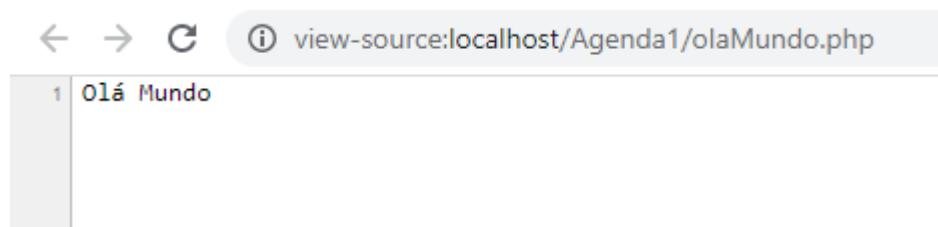


Imagen 16 –Resultado no navegador de exibir o código fonte da página “Olá Mundo”.

Perceba, que os códigos PHP não estão sendo exibidos, qual o motivo para isso? Quando o usuário faz a requisição para o servidor, ele busca a página, interpreta todo o código PHP e envia como resposta apenas o necessário para o navegador interpretar e exibir a página para o usuário que, neste exemplo, é apenas a Frase codificada.

Obs: O USBWebServer vem com uma configuração para não mostrar no site os erros de codificação. Quando pensamos no produto acabado, esta configuração é muito válida, porém, para o processo de aprendizagem trocar essa configuração será muito útil.

Para realizar a alteração, basta entrar na pasta UsbWebServer, e depois em uma pasta denominada **settings**. (Imagen 17)

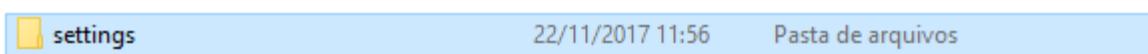


Imagen 17 – Pasta settings.

Dentro desta pasta terá um arquivo denominado **php.ini**, clique com o botão direito sobre ele e escolha a opção “**abrir com**”, escolha o Visual Studio Code ou IDE que você está utilizando para o desenvolvimento e pressione **CTRL + F** (Caso esteja utilizando Visual Studio Code), e coloque para procurar “**display_errors = Off**”, esta configuração estará provavelmente na linha 477, conforme a imagem a seguir.

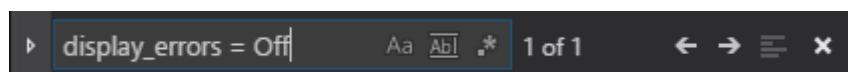


Imagen 18 – display_errors=Off.

Substitua o **Off** por **On** e salve, pronto! Quando acontecer algum erro no código php o servidor enviará uma mensagem, para a tela.



Utilizando o que foi visto até agora....

1. Crie um arquivo PHP na pasta Agenda 1.
2. Mostre na tela os dados a seguir:
 - a. Nome Completo;
 - b. Idade;
 - c. Profissão.



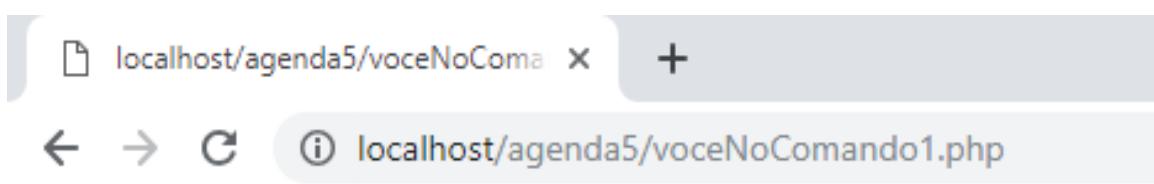
Confira a seguir se você conseguiu resolver os desafios propostos!

O código abaixo deve ser criado no **PHP** e salvo na pasta Agenda 1.

```
<?php  
echo"Nome: Zeca da Silva. Idade: 19. Profissão: Estudante";  
?>
```

Resultado no Navegador

Ao digitar no navegador **localhost/agenda5/voceNoComando1.php** teremos como resposta: "Nome: Zeca da Silva, Idade: 19 e Profissão: estudante", conforme a imagem a seguir.



Nome: Zeca da Silva. Idade: 19. Profissão: Estudante

Imagen 19 –Resultado no navegador do Exercício Você no Comando.

Comentários

Obs.: Da mesma forma que **HTML** e **CSS**, o **PHP** também possui comentários. Você pode adicionar comentários em seu arquivo PHP usando “//” comentário em linha

```
<?php  
//Comentário.  
?>
```

Ou utilizando /* */ para comentário em bloco.

```
<?php  
/*  
     Comentário em Bloco  
  
     */  
?>
```



Estes exercícios devem ser entregues de forma on-line como atividades da agenda.

Questionários online

1. Os delimitadores para indicar ao servidor onde se iniciam e se encerram os códigos PHP são:
 - (A) <?></?>
 - (B) <php> </php>
 - (C) <#php #>
 - (D) <!php !>
 - (E) <?php ?>

2. Quando programamos em PHP quem é o responsável pela interpretação dos códigos:
 - (A) Browser (Navegador)
 - (B) CPU (Unidade Central de Processamento)
 - (C) CPUW (Unidade Central de Processamento WEB)
 - (D) Web Server (Servidor WEB)
 - (E) Web Page (Página WEB)

3. Zeca ao executar os seguintes comandos:

```
<?php  
    echo "Olá"  
    echo "Zeca";  
?>
```

Apresentou o seguinte Erro:

Parse error: syntax error, unexpected 'echo' (T_ECHO), expecting ',' or ';' in C:\Users\..... on line 3;
Qual das alternativas indica o que Zeca errou em sua codificação.

- (A) Não finalizou o primeiro comando echo com ponto e vírgula
- (B) Não finalizou o primeiro comando echo com três pontos.
- (C) Utilizou o ponto e vírgula no segundo comando echo.
- (D) Utilizou dois comandos echo em sequencia.
- (E) Utilizou duas palavras com as primeiras letras em maiúsculo.

4. Zeca ao implementar em php os seguintes comandos:

```
<?php  
    echo "Zeca da Silva";  
    // echo "Estudante";  
?>
```

O resultado em seu navegador (Browser) foi:

- (A) A Frase “// Estudante”
- (B) A Frase “Estudante”
- (C) A Frase “Zeca da Silva Estudante”
- (D) A Frase “Zeca da Silva // Estudante”
- (E) A Frase “Zeca da Silva”

5. O comando echo na linguagem de programação PHP, tem qual função:

- (A) conectar a outro usuário php
- (B) mostrar a data atual do servidor.
- (C) mostrar dados na tela do usuário.
- (D) verificar se mensagem está correta.
- (E) verificar se codificação php está correta.



Não deixe também de assistir o video:



Fonte: Desenvolvedor Back-end: O que é Server-Side e Client-Side - (TekZoom, 2019).

Confira neste vídeo as reais diferenças entre Front-end e Back-end e descubra também o que é Server-Side e Client-Side. Disponível em <https://www.youtube.com/watch?v=bkDJe5UAvfM>. Acessado em 10/02/2020.

Links

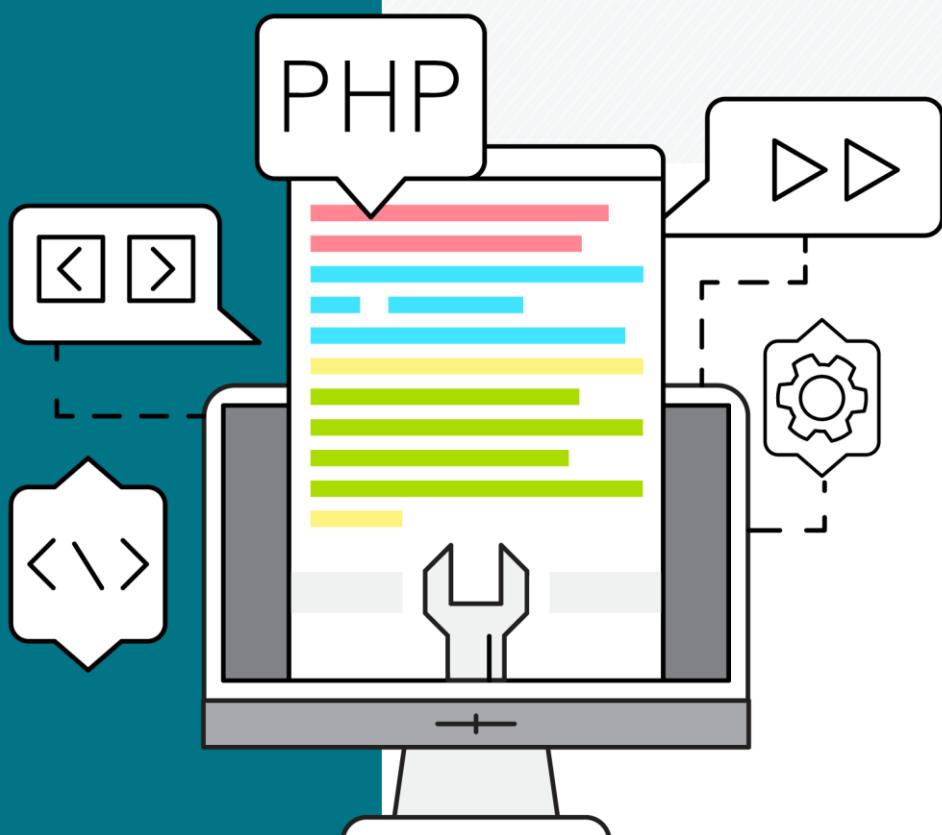
- CAELUM. “Introdução a PHP”. Acessado em 10/02/2020.
<https://www.caelum.com.br/apostila-html-css-javascript/appendice-introducao-a-php/>

Livros

- OGLIO, Pablo D. PHP Programando Com Orientacao a Objetos. 4ª Edição. São Paulo. Novatec. 2018. ISBN: 978-8575226919

AGENDA 2

PHP:
VARIÁVEIS E
MÉTODOS GET
E POST





Na linguagem PHP, também utilizamos variáveis, contudo, diferente de outras linguagens como C#, Java e C++ que precisamos indicar na declaração da variável um tipo (int, double, string), o PHP define o tipo de variável de forma dinâmica, em outras palavras, uma variável pode conter valores de diferentes tipos em diferentes momentos da execução dos comandos.

Então, basicamente, nas linguagens de programação citadas anteriormente, você em algum momento terá que declarar o tipo de dados que a variável recebe, essas linguagens são comumente chamadas de “linguagens fortemente tipadas”. No php a definição de um tipo em variável não existe, quando a variável recebe um valor, o PHP irá determinar automaticamente seu tipo em tempo de execução.

Alguns tipos de dados que a linguagem PHP suporta são:

- Inteiro
- Ponto flutuante
- String - Letras, números e caracteres especiais
- Booleanos

As variáveis no PHP são identificadas pelo caractere “\$” (cifrão). Então para criar uma variável basta colocar o cifrão seguido pelo nome da variável. O php é uma linguagem case-sensitive, ou seja, letras maiúsculas e minúsculas são diferentes, então os nomes das variáveis seguem as mesmas regras, como todos os outros códigos em PHP. Um nome de variável válido em php pode iniciar com uma letra ou underline(sublinhado), seguido de qualquer quantidade de letras, números ou sublinhados. Exemplos:

```
<?php
$nome = "Zeca da Silva";
$salario = 1000.50;
?>
```

Perceba que não indicamos em momento algum qual o tipo de variável, apenas atribuímos o valor na variável, e o php fica responsável pelo restante.

As variáveis numéricas não possuem nenhum mistério, sendo **Inteiro** - Um inteiro é qualquer número sem casas decimais, positivo ou negativo. **Ponto Flutuante** - Números de ponto flutuante, também conhecidos como "floats", "doubles" ou "números reais", números que apresentam fração. Exemplo: 1.0

Obs.: O php determina que o número deixa de ser inteiro para ser ponto fluente no momento em há a necessidade de uso da casa decimal, seguindo essa ideia se colocarmos o valor 1.0 em uma variável o Php a tornará em uma variável de ponto flutuante.

Exemplo:

```
<?php  
  
$valor = 1.0; // Variável depois da atribuição de valor passa a ser do tipo  
float.  
  
?>
```

Obs.: Outro fator muito importante que precisamos levar em consideração é saber que em PHP o separador de casas decimais também é o ponto, assim como em java.

String

Como já sabemos, trata-se do tipo que representa um conjunto de caracteres alfanuméricos (letras, números e caracteres especiais), portanto, é utilizado para armazenar e manipular textos. Para se atribuir valores a uma variável **String**, utilizam-se **aspas simples** ou **duplas**. Apesar de ambas as formas serem igualmente permitidas (aspas simples ou duplas), existem pequenas diferenças em seu uso.

Concatenação

Obs.: Para enviar à tela uma mensagem somando textos e conteúdos das variáveis, a concatenação é realizada por meio do **ponto final “.”**

Exemplo:

```
<?php
$nome = "Zeca da Silva";
$salario = 1000.50;
echo "Nome: ".$nome." Salario: ".$salario;
?>
```

Para entender melhor vamos programar!

No Visual Studio Code, crie um arquivo e o salve dentro da pasta Agenda 2 com o nome de “**desvendandoString**”, não esquecendo de escolher o tipo **PHP**. Veja a codificação deste exemplo:

```
<?php
$nome = "Zeca";
$email = 'zeca.silva@teste.com';

echo $nome."<br>";
echo $email."<br>";

?>
```

Após salvar e executar, perceberá que o resultado será como o representado na imagem 3, demonstrando que, nesse caso, o funcionamento de aspas simples ou duplas tem o mesmo resultado.

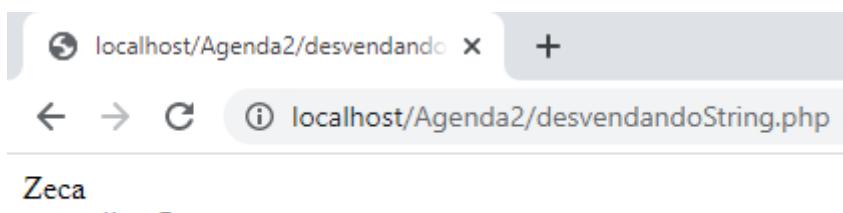


Imagen 4 – Resultado no navegador do arquivo desvendandoString.php.

Agora altere o código para:

```
<?php
$nome = "Zeca";
$email = 'zeca.silva@teste.com';

$texto1 = 'Olá $nome, seu email é: $email';
$texto2 = "Olá $nome, seu email é: $email";
echo$texto1."<br>";
echo$texto2."<br>";
?>
```

Após salvar e utilizar no navegador a mesma **URL**

(localhost/Agenda2/desvendandoString.php), o resultado será como o representado na Imagem 5.

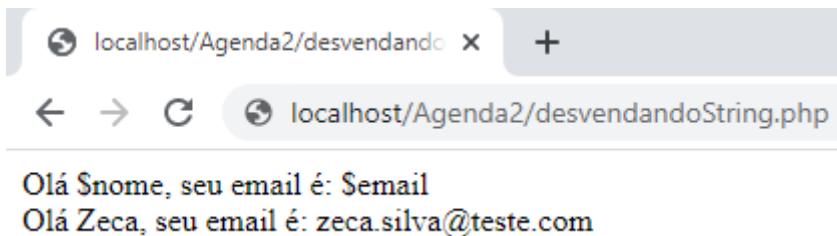


Imagen 5 – Resultado no navegador do arquivo desvendandoString.php após alterações.

Podemos notar que na primeira linha, na qual a string foi construída utilizando a aspa simples, o texto representado pelo navegador é literal:

“Olá \$nome, seu email é: \$email”

Mas quando são utilizadas aspas duplas para a construção da **string**, as variáveis retornam o valor dos seus conteúdos, deixando o resultado desta forma:

“Olá Zeca, seu email é: zeca.silva@teste.com”

Código HTML

Obs.: Nos exemplos utilizados para enviar a mensagem para o navegador podemos perceber que na concatenação do código `echo $nome."
"`; estamos enviando uma codificação **HTML**, então o navegador recebe a informação e interpretara os códigos HTML normalmente, gerando assim uma quebra de linha.

Booleanos

O tipo de dados booleano representa um valor lógico binário, ou seja, tem apenas dois estados possíveis, que pode ser VERDADEIRO ou FALSO, portanto, esses tipos de variáveis são amplamente utilizados para verificação e atribuição de condições. Em PHP, os valores armazenados nas variáveis são verdadeiros (TRUE) e falso (FALSE) representados pelas palavras reservadas TRUE e FALSE.

Mas isso é comum em, basicamente, todas as linguagens, o que muda no PHP? Vamos lá! Lembram da flexibilidade do PHP mencionada algumas vezes. O tipo booleano traz alguns exemplos:

- O valor 1 é considerado verdadeiro, enquanto o valor NULL é considerado falso.
- Valores numéricos diferentes de 0 (zero) são considerados como TRUE, enquanto o zero é considerado FALSE.
- Valores strings preenchidos são considerados TRUE, enquanto strings vazias ("") e o texto "0" é tido como FALSE.
- Arrays vazios ou objetos sem conteúdo são considerados como FALSE, já o inverso é logicamente TRUE.
- O valor NULL é considerado FALSE.



Utilizando o que foi visto até agora....

1. Crie um arquivo PHP na pasta Agenda 2.
 - a. Neste arquivo, crie 5 variáveis para armazenar respectivamente:
 - Nome Completo
 - Idade;
 - Profissão;
 - Salário.
 - b. Exiba, utilizando o Comando echo, os valores armazenados nas variáveis

A seguir, confira se você conseguiu resolver os desafios propostos!

```
<?php  
$nomeCompleto = "Jéssica";  
$idade = '17';  
$profissao = 'Estagiária';  
$salario = 750.00;  
  
echo "Nome Completo: $nomeCompleto<br>";  
echo "Idade: $idade<br>";  
echo "Profissão:". $profissao."<br>";  
echo "Salário: ". $salario."<br>";  
  
?>
```

Resultado no Navegador.

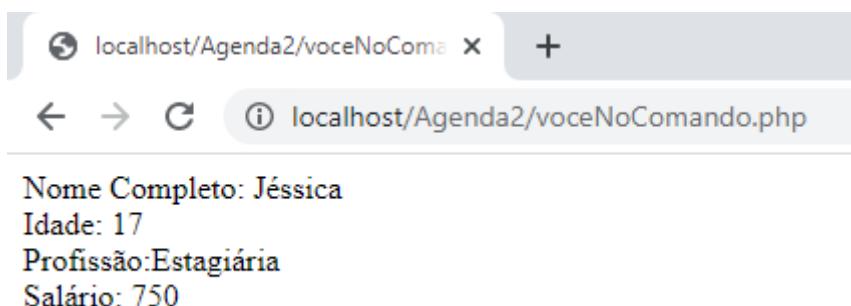


Imagen 6 – Resultado no navegador do exercício proposto.

Formulários, métodos GET e POST

Pense em formulários na web que você utiliza em seu cotidiano. Quando você preenche um formulário e clica no botão “enviar”, o formulário é submetido ao navegador e enviado para o servidor fazer esse processamento (**request**). Para que possamos trabalhar com as *requests* no servidor web, primeiro precisamos dessas requisições enviadas. O vídeo a seguir, explica de forma muito simplificada esse procedimento. Assista!



Fonte: Como funciona uma requisição HTTP? – (SPACE RAIL,2020). Disponível em <https://www.youtube.com/watch?v=fhAXgcD21iE>. Acessado em 22/02/2020.

Para entender ainda mais, vamos programar!

No Visual Studio Code, crie dois arquivos e os salve dentro da pasta **Agenda 2**. O primeiro com o nome de “**request**” e o segundo “**acao**”, não se esquecendo de escolher o tipo **PHP** para ambos os arquivos.

No arquivo *request*, vamos desenvolver um formulário, com um campo de texto e um botão para enviar o conteúdo. A codificação deve ser desenvolvida desta forma:

```

<!DOCTYPE html>
<html lang="pt-BR">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>GET e POST</title>
</head>
<body>
<form id="form1" name="form1" method="get" action="acao.php">
<br>
    Nome <input name="nome" type="text" id="nome" placeholder = "Digite o Nome">
<input name="Enviar" type="submit" id="enviar" value="Enviar">
</form>
</body>
</html>
  
```

A tag **form** contém um atributo **action** com o valor “**acao**”. Com essa configuração o arquivo “**acao.php**” será executado assim que for realizada a ação do clique no formulário. No outro atributo “**method**”, com o valor atribuído **GET**, indica que os dados serão transmitidos diretamente pela **URL** da página. Este método possui algumas limitações, destacando-se:

- **URL** não pode passar de **255 caracteres**, portanto, se você estiver trabalhando com variáveis muito extensas, não é aconselhável o uso esse método.

Como dito, o método **GET** utiliza a **URL** do site para **enviar as requisições**. Há um caractere que **indica o início da criação das variáveis** e outro caractere que **faz a separação entre as variáveis**. Você precisa sempre estar atento ao conceito de que toda variável criada tem o seu valor atribuído, mesmo que esse valor seja nulo.

Para continuarmos, vamos programar o **arquivo ação**, com o seguinte código:

```
<?php
$nome = $_GET['nome'];
echo "Seja bem-vinda!". $nome;
?>
```

Perceba que é utilizada uma variável super global padrão “**\$_GET[]**” e, entre os colchetes e as aspas simples, está escrito nome. Este valor corresponde ao conteúdo atributo “**name**” do input de texto criado no formulário do arquivo “**request.php**”. Desta maneira, a variável “nome” terá o valor que for digitado pelo usuário no campo de texto.

Ao testar, o resultado é apresentado na imagem a seguir:

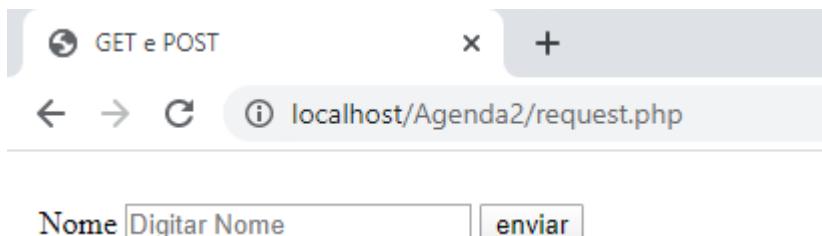


Imagen 7 –Resultado no navegador do arquivo request.php.

Após digitar um nome no campo de texto e clicar no botão enviar, o resultado deverá ser como o apresentado na imagem a seguir.

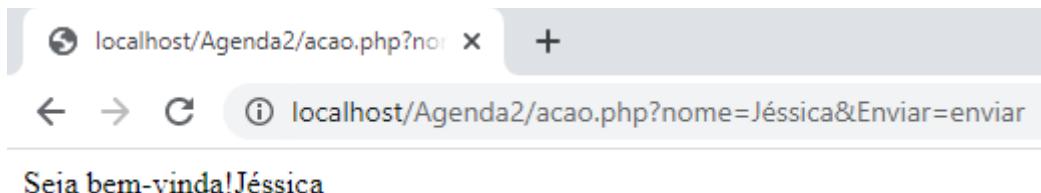


Imagen 8 – Resultado no navegador do arquivo acao.php. Apresenta a frase sejabem-vinda! Jéssica.

Perceba que na **URL** do seu navegador, após o nome do arquivo “**request.php**”, terá “**?nome=Jéssica**”, ou seja, os dados são transferidos, por meio da URL do próprio navegador, como é possível visualizar na imagem a seguir.



Imagen 9 – enviando dados via URL.

No método **POST**, a transferência de dados é realizada de forma oculta junto ao protocolo HTTP. Com este método teremos algumas vantagens como:

- Não há limite de tamanho dos dados que estão sendo enviados, ao contrário do que acontece com o método **GET**;
- Por meio do método **POST**, é possível enviar outros tipos de dados o que não é possível com o método get.

Para testar este método, basta alterar o arquivo “**request.php**”, alterando o valor “**get**” do atributo “**method**” pelo valor “**post**”, resultando no seguinte código:

```
<form id="form1" name="form1" method="post" action="acao.php">
```

Ao executar o código, você obterá o mesmo resultado, porém perceba pela imagem 9 que nenhuma informação será passada pela **URL**:

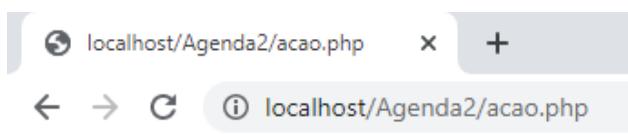


Imagen 10 – resultado do método Post



Utilizando o que foi visto até agora....

1 - Crie um arquivo PHP na pasta Agenda 2.

- a. Neste arquivo, crie um formulário com os campos:
 - Nome Completo
 - Idade;
 - Profissão;
 - Salário.

2 - Crie um arquivo PHP para receber a ação do botão enviar.

- b. Este deverá exibir no navegador uma informação em cada linha.
- c. Utilize o método que você achar melhor.

A seguir, confira se você conseguiu resolver os desafios propostos!

Arquivo Formulário

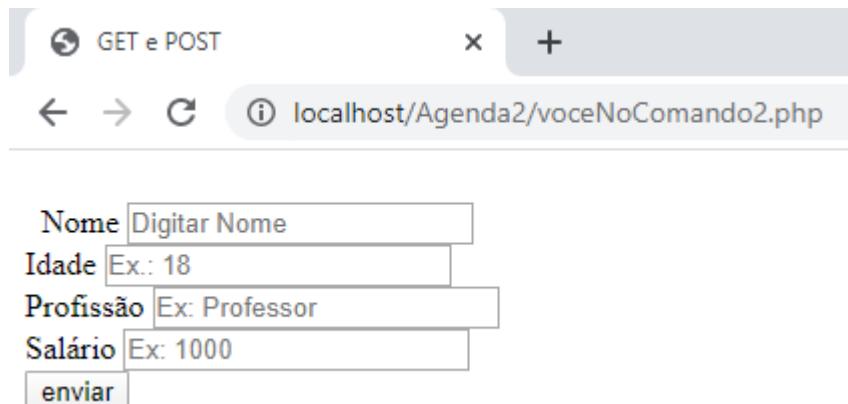
```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<meta http-equiv="X-UA-Compatible" content="ie=edge">
<title>GET e POST</title>
</head>
<body>
<form id="form1" name="form1" method="post" action="voceNoComando2A2.php">
<br>
    Nome <input name="nome" type="text" id="nome" placeholder = "Digite o seu nome"><br>
    Idade<input name="idade" type="text" id="idade" placeholder = "Ex.: 18"><br>
    Profissão<input name="profi" type="text" id="profi" placeholder = "Ex: Professor"><br>
    Salário<input name="sal" type="text" id="sal" placeholder = "Ex: 1000"><br>
    <input name="Enviar" type="submit" id="enviar" value="Enviar">
</form>
</body>
</html>
```

Arquivo de Ação

```
<?php  
echo "Nome: ".$_POST['nome']."<br>";  
echo "Idade: ".$_POST['idade']."<br>";  
echo "Profissão: ".$_POST['profi']."<br>";  
echo "Salário: R$ ".$_POST['sal']."<br>";  
?>
```

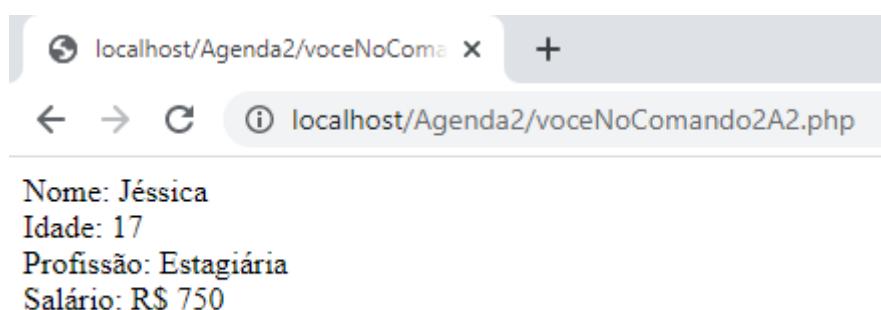
Resultado no Navegador.

Arquivo Formulário



The screenshot shows a browser window with the title 'GET e POST'. The address bar displays 'localhost/Agenda2/voceNoComando2.php'. Below the address bar is a form with four input fields and a submit button. The first field is labeled 'Nome' with placeholder text 'Digite Nome'. The second field is labeled 'Idade' with placeholder text 'Ex.: 18'. The third field is labeled 'Profissão' with placeholder text 'Ex: Professor'. The fourth field is labeled 'Salário' with placeholder text 'Ex: 1000'. Below the fields is a blue 'enviar' button.

Imagen 11 – Possível resultado no navegador do Exercício Você no Comando.

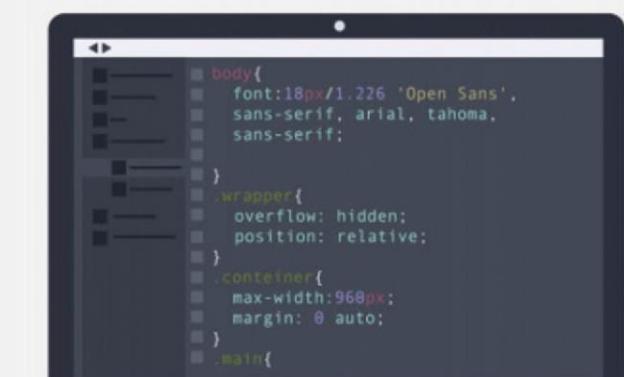


The screenshot shows a browser window with the title 'localhost/Agenda2/voceNoComando2A2.php'. The address bar displays 'localhost/Agenda2/voceNoComando2A2.php'. The main content area of the browser shows the output of the PHP code: 'Nome: Jéssica', 'Idade: 17', 'Profissão: Estagiária', and 'Salário: R\$ 750'.

Imagen 12 – Possível resultado no navegador do Exercício Você no Comando.

AGENDA 3

PHP:
ESTRUTURA DE
DECISÃO E
OPERADORES
GERAIS



```
body {  
    font:18px/1.226 'Open Sans',  
    sans-serif, arial, tahoma,  
    sans-serif;  
}  
.wrapper{  
    overflow: hidden;  
    position: relative;  
}  
.conteiner{  
    max-width:960px;  
    margin: 0 auto;  
}  
.main{
```

GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO WEB II

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



Antes de mergulharmos nas estruturas de decisão, precisamos destacar alguns operadores aritméticos e de atribuição que serão utilizados nas rotinas de decisão. Assim como em outras linguagens de programação, a php também trabalha com esses operadores. A seguir, veja as tabelas com os operadores e suas respectivas funções:

Aritméticos		De Atribuição	
+	Adição	=	Atribuição Simples
-	Subtração	+=	Com adição
*	Multiplicação	-=	Com subtração
/	Divisão	*=	Com multiplicação
%	Módulo (resto da divisão)	/=	Com divisão
		%=	Com módulo
		.=	concatenação

Obs.: Para a precedência matemática também são utilizados apenas os parênteses.

Desvios Condicionais

No PHP, o uso de desvios condicionais e sua sintaxe é muito semelhante às linguagens tipadas, como o Java.

Desvio Condicional Simples

Vamos iniciar com o desvio condicional simples.

```
<?php
// Desvio condicional Simples if(expressao)
{
    [instrucoes];
}

?>
```

O propósito da utilização do desvio condicional simples é realizar uma instrução ou um conjunto de instruções apenas se a condição for satisfeita. Antes de desenvolver essa estrutura e para ser possível a confecção das expressões, precisaremos dos operadores relacionais e dos operadores lógicos do PHP, dispostos nas tabelas a seguir.

Relacionais		Lógicos	
==	Igual a	AND &&	E
!=	Diferente	OR 	OU
>	Maior	!	Inversão (NOT)
>=	Maior ou igual	XOR	Ou exclusivo
<	Menor		
<=	Menor igual		

Imagine que o usuário de um site envie o seu pedido e , ao clicar em “enviar o pedido”, o site informa o nome do usuário e o valor total da compra; porém, neste site de compras há uma regra de negócio de que os fretes para a região Sudeste são gratuitos e deve aparecer apenas

para o usuário que escolher a região Sudeste. Neste caso, o uso do desvio condicional simples é a melhor opção, pois haverá um acréscimo de informação na mensagem somente quando o usuário escolher essa região. Para todas as demais regiões, nada será acrescentado.

Para melhor entendimento, vamos desenvolver um exemplo. Crie dois arquivos PHP, o primeiro denominado “**desvioSimples**” e o segundo “**desvioSimplesAction**”, dentro da pasta “**root**” do seu servidor.

Para o arquivo “**desvioSimples.php**” codifique da seguinte forma:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Desvio Condisional Simples</title>
</head>
<body>
    <div class="w3-container w3-teal">
        <h2>Enviar Pedido</h2>
    </div>
    <form class="w3-container" method="post" action="desvioSimplesAction.php">
        <label class="w3-text-teal"><b>Nome do Usuário</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtNome" type="text">
        <label class="w3-text-teal"><b>Valor total da Compra</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtValorTotal" type="number">
        <label class="w3-text-teal"><b>Escolha a Região:</b></label>
        <select class="w3-input w3-border w3-light-grey" id="regiao" name = "cmbRegiao">
            <option value="Centro-Oeste">Centro-Oeste</option>
            <option value="Nordeste">Nordeste</option>
            <option value="Norte">Norte</option>
            <option value="Sul">Sul</option>
            <option value="Sudeste" selected>Sudeste</option>
        </select>
        <br>
        <button class="w3-btn w3-blue-grey">Enviar</button>
    </form>
</body>
</html>
```

A codificação deve resultar em uma página semelhante a representada a seguir:

Imagem 3. Exemplo de desvio condicional simples "desvioSimples.php".

Com o formulário pronto, o próximo passo é desenvolver a codificação para o arquivo "**desvioSimplesAction.php**", este será executado após a ação do clique no botão enviar.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Mensagem</title>
</head>
<body>
<div class="w3-container w3-teal">
    <h1>
<?php
    echo "" . $_POST['txtNome'] . " ! <br>" ;           echo "Valor total
da Compra: R$ " . $_POST['txtValorTotal'] . "<br>";
    ?>
<?php
    if($_POST['cmbRegiao'] == "Sudeste")
    {
        echo "Neste mês estamos com frete grátis para
o SUDESTE";
    }
    ?>
</h1> </div>
</body>
</html>
```

Neste momento, o usuário terá um resultado na escolha da região sudeste, conforme apresentado pela imagem a seguir:



Imagen 4. Exemplo de desvio condicional simples “desvioSimples.php”.

Para todas as outras regiões será apresentada apenas a mensagem informando o valor total da compra:

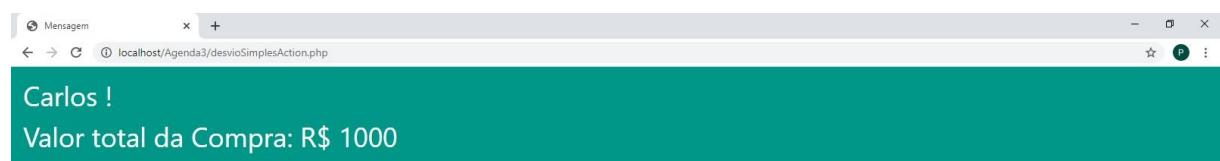


Imagen 5. Exemplo de desvio condicional simples “desvioSimples.php”.

Composto

A estrutura composta traz dois caminhos para o fluxo da página, ou seja, a execução de determinado trecho de código será diferente para quando o resultado da expressão for verdadeiro e para quando o resultado for falso. Segue a sintaxe:

```
// Desvio condicional composto

if(expressao)
{
    [instrucoes]; // Se verdadeiro
} else
{
    [instrucoes]; // Se falso
}
```

Para melhor entendimento, imagine um site para o professor lançar o nome de um aluno e suas três notas bimestrais, e que ao pressionar um botão gere dois possíveis resultados:

“Aprovado”, para a média maior ou igual a 7, ou
“Reprovado”, para média inferior a 7,

Então, para realizar o desenvolvimento deste exemplo, no Visual Studio Code, crie dois arquivos e salve-os dentro da pasta **root** do seu servidor php, sendo o primeiro com o nome de “**desvioComposto**”.

Este arquivo terá um formulário com campos para inserção do nome do aluno e suas respectivas três notas, além de um botão para calcular a média.

O código ficará dessa forma:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Desvio Condicional Composto</title>
</head>
<body>
    <div class="w3-container w3-teal">
        <h2>Calculo de Média e Resultado Final</h2>
    </div>
    <form class="w3container" method="post"
action="desvioCompostoAction.php">
        <label class="w3-text-teal"><b>Nome do Aluno</b></label>
        <input class="w3-input w3-border w3-lightgrey"
name="txtNome" type="text">
        <label class="w3-text-teal"><b>Nota 1</b></label>
        <input class="w3-input w3-border w3-lightgrey"
name="txtN1" type="number">
        <label class="w3-text-teal"><b>Nota 2</b></label>
        <input class="w3-input w3-border w3-lightgrey"
name="txtN2" type="number">
        <label class="w3-text-teal"><b>Nota 3</b></label>
        <input class="w3-input w3-border w3-lightgrey"
name="txtN3" type="number">
        <br>
        <button class="w3-btn w3-blue-grey">Calcular Média</button>
    </form>
</body>
</html>
```

A codificação deve resultar em uma página semelhante à representada a seguir:

The screenshot shows a web browser window with the title "Desvio Condicional Composto". The address bar displays "localhost/agenda3/desvioComposto.php". The main content area is titled "Calculo de Média e Resultado Final". It contains three input fields labeled "Nome do Aluno" and three input fields labeled "Nota 1". Below these fields is a button labeled "Calcular Média".

Imagen 6.Exemplo de desvio condicional simples “desvioComposto.php”.

O segundo arquivo terá o nome de “**desvioCompostoAction**” – Este arquivo receberá três informações, cada uma vindo de um campo do arquivo “**desvioCompostoAction.php**” que serão atribuídas em três variáveis: n1, n2 e n3. A partir dos valores dessas variáveis, será calculado a média. Por meio da estrutura de decisão, será verificado se o aluno foi aprovado ou não:

Se a média foi maior ou igual a 7, então o aluno foi “Aprovado”, senão, se a média menor que 7, então o aluno foi “Reprovado”.

O código resultará em:

```

<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Resultado Final</title>
</head>
<body>
<div class="w3-container w3-teal">
    <h1>
        <?php
            $n1 = $_POST['txtN1'];
            $n2 = $_POST['txtN2'];
            $n3 = $_POST['txtN3'];
            $media = ($n1+$n2+$n3)/3;
            $resultado;
            echo ".$_POST['txtNome']."'! Sua Média foi ".$media."!!! <br>";
        if($media >= 7)
        {
            $resultado = "Aprovado";
        }
        else
        {
            $resultado = "Reprovado";
        }
        echo "Resultado:
        ".$resultado."<br>";
        ?>
    </h1> </div>
</body>
</html>

```

Resultando em duas possibilidades: a primeira, Aprovado (imagem 6)

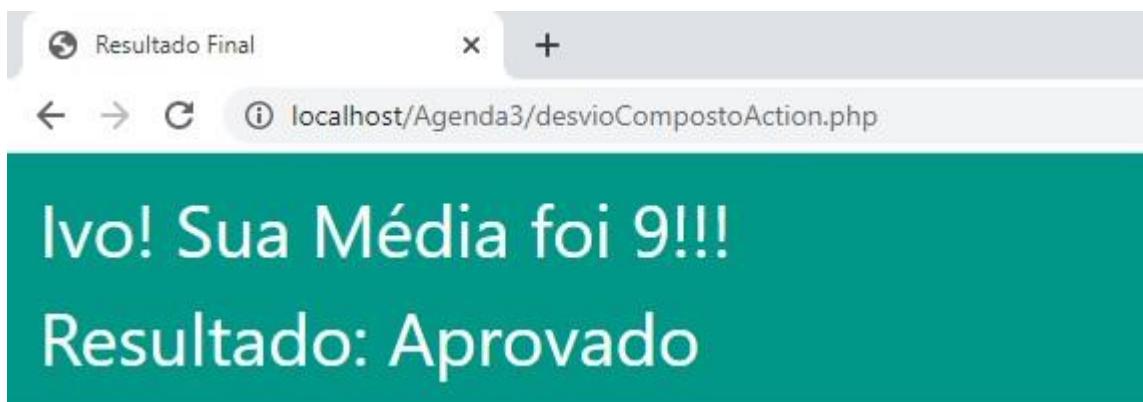


Imagen 6.Exemplo desvio condicional composto caso verdadeiro.

A segunda possibilidade é caso a condição seja falsa. (imagem 7)

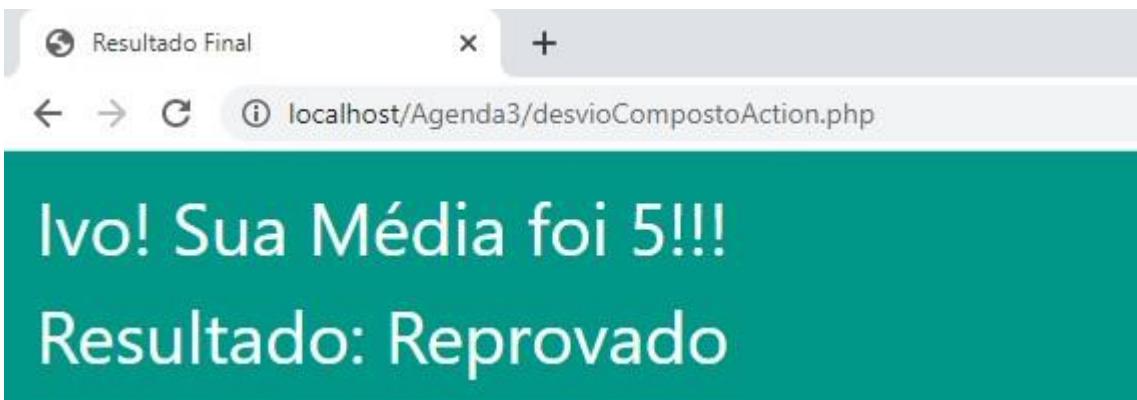


Imagen 7.Exemplo de desvio condicional composto caso falso.



Conclui-se então que o desvio composto oferece dois camindo de fluxo para o site!

Encadeamento de Decisões

Em diversas situações no desenvolvimento de um site, será necessário um conjunto de diferentes resultados possíveis, que dependerá de diversas decisões a serem tomadas para, então, chegar ao conjunto de instruções ideal. Sendo assim, será necessário o uso de um **encadeamento de decisões**.

Para exemplificar melhor, utilizando o exemplo da estrutura composta, será adicionada a possibilidade de o aluno ficar de exame, então:

- o aluno será aprovado se obtiver nota maior ou igual 7;
- o aluno será reprovado se obtiver nota menor que 5;
- o aluno estará em exame se obtiver nota maior ou igual a 5 e menor de 7**

Para isso, basta incrementar a estrutura de decisão composta, o que deve resultar em:

```
if($media >= 7)
{
    $resultado = "Aprovado";
}
else
{
    if($media <
5)
    {
        $resultado = "Reprovado";
    }
}
else
{
    $resultado = "Exame";
}
```

No código fica claro que apenas quando o resultado é falso na primeira condição ($\$média \geq 7$), será executada a próxima condição e, outro detalhe, é que o resultado exame somente ocorrerá caso as duas primeiras condições sejam falsas, então, obtemos uma estrutura Esse encadeamento pode ter várias configurações, não apenas, if...else...if..else, dependendo da necessidade de redirecionamento do seu site. Confira no quadro a seguir:

- elseif

Obs.: No php existe o comando elseif utilizado para quando o encadeamento de condições fica em uma cascata contínua, o resultado é praticamente o mesmo; porém, em uma codificação longa, seu uso pode deixar seu código mais limpo e elegante, diminuindo consideravelmente o número de chaves.

Exemplo:

```
if($media >= 7)
{
    $resultado = "Aprovado";
} elseif($media <
5)
{
    $resultado = "Reprovado";
} else
{
    $resultado = "Exame";
}
```

- switch.

Na linguagem de programação PHP, também é possível utilizar a estrutura switch. Para saber mais, assista ao vídeo a seguir:



Fonte: Estrutura Condicional Switch- GUSTAVO (CURSO EM VÍDEO, 2018).

Estruturas de condição de múltipla escolha em PHP. O Switch case em PHP usa a mesma sintaxe do Java e da Linguagem C e C++. Disponível em: <https://www.youtube.com/watch?v=thElQ5lhM1Q>. Acessado em 18/04/2019.

Operador Ternário

Como é possível notar, ao desenvolver projetos WEB ou em qualquer outra plataforma, é quase improvável não utilizar estruturas condicionais, sejam eles **if**, ou **switches**, porém, em vários pontos de desenvolvimento, essas condições serão tão simples que só terão duas possibilidades, uma para verdadeiro e outra para falso.

Por exemplo, em um site de e-commerce está sendo realizada uma promoção de aniversário que se o usuário comprar acima de R\$500,00, receberá 20% de desconto, caso contrário, ele receberá apenas 10%. A primeira solução seria utilizar um **if** e um **else** (estrutura composta) como, por exemplo:

The screenshot shows a web browser window with the following details:

- Address bar: Operador Ternário
- Address bar: localhost/Agenda3/ternario.php
- Title bar: Valor Final de Compra
- Form fields:
 - Nome do Usuário: An input field.
 - Valor total da Compra: An input field.
- Submit button: Enviar

Imagen 8 -.php

Como é possível notar na imagem 8, o exemplo terá duas passagens de valores, o nome do usuário e total da compra, com isso o resultado seria como na imagem a seguir:

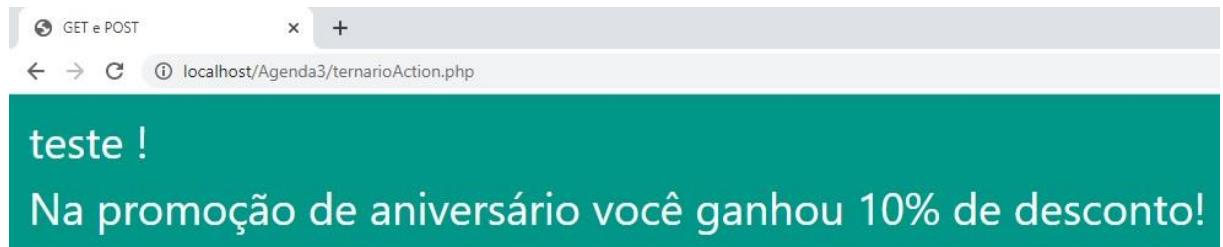


Imagen 9 -.php

E o código com a regra de negócio será:

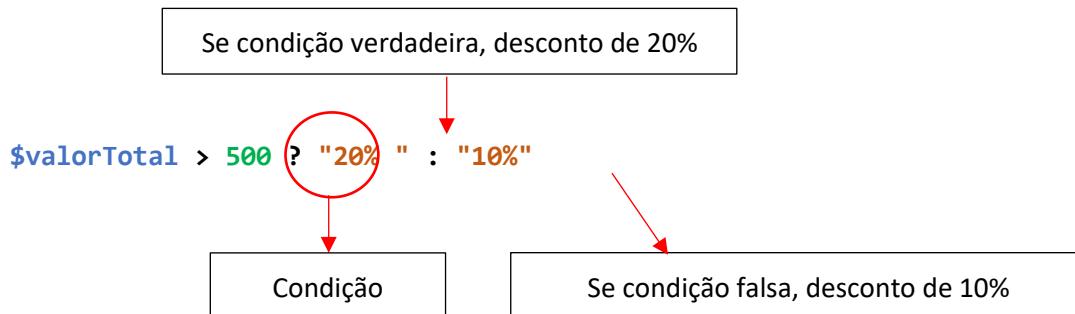
```
<div class="w3-container w3-teal">
    <h1>      <?php      echo
    "'.$_POST['txtNome'].'" ! <br>";
    ?>
    Na promoção de aniversário você ganhou
<?php
    $valorTotal = $_POST['txtValorTotal'];
if ($valorTotal > 500)
{
echo "20% ";
}
else
{
echo "10% ";
}
?> de desconto!
</h1>
</div>
```

Mas que o está sendo codificado é apenas um teste bem simples, com apenas uma linha de código dentro do **if** ou do **else**. Existe uma solução para simplificar e deixar o código bem mais elegante, porém, há uma diferença: o valor após o teste deve ser codificado estritamente em uma linha!

Veja o código a seguir:

```
<div class="w3-container w3-teal">
    <h1>
<?php
    echo "'.$_POST['txtNome'].'" ! <br>";
    ?>
    Na promoção de aniversário você ganhou
<?php
    $valorTotal = $_POST['txtValorTotal'];
echo $valorTotal > 500 ? "20%" : "10%";?>
de desconto!
</h1>
</div>
```

Portanto, inicialmente definimos um teste qualquer, retornando um valor booleano, depois, determinamos o primeiro parâmetro após o ponto de interrogação, que é o valor que deverá ser retornado caso o teste seja verdadeiro (retorno booleano “true”) e o segundo parâmetro, **após os dois pontos**, será retornado caso for falso (retorno booleano “false”).



Pelo fato dessa condição estar dividida em três operações, é dado o nome de **operador ternário**. Mas isso não é uma exclusividade PHP, ela também, está disponível em outras linguagens.

{ Obs.: Segue o código da página inicial para realização de testes }

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Operador Ternário</title>
</head>
<body>
    <div class="w3-container w3-teal">
        <h2>Valor Final de Compra</h2>
    </div>

    <form class="w3-container" method="post" action="ternarioAction.php">
        <label class="w3-text-teal"><b>Nome do Usuário</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtNome" type="text">

        <label class="w3-text-teal"><b>Valor total da Compra</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtValorTotal" type="number">
        <br>

        <button class="w3-btn w3-blue-grey">Enviar</button>
    </form>
</body>
</html>
```

**Utilizando o que foi visto até agora....**

1. Crie um arquivo PHP na pasta root ou Agenda3.
 - a. Neste arquivo, crie um formulário com os campos:
 - Nome;
 - Salário;
 - Quantidade de Dependentes;
2. Crie um arquivo PHP para receber a ação do botão enviar.
 - a. Este deverá calcular e exibir a % de aumento, valor do aumento e novo salário do funcionário, conforme a tabela a seguir:

Salário	Dependentes	% de Aumento
< = 500	< = 5	15%
	> 5	20%
> 500 e < = 1000	< = 5	10%
	> 5	15%
> 1000 e < = 2000	< = 5	10%
	> 5	12%
> 2000	< = 5	08%
	> 5	10%



A seguir, confira se você conseguiu resolver os desafios propostos!

Arquivo Formulário

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Você no Comando</title>
</head>
<body>
    <div class="w3-container w3-teal">
        <h2>Calculo de aumento</h2>
    </div>
    <form class="w3-container" method="post" action="voceNoComandoAction.php">
        <label class="w3-text-teal"><b>Nome Funcionário</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtNome" type="text">
        <label class="w3-text-teal"><b>Salário</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtSal" type="number">
        <label class="w3-text-teal"><b>Quantidade de Dependentes</b></label>
        <input class="w3-input w3-border w3-light-grey" name="txtDep" type="number">
        <button class="w3-btn w3-blue-grey">Calcular</button>
    </form>
</body>
</html>
```

Arquivo de Ação

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Resultado</title>
</head>
<body>
    <div class="w3-container w3-teal">
        <h1>
            <?php
                $sal = $_POST['txtSal'];
                $dep = $_POST['txtDep'];
                $por;
```

```
        if($sal <= 500 &&
$dep <=5)
{
    $por = 15;
}        elseif($sal <= 500
&& $dep > 5)
{
    $por = 20;
}        elseif($sal <= 1000
&& $dep <=5)
{
    $por = 10;
}        elseif($sal <= 1000
&& $dep > 5)
{
    $por = 15;
}        elseif($sal <= 2000
&& $dep <=5)
{
    $por = 10;
}        elseif($sal <= 2000
&& $dep > 5)
{
    $por = 12;
}
elseif($dep <=5)
{
    $por = 8;
}
else
{
    $por = 10;
}
echo
"".$_POST['txtNome']."! <br> ";
echo "Você terá ".$por."% de aumento, resultara no valor de R$ ".($por
* $sal / 100)."<br>";           echo "Seu novo Salário: R$ " .($sal +
$por * $sal / 100)."<br>";
?>
</h1> </div>
</body>
</html>
```

Resultado no Navegador

Arquivo Formulário

The screenshot shows a web browser window with the following details:

- Address bar: localhost/Agenda3/voceNoComando.php
- Page title: Cálculo de aumento
- Form fields:
 - Nome Funcionário: (empty input field)
 - Salário: (empty input field)
 - Quantidade de Dependentes: (empty input field)
- Buttons:
 - A blue "Calcular" button.

Imagen 10 –Possível resultado no navegador do Exercício Você no Comando.

Arquivo de Ação

The screenshot shows a web browser window with the following details:

- Address bar: localhost/Agenda3/voceNoComandoAction.php
- Page content:

Ivo!
Você terá 10% de aumento, resultara no valor de R\$ 200
Seu novo Salário: R\$ 2200

Imagen 11 –Possível resultado no navegador do Exercício Você no Comando.

AGENDA 4

**PHP:
ESTRUTURAS
DE REPETIÇÃO
E FUNÇÕES**



GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

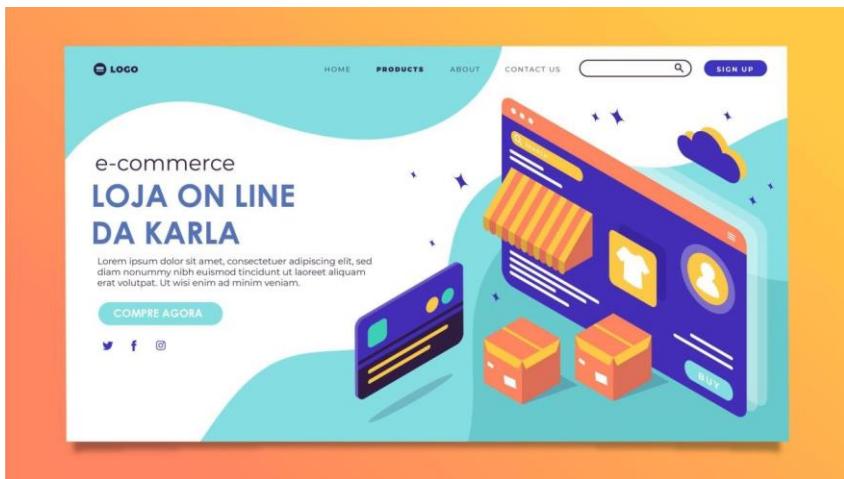


Imagen 3. Web site (FREEPIK)

Uma grande parte de páginas web necessitam de estruturas de repetição e funções no seu desenvolvimento. Para exemplificar, imagine um professor acessando o portal acadêmico de sua escola, para ter acesso às diversas turmas para as quais ministra aula. Ao clicar em um botão dentro dos vários disponíveis, é gerada uma página contendo uma tabela com as respectivas colunas, com o RM (registro de matrícula), Nome do Aluno, Idade e e-mail para contato.

Na sala que ele escolheu existem 30 alunos matriculados, então, a tabela deve possuir 31 tags <tr>, sendo a primeira para os títulos das colunas e as demais uma para cada aluno. Até aqui nada de mais; porém, ele clicou para gerar a mesma tabela em uma outra turma que possui 40 alunos matriculados, então, a tabela agora passa a ter 41 tags <tr>. Neste exemplo, é possível entender que será executada a mesma tabela, mas ela terá o tamanho diferente. Então, para automatizar e oferecer suporte a todas as composições de turmas, podemos criar a tabela a partir da quantidade de alunos na turma utilizando estruturas de repetição. Agora, as funções. Considere que você está desenvolvendo este mesmo portal e que, em uma página, você tenha que repetir uma determinada operação de matemática complexa diversas vezes no decorrer do código, esse conceito traz a possibilidade desta operação ser

implementada dentro de uma função que será chamada para realizar essa operação, quantas vezes for necessário.

Funções

Funções são blocos de código com um nome, que têm como objetivo executar uma tarefa específica e, por meio de seu nome ser invocada e, posteriormente executada, em diversas partes do código. Essa é uma das primeiras e mais utilizadas técnicas para reutilização de código.

A declaração de funções no PHP, como em diversas outras linguagens, é iniciada com a palavra reservada “function”. Logo após, é determinado seu nome e, por fim, se houver, um ou mais parâmetros. Para delimitar o que faz parte ou não da função, utilizam-se chaves. Todo o código que estiver alocado entre essas chaves será executado quando a função for chamada. Observe a sintaxe:

```
<?php  
function foo ($arg_1, $arg_2, /* ..., */ $arg_n)  
{  
    Instruções  
    return $valor_retornado;  
}  
?>
```

Obs.: O PHP não diferencia letras maiúsculas de minúsculas para o nome de funções, ou seja, você pode chamar a função teste() como: Teste(), TESTE() e todos serão reconhecidos como a mesma função.

As funções em php podem possuir retorno ou não, ou seja, o php possui funções void que apenas executam as instruções determinadas e não devolve nenhum valor como resultado e, também, possui funções com retorno, que ao serem executadas, devolvem um valor ao fim de sua execução.

Para melhor entendimento, no visual studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “exemploFuncao” - Neste arquivo, vamos criar um input para o usuário digitar um valor e, ao clicar no botão, ser informado se esse número é par ou ímpar.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Função com e sem retorno.</title>
</head>

<body>
    <div class=" w3-third w3-center w3-animate-top w3-padding">
        <h2 class="w3-container w3-teal " >PAR ou ÍMPAR</h2>
        <form class="w3-
container" method="post" action="exemploFuncaoAction.php">
            <label class="w3-text-teal">Digite Valor</label>
            <input class="w3-input w3-border w3-light-
grey" name="txtValor" type="number" placeholder="Digite ex: 6">
            <br>
            <button class="w3-btn w3-blue-
grey" name="btnCalcular">Verificar</button>
        </form>
    </div>
</body>
</html>
```

O que deve resultar em uma página semelhante à imagem a seguir.



Imagen 4. Imagem do navegador executando exemplo de Função.

Para o segundo arquivo, utilize o nome “exemploFuncaoAction.php” – nele vamos criar duas funções: uma com retorno e outra sem retorno, para observar suas diferenças e os resultados no navegador.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Função com e sem retorno.</title>
</head>
<body>
    <h2 class="w3-container w3-teal ">
        <?php

            function parImparRetorno($valor)
            {
                $resto = $valor % 2;
                if($resto == 0)
                    return "PAR";
                else
                    return "ÍMPAR";
            }
            function parImparVoid($valor)
            {
                $resto = $valor % 2;
                if($resto == 0)
                    echo "PAR";
                else
                    echo "ÍMPAR";
            }
        <?>
    </h2>
</body>
```

```
$t = $_POST['txtValor'];

//Chamada ou Invocação da Função.
echo parImparRetorno($t);
echo "<br>";
parImparVoid($t);

?>
</h2>
</body>
</html>
```

Resultado no Navegador.



Imagen 4.Resultado no navegador.

É possível notar que o resultado no navegador foi o mesmo para as duas chamadas de função; porém, na primeira, todas as instruções resultaram em uma string, que é retornada direto para um comando echo. Já a segunda função é desenvolvida sem retorno, utilizando o comando echo de forma direta dentro da própria função.

Durante o desenvolvimento surgirão situações em que será necessário o uso de cada uma das opções, portanto, saber utilizar as duas é de grande importância.

A seguir, tire mais algumas dúvidas assistindo ao vídeo.



Fonte: Curso de PHP 7 - Aula 31 - Como criar funções (Node Studio Treinamentos(2020).

Criando funções no PHP. Disponível em: <https://www.youtube.com/watch?v=loNQIP5BUEk>. Acessado em 15/05/2020

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES

Funções nativas

No php há uma lista enorme de funções nativas que podem ser visualizadas através do link:

https://www.php.net/manual/pt_BR/indexes.functions.php.

Essas funções são recursos disponíveis que auxiliarão o desenvolvimento dos mais diversos projetos.

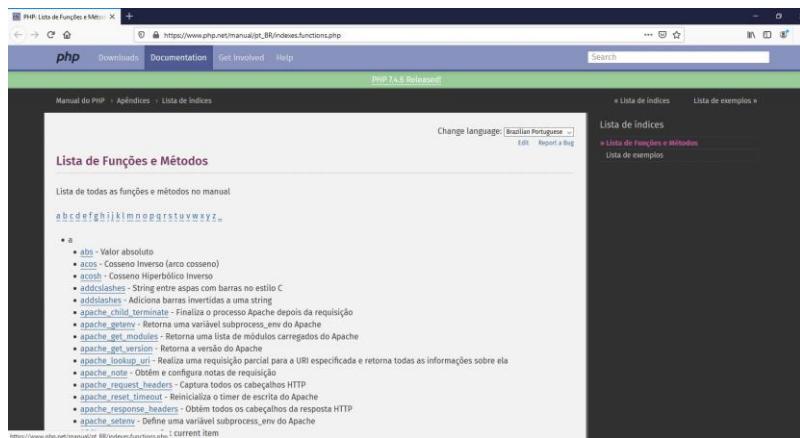


Imagen 5. Imagem da pagina do manual do php(PHP, 2020).

A imagem anterior mostra uma lista de funções, ao clicar em qualquer dos links, será carregada uma nova página expondo os conceitos, estruturas e uso da respectiva função. Porém, existe um detalhe que requer uma atenção especial localizado imediatamente abaixo do nome da função, essa informação é referente às versões do php que suportam essa função.

(PHP 4, PHP 5, PHP 7)

Imagen 6. Imagem de como estão expostas as versões suportadas pela versão do PHP

Vale a pena conferir os links a seguir que demonstram a utilização de algumas funções nativas.

DIEGO BROCANELLI– PHP e seu vasto arsenal de funções nativas. Acessado em 13/05/2020.

<https://www.diegobrocanelli.com.br/php/php-e-seu-vasto-arsenal-de-funcoes-nativas/>

E, também, tire mais algumas dúvidas assistindo ao vídeo.



Fonte: Cuso em Vídeo - Curso PHP Iniciante #16 - Funções String em PHP (Parte 1) (CURSO EM VIDEO, 2020).

Funções String em PHP (Parte 1) - Curso PHP Iniciante #16 - Gustavo Guanabara. Disponível em:

<https://www.youtube.com/watch?v=hQLyyll2Lwl>. Acessado em 15/05/2020.

Estruturas de Repetição

No PHP, as estruturas de repetição seguem o mesmo padrão de outras linguagens, são estruturas que oferecem recursos que permitem executar trechos de código repetidas vezes baseado no resultado de uma condição predeterminada.

A primeira dessas estruturas é a “For”, provavelmente a estrutura mais utilizada entre todas, em desenvolvimento web. Sua característica é possibilitar um loop de repetição com início e fim bem definidos, formando sua estrutura básica que é dividida em três expressões, como podemos ver no código a seguir:

Sintaxe:

FOR

```
For ([inicialização];[condição];[increm. ou decrem.]) {  
    [instrucoes];  
}
```

Normalmente, a primeira expressão é usada para definirmos variáveis de controle para o valor inicial e para a quantidade de repetições em um determinado bloco de código. A segunda expressão, por sua vez, é específica para definir a condição de repetição. Sendo true

(verdadeiro), o laço repetirá mais de uma vez todas as instruções, caso contrário, o laço será finalizado. Por fim, a terceira expressão define comandos que serão executados toda vez em que o bloco de comando seja executado.

Para melhor entendimento, no Visual Studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4. O primeiro com o nome de “exemploFor”. Neste arquivo, vamos criar um formulário com dez botões numerados de 0 a 10, para quando o usuário clicar em um botão, apareça a tabuada do respectivo número como resultado. Então codifique:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Estrutura FOR</title>
</head>

<body>
<div class=" w3-third w3-center w3-animate-top w3-padding">
    <h2 class="w3-container w3-teal">Escolha qual tabuada você deseja Visualizar</h2>
    <form class="w3-container" method="post" action="exemploForAction.php">
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn7">7</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn8">8</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn9">9</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn4">4</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn5">5</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn6">6</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn1">1</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn2">2</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn3">3</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn0">0</button>
    </form>
</div>
```

PHP: ESTRUTURAS DE REPETIÇÃO E FUNÇÕES

```
</form>
</div>
</body>
</html>
```

O que deve resultar em uma página semelhante à imagem a seguir.

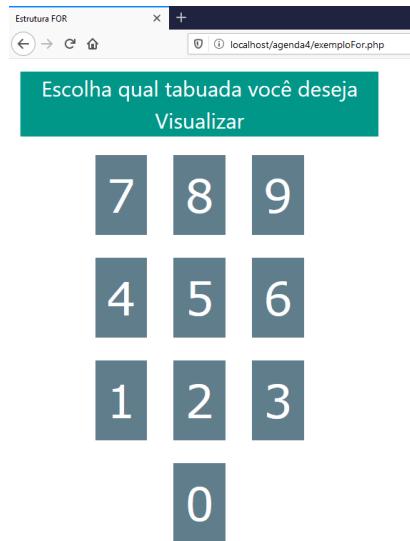


Imagen 7. Imagem do navegador executando exemplo para estrutura for.

Para o segundo arquivo, utilize o nome “exemploForAction.php” – Nele, vamos identificar qual botão foi pressionado pelo usuário, por meio da função nativa do php `isset` cuja utilização pode ser aplicada para verificar se determinado botão foi pressionado ou não, definindo, assim, qual tabuada deverá ser calculada. Por fim, de acordo com o código a seguir, utilizaremos a estrutura de repetição “for” para criar o resultado dentro de uma tabela.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Tabuada</title>
</head>
```

```

<body>

    <?php
        echo '<br><a href="exemploFor.php" class="w3-button w3-teal">Voltar</a><br>';
        $t = -1;
        if(isset($_POST["btn0"]))
            $t = 0;
        elseif(isset($_POST["btn1"]))
            $t = 1;
        elseif(isset($_POST["btn2"]))
            $t = 2;
        elseif(isset($_POST["btn3"]))
            $t = 3;
        elseif(isset($_POST["btn4"]))
            $t = 4;
        elseif(isset($_POST["btn5"]))
            $t = 5;
        elseif(isset($_POST["btn6"]))
            $t = 6;
        elseif(isset($_POST["btn7"]))
            $t = 7;
        elseif(isset($_POST["btn8"]))
            $t = 8;
        elseif(isset($_POST["btn9"]))
            $t = 9;
        echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';
        echo '<table class="w3-table-all w3-hoverable w3-text-black">';
        echo '<tr class="w3-teal ">';
        echo '<th class="w3-center"> Tabuada do '.$t.'</th>';
        echo '</tr>';
        for($i = 0; $i<=10; $i++)
        {
            echo '<tr>';
            echo '<td class="w3-center">'.$t.' X '.$i.' = '.$t*$i.'</td>';
            echo '</tr>';
        }
        echo '</table>';
        echo '</div>';
    ?>
</body>
</html>

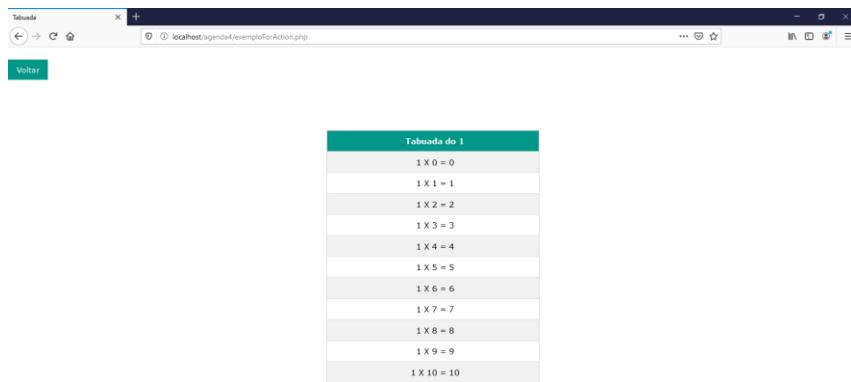
```

É possível notar que foi desenvolvido em comando echo de forma direta na tabela, bem como sua primeira linha, o que resultará em:

Tabuada do 1

Imagen 8. Primeira linha da tabela.

Neste momento, ao invés de fazer o mesmo procedimento para cada uma das respostas da tabuada, o uso da estrutura de repetição `for`, faz com que o processo do cálculo e da criação da tabela seja facilitado, pois, em seu código foi criado um laço de repetição que cria uma linha na tabela para cada cálculo da tabuada, resultando em:



A screenshot of a web browser window titled "Tabuada". The address bar shows "localhost/agenda4/exemploForAction.php". The main content area displays a table titled "Tabuada do 1" with 11 rows. Each row contains the multiplication expression "1 X [number] = [result]" where [number] and [result] are integers from 0 to 10 respectively. The table has alternating light gray and white rows for readability.

Tabuada do 1	
1 X 0 = 0	
1 X 1 = 1	
1 X 2 = 2	
1 X 3 = 3	
1 X 4 = 4	
1 X 5 = 5	
1 X 6 = 6	
1 X 7 = 7	
1 X 8 = 8	
1 X 9 = 9	
1 X 10 = 10	

Imagen 9. Resultado no navegador para quando o usuário escolher o botão com o número 1.

While

A estrutura de repetição `while`, talvez a estrutura mais simples, usa apenas expressão booleana (condição) no início de sua sintaxe. Veja:

WHILE

```
While (expressão) {
    [instrucoes];
}
```

Essa instrução expressa que enquanto (while) a condição for verdadeira o bloco de comandos será executado, até que em um determinado momento a condição não seja satisfeita (falsa), finalizando, assim, sua execução.

Para melhor entendimento, no Visual Studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “estruturaWhile” - Neste arquivo, vamos criar um formulário com um input de texto e um botão para que o usuário digite um valor e, quando clicar no botão, obtenha como resultado a tabuada do número digitado. Então codifique:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Form para tabuada</title>
</head>

<body>
    <div class=" w3-third w3-center w3-animate-top w3-padding">
        <h2 class="w3-container w3-
teal ">Digite qualquer valor para gerar sua tabuada</h2>
        <form class="w3-
container" method="post" action="exemploWhileAction.php">
            <label class="w3-text-
teal "><b>Valor para cálculo da tabuada</b></label>
            <input class="w3-input w3-border w3-light-
grey" name="txtValor" type="number" placeholder="Digite ex: 6">
            <br>
            <button class="w3-btn w3-blue-
grey" name="btnCalcular">Calcular</button>
        </form>
    </div>
</body>
</html>
```

O que deve resultar em:

Digite qualquer valor para gerar sua tabuada

Valor para cálculo da tabuada

Digite ex: 6

Calcular

Imagen 10. Resultado no navegador para o arquivo exemploWhile.php.

Para o segundo arquivo, utilize o nome “exemploWhileAction.php” – Neste arquivo, vamos obter o valor digitado pelo usuário no input por meio da variável global “POST”, este valor determinará qual tabuada será calculada e, por fim, a utilização da estrutura de repetição “while” para criar o resultado dentro de uma tabela de acordo com o código a seguir.

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Tabuada</title>
</head>
<body>

    <?php
        echo '<br><a href="exemploWhile.php" class="w3-button w3-teal">Voltar</a><br>';
        $v = $_POST["txtValor"];
        echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';
        echo '<table class="w3-table-all w3-hoverable w3-text-black">';
        echo '<tr class="w3-teal ">';
        echo '<th class="w3-center"> Tabuada do '.$v.'</th>';
        echo '</tr>';
        $i = 0;
```

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES

```
while($i<=10)
{
    echo '<tr>';
    echo '<td class="w3-center">'.$v.' X '.$i.' = '.$v*$i.'</td>';
    echo '</tr>';
    $i++;
}
echo '</table>';
echo '</div>';

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

Comentado [LSdN1]: https://www.w3schools.com/tags/tag_tr.asp

Perceba que também foi utilizada a estrutura de repetição para que cada uma das respostas da tabuada apareça dentro de uma linha da tabela, gerando um resultado para o usuário idêntico ao anterior.

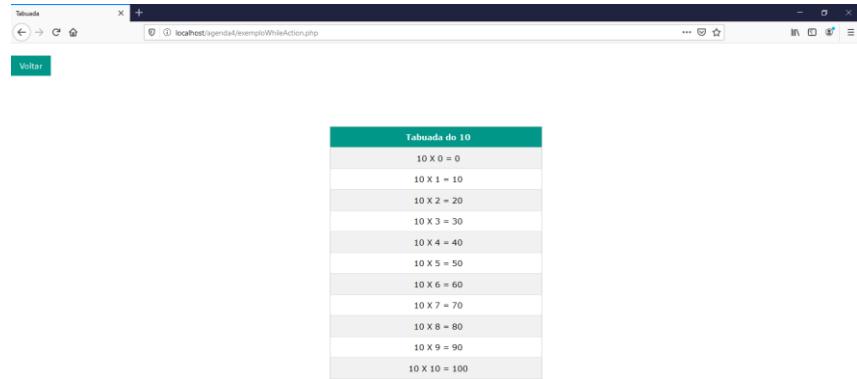


Imagen 11.Resultado no navegador para quando o usuário digitar o número 10.

DO-WHILE

A estrutura de repetição do-while é provável que seja a última que diz respeito à utilização em desenvolvimento web. Esta estrutura possui uma característica de que a primeira iteração é sempre executada, pois sua expressão booleana (condição) é realizada apenas no fim da estrutura. Após a expressão booleana ser executada, caso o resultado seja verdadeiro, será realizado novamente o bloco de instruções, caso contrário, a execução será finalizada. Segue a sintaxe:

DO..WHILE

```
do {  
    [instrucoes];  
} while (expressão);
```

Para melhor entendimento, no visual studio Code, crie dois novos arquivos e os salve dentro da pasta root ou Agenda4, o primeiro com o nome de “estruturasDoWhile” - nele, vamos criar um formulário com apenas um botão para que o usuário, ao clicar gere na sua página todas as tabuadas do 0 ao 10. Para isso codifique:

```
<!DOCTYPE html>  
<html lang="pt-BR">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <meta http-equiv="X-UA-Compatible" content="ie=edge">  
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">  
    <title>Do-WHILE</title>  
</head>  
  
<body>  
    <div class=" w3-third w3-center w3-animate-top w3-padding">  
        <h2 class="w3-container w3-teal">Clique para Gerar todas as Tabuadas do 0 a 10</h2>  
        <form class="w3-container" method="post" action="exemploDoWhileAction.php">  
            <br>  
            <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btnGerar">Tabuadas</button>  
        </form>  
    </div>  
</body>  
</html>
```

O que deve resultar em:



Imagen 12.Resultado no navegador para o arquivo exemplo doWhile.php.

Para o segundo arquivo, utilize o nome “exemploDoWhileAction.php” – Neste arquivo, serão executadas duas estruturas de repetição aninhadas: a primeira será responsável por criar cada uma das 11 tabelas e, a segunda, será responsável por cada uma das 11 linhas de cada uma das tabelas. O que resultará no código a seguir:

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Tabuada</title>
</head>
<body>
    <?php
        echo '<br><a href="exemploDoWhile.php" class="w3-button w3-teal">Voltar</a><br>';
        $i = 0;
        $j = 0;
        do{
            echo '<div class="w3-quarter w3-responsive w3-teal">';
            echo '<table class="w3-table-all w3-hoverable w3-text-black">';
            echo '<tr class="w3-teal ">';
            echo '<th class="w3-center"> Tabuada do '.$j.'</th>';
            echo '</tr>';
            $i = 0;
            do{
                echo '<tr>';
                echo '<td class="w3-center">'.$j.' X '.$i.' = '.$j*$i.'</td>';
                echo '</tr>';
                $i++;
            }while($i<=10);
            echo '</table>';
            echo '</div>';
            $j++;
        }
    </?php>
```

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES

```
    }while($j<=10);
?
</body>
</html>
```

Perceba que o alinhamento das duas estruturas faz com que o bloco de instruções da segunda estrutura seja executado 10 vezes para cada execução da primeira estrutura. O resultado no navegador dever ser semelhante à imagem a seguir.

Tabuada do 0		Tabuada do 1		Tabuada do 2		Tabuada do 3	
0 X 0 = 0		1 X 0 = 0		2 X 0 = 0		3 X 0 = 0	
0 X 1 = 0		1 X 1 = 1		2 X 1 = 2		3 X 1 = 3	
0 X 2 = 0		1 X 2 = 2		2 X 2 = 4		3 X 2 = 6	
0 X 3 = 0		1 X 3 = 3		2 X 3 = 6		3 X 3 = 9	
0 X 4 = 0		1 X 4 = 4		2 X 4 = 8		3 X 4 = 12	
0 X 5 = 0		1 X 5 = 5		2 X 5 = 10		3 X 5 = 15	
0 X 6 = 0		1 X 6 = 6		2 X 6 = 12		3 X 6 = 18	
0 X 7 = 0		1 X 7 = 7		2 X 7 = 14		3 X 7 = 21	
0 X 8 = 0		1 X 8 = 8		2 X 8 = 16		3 X 8 = 24	
0 X 9 = 0		1 X 9 = 9		2 X 9 = 18		3 X 9 = 27	
0 X 10 = 0		1 X 10 = 10		2 X 10 = 20		3 X 10 = 30	
Tabuada do 4		Tabuada do 5		Tabuada do 6		Tabuada do 7	
4 X 0 = 0		5 X 0 = 0		6 X 0 = 0		7 X 0 = 0	
4 X 1 = 4		5 X 1 = 5		6 X 1 = 6		7 X 1 = 7	
4 X 2 = 8		5 X 2 = 10		6 X 2 = 12		7 X 2 = 14	
4 X 3 = 12		5 X 3 = 15		6 X 3 = 18		7 X 3 = 21	
4 X 4 = 16		5 X 4 = 20		6 X 4 = 24		7 X 4 = 28	
4 X 5 = 20		5 X 5 = 25		6 X 5 = 30		7 X 5 = 35	
4 X 6 = 24		5 X 6 = 30		6 X 6 = 36		7 X 6 = 42	

Imagen 13.Resultado no navegador para quando o usuário clicar no botão Gerar Tabuadas.

FOREACH

Como o laço de repetição foreach foi desenvolvido para gerar iterações para o uso em arrays e objetos no PHP, assunto ainda não abordado em php; seu uso ficará condicionado a esses tipos de dados. Para saber mais assista ao vídeo abaixo.



Fonte: *Curso de PHP #10 - Loop FOREACH (CFBCURSOS, 2020)*.

Estrutura de repetição FOREACH bastante interessante, o loop foreach. Disponível em:
<https://www.youtube.com/watch?v=fD0wKp6Cam0>. Acessado em 15/05/2020.

Obs.: Comando break em laço de repetição.

O comando break em uma estrutura de repetição tem a função de parar a execução de laços de repetição. Sua utilização é muito simples, como podemos ver no exemplo a seguir:

Exemplo:

```
<?php  
  
$i = 0;  
  
while (true) {//Loop infinito.  
  
    if ($i > 10) {  
        break;  
    }  
    echo $i."<br>";  
    $i++;  
}  
  
?>
```

O resultado é a parada do loop assim que o \$i for maior do que 4.

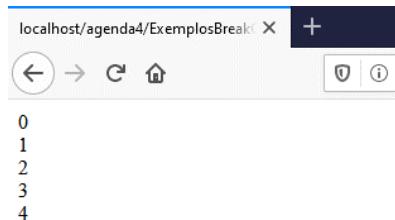


Imagen 14.Resultado no navegador para o uso do comando break.

Obs.: Comando continue em laço de repetição.

O comando continue tem uma função diferente, ele pula a iteração na qual foi executado, passando para a próxima iteração do laço, continuando assim o loop. Seu uso também é bem simples:

Exemplo:

```
<?php  
  
$i = 0;  
  
for($i = 0; $i <=10; $i++)  
{  
    if ($i == 5) {  
        continue;  
    }  
    echo $i."<br>";  
}  
  
?>
```

Observando a imagem a seguir, percebemos que o número 5 não é exibido, então, concluímos que a iteração que mostraria o número 5, foi ignorada pela utilização do comando continue.



localhost/agenda4/ExemplosBreak

0
1
2
3
4
6
7
8
9
10

Imagem 15. Resultado no navegador para o uso do comando continue.



Utilizando o que foi visto até agora, vamos criar uma página completa para mãe de Karla.

1- Crie um arquivo PHP na pasta root ou Agenda4.

- a) Crie uma divisão e coloque a data atual centralizada na Página.
- b) Divida a página em três:
 - Parte 1: Clicar em um botão para gerar a Tabuada respectiva ao número do botão;
 - Parte 2: Digitar o valor e clicar para gerar a tabuada do valor digitado;
 - Parte 3: Clicar em um botão para gerar todas as tabuadas;

2- Crie um arquivo PHP para receber todas as ações possíveis do arquivo anterior e exibir as tabuadas requeridas.

Dicas:

- Procure pela função nativa date().
- Utilize o exemplo de cada estrutura de repetição
- Utilize o comando isset para verificar qual botão foi acionado.

A seguir, confira se você conseguiu resolver os desafios propostos!

Codificação ArquivoFormulário

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Form para tabuada</title>
</head>
<body>
    <div>
        <h2 class="w3-container w3-teal w3-center">Data de Hoje: <?php echo date("d/m/Y");?></h2>
    </div>
    <div>
        <div class=" w3-third w3-center w3-animate-top w3-padding">
```

```
<h2 class="w3-container w3-teal">Escolha qual tabuada você deseja Visualizar</h2>
    <form class="w3-container" method="post" action="voceNoComandoAction.php">
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn7">7</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn8">8</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn9">9</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn4">4</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn5">5</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn6">6</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn1">1</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn2">2</button>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn3">3</button>
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btn0">0</button>
    </form>
</div>
<div class=" w3-third w3-center w3-animate-top w3-padding">
    <h2 class="w3-container w3-teal">Digite qualquer valor para gerar sua tabuada</h2>
    <form class="w3-container" method="post" action="voceNoComandoAction.php">
        <label class="w3-text">
            <b>Valor para cálculo da tabuada</b></label>
            <input class="w3-input w3-border w3-light-grey" name="txtValor" type="number" placeholder="Digite ex: 6">
        <br>
        <button class="w3-btn w3-blue-grey" name="btnCalcular">Calcular</button>
    </form>
</div>
<div class=" w3-third w3-center w3-animate-top w3-padding">
    <h2 class="w3-container w3-teal">Clique para Gerar todas as Tabuadas do 0 a 10</h2>
    <form class="w3-container" method="post" action="voceNoComandoAction.php">
        <br>
        <button class="w3-btn w3-blue-grey w3-margin w3-jumbo" name="btnGerar">Tabuadas</button>
    </form>
</div>
```

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

Resultado no Navegador.

Arquivo Formulário

The screenshot shows a web page titled "Form para tabuada". It displays the current date as "Data Hoje: 15/05/2020". There are three main buttons: "Visualizar", "Gerar", and "Clique para Gerar todas as Tabuadas do 0 a 10". Below these buttons is a grid of buttons for selecting numbers from 0 to 9. At the bottom right of the grid is a large button labeled "Tabuadas".

Imagen 16 –Possível resultado no navegador do Exercício Você no Comando.

Codificação Arquivo de Ação

```
<!DOCTYPE html>
<html lang="pt-BR">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Mensagem</title>
</head>
<body>

    <?php
        echo '<br><a href="voceNoComando.php" class="w3-button w3-teal">Voltar</a><br>';
        if(isset($_POST["btnCalcular"]))
        {
            $v = $_POST["txtValor"];
            echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';
            echo '<table class="w3-table-all w3-hoverable w3-text-black">';
            echo '  <tr class="w3-teal">';
            echo '    <th class="w3-center"> Tabuada do '.$v.'</th>';
            echo '  </tr>';

        }
    </?php
```

```
$i = 0;
while($i<=10)
{
    echo '<tr>';
    echo '<td class="w3-center">'.$v.' X '.$i.' = '.$v*$i.'</td>';
    echo '</tr>';
    $i++;
}
echo '</table>';
echo '</div>';
}
else
{
    if(isset($_POST["btnGerar"]))
    {
        $i = 0;
        $j = 0;
        do{
            echo '<div class="w3-quarter w3-responsive w3-teal">';
            echo '<table class="w3-table-all    w3-hoverable w3-text-
black">';
            echo '<tr class="w3-teal ">';
            echo '<th class="w3-center"> Tabuada do '.$j.'</th>';
            echo '</tr>';
            $i = 0;
            do{
                echo '<tr>';
                echo '<td class="w3-
center">'.$j.' X '.$i.' = '.$j*$i.'</td>';
                echo '</tr>';
                $i++;
            }while($i<=10);
            echo '</table>';
            echo '</div>';
            $j++;
        }while($j<=10);
    }
    else
    {
        $t = -1;
        if(isset($_POST["btn0"]))
            $t = 0;
        elseif(isset($_POST["btn1"]))
            $t = 1;
        elseif(isset($_POST["btn2"]))
            $t = 2;
        elseif(isset($_POST["btn3"]))
            $t = 3;
        elseif(isset($_POST["btn4"]))
            $t = 4;
        elseif(isset($_POST["btn5"]))
            $t = 5;
        elseif(isset($_POST["btn6"]))
            $t = 6;
    }
}
```

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES

```
$t = 6;
elseif(isset($_POST["btn7"]))
    $t = 7;
elseif(isset($_POST["btn8"]))
    $t = 8;
elseif(isset($_POST["btn9"]))
    $t = 9;
echo '<div class="w3-quarter w3-display-middle w3-responsive w3-teal">';
echo '<table class="w3-table-all    w3-hoverable w3-text-black">';
echo '<tr class="w3-teal ">';
echo '<th class="w3-center"> Tabuada do '.$t.'</th>';
echo '</tr>';
for($i = 0; $i<=10; $i++)
{
    echo '<tr>';
    echo '<td class="w3-center">'.$t.' X '.$i.' = '.$t*$i.'</td>';
    echo '</tr>';
}
echo '</table>';
echo '</div>';
}

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

Resultado no Navegador.

Parte 1 e Parte 2

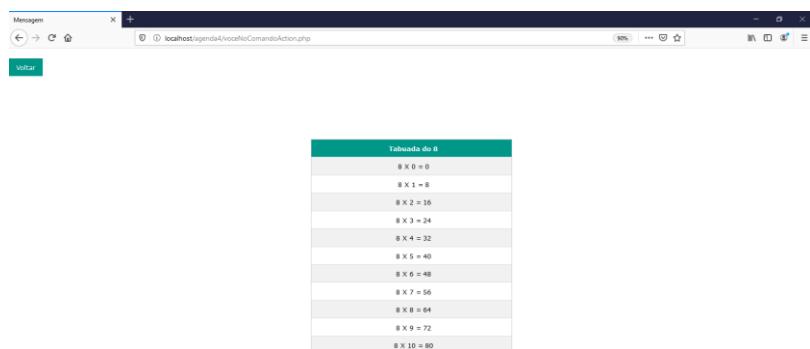


Imagen 17. Possível resultado no navegador do Exercício Você no Comando.

PHP: ESTUTURAS DE REPETIÇÃO E FUNÇÕES

Parte 3

Tabuada do 0	Tabuada do 1	Tabuada do 2	Tabuada do 3
0 X 0 = 0	1 X 0 = 0	2 X 0 = 0	3 X 0 = 0
0 X 1 = 0	1 X 1 = 1	2 X 1 = 2	3 X 1 = 3
0 X 2 = 0	1 X 2 = 2	2 X 2 = 4	3 X 2 = 6
0 X 3 = 0	1 X 3 = 3	2 X 3 = 6	3 X 3 = 9
0 X 4 = 0	1 X 4 = 4	2 X 4 = 8	3 X 4 = 12
0 X 5 = 0	1 X 5 = 5	2 X 5 = 10	3 X 5 = 15
0 X 6 = 0	1 X 6 = 6	2 X 6 = 12	3 X 6 = 18
0 X 7 = 0	1 X 7 = 7	2 X 7 = 14	3 X 7 = 21
0 X 8 = 0	1 X 8 = 8	2 X 8 = 16	3 X 8 = 24
0 X 9 = 0	1 X 9 = 9	2 X 9 = 18	3 X 9 = 27
0 X 10 = 0	1 X 10 = 10	2 X 10 = 20	3 X 10 = 30
Tabuada do 4	Tabuada do 5	Tabuada do 6	Tabuada do 7
4 X 0 = 0	5 X 0 = 0	6 X 0 = 0	7 X 0 = 0
4 X 1 = 4	5 X 1 = 5	6 X 1 = 6	7 X 1 = 7
4 X 2 = 8	5 X 2 = 10	6 X 2 = 12	7 X 2 = 14
4 X 3 = 12	5 X 3 = 15	6 X 3 = 18	7 X 3 = 21
4 X 4 = 16	5 X 4 = 20	6 X 4 = 24	7 X 4 = 28
4 X 5 = 20	5 X 5 = 25	6 X 5 = 30	7 X 5 = 35
4 X 6 = 24	5 X 6 = 30	6 X 6 = 36	7 X 6 = 42

Imagem 18. Possível resultado no navegador do Exercício Você no Comando.

AGENDA 5

PHP - ARRAYS





Arrays

A definição correta para um array em PHP é de um mapa ordenado, ou seja, trata-se de um tipo de dado, assim como integer, float, string ou boolean. Porém, ele pode armazenar mais de um valor, relacionando-os a suas chaves. Simplificando, em um contexto geral, podemos dizer que:

Array é uma variável do php que nos fornece a possibilidade de atribuir diversos valores ao mesmo tempo.

É possível relacionar o array ao conceito e utilização de vetor e matriz (por meio do uso arrays multidimensionais). Como essa variável possui um identificador, torna-se capaz de armazenar mais de um valor na mesma estrutura, e de acordo com o índice é possível gerenciar esses conteúdos (valores). Então, cada array está associado a um índice, que indica a posição de armazenamento de um valor na memória do array.

Obs.: No PHP esse índice pode ser tanto um texto, quanto um número, e seu uso sempre está delimitado entre colchete ([]).

Declarando arrays

Para melhor entendimento vamos programar, então, no visual studio Code.

Crie um arquivo e o salve dentro da pasta root ou Agenda5, com o nome de “decArray.php” – Neste arquivo vamos criar o primeiro array, utilizando o método construtor da linguagem array(). Então codique:

```
<?php
    $sudeste = array("São Paulo", "Minas Gerais", "Rio de Janeiro", "Espírito Santo");
?

```

Para a declaração utilizamos uma variável normal, atribuindo o resultado da função array para cada valor informado, resultando em um índice para ser acessado que pode ser representado graficamente pela imagem a seguir.

0	1	2	3
São Paulo	Minas Gerais	Rio de Janeiro	Espírito Santo

Imagem 3. Representação gráfica dos valores ordenados em um array.

Podemos perceber pela representação gráfica (imagem 3), que os índices são gerados automaticamente, iniciando sempre em 0.

Obs.: É possível também realizar a declaração do array utilizando colchetes. Para o PHP, essas duas formas são iguais, conforme código a seguir:

```
<?php
    $sudeste = ["São Paulo", "Minas Gerais", "Rio de Janeiro", "Espírito Santo"
"];
?>
```

Outras maneiras para declarar o mesmo array seriam:

Diretamente:

```
$sudeste[] = "São Paulo";
$sudeste[] = "Minas Gerais";
$sudeste[] = "Rio de Janeiro";
$sudeste[] = "Espírito Santo";
```

Ou ainda você pode definir os índices, mesmo que fora de ordem:

```
$sudeste[1] = "Minas Gerais";
$sudeste[0] = "São Paulo";
$sudeste[2] = "Rio de Janeiro";
$sudeste[3] = "Espírito Santo";
```

Para acessar os valores do array, basta utilizar seu nome seguido do índice entre colchetes, como exemplo a seguir:

```
<?php
    echo $sudeste[0];
?>
```

O resultado no navegador para todos os exemplos anteriores seria:

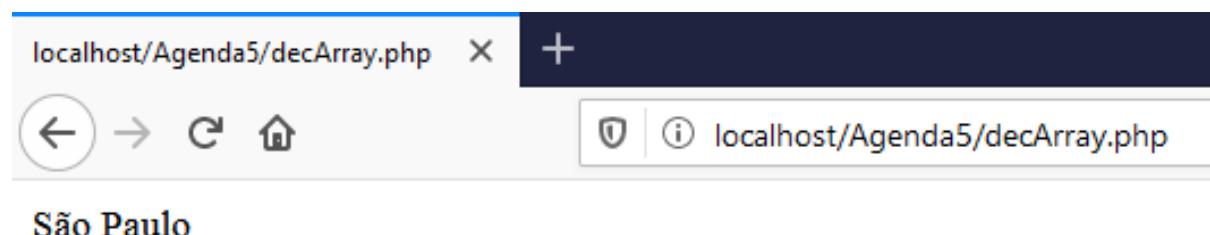


Imagen 4. Navegador o primeiro valor do array através do comando echo.html.

Declarando Array Associativo (arrays com índices textuais)

Você também pode definir índices (*keys* ou *chaves*) como *strings*. Esse tipo de array é denominado associativo e sua declaração precisa sempre “associar” o índice com o seu respectivo valor. Pode ser declarado por meio de um construtor:

```
$filme = array('titulo' => 'Uma mente Brilhante', 'duracao' => '135min', 'genero' => 'drama');
```

Ou até mesmo de forma direta:

```
$filme['titulo'] = 'Uma mente Brilhante';
$filme['duracao'] = '135min';
$filme['genero'] = 'drama';
```

Para melhor visualização da declaração desse array associativo, observe a representação gráfica a seguir:

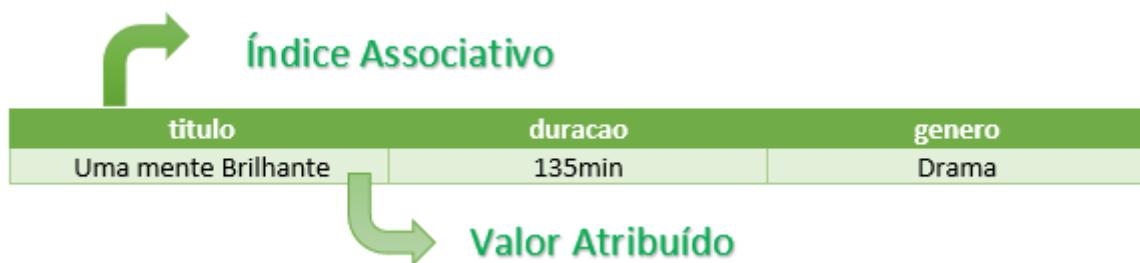


Imagen 5. Representação gráfica dos valores ordenados em um array.

Para acessar os valores do array, basta utilizar o mesmo padrão: nome do array, seguido do índice entre colchetes, que neste caso será uma string, como exemplo a seguir:

```
<?php
echo $filme['titulo'].'<br>';
echo $filme['duracao'].'<br>';
echo $filme['genero'].'<br>

?>
```

Resultado no navegador:

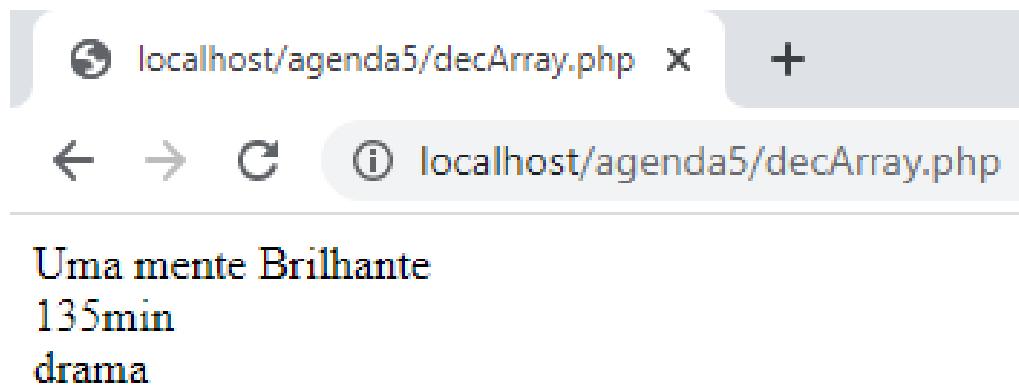


Imagen 6. Navegador exibindo todos os valores do array associativo “filme” por meio do comando echo.html.

Iterando um array com Estruturas de Repetição

Umas das principais vantagens na utilização de um array é a possibilidade de automatização de tarefas em alguma funcionalidade e proposta da web site de manipular as informações no array com utilização de algoritmos. Essas vantagens acontecem principalmente por meio da utilização de estruturas de repetição.

Para melhor entendimento, crie um arquivo no visual studio Code e o salve dentro da pasta root ou Agenda5, com o nome de “exemploForArray” - Neste arquivo vamos criar um formulário com dois inputs, uma combobox e um botão.

O mais importante para esse exemplo é mostrar que a partir de um array populado com todos os estados do país, por exemplo, podemos criar um combobox que ofereça possibilidades como a de um usuário escolher o estado para o envio de mercadoria. Veja a codificação:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Arrays</title>
</head>
<body>
<div class=" w3-third w3-center w3-animate-top w3-padding">
    <form class="w3-
    container" method="post" action="atividadeOnlineAction.php">
        <label class="w3-text-teal"><b>Nome do Cliente</b></label>
        <input class="w3-input w3-border w3-light-
    grey" name="txtNome" type="text">
        <label class="w3-text-teal"><b>Valor da Compra</b></label>
        <input class="w3-input w3-border w3-light-
    grey" name="txtValorCompra" type="number">
        <label class="w3-text-teal"><b>Estado para envio:</b></label>
        <select class="w3-input w3-border w3-light-grey" name = "cmbEstados">
            <?php
                $estados = array("Acre","Alagoas","Amapá","Amazonas","Bahia",
Ceará",
                    "Espírito Santo","Goiás","Maranhão","Mato Grosso","Mato Grosso
do Sul",
                    "Minas Gerais","Pará","Paraíba","Paraná","Pernambuco","Piauí",
"Rio de Janeiro",
                    "Rio Grande do Norte","Rio Grande do Sul","Rondônia","Roraima"
,"Santa Catarina",
                    "São Paulo","Santa Catarina", "Sergipe", "Tocantins","Distrito
Federal");
            <br>
            for($i = 0; $i < count($estados); $i++)
            {
                echo '<option value="'. $i .'>' . $estados[$i] . '</option>';
            }
        ?>
    </select>
    <br>
    <button class="w3-btn w3-blue-grey">Enviar</button>
</form>
</div>
</html>
```

O resultado em um navegador pode ser visualizado a seguir:

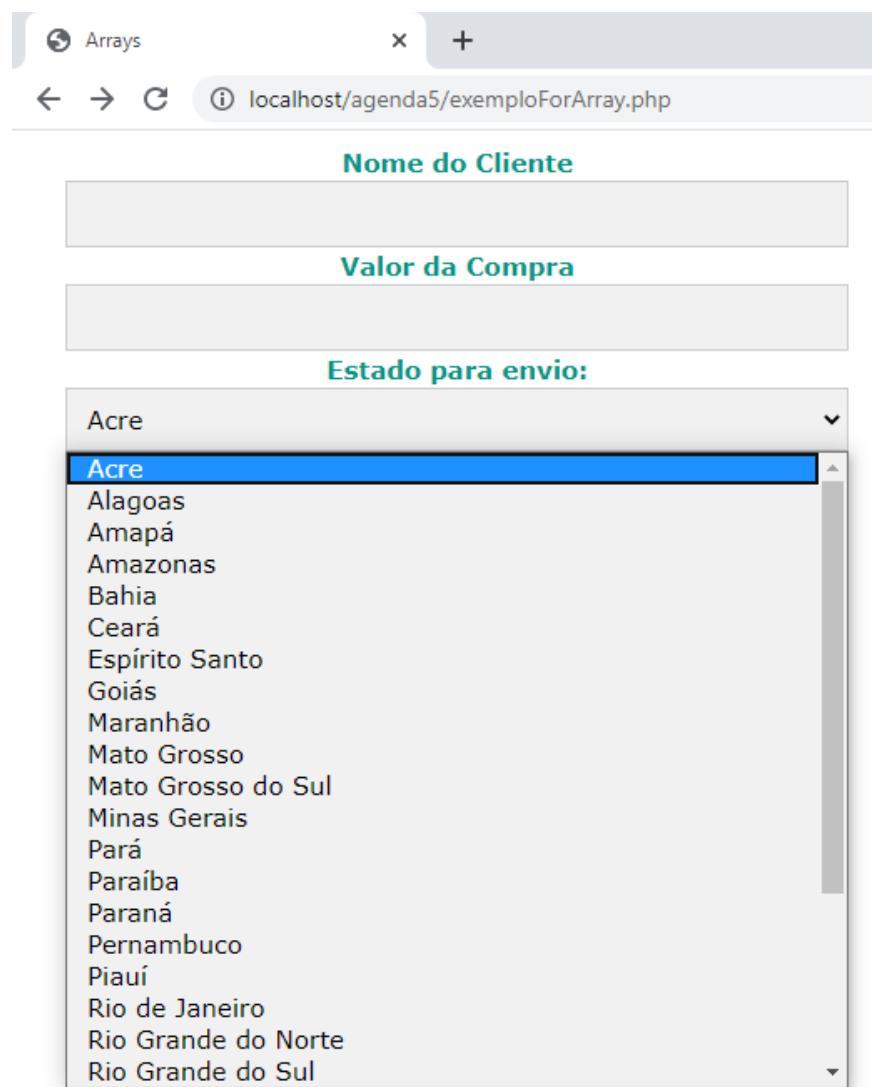


Imagen 7. Resultado no Navegador do combobox criado a partir do array estados com a estrutura de repetição for.

Porém em PHP, a estrutura mais comum para a iteração de arrays, certamente, é por meio da utilização da Estrutura de Repetição **foreach**. Para o mesmo resultado do exemplo anterior podemos codificar dessa forma:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Arrays</title>
</head>
<body>
<div class=" w3-third w3-center w3-animate-top w3-padding">
```

```

<form class="w3-container" method="post" action="atividadeOnlineAction.php">
    <label class="w3-text-teal"><b>Nome do Cliente</b></label>
    <input class="w3-input w3-border w3-light-grey" name="txtNome" type="text">
    <label class="w3-text-teal"><b>Valor da Compra</b></label>
    <input class="w3-input w3-border w3-light-grey" name="txtValorCompra" type="number">
    <label class="w3-text-teal"><b>Estado para envio:</b></label>
    <select class="w3-input w3-border w3-light-grey" name = "cmbEstados">
        <?php
            $estados = array("Acre", "Alagoas", "Amapá", "Amazonas", "Bahia", "Ceará",
                "Espírito Santo", "Goiás", "Maranhão", "Mato Grosso", "Mato Grosso do Sul",
                "Minas Gerais", "Pará", "Paraíba", "Paraná", "Pernambuco", "Piauí",
                "Rio de Janeiro",
                "Rio Grande do Norte", "Rio Grande do Sul", "Rondônia", "Roraima",
                "Santa Catarina",
                "São Paulo", "Santa Catarina", "Sergipe", "Tocantins", "Distrito Federal");
            foreach($estados as $estado)
            {
                echo '<option>' . $estado . '</option>';
            }
        ?>
    </select>
    <br>
    <button class="w3-btn w3-blue-grey">Enviar</button>
</form>
</div>
</html>

```

Conforme visto, o resultado será exatamente o mesmo, mas perceba que a declaração dessa estrutura é um pouco mais fácil de entender. Podemos interpretá-la simplesmente fazendo a leitura da estrutura:

Para cada um dos **\$estados**, será tratado como um **\$estado**, ou seja, para cada iteração da estrutura de repetição, o valor da nova variável **\$estado** será de um valor do array identificado por meio de um índice diferente, começando do primeiro até o último, para maior consolidação da interpretação.

Observe representação a seguir:

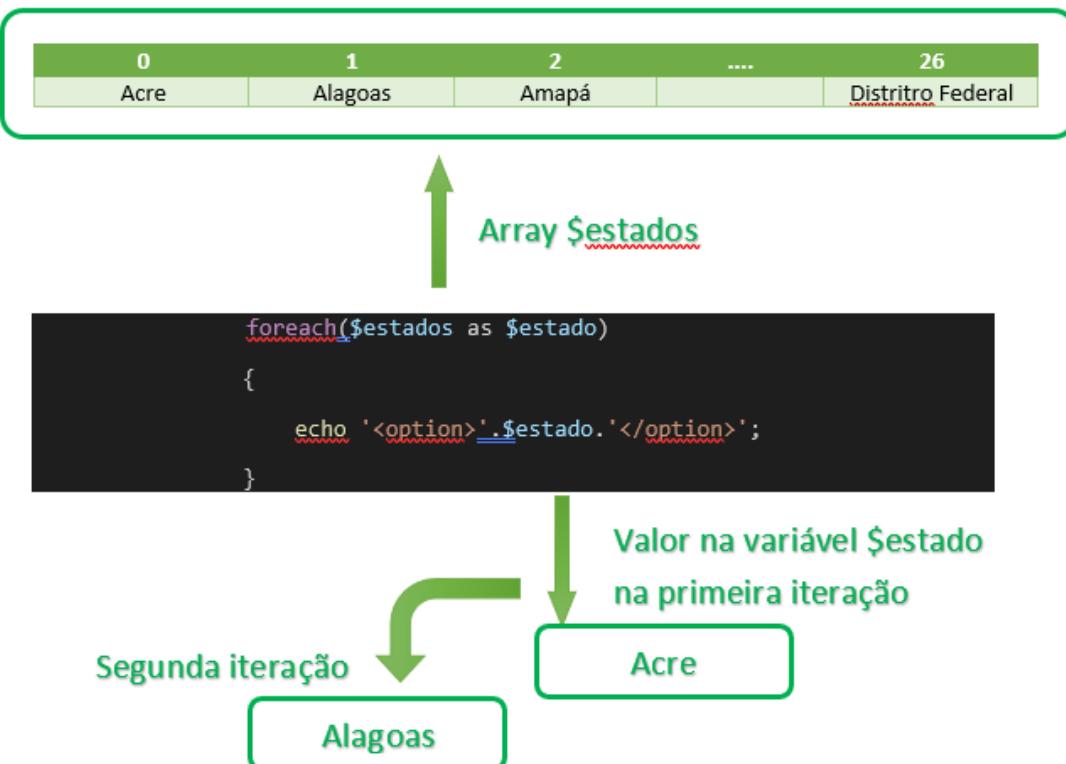


Imagen 8. Representação da execução da primeira e segunda iteração de um comando Foreach no array estado.

Funções para arrays

Existem diversas funções nativas do PHP que oferecem diversos recursos para serem utilizados em array. A seguir uma sequência de quatro vídeos que explica as principais funções e suas utilizações.

Funções de Arrays – Parte #1



Fonte: Node Studio Treinamentos - Curso de PHP 7 - Aula 15 - Funções de Arrays #1. Disponível em <https://www.youtube.com/watch?v=ys5KjCdvSAg>. Acessado em 15/06/2020.

Funções de Arrays – Parte #2



Fonte: Node Studio Treinamentos - Curso de PHP 7 - Aula 16 - Funções de Arrays #2. Disponível em <https://www.youtube.com/watch?v=O5mKRNvoWbE>. Acessado em 15/06/2020.

Funções de Arrays – Parte #3



Fonte: Node Studio Treinamentos - Curso de PHP 7 - Aula 16 - Funções de Arrays #3. Disponível em <https://www.youtube.com/watch?v=4XcUwfCpshg>. Acessado em 15/06/2020.

Funções de Arrays – Parte #4



Fonte: Node Studio Treinamentos - Curso de PHP 7 - Aula 18 - Funções de Arrays #4. Disponível em https://www.youtube.com/watch?v=Zz-5WjcP8_g. Acessado em 15/06/2020.

Arrays Multidimensionais.

Para entender de forma bem simples, trata-se de um Array que tem vinculado um outro Array para cada uma de suas posições, ou seja, uma array dentro de outro array.

O PHP permite construir estruturas com quantas dimensões forem necessárias, mas é certo que, em quase sua totalidade o desenvolvimento será por meio do uso de estruturas que tenham até 2 dimensões, ou seja, estruturas em formato de tabelas ou planilhas.

Para melhor entendimento, no visual studio Code, crie um arquivo e o salve dentro da pasta root ou Agenda5, com o nome de “exemploDeclaracaoArrayMultidimensional” - Neste arquivo vamos criar um array bidimensional chamado produto. Nele deve constar o nome do produto e seu respectivo valor.

Para arrays multidimensionais também existem várias formas de declaração, então vamos codificar.

Obs.: O uso de strings para índices também são permitidos.

```
$produtos = array(
    array("Processador", "900"),
    array("Mouse", "15"),
    array("Teclado", "20"),
    array("Impressora", "500"),
    array("Monitor", "450"),
    array("Placa de Vídeo", "1500"),
    array("Memória RAM 8G", "500"),
    array("Placa Mãe", "600"),
    array("Mouse Pad", "25"),
    array("SSD", "245"),
);
```

Para essa declaração pode utilizar a imagem a seguir para sua apresentação gráfica.

Índice	0	1
0	Processador	900
1	Mouse	15
2	Teclado	20
3	Impressora	500
4	Monitor	450
5	Placa de Vídeo	1500
6	Memória RAM 8G	500
7	Placa Mãe	600
8	Mouse Pad	25
9	SSD	245

Imagen 9. Representação gráfica dos valores em um array bidimensional.

Para acessar os valores do array bidimensional, basta utilizar seu nome seguido dos índices entre colchetes, como exemplo a seguir:

```
<?php
    echo $produtos[0][0];
?>
```

Resultado da imagem no navegador:

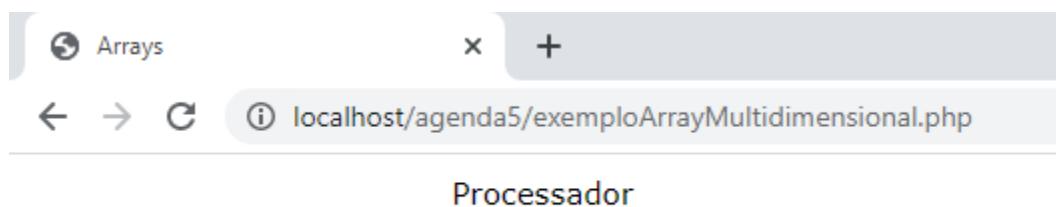


Imagem 10. Navegador exibindo o primeiro valor do array bidimensional produto por meio do comando echo.html.

Uma alternativa para a declaração, que inclusive, é muito utilizada para armazenamento de tabelas de banco de dados em array no php. Vamos colocar as “colunas” com índices associativos (strings) fazendo referência a seu conteúdo.

Exemplo:

```
$produtosAssociativo = array(
    array("nome"=> "Processador", "valor"=> "900" ),
    array("nome"=> "Mouse", "valor"=> "15" ),
    array("nome"=> "Teclado", "valor"=> "20" ),
    array("nome"=> "Impressora", "valor"=> "500" ),
    array("nome"=> "Monitor", "valor"=> "450" ),
    array("nome"=> "Placa de Vídeo", "valor"=> "1500" ),
    array("nome"=> "Memória RAM 8G", "valor"=> "500" ),
    array("nome"=> "Placa Mãe", "valor"=> "600" ),
    array("nome"=> "Mouse Pad", "valor"=> "25" ),
    array("nome"=> "SSD", "valor"=> "245" ),
);
```

Para esta declaração uma representação gráfica adequada seria:

Índice	nome	valor
0	Processador	900
1	Mouse	15
2	Teclado	20
3	Impressora	500
4	Monitor	450
5	Placa de Vídeo	1500
6	Memória RAM 8G	500
7	Placa Mãe	600
8	Mouse Pad	25
9	SSD	245

Para acessar os valores do array bidimensional, basta utilizar o nome seguido dos índices entre colchetes, conforme exemplo:

```
echo $produtosAssociativo[0]['nome'];
?>
```

O resultado seria o mesmo que o exemplo anterior exibido anteriormente por meio da imagem 10.

Iteração em Array Bidimensional.

De forma bem simples e objetiva, podemos tratar o array bidimensional como uma matriz, permitindo assim o uso de estruturas de repetição de diversas formas. Porém, o mais comum é usar a estrutura foreach.

Para melhor entendimento, no visual studio Code, crie um arquivo e o salve dentro da pasta root ou Agenda5, com o nome de “iteracaoArrayMultidimensional” - Neste arquivo vamos utilizar o array bidimensional “\$produtos” do exemplo anterior.

A partir de seu conteúdo, vamos criar uma tabela de produtos para serem exibidas no navegador, por meio de dois exemplos de uso da estrutura foreach.

Veja o código do primeiro exemplo:

```
echo '<table class="w3-table-all w3-hoverable w3-text-black">';
echo '<tr class="w3-teal ">';
echo '<th class="w3-center"> Nome</th>';
echo '<th class="w3-center"> Valor</th>';
echo '</tr>';
foreach($produtos as $produto)
{
    echo '<tr>';
    foreach($produto as $item)
    {
        echo '<td class="w3-center">'.$item.'</td>';

    }
    echo '</tr>';
}
echo '</table>';
```

Observe que neste exemplo estamos utilizando duas estruras de repetição foreach. Para cada iteração o primeiro transforma o array bidimensional (`$produtos`) em um array unidimensional (`$produto`) e o segundo foreach transformando o array unidimensional (`$produto`) em uma variável.

Veja a representação gráfica da primeira iteração de cada foreach:

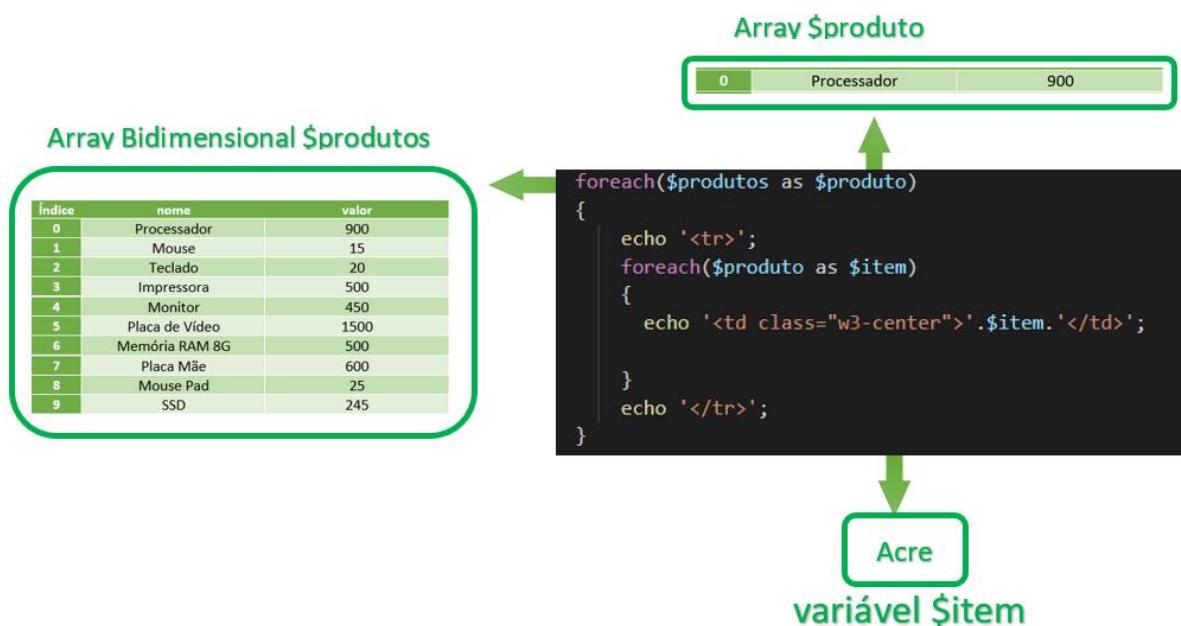


Imagen 12. Representação gráfica da primeira iteração dos foreach no array `$produto`.

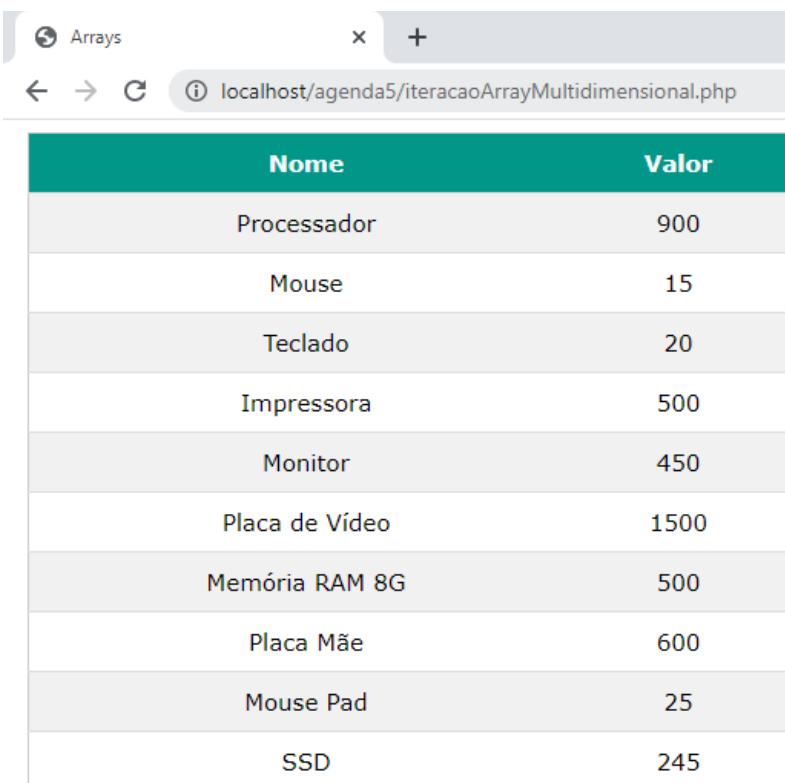
Para o segundo exemplo, deixamos um formatado que é usado de forma quase unânime quando os dados são oriundos de uma consulta a banco de dados.

```
echo '<table class="w3-table-all w3-hoverable w3-text-black">';
    echo '<tr class="w3-teal ">';
        echo '<th class="w3-center"> Nome</th>';
        echo '<th class="w3-center"> Valor</th>';
        echo '</tr>';
    foreach($produtosAssociativo as $produto)
    {
        echo '<tr>';
        echo '<td class="w3-
center">'.$produto['nome'].'</td>';
        echo '<td class="w3-
center">'.$produto['valor'].'</td>';
        echo '</tr>';

    }
echo '</table>';
```

Observe que neste exemplo foi utilizado apenas um foreach, em que para cada iteração é transformado o array bidimensional (`$produtosAssociativo`) em um array unidimensional (`$produto`) e dentro desse, é acionada, para cada echo, uma posição diferente do array (nome e valor).

O resultado no navegador para ambos os exemplos pode ser observado na representação da imagem a seguir:



A screenshot of a web browser window titled "Arrays". The address bar shows "localhost/agenda5/iteracaoArrayMultidimensional.php". The page displays a table with two columns: "Nome" (Name) and "Valor" (Value). The data rows are:

Nome	Valor
Processador	900
Mouse	15
Teclado	20
Impressora	500
Monitor	450
Placa de Vídeo	1500
Memória RAM 8G	500
Placa Mãe	600
Mouse Pad	25
SSD	245

Imagen 13. Navegador exibindo o resultado do uso da estrutura de repetição para a exibição do conteúdo de um array bidimensional em formato de tabela.html.



Joseph, recebeu de seu professor o desafio de criar uma tabela por meio de um array multidimensional, populado com todos os nomes e siglas dos estados brasileiros, inclusive o Distrito Federal. Essa tabela será exibida em um site de uma escola municipal de ensino fundamental.



Imagen 14. Joseph recebendo o desagio do professor (MACROVECTOR / FREEPIK,2020)

Utilizando o que foi visto até agora, crie a página solicitada:

- 1 - Crie um arquivo PHP na pasta root ou Agenda5.
- 2 - Crie uma tabela utilizando uma estrutura de repetição.

Dicas:

- Utilize a Estrutura de Repetição Foreach
- Os índices são: "estado" e "sigla".

Array disponibilizado pelo professor de Joseph

```
$estados = array(  
    array("estado"=> "Acre","sigla"=> "AC" ),  
    array("estado"=> "Alagoas","sigla"=> "AL" ),  
    array("estado"=> "Amapá","sigla"=> "AP" ),  
    array("estado"=> "Amazonas","sigla"=> "AM" ),  
    array("estado"=> "Bahia","sigla"=> "BA" ),  
    array("estado"=> "Ceará","sigla"=> "CE" ),  
    array("estado"=> "Espírito Santo","sigla"=> "ES" ),  
    array("estado"=> "Goiás","sigla"=> "GO" ),  
    array("estado"=> "Maranhão","sigla"=> "MA" ),  
    array("estado"=> "Mato Grosso","sigla"=> "MT" ),  
    array("estado"=> "Mato Grosso do Sul","sigla"=> "MS" ),  
    array("estado"=> "Minas Gerais","sigla"=> "MG" ),  
    array("estado"=> "Pará","sigla"=> "PA" ),  
    array("estado"=> "Paraíba","sigla"=> "PB" ),  
    array("estado"=> "Paraná","sigla"=> "PR" ),  
    array("estado"=> "Pernambuco","sigla"=> "PE" ),  
    array("estado"=> "Piauí","sigla"=> "PI" ),  
    array("estado"=> "Rio de Janeiro","sigla"=> "RJ" ),  
    array("estado"=> "Rio Grande do Norte","sigla"=> "RN" ),  
    array("estado"=> "Rio Grande do Sul","sigla"=> "RS" ),  
    array("estado"=> "Rondônia","sigla"=> "RO" ),  
    array("estado"=> "Roraima","sigla"=> "RR" ),  
    array("estado"=> "Santa Catarina","sigla"=> "SC" ),  
    array("estado"=> "São Paulo","sigla"=> "SP" ),  
    array("estado"=> "Sergipe","sigla"=> "SE" ),  
    array("estado"=> "Tocantins","sigla"=> "TO" ),  
    array("estado"=> "Distrito Federal","sigla"=> "DF" ));
```

A seguir, confira se você conseguiu resolver os desafios propostos!

Código para criação da Tabela

```
echo '<table class="w3-table-all w3-hoverable w3-text-black">';
    echo '<tr class="w3-teal ">';
        echo '<th class="w3-center"> Estado</th>';
        echo '<th class="w3-center"> Sigla</th>';
    echo '</tr>';
    foreach($estados as $estado)
    {
        echo '<tr>';
            echo '<td class="w3-
center">'.$estado['estado'].'</td>';
            echo '<td class="w3-
center">'.$estado['sigla'].'</td>';
        echo '</tr>';

    }
echo '</table>';
```

Codificação Completa:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <title>Arrays</title>
</head>
<body>
<div class="w3-third w3-center w3-animate-top w3-padding">
    <?php
        $estados = array(
            array("estado"=> "Acre","sigla"=> "AC" ),
            array("estado"=> "Alagoas","sigla"=> "AL" ),
            array("estado"=> "Amapá","sigla"=> "AP" ),
            array("estado"=> "Amazonas","sigla"=> "AM" ),
            array("estado"=> "Bahia","sigla"=> "BA" ),
            array("estado"=> "Ceará","sigla"=> "CE" ),
            array("estado"=> "Espírito Santo","sigla"=> "ES" ),
            array("estado"=> "Goiás","sigla"=> "GO" ),
            array("estado"=> "Maranhão","sigla"=> "MA" ),
            array("estado"=> "Mato Grosso","sigla"=> "MT" ),
            array("estado"=> "Mato Grosso do Sul","sigla"=> "MS" ),
```

```
array("estado"=> "Minas Gerais","sigla"=> "MG" ),
array("estado"=> "Pará","sigla"=> "PA" ),
array("estado"=> "Paraíba","sigla"=> "PB" ),
array("estado"=> "Paraná","sigla"=> "PR" ),
array("estado"=> "Pernambuco","sigla"=> "PE" ),
array("estado"=> "Piauí","sigla"=> "PI" ),
array("estado"=> "Rio de Janeiro","sigla"=> "RJ" ),
array("estado"=> "Rio Grande do Norte","sigla"=> "RN" ),
array("estado"=> "Rio Grande do Sul","sigla"=> "RS" ),
array("estado"=> "Rondônia","sigla"=> "RO" ),
array("estado"=> "Roraima","sigla"=> "RR" ),
array("estado"=> "Santa Catarina","sigla"=> "SC" ),
array("estado"=> "São Paulo","sigla"=> "SP" ),
array("estado"=> "Sergipe","sigla"=> "SE" ),
array("estado"=> "Tocantins","sigla"=> "TO" ),
array("estado"=> "Distrito Federal","sigla"=> "DF" ));

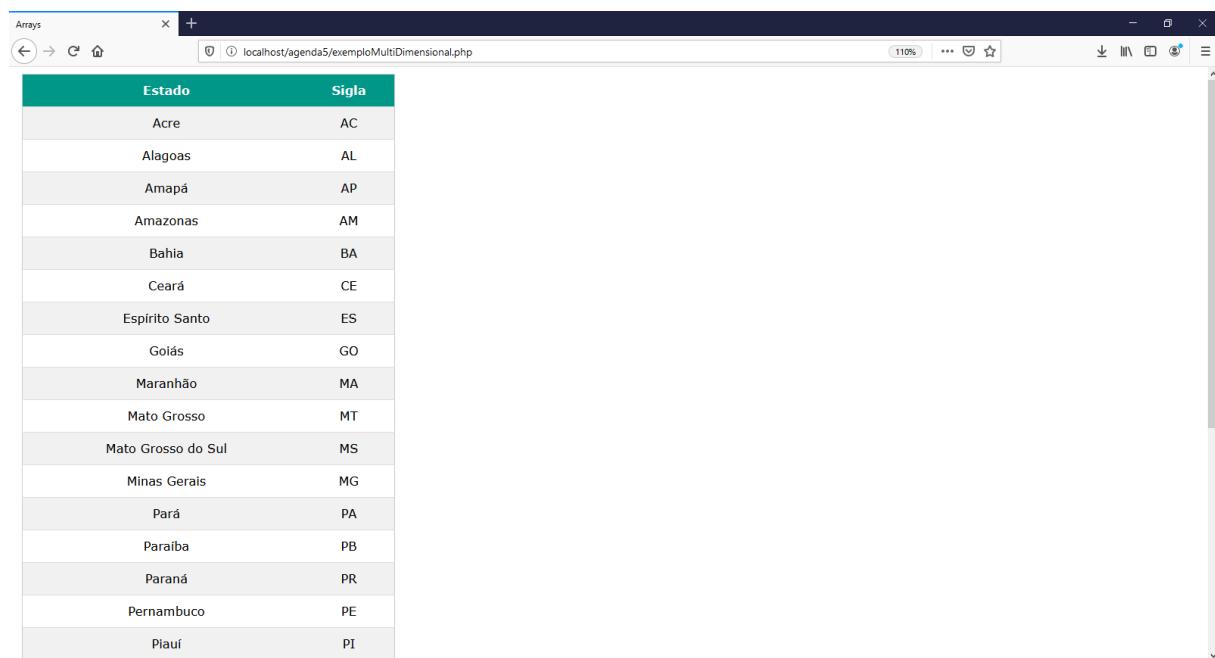
echo '<table class="w3-table-all w3-hoverable w3-text-
black">';
echo '<tr class="w3-teal ">';
echo '<th class="w3-center"> Estado</th>';
echo '<th class="w3-center"> Sigla</th>';
echo '</tr>';
foreach($estados as $estado)
{
    echo '<tr>';
    echo '<td class="w3-
center">' . $estado['estado'] . '</td>';
    echo '<td class="w3-
center">' . $estado['sigla'] . '</td>';
    echo '</tr>';

}
echo '</table>';

?>
<br>

</div>
</html>
```

Resultado no Navegador:



A screenshot of a web browser window titled "Arrays". The address bar shows "localhost/agenda5/exemploMultiDimensional.php". The main content is a table with two columns: "Estado" and "Sigla". The table lists all 26 states of Brazil with their corresponding abbreviations. The table has a green header row and alternating light gray and white rows for the data.

Estado	Sigla
Acre	AC
Alagoas	AL
Amapá	AP
Amazonas	AM
Bahia	BA
Ceará	CE
Espírito Santo	ES
Goiás	GO
Maranhão	MA
Mato Grosso	MT
Mato Grosso do Sul	MS
Minas Gerais	MG
Pará	PA
Paraíba	PB
Paraná	PR
Pernambuco	PE
Piauí	PI

Imagen 15 – Possível resultado no navegador do Exercício Você no Comando.



Envio de Atividade



Imagen 16. Joseph comemorando sua primeira proposta de trabalho remunerado. (PCH.VECTOR / FREEPIK, 2020).

A Diretora da escola ficou encantada com o resultado do trabalho de Joseph e pediu para ele desenvolver uma página para exibir as notas do oitano ano A.

AGENDA 6

PHP:
PERSISTÊNCIA
DE DADOS
PARTE 1 -
MYSQLI



GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



Mysqli

Para ser possível o acesso e manipulação de uma base de dados através da linguagem de programação PHP, é necessário um Driver (MYSQLI), para realizar a ponte entre linguagem e base de dados.

Formalizando a extensão MySQLi é um driver de banco de dados relacional usado na linguagem PHP, fornecendo uma interface com os bancos de dados MySQL. Então como o nome indica, o uso dessa extensão é possível apenas para bancos MySQL.

O PHP costumava utilizar originalmente o driver homônimo do banco MySQL, que suporta as versões mais antigas do MySQL. No entanto, essa extensão foi descontinuada e substituída por MySQLi.

Obs.1: As funções MySQLi funcionam apenas com **PHP 5** (ou superior) e **MySQL 4.1.3** (ou superior).

Obs.2: MySQLi - o “i” vem da palavra em língua inglesa *improved* ou melhorado.

Vamos começar! Antes de qualquer programação, será necessário criar uma base de dados e uma tabela. Para exemplo vamos utilizar o banco de dados MySQL disponível no USBWebServer e em conjunto vamos utilizar MySQL Workbench que é uma ferramenta visual para design de banco de dados, dentro dela podemos desenvolver, administrar, criar e manter, tudo em um único ambiente de desenvolvimento integrado para o sistema de banco de dados MySQL. Download pode ser realizado através do link: <https://dev.mysql.com/downloads/workbench/>.

Obs.1: Para a criação do banco de dados vocês está livre para usar a ferramenta que for mais amigável para você. Um outro exemplo é o phpmyadmin, que para acessá-lo, basta clicar no botão de mesmo nome no painel do USBWebserver



Imagem 3. Destaque do botão *phpmyAdmin*.

Esta ação encaminhará pra uma página para realizar um login, com o usuário é “root” e senha “usbw” (utilizando o MySql do USBWebserver).

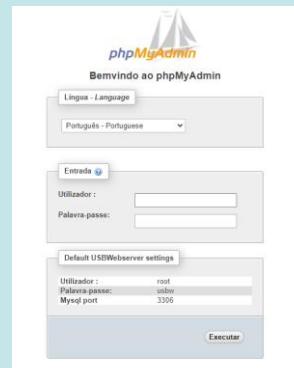


Imagen 4. Tela de login.

Após a realização do login, você será redirecionado para a tela que oferece todos recursos necessários para criar e gerenciar sua base de dados.

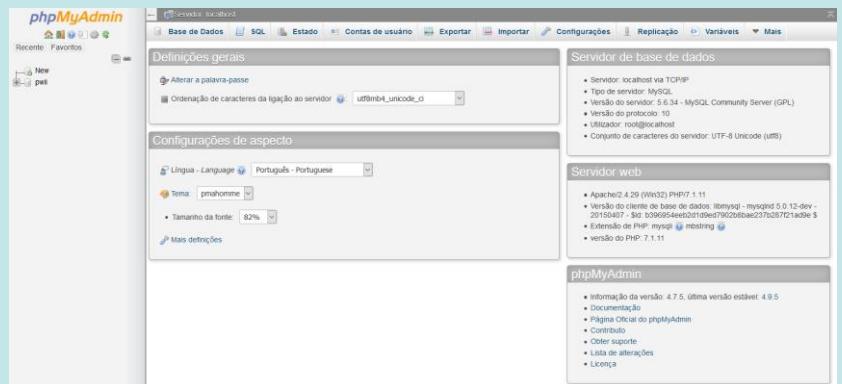


Imagen 5. Tela principal do phpmyadmin.

Para começarmos vamos precisar de uma base de dados, caso queira utilizar o USBWebServer, o usuário é “root” e senha “usbw”, então crie uma base de dados denominada “pwii”.

```
CREATE DATABASE `pwii`;
```

Ou

```
CREATE SCHEMA `pwii` ;
```

Também crie uma tabela nesta base denominada amigos, que tenha os campos:

- id (chave primária e auto incremento).
- Nome.
- Apelido.
- Email.

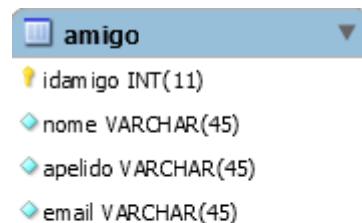


Imagen 6.
Diagrama Banco
de dados.

ou utilize o script a seguir.

```
CREATE TABLE `pwii`.`amigo` (
  `idamigo` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `apelido` VARCHAR(45) NOT NULL,
  `email` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idamigo`));
```

Para este exemplo, será necessário criar diversos arquivos para a abordagem das quatro operações básicas (inserir, excluir, atualizar e deletar). Vamos começar, no visual studio Code, dentro da pasta Agenda6 (ou outra pasta de sua preferência), crie o primeiro arquivo com o nome de “index.php” - Neste arquivo vamos criar dois links, que irão nos direcionar para a realização das ações relacionadas ao banco de dados. Então codifique.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
  <title>Projeto - MYSQLI</title>
</head>
<body>
<div class="w3-padding w3-text-grey w3-half w3-display-middle w3-center">
  <h1 class="w3-center w3-teal w3-round-large w3-margin">Projeto Lista de Amigos</h1>
  <div class="w3-row">
    <div class="w3-col w3-button w3-teal w3-cell w3-round-large" style="width:45%;">
      <a href="cadastro.php" style="text-decoration: none;">
        <i class="fa fa-user-plus" style="font-size: 10.5em"></i>
        <p style="font-size: 2em">Adicionar </p>
      </a>
    </div>
    <div class="w3-col w3-button w3-teal w3-cell w3-round-large w3-right" style="width:45%;">
      <a href="listar.php" style="text-decoration: none;">
        <i class="fa fa-vcard-o" style="font-size: 10.5em"></i>
        <p style="font-size: 2em">Listar</p>
      </a>
    </div>
  </div>
</div>
</body>
</html>
```

Como é possível perceber pelo código anterior, os dois links criados nos encaminham para dois arquivos diferentes: `cadastro.php` e `listar.php`.

O resultado no Navegador:

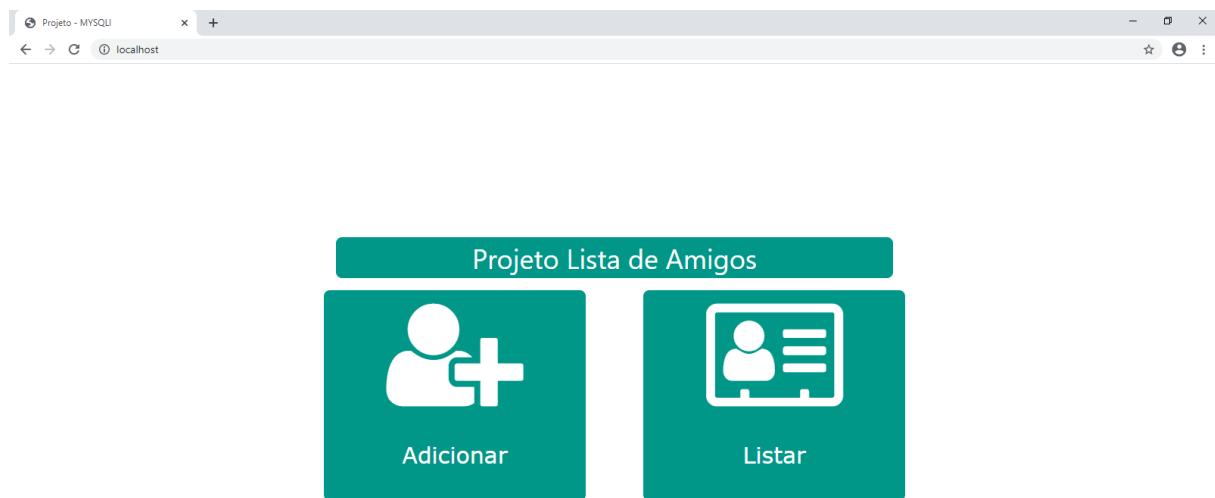


Imagen 7. Resultado no Navegador.

Para continuidade, criaremos agora o arquivo “`cadastro.php`”, na mesma pasta do projeto. Neste arquivo vamos criar um formulário com quatro campos, sendo eles repectivos para: ID, Nome, Apelido e e-mail, ao final, também será criado um botão para adicionar o amigo, a sua base de dados. Código a seguir:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Cadastro - MYSQLI</title>
</head>
<body>
<a href="index.php" class="w3-display-topleft">
    <i class="fa fa-arrow-circle-left w3-large w3-teal w3-button w3-xxlarge"></i>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
    <h1 class="w3-center w3-teal w3-round-large w3-margin">Cadastro de Amigos</h1>
    <form action="cadastroAction.php" class="w3-container" method='post'>
```

```

        <label class="w3-text-teal" style="font-
weight: bold;">Código</label>
        <input name="txtID" class="w3-input w3-grey w3-
border" disabled><br>
        <label class="w3-text-teal" style="font-
weight: bold;">Nome</label>
        <input name="txtNome" class="w3-input w3-light-grey w3-
border"><br>
        <label class="w3-text-teal" style="font-
weight: bold;">Apelido</label>
        <input name="txtApelido" class="w3-input w3-light-grey w3-
border"><br>
        <label class="w3-text-teal" style="font-
weight: bold;">Email</label>
        <input name="txtEmail" class="w3-input w3-light-grey w3-
border"><br>
        <button name="btnAdicionar" class="w3-button w3-teal w3-cell w3-
round-large w3-right w3-margin-right">
            <i class="w3-xxlarge fa fa-user-plus"></i> Adicionar
        </button>
    </form>
</div>

```

Obs.: Logo no início do corpo da página foi criado um link, este tem a função de retorno para a página inicial, fazendo com que a navegação em nosso site fique mais fluída e completa.

Resultado no Navegador:

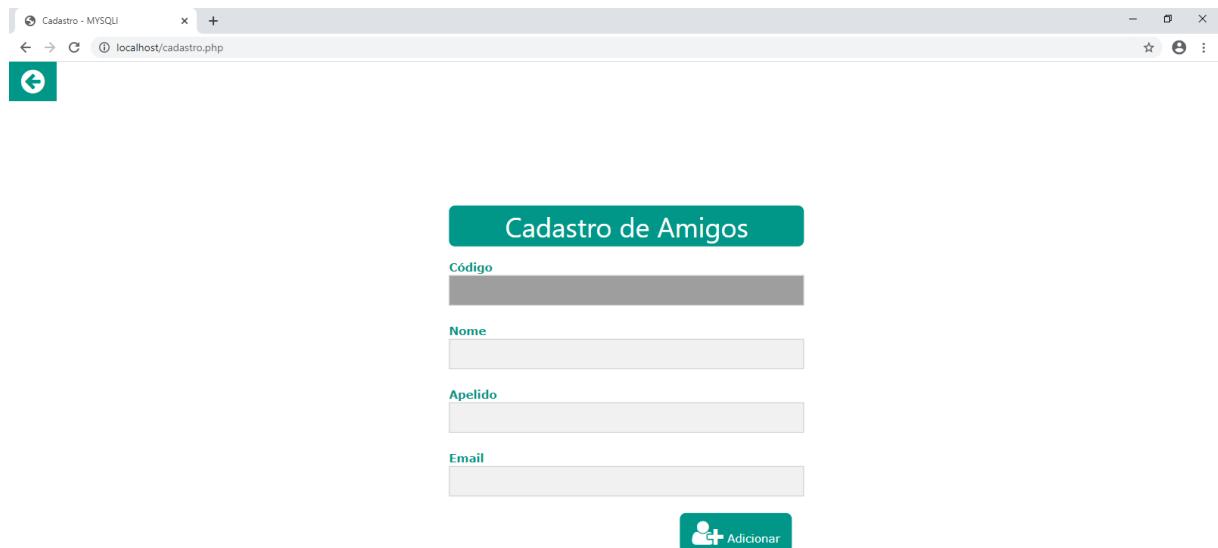


Imagen 8. Resultado no Navegador.

A action do formulário do código anterior, está definida com o arquivo “cadastroAction.php”. Arquivo que devemos criar. Então para a continuidade, crie o arquivo “cadastroAction”, também dentro da mesma pasta do projeto. Neste arquivo, vamos interagir pela primeira vez com a nossa base de dados criada anteriormente, realizando a conexão com o banco, criando e executando a sentença sql de insert, e para finalizar enviar mensagens de sucesso ou fracasso, a depender do resultado da nossa inserção de dados na base.

Para criar a conexão, há uma boa prática de criar variáveis para armazenar o nome do servidor, nome do usuário, senha e por fim o nome da base de dados. Após a definição dos valores para cada uma das variáveis, criamos uma instância do mysqli, que nos fornecerá recursos para realizar a inserção no banco de dados.

```
$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}
```

Obs.: Ao observar o código anterior, percebemos que há um if, para verificar se a conexão foi realizada com sucesso.

O próximo passo será a criação da sentença sql, para a inserção de um registro na tabela amigo, utilizando os dados oriundos do formulário do arquivo “cadastro.php”.

```
$sql = "INSERT INTO amigo (nome, apelido, email)
        VALUES ('".$_POST['txtNome']."' , '".$_POST['txtApelido']."' , '".$_POST
['txtEmail']."' )";

if ($conexao->query($sql) === TRUE) {
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-teal">Amigo Salvo com sucesso! </h1>
        </a>
    ';
}
```

```

} else {
    echo '
<a href="index.php">
    <h1 class="w3-button w3-teal">ERRO! </h1>
</a>
';
}

```

A execução da query sql, está sendo realizada dentro de uma estrutura condicional “if”, verificando obtenção de sucesso em sua ação, fornecendo assim uma mensagem de sucesso em caso verdadeiro ou uma mensagem de falha em caso falso.

O código completo deste arquivo, com os recursos em html e css deve ficar:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Cadastro - MYSQLI</title>
</head>
<body>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">
    <?php
        $servername = "localhost";
        $username = "root";
        $password = "usbw";
        $dbname = "pwii";
        $conexao = new mysqli($servername, $username, $password, $dbname);
        if ($conexao->connect_error) {
            die("Connection failed: " . $conexao->connect_error);
        }
        $sql = "INSERT INTO amigo (nome, apelido, email)
VALUES ('".$_POST['txtNome']."' , '".$_POST['txtApelido']."' , '".$_POST
['txtEmail']."' )";

        if ($conexao->query($sql) === TRUE) {
            echo '
<a href="index.php">
    <h1 class="w3-button w3-teal">Amigo Salvo com sucesso! </h1>
</a>
';

        } else {
            echo '
<a href="index.php">

```

```
<h1 class="w3-button w3-teal">ERRO! </h1>
</a>
';
}
$conexao->close();
?>
</div>
</body>
</html>
```

Claro não podemos esquecer de fechar a conexão com o banco.

```
$conexao->close();
```

Obs.: Tanto para a mensagem de sucesso, quanto para mensagem de falha, terão links direcionando para a página inicial “index.php”.

O resultado no navegador para a inserção no banco com sucesso será:

Amigo Salvo com sucesso!

Imagen 9. Resultado no Navegador para mensagem de sucesso.

Obs: A mensagem de falha segue o mesmo design.

Em continuidade ao projeto, a tarefa de agora é criar o arquivo “listar.php” (arquivo do segundo link da página index), então crie o arquivo “listar”, na mesma pasta do projeto.

Neste arquivo vamos criar uma tabela com seis campos, sendo eles repectivos para ID, Nome, Apelido, e-mail, remover amigo e atualizar amigo, estes dois últimos serão links que redirecionarão para páginas respectivas de exclusão e atualização de amigos. Para a montagem desta tabela, vamos utilizar os dados salvos no banco de dados em vemos precisar novamente:

- Criar instância Mysqli
- Verificar conexão.
- Criar sentença SQL

```
$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}
$sql = "SELECT * FROM amigo" ;
```

Tudo seguindo o mesmo padrão da inserção de dados na base. No entanto, para a setença select, será retornado uma matriz de dados (array), ou seja, precisamos atribuir o resultado em uma variável, também sem esquecer a confirmação para a certeza se foi retornado o resultado esperado.

```
$resultado = $conexao->query($sql);
if($resultado != null)
```

Em caso de retorno, uma a estrutura de repetição foreach (como desenvolvido na AGENDA 5), será utilizada para montar a tabela, listando todos os amigos.

```
foreach($resultado as $linha) {
    echo '<tr>';
    echo '<td>'.$linha['idamigo'].'</td>';
    echo '<td>'.$linha['nome'].'</td>';
    echo '<td>'.$linha['apelido'].'</td>';
    echo '<td>'.$linha['email'].'</td>';
    echo '<td><a href="excluir.php?id='.$linha['idamigo'].'&nome='.$linha['nome'].'&apelido='.$linha['apelido'].'&email='.$linha['email'].'"><i class="fa fa-user-times w3-large w3-text-teal"></i> </a></td></td>';
    echo '<td><a href="atualizar.php?id='.$linha['idamigo'].'&nome='.$linha['nome'].'&apelido='.$linha['apelido'].'&email='.$linha['email'].'"><i class="fa fa-refresh w3-large w3-text-teal"></i></a></td></td>';
    echo '</tr>';
}
```

Neste código, perceba que também já criamos os links para as páginas de exclusão e atualização de amigos, e que enviaremos os dados através da url, utilizando os conceitos do método get, desenvolvido nas primeiras agendas.

O código completo desta página deve ficar.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Listagem de Amigos - MYSQLI</title>
</head>
<body>
<a href="index.php" class="w3-display-topleft">
    <i class="fa fa-arrow-circle-left w3-large w3-teal w3-button w3-xxlarge"></i>
</a>
<?php
    $servername = "localhost";
    $username = "root";
    $password = "usbw";
    $dbname = "pwii";
    $conexao = new mysqli($servername, $username, $password, $dbname);
    if ($conexao->connect_error) {
        die("Connection failed: " . $conexao->connect_error);
    }
    echo '
        <div class="w3-paddingw3-content w3-half w3-display-topmiddle w3-margin">
            <h1 class="w3-center w3-teal w3-round-large w3-margin">Listagem de Amigos</h1>
            <table class="w3-table-all w3-centered">
                <thead>
                    <tr class="w3-center w3-teal">
                        <th>Código</th>
                        <th>Nome</th>
                        <th>Apelido</th>
                        <th>Email</th>
                        <th>Excluir</th>
                        <th>Atualizar</th>
                    </tr>
                <thead>
                ';
    $sql = "SELECT * FROM amigo" ;
```

```

$resultado = $conexao->query($sql);
if($resultado != null)
foreach($resultado as $linha) {
    echo '<tr>';
    echo '<td>' . $linha['idamigo'] . '</td>';
    echo '<td>' . $linha['nome'] . '</td>';
    echo '<td>' . $linha['apelido'] . '</td>';
    echo '<td>' . $linha['email'] . '</td>';
    echo '<td><a href="excluir.php?id=' . $linha['idamigo'] . '&nome=' . $linha['nome'] . '&apelido=' . $linha['apelido'] . '&email=' . $linha['email'] . '"><i class="fa fa-user-times w3-large w3-text-teal"></i> </a></td></td>';
    echo '<td><a href="atualizar.php?id=' . $linha['idamigo'] . '&nome=' . $linha['nome'] . '&apelido=' . $linha['apelido'] . '&email=' . $linha['email'] . '"><i class="fa fa-refresh w3-large w3-text-teal"></i></a></td></td>';
    echo '</tr>';
}
echo '
</table>
</div>';
$conexao->close();
?>
</div>
</body>
</html>

```

Obs.: Logo no início do corpo da página foi criado um link, este tem a função de retorno para a página, inicial, fazendo com que a navegação em nosso site fique mais fluída.

O resultado no navegador para será como representado na imagem a seguir.

Código	Nome	Apelido	Email	Excluir	Atualizar
1	Gabriela	Gabi	gabi@gmail.com		

Imagen 10. Resultado no Navegador para lista de amigos com apenas o contato da gabriela.

Agora é o momento para a criação de mais dois arquivos: excluir e excluir action, para o primeiro criaremos um formulário com os campos: nome, apelido e email (todos desabilitados), um botão para confirmar a exclusão e um link para cancelar a exclusão. O resultado deve ser como o apresentado na imagem a seguir.



EXLUIR - ID: 1

Nome	Gabriela
Apelido	Gabi
Email	gabi@gmail.com

 Confirmar Exclusão.

Imagen 11. Resultado no Navegador para exclusão de um item da lista de amigos.

Para a construção deste arquivo, codifique.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Excluir - MYSQLI</title>
</head>
<body>
<a href="index.php" class="w3-display-topmiddle w3-red w3-center w3-padding w3-button" style="text-decoration:none; ">
    <i class="fa fa-ban" style="font-size:5em"></i>
    <p style="font-weight:bold;">CANCELAR EXCLUSÃO</p>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
    <h1 class="w3-center w3-teal w3-round-large w3-margin">EXLUIR - ID: <?php echo " ".$_GET['id']?> </h1>
    <form action="excluirAction.php" class="w3-container w" method='post'>
        <input name="txtID" class="w3-input w3-grey w3-border" type="hidden" value="<?php echo $_GET['id']?>">
```

```

<br>
<label class="w3-text-teal" style="font-weight: bold;">Nome</label>
<input name="txtNome" class="w3-input w3-border w3-grey" disabled value="<?php echo $_GET['nome']?>">
<br>
<label class="w3-text-teal" style="font-weight: bold;">Apelido</label>
<input name="txtApelido" class="w3-input w3-border w3-grey" disabled value="<?php echo $_GET['apelido']?>">
<br>
<label class="w3-text-teal" style="font-weight: bold;">Email</label>
<input name="txtEmail" class="w3-input w3-border w3-grey" disabled value="<?php echo $_GET['email']?>">
<br>
<button name="btnExcluir" class="w3-button w3-teal w3-cell w3-round-large w3-right">
    <i class="w3-xxlarge fa fa-check"></i> Confirmar Exclusão.
</button>
</form>
</div>
</body>
</html>

```

Perceba com o código, que preenchemos os dados do formulário com os dados obtidos via método get, enviados do arquivo “lista.php”.

Como o atributo action deste form está indicando o arquivo “excluirAction”, precisamos desenvolvê-lo. Então devemos:

- Criar instância Mysqli.
- Verificar conexão.
- Criar sentença DELETE.
- Executar a sentença verificando se mesma obteve sucesso.
- Gerar mensagens de sucesso e falha com link para o arquivo listar.php.
- Fechar a conexão.

Então codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">

```

```

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<title>Exclusão - MYSQLI</title>
</head>
<body>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" id="eProfissional">
<?php
    $servername = "localhost";
    $username = "root";
    $password = "usbw";
    $dbname = "pwii";
    $conexao = new mysqli($servername, $username, $password, $dbname);
    if ($conexao->connect_error) {
        die("Connection failed: " . $conexao->connect_error);
    }
    $sql = "DELETE FROM amigo WHERE idamigo = '".$_POST['txtID']."' ";
    if ($conexao->query($sql) === TRUE) {
        echo '
            <a href="listar.php">
                <h1 class="w3-button w3-teal">Amigo Excluido com sucesso! </h1>
            </a>
        ';
    } else {
        echo '
            <a href="listar.php">
                <h1 class="w3-button w3-teal">ERRO! </h1>
            </a>
        ';
    }
    $conexao->close();
?>
</div>

```

O resultado de um amigo excluído será como demonstrado na imagem a seguir.

Amigo Excluido com sucesso!

Imagen 12. Resultado no Navegador para mensagem de sucesso.

Obs: A mensagem de falha segue o mesmo design.

Para a última das quatro operações básicas, sobrou a atualização dos dados. Para isso vamos criar mais dois arquivos: “atualizar.php” e “atualizarAction.php”. Para o primeiro vamos criar um formulário com os campos (inputs): o nome, apelido e email, que também receberá os dados da url através da utilização do método get, direto do arquivo listar.php, oferecendo a possibilidade do usuário realizar alterações nos dados e assim executar o update, através do clique no botão “atualizar”, confirmando a alteração dos dados

O resultado no navegador deve ser como o apresentado na imagem a seguir.

Imagen 13. Resultado no Navegador para mensagem de sucesso.

A codificação para a construção dessa página, será.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Atualizar - MYSQLI</title>
</head>
<body>
<a href="index.php" class="w3-display-topleft">
    <i class="fa fa-arrow-circle-left w3-large w3-teal w3-button w3-xxlarge"></i>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
    <h1 class="w3-center w3-teal w3-round-large w3-margin">Atualizar - ID: <?php echo " ".$_GET['id']?> </h1>
```

```

<form action="atualizarAction.php" class="w3-container" method='post'>

    <input name="txtID" class="w3-input w3-grey w3-border" type="hidden" value="<?php echo $_GET['id']?>">
    <br>
    <label class="w3-text-teal" style="font-weight: bold;">>Nome</label>
    <input name="txtNome" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['nome']?>">
    <br>
    <label class="w3-text-teal" style="font-weight: bold;">>Apelido</label>
    <input name="txtApelido" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['apelido']?>">
    <br>
    <label class="w3-text-teal" style="font-weight: bold;">>Email</label>
    <input name="txtEmail" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['email']?>">
    <br>
    <button name="btnAtualizar" class="w3-button w3-teal w3-cell w3-round-large w3-right">
        <i class="w3-xxlarge fa fa-refresh"></i> Atualizar
    </button>
</form>
</div>
</body>
</html>

```

Como o atributo action deste form, indica o arquivo “atualizarAction.php”, precisamos desenvolvê-lo. Para isso devemos:

- Criar instância Mysqli
- Verificar conexão.
- Criar sentença Update
- Executar a sentença verificando se mesma obteve sucesso
- Gerar mensagens de sucesso e falha com link para o arquivo listar.php
- Fechar conexão

Então codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">

```

```

<link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
<title>Atualização - MYSQLI</title>
</head>
<body>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" >
<?php
    $servername = "localhost";
    $username = "root";
    $password = "usbw";
    $dbname = "pwii";
    $conexao = new mysqli($servername, $username, $password, $dbname);
    if ($conexao->connect_error) {
        die("Connection failed: " . $conexao->connect_error);
    }
    $sql = "UPDATE amigo SET nome = '". $_POST['txtNome']."' , apelido = '' .
$_POST['txtApelido'].' ,
email='". $_POST['txtEmail']."' WHERE idamigo =". $_POST['txtID']. ";" ;

    if ($conexao->query($sql) === TRUE) {
        echo '
            <a href="listar.php">
                <h1 class="w3-button w3-teal">Amigo Atualizado com sucesso! </h1>
            </a>
        ';
        $id = mysqli_insert_id($conexao);

    } else {
        echo '
            <a href="listar.php">
                <h1 class="w3-button w3-teal">ERRO! </h1>
            </a>
        ';
    }
    $conexao->close();
?>
</div>
</body>
</html>

```

O resultado de um amigo atualizado será como demonstrado na imagem a seguir.

Amigo Atualizado com sucesso!

Imagen 14. Resultado no Navegador para mensagem de sucesso.

Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas agendas e módulo anteriores, há outros métodos e conceitos para fazer todas as operações contidas nessa agenda.



ATENÇÃO - Será disponibilizado o arquivo usbwebserver - Agenda6, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando e itens descritos na atividade online disponíveis para uso e consulta.



Utilizando o que foi visto até agora, foi criada uma tabela no **banco de dados** com o nome: **jogos**, com os atributos idjogo (auto incremento), nome e fabricante, conforme diagrama a seguir.

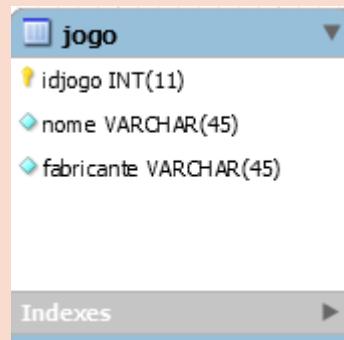


Imagen 15. Diagrama Banco de dados.

1. Obtenha os dados da tabela, utilizando o driver Mysqli
2. Crie uma tabela utilizando os dados obtidos através da consulta ao banco de dados e exiba no navegador.

Dicas:

- Utilize os conceitos e exemplos de estrutura de repetição da Agenda 5.
- De preferência ao uso da estrutura de repetição Foreach.

Caso esteja com dificuldade a seguir script sql para criação da tabela jogo.

```
CREATE TABLE `pwii`.`jogo` (
  `idjogo` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `fabricante` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idjogo`));
```

Caso esteja com dificuldade, a seguir script sql inserção de alguns registros de jogos.

```
INSERT INTO `pwii`.`jogo` (`nome`, `fabricante`) VALUES ('FIFA 2020', 'EA');
INSERT INTO `pwii`.`jogo` (`nome`, `fabricante`) VALUES ('FINAL FANTASY', 'Square Enix');
INSERT INTO `pwii`.`jogo` (`nome`, `fabricante`) VALUES ('GTA', 'Rockstar Games');
INSERT INTO `pwii`.`jogo` (`nome`, `fabricante`) VALUES ('CS: GO', 'Valve');
```

O resultado deverá ser exibido ao usuário em forma de tabela conforme demonstra a imagem a seguir.

Código	Nome	Fabricante
1	FIFA 2020	EA
2	FINAL FANTASY	Square Enix
3	GTA	Rockstar Games
4	CS: GO	Valve

Imagen 16. Imagem tabela de lista de jogos.

Confira abaixo se você conseguiu resolver o desafio propostos!

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
```

```

<title>Tabela Jogos - MYSQLI</title>
</head>
<body class="w3-black">
    <div class="w3-paddingw3-content w3-half w3-display-topmiddle w3-
margin">
        <h1 class="w3-center w3-orange w3-round-large w3-
margin">Listagem de Jogos</h1>
        <table class="w3-table-all w3-centered w3-text-black">
            <thead>
                <tr class="w3-center w3-orange ">
                    <th>Código</th>
                    <th>Nome</th>
                    <th>Fabricante</th>
                </tr>
            <thead>
            <?php
                $servername = "localhost";
                $username = "root";
                $password = "usbw";
                $dbname = "pwii";
                $conexao = new mysqli($servername, $username, $password, $dbname);
                if ($conexao->connect_error) {
                    die("Connection failed: " . $conexao->connect_error);
                }
                $sql = "SELECT * FROM jogo" ;
                $resultado = $conexao->query($sql);
                if($resultado != null)
                    foreach($resultado as $linha) {
                        echo '<tr>';
                        echo '<td>' . $linha['idjogo'] . '</td>';
                        echo '<td>' . $linha['nome'] . '</td>';
                        echo '<td>' . $linha['fabricante'] . '</td>';
                        echo '</tr>';
                    }
                    $conexao->close();
                ?>
            </table>
        </div>
    </body>
</html>

```



ATENÇÃO - Será disponibilizado o arquivo usbwebserver - Agenda6, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando e itens descritos na atividade online disponíveis para uso e consulta.



Envio de Atividade



Imagen 17. Gabriela comemorando a oportunidade.

(STUDIOGHOST / FREEPIK, 2020).

Inês, professora de Inglês, ficou sabendo que Gabriela estudava programação, ficou interessada e pediu para ela desenvolver uma página para exibir as notas de cada um dos 4 módulos de inglês, de cada aluno concluinte da escola, oferecendo para Gabriela desconto em sua mensalidade.

Gabriela ficou muito interessada e começou a buscar informações para desenvolver o projeto. Por sorte a escola, possui um banco de dados, que contém uma tabela, com o código do aluno, seu nome e as notas dos quatro módulos de cada um dos alunos concluintes.

Utilizando o que foi visto até agora, vamos criar a página solicitada, com a seguinte tabela.

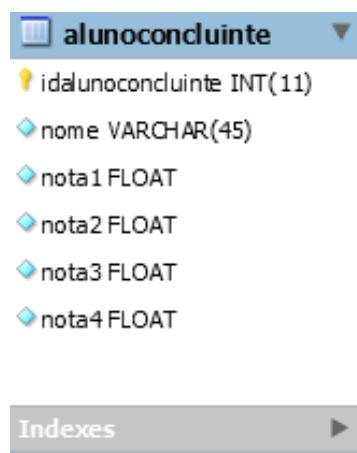


Imagen 18. Diagrama da tabela alunoconcluinte.

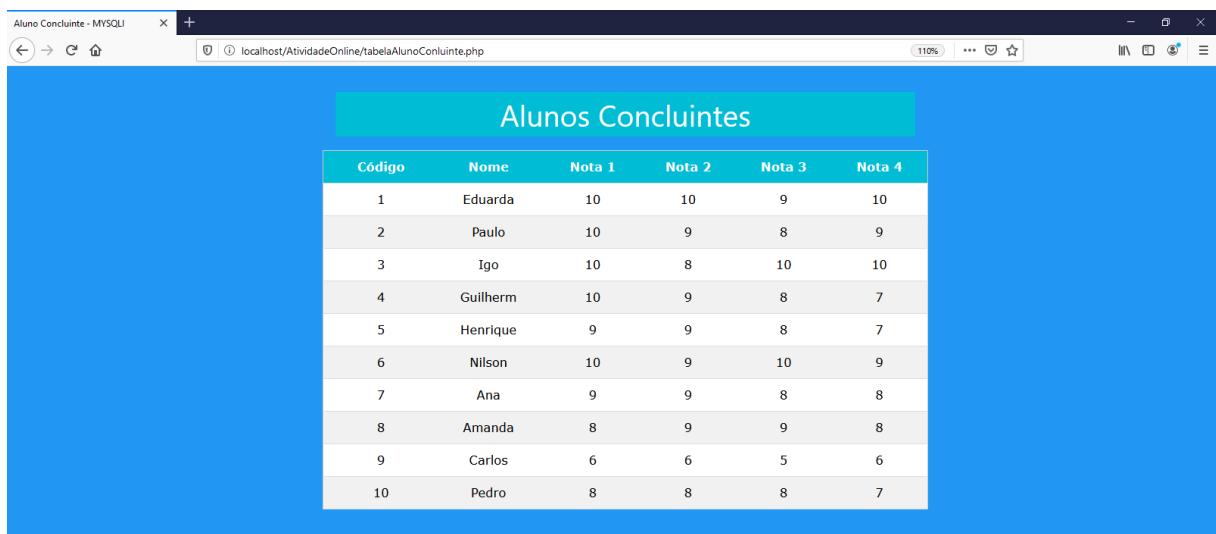
Caso esteja com dificuldade a seguir script sql para criação da tabela AlunoConcluinte

```
CREATE TABLE `pwii`.`alunoconcluinte` (
  `idalunoconcluinte` INT(11) NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `nota1` FLOAT NOT NULL,
  `nota2` FLOAT NOT NULL,
  `nota3` FLOAT NOT NULL,
  `nota4` FLOAT NOT NULL,
  PRIMARY KEY (`idalunoconcluinte`));
```

Caso esteja com dificuldade, a seguir script sql inserção de alguns registros de alunos.

```
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Eduarda', '10', '10', '9', '10');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Paulo', '10', '9', '8', '9');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Igo', '10', '8', '10', '10');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Guilherm', '10', '9', '8', '7');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Henrique', '9', '9', '8', '7');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Nilson', '10', '9', '10', '9');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Ana', '9', '9', '8', '8');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Amanda', '8', '9', '9', '8');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Carlos', '6', '6', '5', '6');
INSERT INTO `pwii`.`alunoconcluinte`(`nome`, `nota1`, `nota2`, `nota3`, `nota4`) VALUES ('Pedro', '8', '8', '8', '7');
```

O resultado deverá ser exibido ao usuário em forma de tabela conforme demonstra a imagem a seguir.



Código	Nome	Nota 1	Nota 2	Nota 3	Nota 4
1	Eduarda	10	10	9	10
2	Paulo	10	9	8	9
3	Igo	10	8	10	10
4	Guilherm	10	9	8	7
5	Henrique	9	9	8	7
6	Nilson	10	9	10	9
7	Ana	9	9	8	8
8	Amanda	8	9	9	8
9	Carlos	6	6	5	6
10	Pedro	8	8	8	7

Imagen 19. Exemplo de tabela solicita para Gabriela.

Vamos ajudar Gabriela a Desenvolver esta atividade!

Dicas:

- Você pode utilizar html e css para deixar a página mais agradável
- Utilize o exercício você no comando como base para a resolução desse exercício.



ATENÇÃO - Será disponibilizado o arquivo usbwebserver - Agenda6, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando e itens descritos na atividade online disponíveis para uso e consulta.



Vídeos:

Não deixe de assistir aos vídeos:

phpMyAdmin: curso rápido em 17 min



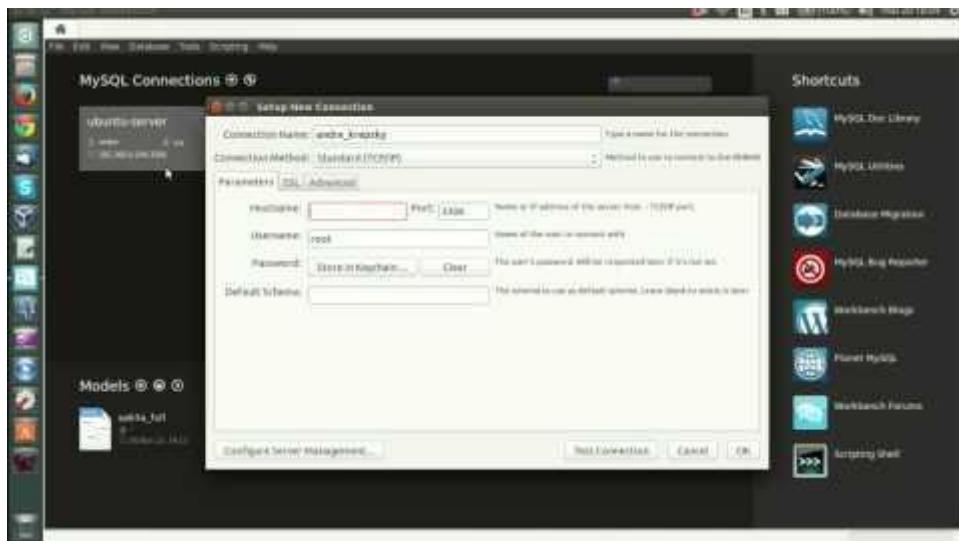
HomeHost - *phpMyAdmin: curso rápido em 17 min | MySQL | Insert, Select, Update, Delete, Create table e Backup.* Disponível em <https://www.youtube.com/watch?v=kviT7G14gqk>. Acessado em 30/06/2020.

Criando conexão no MySQL Workbench 6.



Professor DB - 01 - *Criando conexão no MySQL Workbench 6.* Disponível em <https://www.youtube.com/watch?v=xzAdW7fW0p8>. Acessado em 30/06/2020.

CREATE TABLE: Construção da tabela Titulos



Professor DB - REATE TABLE: Construção da tabela Titulos. Disponível em <https://www.youtube.com/watch?v=0Uldzp4UI0w>. Acessado em 30/06/2020.

Links:

Thiago Belem – Guia prático de MySQLi no PHP. Acessado em 09/07/2020.

<http://blog.thiagobelem.net/guia-pratico-de-mysqli-no-php>.

Maurício Programador – Lista de Comandos PHP para MySQLi. Acessado em 09/07/2020.

<http://www.mauricioprogramador.com.br/posts/lista-de-comandos-php-para-mysqli>

Livro:

OGLIO, Pablo D. **PHP: Programando Com Orientacao a Objetos**. 4ª Edição. São Paulo. Novatec. 2018. ISBN: 978-8575226919

AGENDA 7

PHP:
PERSISTÊNCIA
DE DADOS
PARTE 2 - PDO



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



PDO

PDO conceitualmente trata-se de uma interface PHP Data Objects (PDO), que ao ser implementada, fornece uma camada de abstração em relação a conexão com o banco de dados, trazendo o grande benefício de efetuar a conexão com diversos bancos de dados da mesma forma, alterando basicamente sua string de conexão.

Podemos observar através da tabela a seguir, as semelhanças e diferenças entre PDO e MySQLi.

	PDO	MySQLi
Suporte a Bancos de Dados	12 drivers diferentes	Somente MySQL
API	Orientada a Objetos	Orientada a Objetos + Procedural
Conexão	Fácil	Fácil
Parâmetros Nomeados	Sim	Não
Mapeamento de Objetos	Sim	Sim
Sentenças Preparadas (lado do cliente)	Sim	Não
Performance	Rápido	mais Rápido
Procedimentos Armazenados	Sim	Sim

Obs.: Em alguns casos, o PDO deve ser habilitado no servidor para ter seu funcionamento adequado. No caso do USBWebServer, é possível ser habilitado a partir do arquivo php.ini, localizado na pasta settings, removendo o comentário (;) da extension=php_pdo_mysql.dll.

```
908  extension=php_openssl.dll
909 ;extension=php_pdo_firebird.dll
910 extension=php_pdo_mysql.dll
911 ;extension=php_pdo_oci.dll
912 ;extension=php_pdo_odbc.dll
913 extension=php_pdo_pgsql.dll
914 extension=php_pdo_sqlite.dll
```

Imagen 3. extension=php_pdo_mysql.dll, sem comentário

Em outros servidores talvez também seja necessário, a ativação do: extension=php_pdo.dll , que também é realizada removendo o (;) .

Vamos começar! Seguindo mesmo padrão da agenda anterior, será necessário criar uma base de dados e uma tabela. Continuaremos a utilizar o banco de dados MySQL disponível no USBWebServer com o MySQL Workbench.

Para começarmos a base de dados denominada “pwii” ou você pode usar a mema base da agenda anterior.

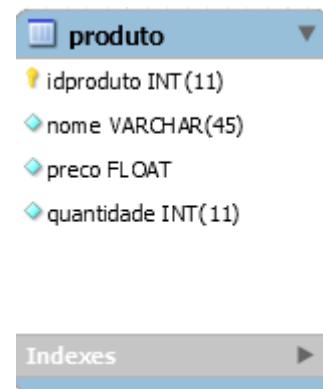
```
CREATE DATABASE `pwii`;
```

Ou

```
CREATE SCHEMA `pwii` ;
```

Também crie uma tabela nesta base denominada produto, que contenha os campos:

- id (chave primária e auto incremento).
- Nome.
- Preco.
- quantidade.



Conforme mostra o diagrama a seguir.

Imagen 6.
Diagrama
Banco de dados.

ou utilize o script a seguir:

```
CREATE TABLE `pwii`.`produto` (
  `idproduto` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `preco` FLOAT NOT NULL,
  `quantidade` INT NOT NULL,
  PRIMARY KEY (`idproduto`));
```

Neste exemplo, também será necessário criar diversos arquivos.

No visual studio Code, dentro da pasta Agenda7 (ou outra pasta de sua preferência), crie o primeiro arquivo com o nome “index.php” - Neste arquivo vamos criar quatro links, que terá o propósito de direcionar a navegação para a realização das ações relacionadas ao banco de dados. Então codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
        <title>Projeto - PDO</title>
</head>
<body class="w3-black">
<h1 class=" w3-center w3-purple w3-margin w3-padding w3-round-large">Projeto Lista de Produtos</h1>
<div class="w3-container w3-col w3-row w3-center ">
    <div class="w3-col w3-button w3-blue w3-cell w3-round-large w3-margin-right w3-margin-left" style="width:23%;">
        <a href="cadastro.php" style="text-decoration: none;">
            <i class="fa fa-cube" style="font-size: 10.5em"></i>
            <p style="font-size: 2em">Adicionar</p>
        </a>
    </div>
    <div class="w3-col w3-button w3-red w3-cell w3-round-large w3-margin-right w3-text-white" style="width:23%;">
        <a href="listar.php" style="text-decoration: none;">
            <i class=" fa fa-trash" style="font-size: 10.5em"></i>
            <p style="font-size: 2em">Remover</p>
        </a>
    </div>
    <div class="w3-col w3-button w3-deep-purple w3-cell w3-round-large w3-margin-right" style="width:23%;">
        <a href="listar.php" style="text-decoration: none;">
            <i class=" fa fa-refresh" style="font-size: 10.5em"></i>
            <p style="font-size: 2em">Atualizar</p>
        </a>
    </div>
    <div class="w3-col w3-button w3-indigo w3-cell w3-round-large w3-margin-right" style="width:23%;">
        <a href="listar.php" style="text-decoration: none;">
            <i class=" fa fa-list-alt" style="font-size: 10.5em"></i>
            <p style="font-size: 2em">Listar</p>
        </a>
    </div>
</div>
</body>
</html>

```

Como é possível perceber pelo código anterior, os quatros links criados nos encaminham para 2 arquivos diferentes: `cadastro.php` (link adicionar) e `listar.php` (links Listar, Atualizar e Exlcuir).

O resultado no Navegador será como o representado na imagem a seguir.

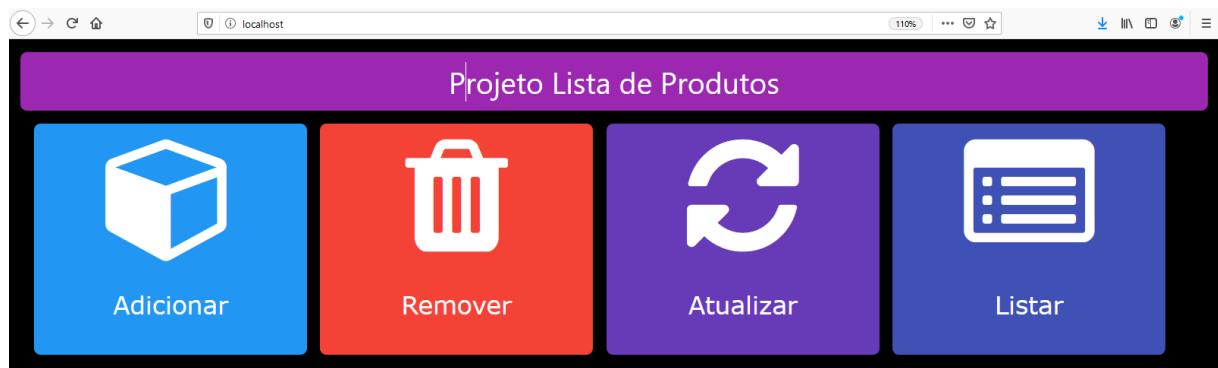


Imagen 7. Resultado no Navegador.

Continuando, crie o arquivo “cadastro.php”, na mesma pasta do projeto. Este arquivo terá um formulário com quatro campos: ID, Nome, Preço e quantidade, ao final, também será criado um botão para adicionar o produto.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Cadastro - PDO</title>
</head>
<body class="w3-black">
<a href="index.php" class="w3-display-topleft ">
    <i class="fa fa-arrow-circle-left w3-large w3-blue w3-button w3-xxlarge"></i>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
    <h1 class="w3-center w3-blue w3-round-large w3-margin">Cadastro de Produtos</h1>
    <form action="cadastroAction.php" class="w3-container" method='post'>
        <label class="w3-text-blue" style="font-weight: bold;">Código</label>
        <input name="txtID" class="w3-input w3-grey w3-border" disabled><br>
        <label class="w3-text-blue" style="font-weight: bold;">Nome</label>
```

```

        <input name="txtNome" class="w3-input w3-light-grey w3-border"><br>
            <label class="w3-text-blue" style="font-weight: bold;">Preço</label>
                <input name="txtPreco" class="w3-input w3-light-grey w3-border"><br>
                    <label class="w3-text-blue" style="font-weight: bold;">Quantidade</label>
                        <input name="txtQtd" class="w3-input w3-light-grey w3-border"><br>
                            <button name="btnAdd" class="w3-button w3-blue w3-cell w3-round-large w3-right w3-margin-right">
                                <i class="w3-xxlarge fa fa-plus-square"></i> Adicionar
                            </button>
                        </form>
                    </div>
                </body>
            </html>

```

Resultado no navegador:



Imagen 8. Resultado no Navegador.

Neste código, a action do formulário tem como valor atribuído o arquivo “cadastroAction.php”, Arquivo de vamos criar na mesma pasta do nosso projeto. Neste arquivo, terá a conexão com o banco de dados através do PDO, a criação e execução da sentença sql de insert, e por fim uma mensagem de sucesso ou fracasso.

Para criar a conexão, continuamos com a boa prática criar variáveis para armazenar o nome do servidor, nome do usuário, senha e por fim o nome da base de dados. Neste momento criamos uma instância do driver PDO.

```
$host = "localhost";
$usuario = "root";
$senha = "usbw";
$bd = "pwii";
try{
    $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306; charset=utf8", $usuario, $senha);
    $conecta->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
} catch(PDOException $e){
    echo "falha ao conectar: ". $e->getMessage();
}
```

Ao observar o código anterior, percebemos que dentro do método construtor, há uma string com a indicação “mysql:”, essa indicação faz com que a instância seja feita para uma base de dados que utilize o SGDB Mysql. Perceba também que foi determinada uma porta e charset utilizado para salvar os dados na base.

Perceba também que neste exemplo, foi acionada um método setAttribute, sua principal função é alterar o modo que o PDO apresenta seus erros, fazendo com que seja lançada uma excessão, para quando eventuais problemas surgirem, assim, já justificamos o uso do try...catch, que pegará a excessão lançada, e oferta recursos para manipularmos essas informações sobre o erro em questão.

O próximo passo, será a criação da sentença sql inserção de um registro na tabela amigo, utilizando os dados oriundos do formulário do arquivo cadastro.

```
$sql = "INSERT INTO produto (nome, preco, quantidade)
VALUES ('".$_POST['txtNome']."' , '".$_POST['txtPreco']."' , '".$_POST['txtQtd']."' )";

try{
    $resultado = $conecta->query($sql);
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-blue">Produto Salvo com sucesso! </h1>
        </a>
    ';
}
```

```

}catch(PDOException $e){
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-blue">ERRO! </h1>
        </a>
    ';
}

```

A execução da query sql, está sendo realizada dentro de uma estrutura try, verificando se mesma obteve sucesso em sua ação, gerando assim uma mensagem de sucesso em caso verdadeiro, e uma mensagem de cado se falha (alguma exceção aconteça).

O código completo deste arquivo, com os recursos em html e css deve ficar:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Cadastro - PDO</title>
</head>
<body class="w3-black">
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">
    <?php
        $host = "localhost";
        $usuario = "root";
        $senha = "usbw";
        $bd = "pwii";
        try{
            $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306; charset=utf8", $usuario, $senha);
            $conecta->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        }catch(PDOException $e){
            echo "falha ao conectar: ". $e->getMessage();
        }

        $sql = "INSERT INTO produto (nome, preco, quantidade)
VALUES ('".$_POST['txtNome']."' , '".$_POST['txtPreco']."' , '".$_POST['txtQtd']."' )";

        try{
            $resultado = $conecta->query($sql);
            echo '
                <a href="index.php">
                    <h1 class="w3-button w3-blue">Produto Salvo com sucesso! </h1>

```

```
        </a>
        ';
    }catch(PDOException $e){
        echo '
            <a href="index.php">
                <h1 class="w3-button w3-blue">ERRO! </h1>
            </a>
        ';
    }
?>
</div>
</body>
</html>
```

Obs.: Tanto para a mensagem de sucesso, quanto para mensagem de falha, terão links direcionando para a página inicial “index.php”.

O resultado no navegador para a inserção no banco com sucesso será:

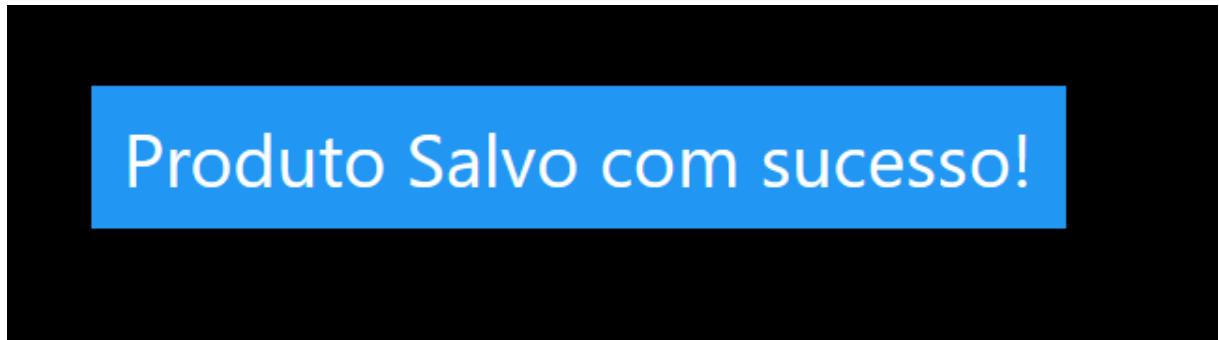


Imagen 9. Resultado no Navegador para mensagem de sucesso.

Obs: A mensagem de falha segue o mesmo design.

Dando continuidade, nossa tarefa agora é criar o arquivo “listar.php” (arquivo do quarto link da página index), então crie o arquivo “listar”, na mesma pasta do projeto. Neste arquivo vamos criar uma tabela com seis campos, sendo eles repectivos para ID, Nome, Preco, Quantidade, remover e atualizar produto, estes dois últimos serão links que redirecionarão para páginas respectivas de exclusão e atualização de amigos.

Para a montagem desta tabela, vamos utilizar os dados salvos no banco de dados em vamos precisar novamente:

- Criar instância Mysqli
- Verificar conexão.
- Criar sentença SQL

Tudo seguindo o mesmo padrão da inserção de dados na base. No entanto, para a setença select, será retornado uma matriz de dados, ou seja, precisamos atribuir o resultado em uma variável.

Neste código, também já criamos os links para as páginas de exclusão e atualização de amigos, perceba que enviamos os dados através da url, utilizando os conceitos do método get.

O código completo desta página deve ficar.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Lista de Produtos - PDO</title>
</head>
<body class="w3-black">
<a href="index.php" class="w3-display-topleft">
    <i class="fa fa-arrow-circle-left w3-large w3-indigo w3-button w3-xxlarge"></i>
</a>
<?php
    $host = "localhost";
    $usuario = "root";
    $senha = "usbw";
    $bd = "pwii";
    try{
        $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306; charset=utf8", $usuario, $senha);
        $conecta-
>setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

```

        echo '
            <div class="w3-padding w3-content w3-half w3-display-
topmiddle w3-margin">
                <h1 class="w3-center w3-indigo w3-round-large w3-
margin">Lista de Produtos</h1>
                <table class="w3-table-all w3-centered w3-text-black">
                    <thead>
                        <tr class="w3-center w3-indigo">
                            <th>Código</th>
                            <th>Nome</th>
                            <th>Preço</th>
                            <th>Quantidade</th>
                            <th>Excluir</th>
                            <th>Atualizar</th>
                        </tr>
                    <thead>
                    ';
        $sql = "SELECT * FROM produto" ;
        $resultado = $conecta->query($sql);
        if($resultado != null)
        foreach($resultado as $linha) {
            echo '<tr>';
            echo '<td>' . $linha['idproduto'] . '</td>';
            echo '<td>' . $linha['nome'] . '</td>';
            echo '<td>' . $linha['preco'] . '</td>';
            echo '<td>' . $linha['quantidade'] . '</td>';
            echo '<td><a href="excluir.php?id=' . $linha['idproduto'] . '&
nome=' . $linha['nome'] . '&preco=' . $linha['preco'] . '&quantidade=' . $linha['quantid
ade'] . '"><i class="fa fa-user-times w3-large w3-text-indigo"></i> </a></td>';
            echo '<td><a href="atualizar.php?id=' . $linha['idproduto'] . '&
nome=' . $linha['nome'] . '&preco=' . $linha['preco'] . '&quantidade=' . $linha['quant
idade'] . '"><i class="fa fa-refresh w3-large w3-text-indigo"></i> </a></td>';
            echo '</tr>';
        }
        echo '
            </table>
        </div>';
    }catch(PDOException $e){
        echo "falha ao conectar: ". $e->getMessage();
    }
    ?>
</div>
</body>
</html>

```

Obs.: Logo no início do corpo da página foi criado um link, este tem a função de retorno para a página, inicial, fazendo com que a navegação em nosso site fique mais fluído.

O resultado no navegador para a inserção no banco com sucesso será:

A screenshot of a web browser window titled "Lista de Produtos". The URL in the address bar is "localhost/listar.php". The page displays a table with one row of data:

Código	Nome	Preço	Quantidade	Excluir	Atualizar
1	Mouse	150	4		

Imagen 10. Resultado no Navegador para lista de produto.

Agora é o momento para a criação de mais dois arquivos: excluir e excluir action, para o primeiro vamos criar um formulário com campos nome, preço e quantidade desabilitados, um botão para confirmar a exclusão e um link para cancelar a exclusão, o resultado deve ser como o apresentado na imagem a seguir.

A screenshot of a web browser showing a delete confirmation dialog. The dialog has a red background with a white "X" icon and the text "CANCELAR EXCLUSÃO". Below the dialog, there is a blue button labeled "EXLUIR - ID: 1". The main content area shows three input fields with disabled inputs:

Nome
Mouse

Preço
150

Quantidade
4

At the bottom right is a blue button with a checkmark icon labeled "Confirmar Exclusão."

Imagen 11. Resultado no Navegador para exclusão de um item da lista de produtos.

Para a construção dessa página, codifique.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Excluir - PDO</title>
</head>
<body class="w3-black">
<a href="index.php" class="w3-display-topmiddle w3-red w3-center w3-padding w3-button" style="text-decoration:none; ">
    <i class="fa fa-ban" style="font-size:5em"></i>
    <p style="font-weight:bold;">CANCELAR EXCLUSÃO</p>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
    <h1 class="w3-center w3-indigo w3-round-large w3-margin">EXLUIR - ID: <?php echo " ".$_GET['id']?> </h1>
    <form action="excluirAction.php" class="w3-container w" method='post'>

        <input name="txtID" class="w3-input w3-grey w3-border" type="hidden" value="<?php echo $_GET['id']?>">
        <br>
        <label class="w3-text-indigo" style="font-weight: bold;">Nome</label>
        <input name="txtNome" class="w3-input w3-grey w3-border" disabled value="<?php echo $_GET['nome']?>">
        <br>
        <label class="w3-text-indigo" style="font-weight: bold;">Preco</label>
        <input name="txtPreco" class="w3-input w3-gray w3-border" disabled value="<?php echo $_GET['preco']?>">
        <br>
        <label class="w3-text-indigo" style="font-weight: bold;">Quantidade</label>
        <input name="txtQtd" class="w3-input w3-gray w3-border" disabled value="<?php echo $_GET['quantidade']?>">
        <br>
        <button name="btnExcluir" class="w3-button w3-indigo w3-cell w3-round-large w3-right">
            <i class="w3-xxlarge fa fa-check"></i> Confirmar Exclusão.
        </button>
    </form>
</div>
</body>
</html>
```

Como o atributo action deste form, indica o arquivo “excluirAction”, precisamos desenvolvê-lo. Para isso vamos:

- Criar instância Mysqli
- Verificar conexão.
- Criar sentença DELETE
- Executar a sentença verificando se mesma obteve sucesso
- Gerar mensagens de sucesso e falha com link para o arquivo listar.php.

Neste exemplo, vamos modificar um pouco a forma como montamos a sentença SQL, através de um recurso de preparação de parâmetro.

```
$sql = $conecta->prepare("DELETE FROM produto WHERE idproduto = ?;");  
$sql->bindParam(1, $_POST['txtID']);  
$sql->execute();
```

Conforme podemos observar no código anterior, começamos preparando a string no SQL utilizando o método prepare, que tem a função apenas de iniciar uma sentença SQL, perceba a presença de um caracter de interrogação (?), este caracter será substituído pelos valores adicionados, através do uso método bindParam.

O método bindParam neste exemplo utiliza duas passagens de parâmetros. A primeira (1) trata-se da posição da interrogação que será substituído (caso existam mais de uma interrogação, para cada uma delas é dada um número em ordem crescente da direita para esquerda) e o segundo parâmetro (\$_POST['txtID']) destina-se ao valor que será inserido na sentença sql.

Obs.: Pode existir um terceiro parâmetro, referente ao tipo de valor que será enviado.

Ficou alguma dúvida? Complemente seus estudos e assista ao vídeo:

Prepared Statements no PHP Orientado a Objetos com PDO.



UpInside - Prepared Statements no PHP Orientado a Objetos com PDO (aula #33)- Disponível em:

<https://www.youtube.com/watch?v=l0Xf4hyWWqw>. Acessado em 12/07/2020.

Então codifique.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Exclusão - PDO</title>
</head>
<body class="w3-black">
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">
    <?php
        $host = "localhost";
        $usuario = "root";
        $senha = "usbw";
        $bd = "pwii";
        try{
            $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306; charset=utf8", $usuario, $senha);
            $conecta->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        }catch(PDOException $e){
            echo "falha ao conectar: ". $e->getMessage();
        }
        try{
            $sql = $conecta-
>prepare("DELETE FROM produto WHERE idproduto = ?");
            $sql->bindParam(1,$_POST['txtID']);
        }
    </?php
</div>
</body>
</html>
```

```

    $sql->execute();
    echo '
        <a href="listar.php">
            <h1 class="w3-button w3-indigo">Produto Excluido com sucesso! </h1>
        </a>
    ';
}catch(PDOException $e){
    echo '
        <a href="listar.php">
            <h1 class="w3-button w3-indigo">ERRO! </h1>
        </a>
    ';
}
?>
</div>
</body>
</html>

```

O resultado de um amigo excluído será como demonstrado na imagem a seguir.



Imagen 12. Resultado no Navegador para mensagem de sucesso.

Obs: A mensagem de falha segue o mesmo design.

Para a última das quatro operações básicas, sobrou a atualização dos dados. Para isso vamos criar mais dois arquivos: “atualizar” e “atualizarAction”. Primeiro vamos criar um formulário com inputs para o nome, preço e quantidade, que receberá os dados oriundos do método get do arquivo “listar.php”, ofertando a possibilidade do usuário realizar alterações e o update através de botão “atualizar”, o resultado deve ser como o apresentado na imagem a seguir.

Atualizar - ID: 6

Nome

Preço

Quantidade



Imagen 13. Resultado no Navegador.

A codificação para a construção dessa página, será.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Atualizar - PDO</title>
</head>
<body class="w3-black">
<a href="index.php" class="w3-display-topleft">
    <i class="fa fa-arrow-circle-left w3-large w3-deep-purple w3-button w3-xxlarge"></i>
</a>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-middle">
```

```

<h1 class="w3-center w3-deep-purple w3-round-large w3-margin">Atualizar - ID: <?php echo ". $_GET['id'] ?> </h1>
<form action="atualizarAction.php" class="w3-container" method='post'>

    <input name="txtID" class="w3-input w3-grey w3-border" type="hidden" value="<?php echo $_GET['id'] ?>">
    <br>
    <label class="w3-text-deep-purple " style="font-weight: bold;">Nome</label>
        <input name="txtNome" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['nome'] ?>">
        <br>
        <label class="w3-text-deep-purple " style="font-weight: bold;">Preço</label>
            <input name="txtPreco" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['preco'] ?>">
            <br>
            <label class="w3-text-deep-purple " style="font-weight: bold;">Quantidade</label>
                <input name="txtQtd" class="w3-input w3-light-grey w3-border" value="<?php echo $_GET['quantidade'] ?>">
                <br>
                <button name="btnAtualizar" class="w3-button w3-deep-purple w3-cell w3-round-large w3-right">
                    <i class="w3-xxlarge fa fa-refresh"></i> Atualizar
                </button>
            </form>
        </div>
    </body>
</html>

```

Como o atributo action deste form, indica o arquivo “atualizarAction”, precisamos desenvolvê-lo. Para isso vamos:

- Criar instância PDO
- Verificar conexão.
- Criar sentença Update
- Executar a sentença verificando se mesma obteve sucesso
- Gerar mensagens de sucesso e falha com link para o arquivo listar.php

Então codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
        <title>Atualização - PDO</title>
</head>
<body class="w3-black">
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">
    <?php
        $host = "localhost";
        $usuario = "root";
        $senha = "usbw";
        $bd = "pwii";
        try{
            $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306; charset=utf8", $usuario, $senha);
            $conecta->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        }catch(PDOException $e){
            echo "falha ao conectar: ". $e->getMessage();
        }
        try{
            $sql = $conecta->prepare("UPDATE produto SET nome = ?, preco = ?, quantidade=? WHERE idproduto = ?;");
            $sql->bindParam(1,$_POST['txtNome']);
            $sql->bindParam(2,$_POST['txtPreco']);
            $sql->bindParam(3,$_POST['txtQtd']);
            $sql->bindParam(4,$_POST['txtID']);
            $sql->execute();
            echo '
                <a href="listar.php">
                    <h1 class="w3-button w3-deep-purple ">Produto Atualizado com sucesso! </h1>
                </a>
            ';
        }catch(PDOException $e){
            echo '
                <a href="listar.php">
                    <h1 class="w3-button w3-deep-purple ">ERRO! </h1>
                </a>
            ';
        }
    ?>
</div>
</body>
</html>

```

O resultado de um produto atualizado será como demonstrado na imagem a seguir.

Produto Atualizado com sucesso!

Imagen 14. Resultado no Navegador para mensagem de sucesso.

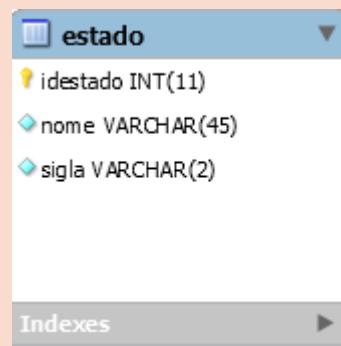
Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nessa agenda.



ATENÇÃO - Será disponibilizado o arquivo **usbwebserver – Agenda7**, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema e você no comando.



Utilizando o que foi visto até agora, foi criada uma tabela no **banco de dados** com o nome: **estado**, com os atributos **ideestado** (auto incremento), **nome** e **sigla**, conforme diagrama a seguir.



*Imagen 16.
Diagrama
Banco de dados.*

1. Obtenha os dados da tabela, utilizando o driver PDO
2. Crie uma tabela no navegador utilizando os dados obtidos através da consulta ao banco de dados e exiba no navegador.

Dicas:

- Utilize os conceitos e exemplos de estrutura de repetição da Agenda 5.
- De preferência ao uso da estrutura de repetição Foreach.

Caso esteja com dificuldade a seguir script sql para criação da tabela estado

```
CREATE TABLE `pwii`.`estado` (
  `idestado` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `sigla` VARCHAR(2) NOT NULL,
  PRIMARY KEY (`idestado`));
```

Caso esteja com dificuldade, a seguir script sql inserção de todos registros de estados.

```
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Acre','AC');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Alagoas','AL');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Amapá','AP');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Amazonas','AM');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Bahia','BA');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Ceará','CE');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Espírito Santo','ES');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Goiás','GO');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Maranhão','MA');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Mato Grosso','MT');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Mato Grosso do Sul','MS');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Minas Gerais','MG');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Pará','PA');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Paraíba','PB');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Paraná','PR');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Pernambuco','PE');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Piauí','PI');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Rio de Janeiro','RJ');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Rio Grande do Norte','RN');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Rio Grande do Sul','RS');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Rondônia','RO');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Roraima','RR');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Santa Catarina','SC');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('São Paulo','SP');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Sergipe','SE');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Tocantins','TO');
INSERT INTO `pwii`.`estado`(`nome`, `sigla`) VALUES ('Distrito Federal','DF');
```

O resultado deverá ser exibido ao usuário em forma de tabela conforme demonstra a imagem a seguir.



Código	Nome	Sigla
1	Acre	AC
2	Alagoas	AL
3	Amapá	AP
4	Amazonas	AM
5	Bahia	BA
6	Ceará	CE
7	Espírito Santo	ES
8	Goiás	GO
9	Maranhão	MA
10	Mato Grosso	MT
11	Mato Grosso do Sul	MS
12	Minas Gerais	MG
13	Pará	PA
14	Paraíba	PB

Imagen 16. Imagem tabela de lista de todos os Estados.

Confira abaixo se você conseguiu resolver o desafio propostos!

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title>Tabela de Estados - PDO</title>
</head>
<body style="background-color: blue;">
    <div class="w3-padding w3-content w3-half w3-display-topmiddle w3-margin">
        <h1 class="w3-center w3-yellow w3-round-large w3-margin">Lista de Estados</h1>
        <table class="w3-table-all w3-centered w3-text-black">
            <thead>
                <tr class="w3-center w3-green">
                    <th>Código</th>
                    <th>Nome</th>
                    <th>Sigla</th>
                </tr>
            <thead>
            <?php
```

```

$host = "localhost";
$usuario = "root";
$senha = "usbw";
$bd = "pwii";
try{
    $conecta = new PDO("mysql:dbname=$bd; host=$host; port=3306;
charset=utf8", $usuario, $senha);
    $conecta-
>setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
}catch(PDOException $e){
    echo "falha ao conectar: ". $e->getMessage();
}
try{
    $sql = "SELECT * FROM estado";
    $resultado = $conecta->query($sql);

    foreach($resultado as $linha) {
        echo '<tr>';
        echo '<td>' . $linha['idestado'] . '</td>';
        echo '<td>' . $linha['nome'] . '</td>';
        echo '<td>' . $linha['sigla'] . '</td>';
        echo '</tr>';
    }
}catch(PDOException $e){
    echo '
        <h1 class="w3-button w3-indigo">ERRO! </h1>
    ';
}
?>
</table>
</div>
</body>
</html>

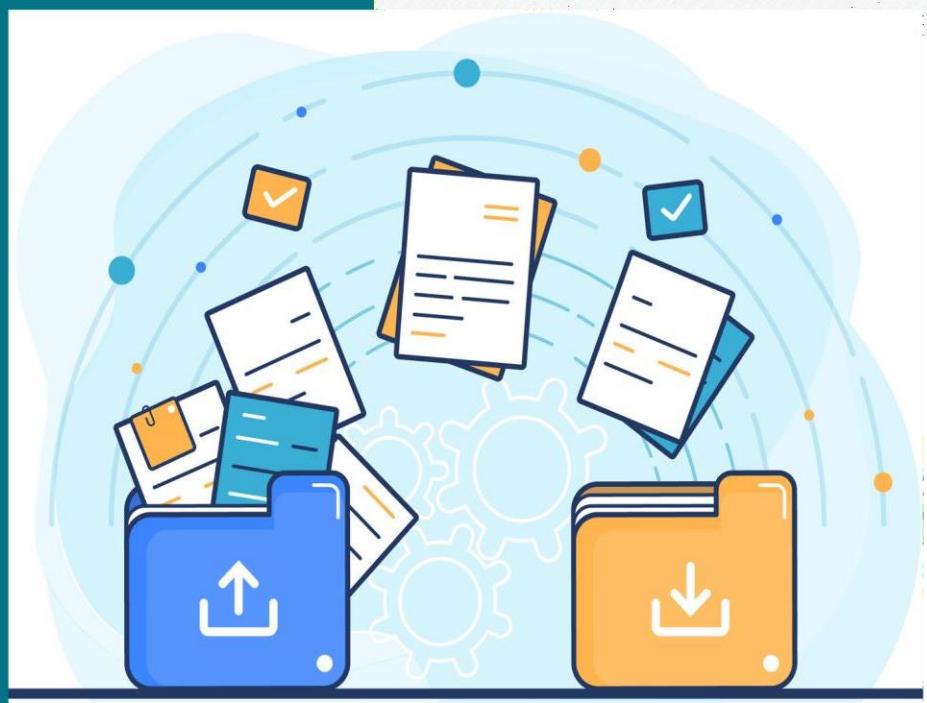
```



ATENÇÃO - Será disponibilizado o arquivo usbwebserver – Agenda7.rar, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando disponíveis para uso e consulta.

AGENDA 8

**PHP:
IMPORTAÇÃO
DE ARQUIVOS,
VARIÁVEIS DE
SESSÃO, COOKIES
E HOSPEDAGEM**



GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO

CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

PROGRAMAÇÃO MOBILE I

Expediente

Autor:

Paulo Eduardo Cardoso Andrade

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: Flávio Biazim



Importação de Arquivo (Require e Include)

No PHP, há disponível quatro funções com o objetivo de importar arquivos, e cada uma exerce uma tarefa específica. As quatro funções são: include (), include_once(), require() ou require_once(). As funções que contém a palavra “once” evita a importação de um arquivo que já tenha sido importado anteriormente, em alguma etapa do projeto.

Obs: Um dos grandes problemas em importar um arquivo que já está importado, é que as variáveis contidas no escopo desse arquivo serão resetadas, assim, os valores contidos nas variáveis já definidas em outro arquivo que já utilizou a importação, serão perdidos. Isso é somente um dos problemas, podemos encontrar outros também, no entanto, em alguns casos esse problema é utilizado como solução.

As funções com nome include do PHP têm como objetivo adicionar um arquivo dentro de outro quando acessado. Em qualquer problema encontrado na inclusão, será apresentado um aviso (Warning). Esse aviso vai indicar que não foi possível realizar sua inclusão e exibirá a página normalmente (sem a inclusão do arquivo). As funções com nome required PHP tem o mesmo objetivo das funções include, a diferença está, em situações em que o arquivo que está sendo incluído não exista ou não seja encontrado, um Fatal Error (erro fatal) é gerado, interrompendo a execução de qualquer codificação que venha após a linha da função require.

Obs: outra divergência é o fato das funções require não aceitarem parâmetros via GET para o arquivo chamado. Caso você utilize este parâmetro, ele será ignorado. Já, as funções de nome include, aceitam parâmetros via GET.

Com os conceitos em mente, vamos começar! Para exemplificar, vamos fazer atualização do projeto da agenda 6 – Lista de Amigos.

Inicialmente vamos criar uma página de login, para ser possível restringir o acesso aos dados da lista de amigos. Então crie uma tabela denominada usuário, que contenha os campos:

- id (chave primária e auto incremento).
- Nome.
- senha.

Conforme mostra o diagrama a seguir.

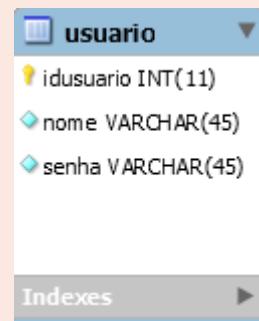


Imagem 3. Diagrama Banco de dados.

Nesta tabela, vamos adicionar um registro de login através do comando SQL:

```
INSERT INTO `pwii`.`usuario` (`nome`, `senha`) VALUES ('gabi', 'gabi123');
```

Obs: Utilizamos para o campo nome o valor “gabi” e para o campo senha “gabi123”, mas você pode utilizar qualquer valor, lembre-se apenas que estes valores serão utilizados para login no site.

Você pode copiar o projeto da pasta utilizada na agenda 6, para uma pasta chamada Agenda8 (ou outra pasta de sua preferência) ou continuar utilizando a mesma pasta do projeto anterior, isso tudo fica a seu critério.

Para iniciar, o atual arquivo “index” deverá ser renomeado para “principal”, assim, podemos no visual studio Code, dentro da pasta Agenda8 (ou outra pasta de sua preferência), criar o primeiro arquivo da atualização com o nome “index.php” - Neste arquivo vamos criar a página de login, que vai consistir em um formulário, com dois campos (nome e senha) e um botão, que terá o propósito de fornecer acesso a página recentemente nomeada como principal. Então codifique.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body>
    <div class="w3-container w3-round-xxlarge w3-display-middle w3-card-4 w3-third " style="">

        <div class="w3-center">
            <br>
            
        </div>

        <form class="w3-container " action="loginAction.php" method="post">
            <div class="w3-section">
                <label style="font-weight: bold;">Usuário</label>
                <input class="w3-input w3-border w3-margin-bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
                <label style="font-weight: bold;">Senha</label>
                <input class="w3-input w3-border" type="text" placeholder="Digite a Senha" name="txtSenha" required>
                <button class="w3-button w3-block w3-teal w3-section w3-padding" type="submit">Entrar</button>
            </div>
        </form>
        <br>
    </div>
</body>
</html>
```

O resultado no Navegador será como o representado na imagem a seguir.

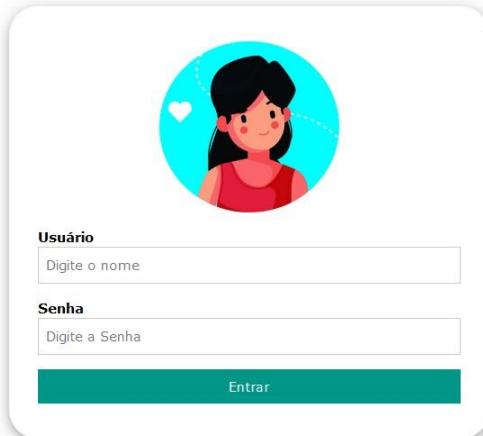


Imagen 4. Resultado no Navegador.

Obs: Ao digitar a senha perceba que ela está sendo exibida, para não exibir basta alterar o type do input pra “password”.

Imagen 5. Input configurado como password.

Como é possível perceber pelo código anterior, a action do formulário indica para o arquivo “loginAction.php”, que será o próximo arquivo que vamos desenvolver.

Então, no visual studio Code, dentro da pasta Agenda8 (ou outra pasta de sua preferência), crie o arquivo com o nome “loginAction.php” - Neste arquivo, será realizada a conexão com o banco de dados através do Mysqli, a criação e execução da sentença sql de select, verificando se existe ou não o usuário, e por fim verificando se a senha digitada no formulário é a mesma gravada no banco de dados. Para isso Codifique.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body>
    <div class="w3-padding w3-content w3-text-grey w3-third w3-display-
middle" >
        <?php
            $nome = $_POST['txtNome'];
            $senha = $_POST['txtSenha'];
            $servername = "localhost";
            $username = "root";
            $password = "usbw";
            $dbname = "pwii";
            $conexao = new mysqli($servername, $username, $password, $dbname);
            if ($conexao->connect_error) {
                die("Connection failed: " . $conexao->connect_error);
            }
            $sql = "SELECT * FROM usuario WHERE nome = '". $nome . "' ;";
            $resultado = $conexao->query($sql);
            //echo $sql;
            $linha = mysqli_fetch_array($resultado);
            if($linha != null)
            {
                if($linha['senha'] == $senha)
                {
                    echo '
                        <a href="principal.php">
                            <h1 class="w3-button w3-teal">' . $nome . ', Seja Bem-
Vinda! </h1>
                        </a>
                    ';
                }
                else
                {
                    echo '
                        <a href="index.php">
                            <h1 class="w3-button w3-teal">Login Inválido! </h1>
                        </a>
                    ';
                }
            }
            else
            {
                echo '
                    <a href="index.php">

```

```

        <h1 class="w3-button w3-teal">Login Inválido! </h1>
    </a>
    ';
}

$conexao->close();
?>
</div>

</body>
</html>

```

Perceba que existe um desvio condicional, que em caso a senha digitada no formulário e a senha do banco sejam as mesmas, será exibido um link que redirecionará o usuário para a página principal (imagem 6).

gabi, Seja Bem-Vinda!

Imagen 6. Login Realizado com sucesso.

Em caso negativo (senhas diferentes), será exibido um link que retornará para página principal.

Login Inválido!

Imagen 7. Login inválido.

Pronto, nosso login está criado! Porém perceba o seguinte, toda vez que vamos realizar o acesso ao banco de dados, seja, para inserir, deletar, atualizar ou buscar uma informação precisamos passar os dados de conexão. Então, estamos utilizando acesso ao banco em pelo menos 4 arquivos. Vamos imaginar que por algum motivo a senha para login no banco, mudou, neste projeto precisaríamos ir aos quatro arquivos e fazer a alteração um por vez, mas se utilizarmos o conceito de importação de arquivos (include ou require), podemos criar um arquivo chamado “conexaoBD” e dentro dele programar:

```
<?php

$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}

?>
```

E substituir em todos os quatro arquivos esse mesmo código por:

```
require_once 'conexaoBD.php';
```

ou

```
require_once ('conexaoBD.php');
```

Assim, em uma possível troca de senha, precisaríamos apenas alterar o arquivo “conexaoBD”, e todo nosso projeto seria “atualizado” e utilizaria a senha correta. Além disso, diminui a quantidade de códigos digitados, evita esquecimentos e erros. Deve resultar em um código semelhante com:

```
<?php
$nome = $_POST['txtNome'];
$senha = $_POST['txtSenha'];
require_once 'conexaoBD.php';
$sql = "SELECT * FROM usuario WHERE nome = '". $nome . "' ;";
$resultado = $conexao->query($sql);
//echo $sql;
$linha = mysqli_fetch_array($resultado);
if($linha != null)
{
    if($linha['senha'] == $senha)
    {
        echo '
            <a href="principal.php">
                <h1 class="w3-button w3-teal">' . $nome . ', Seja Bem-
Vinda! </h1>
            </a>
        ';
    }
}
```

```

        else
        {
            echo '
                <a href="index.php">
                    <h1 class="w3-button w3-teal">Login Inválido! </h1>
                </a>
            ';
        }
    }
else
{
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-teal">Login Inválido! </h1>
        </a>
    ';
}

$conexao->close();
?>

```

Outra boa utilização, é criar arquivos de cabeçalho e rodapé, e utilizar a importação de arquivos, assim quando alterar nester arquivos, todo o site será atualizado.

Então no visual studio code, crie dois arquivos salvando-os na mesma pasta do projeto. O primeiro denominado “cabecalho.php” com o seguinte código.

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body>

```

O Segundo arquivo denominado de rodapé, com o seguinte código:

```

</body>
</html>

```

Obs: Este exemplo é bem simples e a quantidade de código no rodapé é mínima, porém, é possível em sites mais complexos ter um rodapé bem mais complexo e recheado de código e informações relevantes.

Para finalizar atualize todos os arquivos do projeto, substituindo todos os códigos iguais por:

```
<?php require_once ('cabecalho.php'); ?>
```

No Cabeçalho e para o rodapé:

```
<?php require_once ('rodape.php'); ?>
```

Para melhor exemplificar, a seguir o código atualizado do arquivo “index.php” com os requires.

```
<?php require_once ('cabecalho.php'); ?>
<div class="w3-container w3-round-xxlarge w3-display-middle w3-card-4 w3-third " style="">

    <div class="w3-center">
        <br>
        
    </div>

    <form class="w3-container " action="loginAction.php" method="post">
        <div class="w3-section">
            <label style="font-weight: bold;">Usuário</label>
            <input class="w3-input w3-border w3-margin-bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
            <label style="font-weight: bold;">Senha</label>
            <input class="w3-input w3-border" type="password" placeholder="Digite a Senha" name="txtSenha" required>
            <button class="w3-button w3-block w3-teal w3-section w3-padding" type="submit">Entrar</button>
        </div>
    </form>
    <br>

</div>
<?php require_once ('rodape.php'); ?>
```

Com esta etapa concluída, realize um teste! Digitando no navegador “localhost/nomeDaPastaDoSeuProjeto/principal.php”, você vai perceber que foi possível burlar o login, para isso precisamos estudar o próximo tópico SESSION!

SESSION e suas variáveis

Na internet há um problema: um servidor web não sabe quem é você ou o que você faz quando você acessa um site, porque o endereço HTTP não mantém o estado ou condição que você está no site. Para resolver este problema, entra em ação as variáveis de sessões, elas conseguem realizar o armazenamento de informações do usuário para serem utilizadas em várias páginas (por exemplo, nome de usuário, cor favorita etc.). Desta forma as variáveis de sessão contêm informações sobre um único usuário e estão disponíveis para todas as páginas durante seu acesso.

Para iniciar uma sessão em php você precisa da função session_start(), e as variáveis de sessão serão configuradas com a variável global PHP: \$_SESSION.

```
<?php
    session_start();
    $_SESSION["Nome"] = "Gabi";
    $_SESSION["Idade"] = 17;
?>
```

No Código acima foram criadas duas variáveis de sessão a primeira denominada Nome e atribuindo o valor dentro da mesma, a segunda denominada idade e atribuição do valor 17 na mesma.

Para obter os valores contido nas variáveis em outras páginas, basta iniciar novamente a sessão, e utilizá-las como o exemplo a seguir.

```
<?php
    session_start();
    echo "Nome: ".$_SESSION["Nome"]. "e idade: ".$_SESSION["Idade"];
?>
```

Com esse conceito explanado, vamos atualizar nosso projeto e proteger o acesso via url de usuário não identificado, atualizando o arquivo “loginAction”. A atualização está em iniciar a sessão (“session_start();”) e dentro da estrutura condicional em caso verdadeiro para login e senha corretos, o valor da variável \$nome, dentro de uma variável de sessão denominada “logado”, resultando código a seguir.

```
<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" >
    <?php
        session_start();
        $nome = $_POST['txtNome'];
        $senha = $_POST['txtSenha'];
        require_once 'conexaoBD.php';
        $sql = "SELECT * FROM usuario WHERE nome = '". $nome . "' ;";
        $resultado = $conexao->query($sql);
        //echo $sql;
        $linha = mysqli_fetch_array($resultado);
        if($linha != null)
        {
            if($linha['senha'] == $senha)
            {
                echo '
                    <a href="principal.php">
                        <h1 class="w3-button w3-teal">' . $nome . ', Seja Bem-
                Vinda!  </h1>
                    </a>
                ';
                $_SESSION['logado'] = $nome;
            }
            else
            {
                echo '
                    <a href="index.php">
                        <h1 class="w3-button w3-teal">Login Inválido! </h1>
                    </a>
                ';
            }
        }
    else
    {
        echo '
            <a href="index.php">
                <h1 class="w3-button w3-teal">Login Inválido! </h1>
            </a>
        ';
    }
    $conexao->close();
```

```
?>
</div>
<?php require_once ('rodape.php'); ?>
```

Agora, crie um arquivo chamado “verificarAcesso”, este arquivo sempre vai verificar se há uma sessão aberta e se a variável logado está iniciada. Caso ela não esteja, utilizando a função nativa header, redirecionaremos o acesso para outro arquivo denominado acessoNegado. O código para esse arquivo de ser.

```
<?php
if(!isset($_SESSION))
{
    session_start();
}
if((!isset ($_SESSION['logado']) == true))
{
    header('location:acessoNegado.php');
    die();
}
?>
```

Por fim, é necessário criar o arquivo “acessoNegado.php”, que terá apenas uma mensagem de Acesso Negado e um link para a página index, para a realização do login. Segue o código.

```
<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle">

<?php
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-teal">Acesso NEGADO! </h1>
            <br>
            <h2 class="w3-button w3-
teal">É necessária a Realização do Login Inválido! </h2>
        </a>
    ';
?>
</div>
<?php require_once ('rodape.php'); ?>
```

Por fim, precisamos atualizar todos os arquivos inserindo na primeira linha o seguinte código:

```
<?php require_once ('verificarAcesso.php');?>
```

Esta simples linha de código fará, com que sempre que um usuário tente acessar via url as páginas do nosso projeto, seja verificado o acesso do mesmo. E caso ele não tenha realizado o login, sempre aparecerá a mensagem “Acesso Negado”.

Para exemplificar segue o código do arquivo principal.php atualizado.

```
<?php require_once ('verificarAcesso.php');?>
<?php require_once ('cabecalho.php'); ?>
<div class="w3-padding w3-text-grey w3-half w3-display-middle w3-center">

    <h1 class="w3-center w3-teal w3-round-large w3-margin">Projeto Lista de Amigos</h1>
    <div class="w3-row">
        <div class="w3-col w3-button w3-teal w3-cell w3-round-large" style="width:45%;">
            <a href="cadastro.php" style="text-decoration: none;">
                <i class=" fa fa-user-plus" style="font-size: 10.5em"></i>
                <p style="font-size: 2em">Adicionar </p>
            </a>
        </div>
        <div class="w3-col w3-button w3-teal w3-cell w3-round-large w3-right" style="width:45%;">
            <a href="listar.php" style="text-decoration: none;">
                <i class="fa fa-vcard-o" style="font-size: 10.5em"></i>
                <p style="font-size: 2em">Listar</p>
            </a>
        </div>
    </div>
</div>
<?php require_once ('rodape.php'); ?>
```

Agora podemos observar o resultado na imagem a seguir, ao tentar acessar essa página através da url.



Imagen 8. Acesso Negado.

Obs: Para saber mais sobre função header para redirecionamento, vale a leitura do link a seguir.

<https://www.hostinger.com.br/tutoriais/redirecionamento-php/>

Por fim vamos criar o logout para quando o usuário decidir deixar nossa página, crie um arquivo chamado “logoutAction.php”, este arquivo vai basicamente remover a variável de sessão “logado” e redirecionar (função header) para a página inicial “index”, o que tornará o login novamente necessário.

```
<?php require_once ('verificarAcesso.php'); ?>
<?php
    unset( $_SESSION['logado'] );
    header("location:index.php");
?>
```

Para o usuário realizar o logout, a criação de um botão no arquivo “principal.php” se faz necessária, o que poder ser realizao inserindo o código a seguir logo depois do: <?php require_once ('cabecalho.php');?>.

```
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-
topright">
    <form action="logoutAction.php" class="w3-container" method='post'>
        <button name="btnLogout" class="w3-button w3-red w3-cell w3-round-
large w3-right w3-margin-right">
            <i class="w3-xxlarge fa fa-times-rectangle"> </i> Logout
        </button>
    </form>
</div>
```

Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nessa agenda.

ATENÇÃO - [Baixe aqui o arquivo usbwebserver – Agenda8](#), nele você encontra todos os arquivos php e tabelas no banco de dados do mergulhando no tema e você no comando.

Com esta etapa concluída, realize diversos testes, implemente diversas atualizações isso vai te trazer o desenvolvimento de mais competências!

Cookie

Cookie são dados ou grupo de dados que browser (navegador) e o servidor Apache trocam dados e informações entre si, enquanto o usuário utiliza as páginas do seu site. Estes dados são salvos em arquivo de texto direto no computador do usuário.

Para criar e manipular um cookie você precisa basicamente aprender basicamente apenas uma função (`setcookie()`), para o primeiro parâmetro deve-se definir um nome para o cookie, o segundo por sua vez refere-se ao valor do cookie.

```
<?php  
setcookie('usuario', 'gabi');  
?>
```

Há também um terceiro parâmetro utilizado para determinar um tempo de “vida” para esse cookie.

```
<?php  
// Criando um cookie para durar um dia  
setcookie('usuario', 'gabi', (time() + (24 * 3600)));  
?>
```

Obs.: Neste exemplo, o cookie está obtendo a hora do servidor, e adicionando a quantidade de segundos de um dia, fazendo com que o cookie tenha a duração por 24 horas (1 dia) em segundos.

Assista o vídeo, e entenda mais sobre cookies e suas utilizações:

Cookie em PHP



Node Studio Terinamentos - Curso de PHP 7 - Aula 52 - \$_COOKIE- Disponível em:
<https://www.youtube.com/watch?v=ilHJx8hB8yA> . Acessado em 29/07/2020.

Hospedagem Registro de Dominios e mais

Para ser possível, acessar sua página, site ou porta de qualquer lugar com internet é necessário alguns recursos: como um domínio e hospedagem. Existem diversas formas para obter esse recursos a seguir uma sequência de 3 vídeos que explica de forma simples e objetiva o mínimo necessário. E também mais alguns vídeos com opções para hospedagens e criação de domínios.

Registro de Domínio | O que é um domínio?



Fonte: Hostnet Internet - Registro de Domínio | O que é um domínio?.

Disponível <https://www.youtube.com/watch?v=bFpSJrzzqPU>. Acessado em 15/07/2020.

Registro de Domínio | Por que eu preciso de um domínio?



Fonte: Hostnet Internet - Registro de Domínio | Por que eu preciso de um domínio?.

Disponível <https://www.youtube.com/watch?v=OcJk8P5imS8>. Acessado em 15/07/2020.

Registro de Domínio | 10 Dicas Para Escolher o Nome do Seu Domínio



Fonte: Hostnet Internet - Registro de Domínio | O que é um domínio?.

Disponível <https://www.youtube.com/watch?v=y5oVU34R0Zk>. Acessado em 15/07/2020.

Hospedagem de Site com Domínio Grátis e SSL (como registrar um domínio na HostGator)



Fonte: Hospedagem de Site com Domínio Grátis e SSL (como registrar um domínio na HostGator).

Disponível em: <https://www.youtube.com/watch?v=DeosapvOQrE>. Acessado em 29/07/2020.

Curso de HTML5 - 35 - Como Hospedar um Site - by Gustavo Guanabara



Fonte: Filipe Deschamps - Curso em Vídeo Curso de HTML5 - 35 - Como Hospedar um Site - by Gustavo Guanabara.

Disponível em <https://www.youtube.com/watch?v=O5mKRNvoWbE>. Acessado em 20/07/2020.

Criando e hospedando sites com a godaddy!



Fonte: Loop Infinito. CRIANDO E HOSPEDANDO SITES COM A GODADDY!.

Disponível em <https://www.youtube.com/watch?v=X8jNCkVPRjQ>. Acessado em 29/06/2020.

Obs: O formato dessa agenda abordou o conceito utilizando apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nessa agenda.

ATENÇÃO - Será disponibilizado o arquivo usbwebserver – Agenda8, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema e você no comando.



Utilizando o que foi visto até agora, foi disponibilizado uma página com o nome, disciplina e dados de vários professores. Nossa missão é restringir o acesso a essa página, através de identificação. Esta identificação deve ser validada através da tabela (professor) criada no **banco de dados** com os atributos: idprofessor, nome, disciplina e senha, conforme diagrama a seguir.

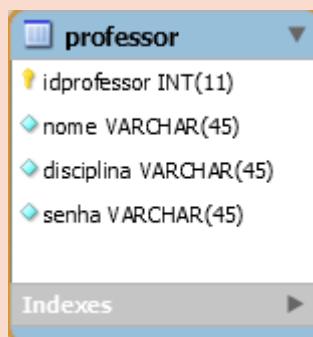


Imagen 16. Diagrama Banco de dados.

1. Obtenha os dados da tabela, utilizando o driver PDO ou Mysqli
2. Crie uma Página para um formulário para login e senha
3. Utilize Session para impedir o acesso via URL da página principal
4. Crie um botão para realizar logout na página em que os professores estão listados
5. Crie arquivos para auxiliar no desenvolvimento

Dicas:

- Utilize os conceitos e exemplos de estrutura de repetição da Agenda 5.
- De preferência ao uso da estrutura de repetição Foreach.
- Agenda 6 ou 7 para acesso ao banco de dados.
- Os conceitos de importação podem te ajudar a economizar tempo

Caso esteja com dificuldade a seguir script sql para criação da tabela estado.

```
CREATE TABLE `pwii`.`professor` (
  `idprofessor` INT NOT NULL AUTO_INCREMENT,
  `nome` VARCHAR(45) NOT NULL,
  `disciplina` VARCHAR(45) NOT NULL,
  `senha` VARCHAR(45) NOT NULL,
  PRIMARY KEY (`idprofessor`));
```

Caso esteja com dificuldade, a seguir script sql inserção de todos registros de estados.

```
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Paulo'
, 'Programação WEB II', 'paulo123');
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Guilherme',
'Programação Mobile', 'guilherme123');
INSERT INTO `pwii`.`professor` (`nome`, `disciplina`, `senha`) VALUES ('Eliana'
, 'Lógica de Programação', 'eliana123');
```

Caso esteja com dificuldade, Código da página dos professores.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
  <title>Listagem Professores</title>
</head>
<body class="w3-black">
```

```

<div class="w3-paddingw3-content w3-half w3-display-topmiddle w3-margin">
    <h1 class="w3-center w3-green w3-margin">Professores</h1>
    <table class="w3-table-all w3-centered w3-text-black">
        <thead>
            <tr class="w3-center w3-green ">
                <th>Código</th>
                <th>Nome</th>
                <th>Disciplina</th>
            </tr>
        <thead>
        <?php
            $servername = "localhost";
            $username = "root";
            $password = "usbw";
            $dbname = "pwii";
            $conexao = new mysqli($servername, $username, $password, $dbname);
            if ($conexao->connect_error) {
                die("Connection failed: " . $conexao->connect_error);
            }

            $conexao->set_charset("utf8");
            if ($conexao->connect_error) {
                die("Connection failed: " . $conexao->connect_error);
            }
            $sql = "SELECT * FROM professor" ;
            $resultado = $conexao->query($sql);
            if($resultado != null)
                foreach($resultado as $linha) {
                    echo '<tr>';
                    echo '<td>' . $linha['idprofessor'] . '</td>';
                    echo '<td>' . $linha['nome'] . '</td>';
                    echo '<td>' . $linha['disciplina'] . '</td>';
                    echo '</tr>';
                }
            $conexao->close();
        ?>
    </table>

```

A seguir, confira se você conseguiu resolver os desafios propostos!



Para a Resposta foram criados os seguintes arquivos:

- Index - contém o formulário para login e action direcionando para o arquivo “LoginAction”.

- LoginAction - contém a consulta no banco para verificação se há o professor ou não, em caso verdadeiro exibirá mensagem de login realizado, com link para página de professores, em caso negativo uma mensagem de falhar com link para a página index.
- LogoutAction - contém a codificação para o logout do usuário conectado e o redirecionando para a página index.
- conexaoBD - criado para concentrar os dados e conexão em um único arquivo.
- acessoNegado - utilizado para quando o usuário tentar realizar o acesso a página sem login.

A seguir, código e imagens de tudo o que foi desenvolvido.



Imagen 17. Index.

Código:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body class = "w3-black">
    <div class="w3-container w3-round-xxlarge w3-display-middle w3-third " style="">
```

```

<form class="w3-container" action="loginAction.php" method="post">
    <div class="w3-section">
        <label style="font-weight: bold;">Usuário</label>
        <input class="w3-input w3-border w3-margin-bottom" type="text" placeholder="Digite o nome" name="txtNome" required>
        <label style="font-weight: bold;">Senha</label>
        <input class="w3-input w3-border" type="text" placeholder="Digite a Senha" name="txtSenha" required>
        <button class="w3-button w3-block w3-green w3-section w3-padding" type="submit">Entrar</button>
    </div>
</form>
<br>

</div>
</div>
</body>
</html>

```

Login Realizado com Sucesso!

Login Inválido!

Imagen 18. LoginAction.

Código:

```

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body class = "w3-black">
<div class="w3-padding w3-content w3-text-grey w3-third w3-display-middle" >

```

```
<?php
session_start();
$nome = $_POST['txtNome'];
$senha = $_POST['txtSenha'];
require_once 'conexaoBD.php';
$sql = "SELECT * FROM professor WHERE nome = '". $nome . "' ;";
$resultado = $conexao->query($sql);
//echo $sql;
$linha = mysqli_fetch_array($resultado);
if($linha != null)
{
    if($linha['senha'] == $senha )
    {
        echo '
            <a href="professor.php">
                <h1 class="w3-button w3-
green">Login Realizado com Sucesso! </h1>
            </a>
        ';
        $_SESSION['logado'] = $nome;
    }
    else
    {
        echo '
            <a href="index.php">
                <h1 class="w3-button w3-green">Login Inválido! </h1>
            </a>
        ';
    }
}
else
{
    echo '
        <a href="index.php">
            <h1 class="w3-button w3-green">Login Inválido! </h1>
        </a>
    ';
}
$conexao->close();
?>
</div>
</body>
</html>
```

Professores

Código	Nome	Disciplina
1	Paulo	Programação WEB II
2	Guilherme	Programação Mobile
3	Eliana	Lógica de Programação

 Logout

Imagen 19. Professor com atualização.

Código:

```

<?php
if(!isset($_SESSION))
{
    session_start();
}
if((!isset($_SESSION['logado']) == true))
{
    header('location:acessoNegado.php');
    die();
}
?>

<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
        <title>Listagem Professores</title>
</head>
<body class="w3-black">
    <div class="w3-padding w3-content w3-half w3-display-topmiddle w3-margin">

```

```

<h1 class="w3-center w3-green w3-margin">Professores</h1>
<table class="w3-table-all w3-centered w3-text-black">
<thead>
    <tr class="w3-center w3-green">
        <th>Código</th>
        <th>Nome</th>
        <th>Disciplina</th>
    </tr>
<thead>
<?php
    require_once 'conexaoBD.php';
    $conexao->set_charset("utf8");
    if ($conexao->connect_error) {
        die("Connection failed: " . $conexao->connect_error);
    }
    $sql = "SELECT * FROM professor" ;
    $resultado = $conexao->query($sql);
    if($resultado != null)
        foreach($resultado as $linha) {
            echo '<tr>';
            echo '<td>' . $linha['idprofessor'] . '</td>';
            echo '<td>' . $linha['nome'] . '</td>';
            echo '<td>' . $linha['disciplina'] . '</td>';
            echo '</tr>';
        }
        $conexao->close();
    ?>
</table>

</div>
<div class="w3-padding w3-content w3-text-grey w3-third w3-margin w3-display-bottomright">
    <form action="logoutAction.php" class="w3-container" method='post'>
        <button name="btnLogout" class="w3-button w3-red w3-cell w3-round-large w3-right w3-margin-right">
            <i class="w3-xxlarge fa fa-times-rectangle"> </i> Logout
        </button>
    </form>
</div>
</body>
</html>

```

Código Conexão BD:

```

<?php

$servername = "localhost";
$username = "root";
$password = "usbw";
$dbname = "pwii";

```

```
$conexao = new mysqli($servername, $username, $password, $dbname);
if ($conexao->connect_error) {
    die("Connection failed: " . $conexao->connect_error);
}

?>
```

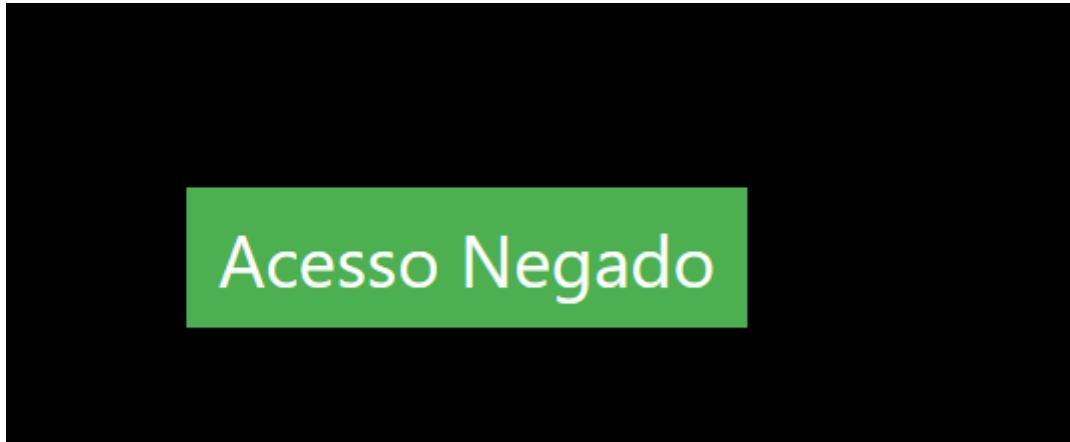


Imagen 20. Professor com atualização.

Código Acesso Negado:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <link rel="stylesheet" href="https://www.w3schools.com/w3css/4/w3.css">
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">
    <title></title>
</head>
<body class ="w3-black">
<div class="w3-padding w3-content w3-third w3-display-middle">

    <?php
        echo '
            <a href="index.php">
                <h1 class="w3-button w3-green">Acesso Negado</h1>
            </a>
        ';
    ?>
</div>
</body>
</html>
```



Obs: O formato dessa da resposta abordou apenas o conteúdo desenvolvido até o momento nas disciplinas, há outros métodos para fazer todas as operações contidas nesse exercício.



ATENÇÃO - Será disponibilizado o arquivo **usbwebserver – Agenda8.rar**, nele terá todos os arquivos php e tabelas no banco de dados do mergulhando no tema, você no comando disponíveis para uso e consulta.

AGENDA 9

**UTILIZANDO
O KODULAR
PARA
DESENVOLVIMENTO
MOBILE**

GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Atualização técnica:

Rogério Galdiano de Freitas

Revisão Técnica:

Eliana Cristina Nogueira Barion

Kelly Dal Pozzo

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação: *Flávio Biazim*

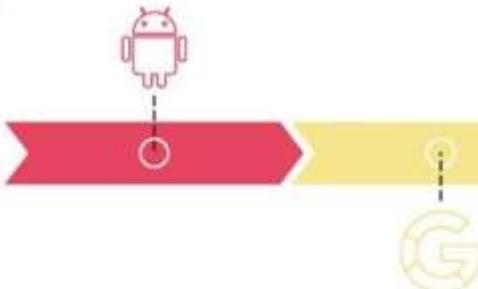


Como todos já sabem, um dispositivo computacional, seja ele qual for, é composto pela sua parte física, denominada **Hardware** e sua parte lógica, chamada de **Software**.

Não é diferente com os dispositivos Mobile, como celulares ou Smartphones, Tablets, etc. Não adiantaria tanta potência e design, sem um sistema operacional para fazer o dispositivo ganhar vida!

Com a evolução dos equipamentos, a indústria necessitava de um sistema genérico, que funcionasse em diversos *Hardwares* diferentes, e que estivesse em constante manutenção e evolução, assim como os dispositivos físicos.

Em setembro de **2003**, para satisfazer as necessidades da indústria, nascia o sistema operacional Android, uma derivação do sistema operacional Linux. A empresa responsável pela criação e desenvolvimento do sistema, diferente do que você imagina, era a empresa **Android Inc.**



A Google pretendia entrar no mercado de dispositivos móveis e, em **2007**, começou a oferecer um sistema com atualização constante para as principais empresas de Hardware do mercado e, logo, começou a fechar acordos comerciais, embarcando seu sistema nesses dispositivos.

Em 17 de agosto de **2005**, a Google, poderosa empresa de tecnologia e atual detentora do desenvolvimento do projeto Android, comprou a **Android Inc.**, juntamente com o projeto do sistema operacional, contratando toda a equipe de desenvolvimento do sistema da empresa recém adquirida.

Outras empresas do setor de telefonia foram convidadas a conhecer e a fazer parte da aceitação ao novo sistema, como também, empresas de desenvolvimento de sistemas. Assim, iniciava-se a ampla disseminação do Android em todo mundo.

Figura 1 - Origem da sistema Operacional Android



O sistema operacional Android, tem seu código aberto pela Google, e muitas empresas fabricantes de aparelhos, já trabalham alterando esse código, customizando e otimizando para se adequar aos seus produtos. Alguns dispositivos hoje, são lançados contendo uma versão do sistema que possui seu código aberto e parte sendo proprietário da empresa do aparelho, em virtude das customizações e otimizações.

Com sua grande utilização, o sistema operacional Android abriu as portas de um novo mercado, o mercado de desenvolvimento Mobile. O Android, possui uma loja de aplicativos, chamada de “Play Store”, que são comercializados aos seus alunos que buscam ferramentas para o lazer, trabalho, estudo, etc. E com a popularização da loja virtual da empresa Google, muitos desenvolvedores, tiveram a oportunidade de publicar seus trabalhos e ferramentas para todo o mundo, onde muitos saíram do anonimato e se tornando milionários.

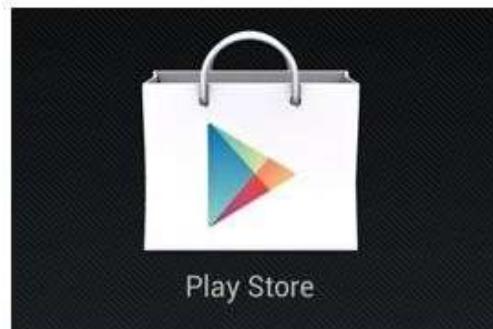


Figura 2 - Aplicativo de compras da loja da empresa Google, chamado de Play Store.

Uma outra grande forma de trabalho para o desenvolvedor, é criar projetos para fins particulares, ou seja, desenvolver aplicativos para empresas como bancos, empresas de vendas, etc.

Para atender cada vez mais a necessidade do mercado de desenvolvimento, foi criada a ferramenta Kodular, para auxiliar e facilitar o desenvolvimento de aplicativos, seja ele para comercialização na loja da empresa Google, ou para atender empresas específicas.



No tópico a seguir, apresentamos o Kodular e assim já estaremos aptos para instalar e executar o nosso primeiro projeto desenvolvido para o sistema operacional Android!

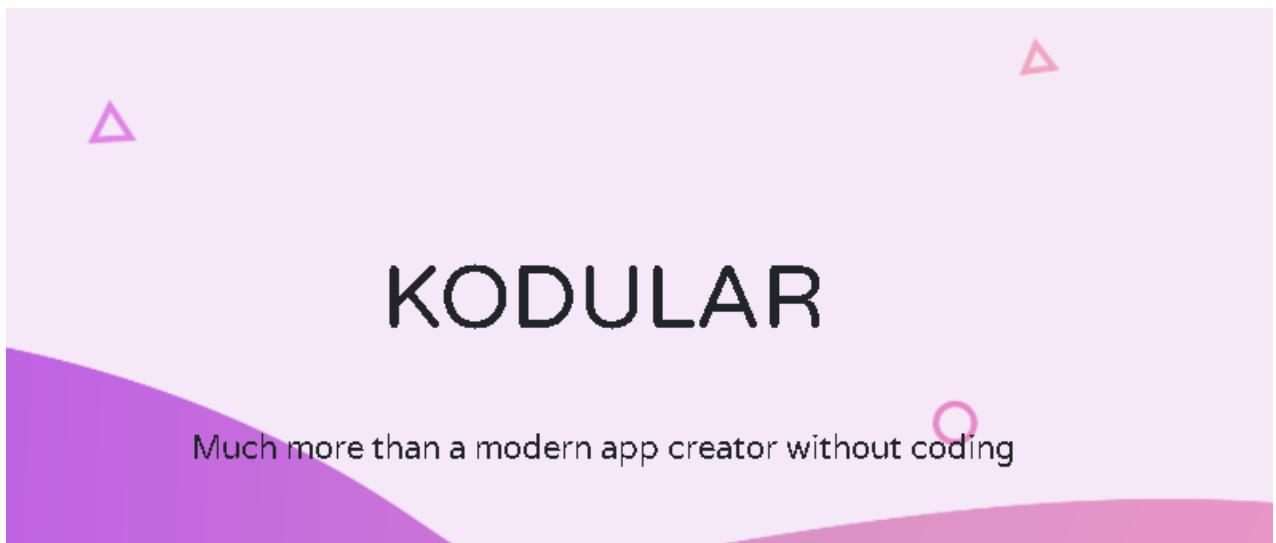


Figura 3 - Tela Inicial da Plataforma Kodular

Com o surgimento e aceitação do sistema operacional Android, o mercado necessitava de ferramentas para atender o desenvolvimento de aplicativos, gerados a partir da linguagem Java.

Por algum tempo, os desenvolvedores utilizavam a ferramenta Eclipse. Para conseguir operar essa ferramenta, era necessário muitos ajustes e configurações. Muitos desenvolvedores abandonaram os estudos nessa área, pelo simples fato de não conseguir, configurar o ambiente e produzir o seu primeiro projeto. Tornando uma experiência frustrante, seu primeiro contato com esse universo da programação Mobile.

Atualmente, existe uma tendência de desenvolvimento de aplicativos através da plataforma Kodular, na qual os criadores deste ambiente já possuem uma certa experiência, porque é uma evolução do Makeroid, uma plataforma baseada na App Inventor, criada no MIT (Massachusetts Institute of Technology). E para o desenvolvimento do aplicativo é utilizado a linguagem de programação em blocos. Resumidamente uma linguagem de programação baseada em blocos funciona como um quebra-cabeça, no qual cada peça é um comando e quando montamos uma sequência de peças conseguimos escrever um programa.

Muitas empresas e desenvolvedores, defendem a ideia de que para a criação de um bom aplicativo, é fundamental que ele tenha desempenho e baixa utilização dos recursos do dispositivo Mobile.

Desta forma, levantam a bandeira de que o desenvolvimento ideal é aquele realizado na ferramenta exclusiva da plataforma, pois essa ferramenta não trabalha com conversões, ela é nativa e desenvolvida para atender as necessidades de um único sistema operacional. É válido lembrar que isso não é uma lei, é apenas a opinião de uma parcela de desenvolvedores.

Por outro lado, o mercado de ferramentas de desenvolvimento multiplataforma cresce, pois otimizam o tempo gasto na produção de aplicativos.

Devidamente apresentada nossa ferramenta de desenvolvimento **Kodular**, que além de robusta e sensacional, conta com a licença de uso gratuita! Vamos aproveitar essa oportunidade ao máximo? Chegou a hora de aprender!

Acesse o site - <https://creator.kodular.io/>

Kodular

O desenvolvimento de aplicativos cresce a cada ano, gerando à falta de mão de obra qualificada. Isto sem mencionar, que a plataforma Android Studio exige um hardware robusto e possui várias etapas para a sua configuração completa. O que pode desanimiar o aluno iniciante nesta trajetória. Portanto a plataforma de desenvolvimento **Kodular** oferta uma ambiente com grande facilidade de manuseio e gratuidade.

O **Kodular** permite a transformação das nossas ideias em aplicativos para o sistema operacional Android, através da programação em blocos. Que permite o simples passo a passo de arrastar, conectar e soltar os blocos (objetos). Não havendo a necessidade de conhecimento de uma linguagem de programação específica, mas sim de uma ótima lógica de programação, para definir a junção dos blocos no desenvolvimento do código do aplicativo.



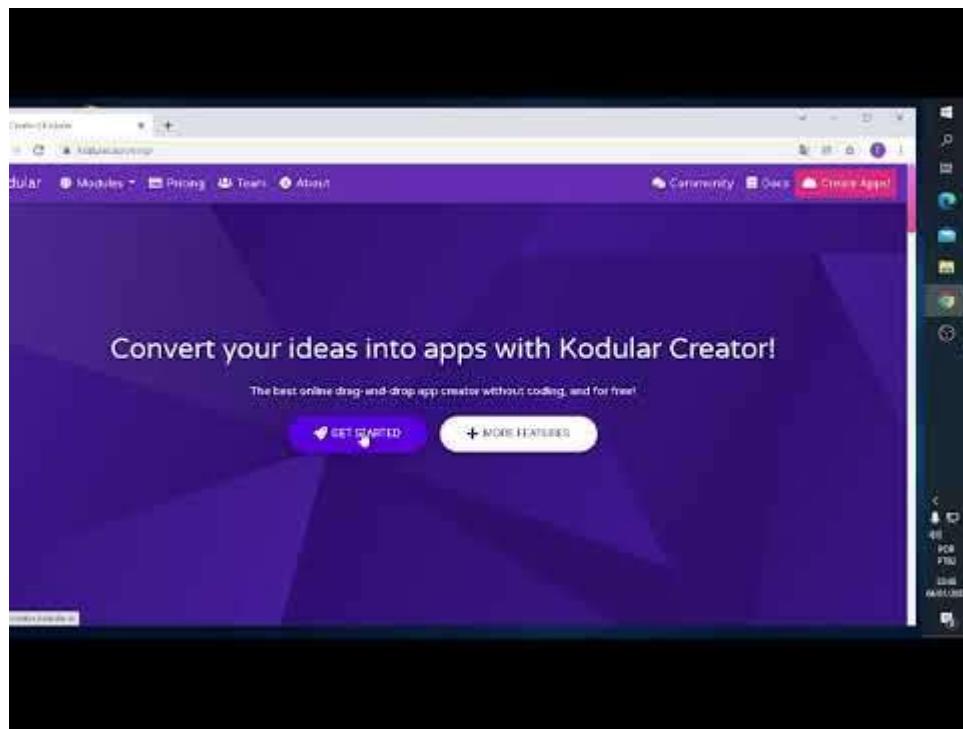
Marcelo, vamos criar nosso primeiro projeto no Kodular?

Para começar a utilizá-la, acesse a plataforma de desenvolvimento digitando na barra de endereço de seu navegador o seguinte link: <https://creator.kodular.io/>

Após acessar o endereço, será apresentado um vídeo com passo a passo para acessarmos a plataforma de maneira fácil. Assista ao vídeo:

Agenda 09 - Acessando a plataforma Kodular, disponível em:

<https://www.youtube.com/watch?v=WliqcNPdW9E>



Ao clicar no botão **Próxima**, a plataforma Kodular solicitará a permissão de autorização para acessar ao ambiente de desenvolvimento. Basta clicar no botão **Authorize**.

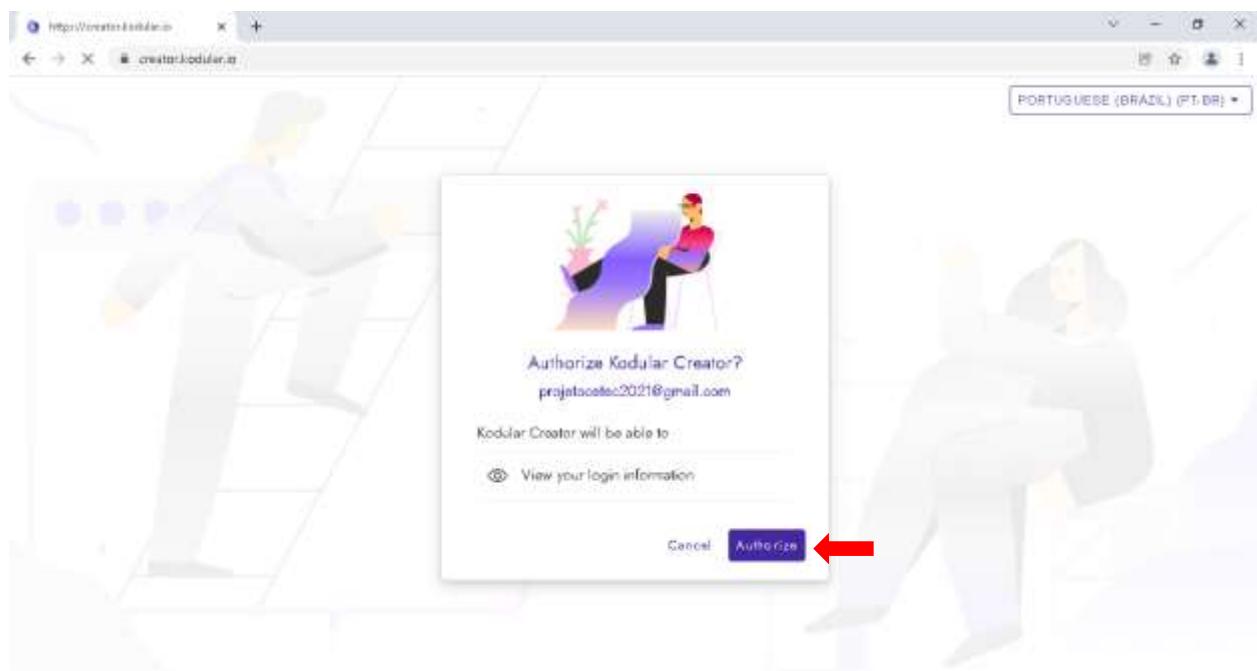


Figura 4 - Pedido de Autorização

No próximo passo, será apresentado o termo de acesso com seus direitos e deveres. Portanto o aluno deverá clicar no botão na parte inferior da janela **I ACCEPT THE TERMS SERVICE**.

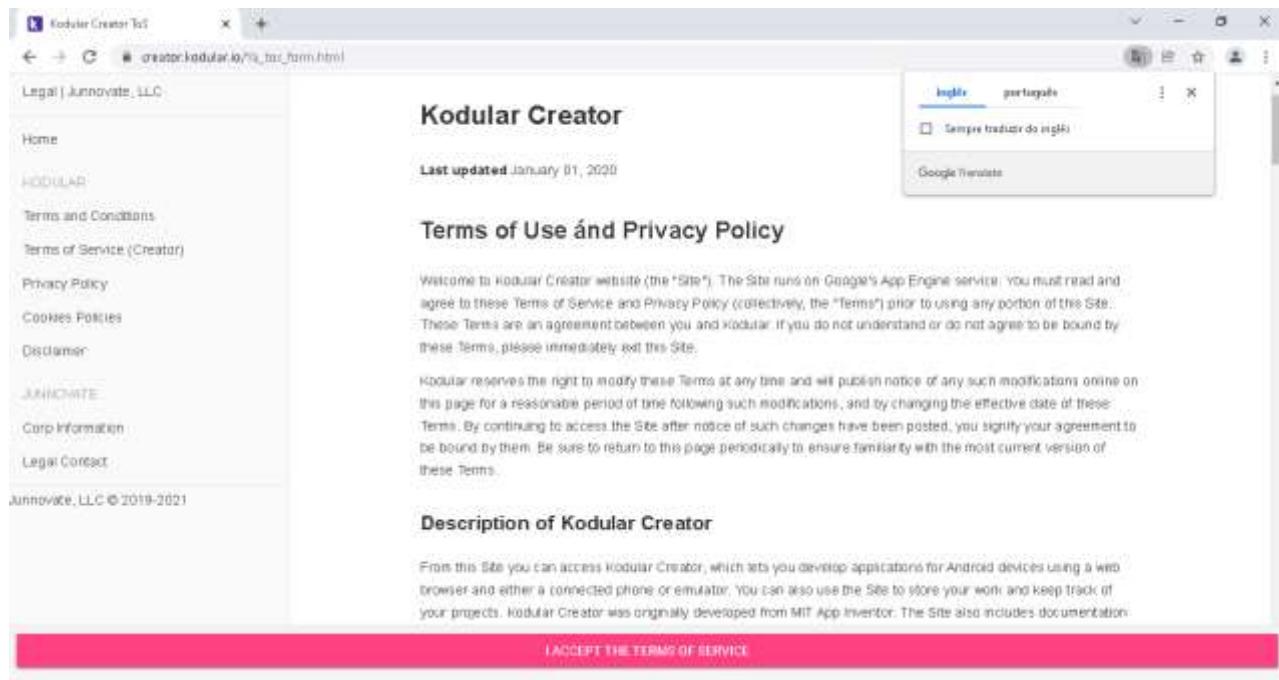


Figura 5 - Autorizando o acesso a plataforma Kodular.

Ao efetuar todos os procedimentos de identificação e autorização, o aluno poderá desfrutar do ambiente de desenvolvimento **Kodular**. Clique no botão **Let's go!**

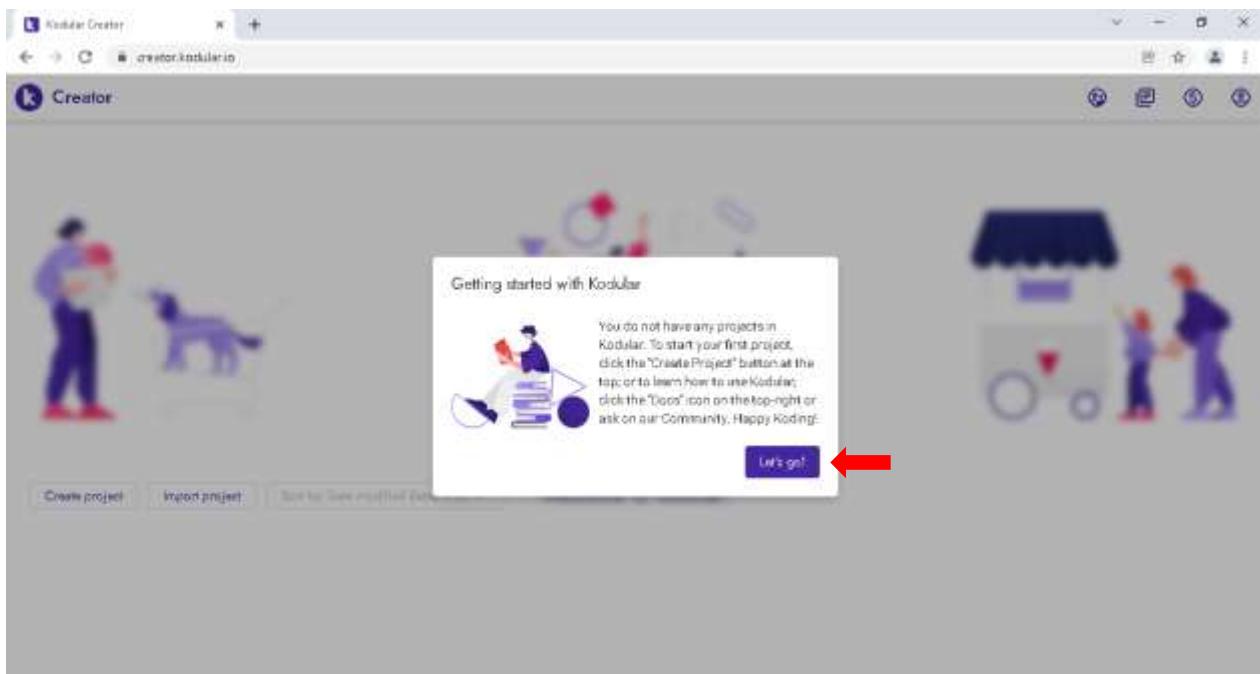


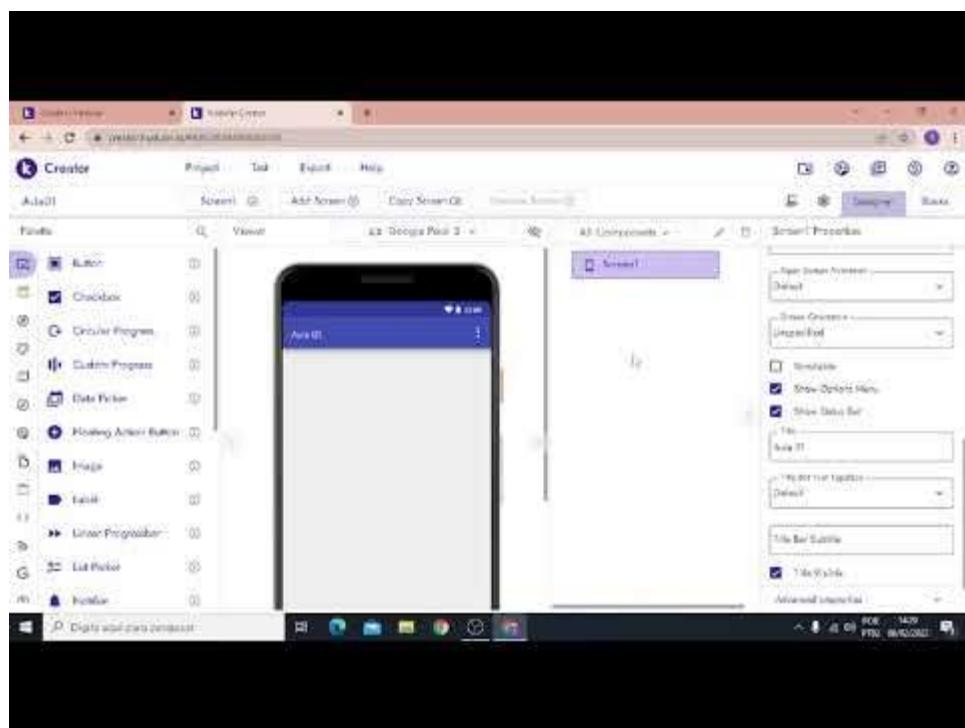
Figura 6 - Vamos começar a explorar a criação de aplicativos.

Ao acessar o ambiente de desenvolvimento, o aluno deverá clicar no botão **Create project** para construir o primeiro projeto.

Para iniciar a criação deste novo projeto, a plataforma solicita o nome do projeto, lembre que o nome do projeto não poderá ser renomeado. Então devemos pensar muito em relação ao nome do projeto antes da sua criação. Sendo que o nome do projeto não pode ser formado por caracteres especiais, acentuação ou espaço em branco. Portanto o nome do projeto deverá ser **Aula01**. Mas para que aluno possa ter uma maior segurança na construção deste projeto, estamos disponibilizando o vídeo para que as dúvidas possam ser solucionadas.

Assista ao video: Agenda 09 – Criando um novo projeto, disponivel em :

<https://www.youtube.com/watch?v=lsGEwJ6y4al>



Pode ser um incômodo criar e instalar o arquivo APK do seu projeto no dispositivo móvel, principalmente pelo motivo que sempre que você fizer uma alteração será necessário efetuar a instalação novamente. Por isso, o **Kodular** possui um recurso que facilita muito o processo de criação de aplicativos - o companheiro Kodular. O companheiro pode ser instalado no seu dispositivo como qualquer aplicativo Android normal. Depois de conectar o companheiro ao construtor, por WiFi ou USB, as alterações feitas no designer serão exibidas instantaneamente no seu dispositivo.

- Acesse a loja do **Play Store** no seu dispositivo móvel
- Pesquise pelo aplicativo **Kodular Companion**

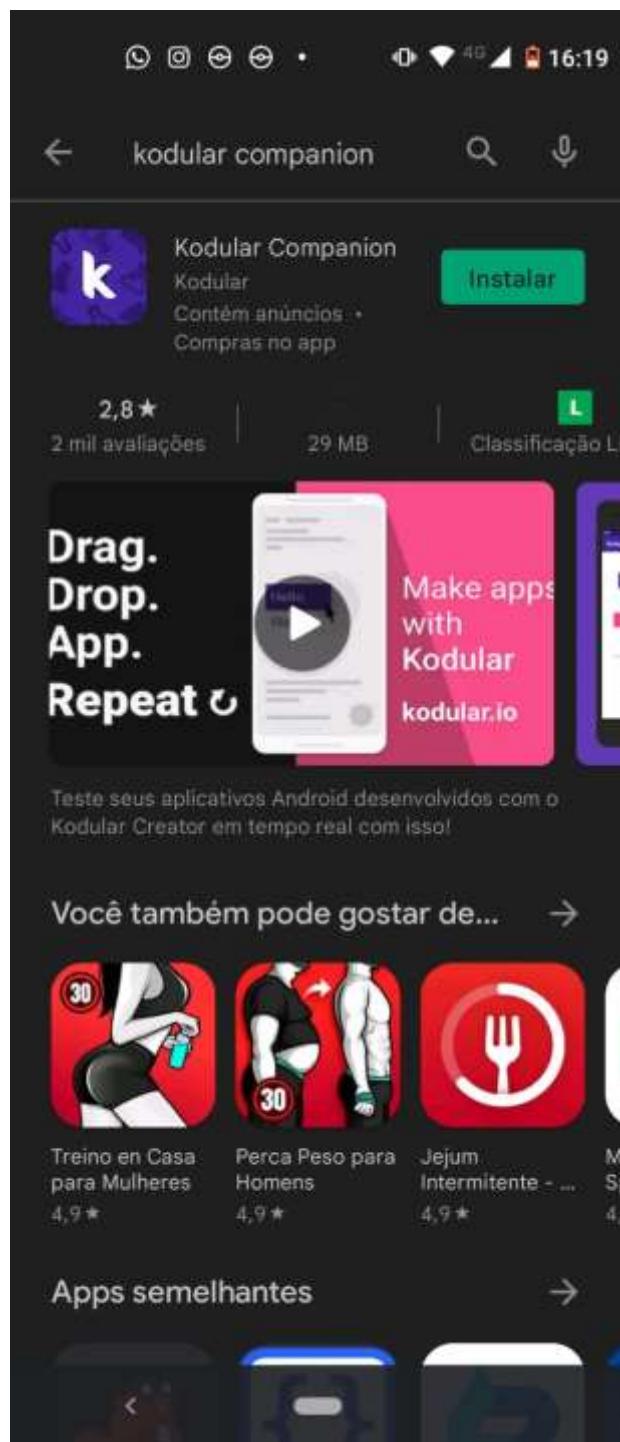


Figura 7 - Play Store - Kodular Companion

- Faça a instalação do aplicativo, clicando no botão **Instalar**
- Abra o aplicativo no dispositivo Móvel
- Retorne a janela do **Kodular** em seu computador e/ou notebook.

- Clique no menu **Test**, opção **Connect to companion**

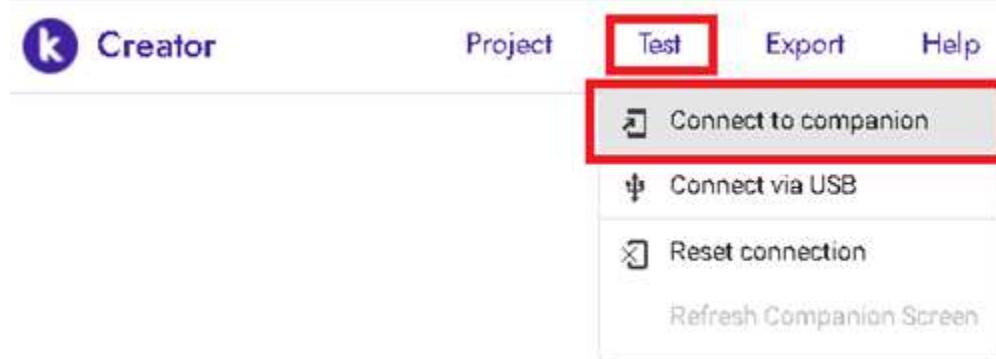


Figura 8 - Criando a conexão companion.

A plataforma **Kodular** irá apresentar um **qrcode** para o aplicativo escanear e acessar o seu projeto. Mas aconselho ao aluno digitar o código ao invés de escanear, pois houve algumas atualizações e o bug surgiu neste processo de escanear.

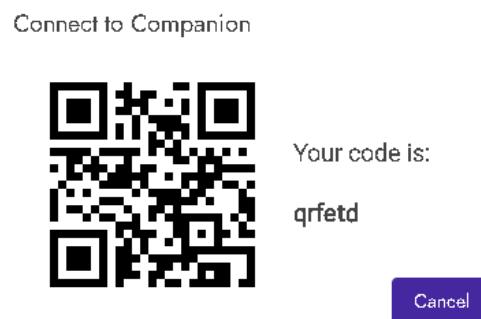


Figura 9 - Qrcode criado com sucesso.

- Digite o código no aplicativo **Kodular Companion** e clique na seta.

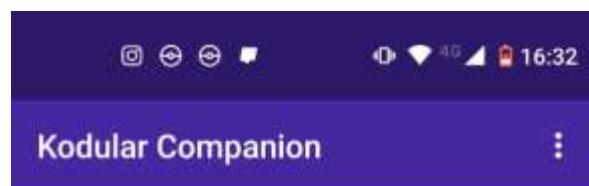


Figura 10 - Aplicativo Kodular Companion

- O dispositivo móvel irá abrir o seu projeto, ou seja, o aplicativo será executado no seu celular.

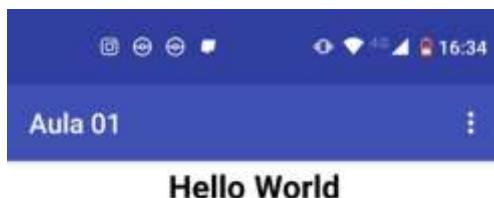


Figura 11 - Aplicativo executando.

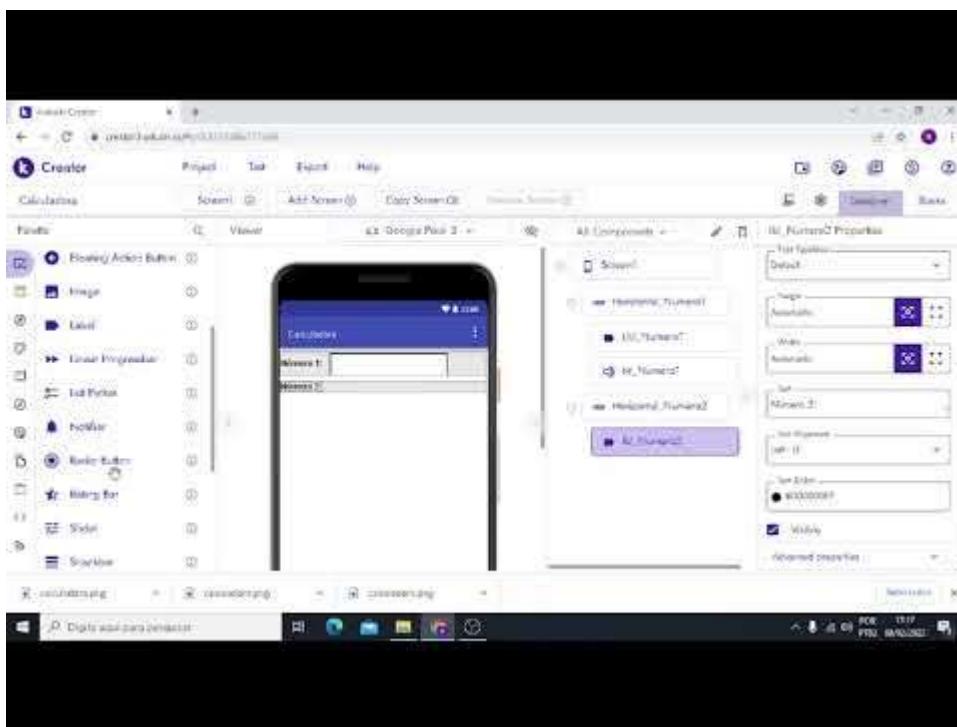
Desenvolvendo o projeto Calculadora de soma entre dois valores

Vamos seguir mergulhando nos estudos do **Kodular**. Agora iremos montar um aplicativo que faça a soma de dois números. Para isso, será necessário o uso da programação em blocos. Mas muita calma, primeiro passo será a construção do layout da nossa calculadora.

Para que a construção do projeto seja realizada de forma plena, o aluno deverá acompanhar o seguinte vídeo.

Agenda 09 – Criando projeto Calculadora, disponível em:

<https://www.youtube.com/watch?v=gsgvYulPMv4>



Terminamos o projeto da Calculadora, agora vamos explicar a forma de transferir o seu aplicativo para o dispositivo móvel. A instalação de forma permanente do aplicativo, para isso precisamos gerar o arquivo .apk que nada mais que o arquivo de instalação para o sistema operacional Android.

- Clique no menu **Export**, opção **Android App (.apk)**.

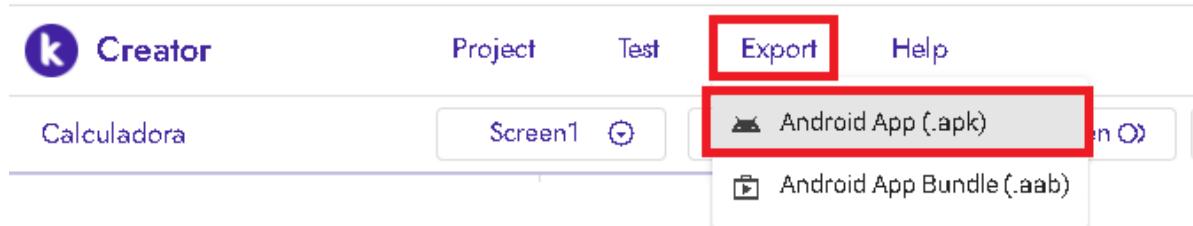
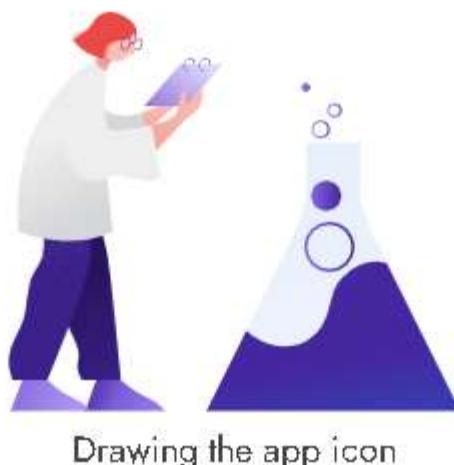


Figura 12 - Menu Export, opção Android app (.apk)

Aguarde alguns instantes, pois a plataforma está gerando o arquivo .apk com todos os recursos utilizados durante a construção do aplicativo.



Drawing the app icon

0%

Figura 13 - Gerando o arquivo .apk

Ao gerar o arquivo .apk para facilitar a plataforma Kodular permite o uso do aplicativo kodular companion para baixar e instalar o aplicativo no dispositivo móvel.

Android App for "Calculadora"

Scan the QR code on your phone to install the app or download the APK file to your computer.

Note: This link is valid only for 10 minutes. It is recommended to export your app as an Android App Bundle for distribution via Google Play.

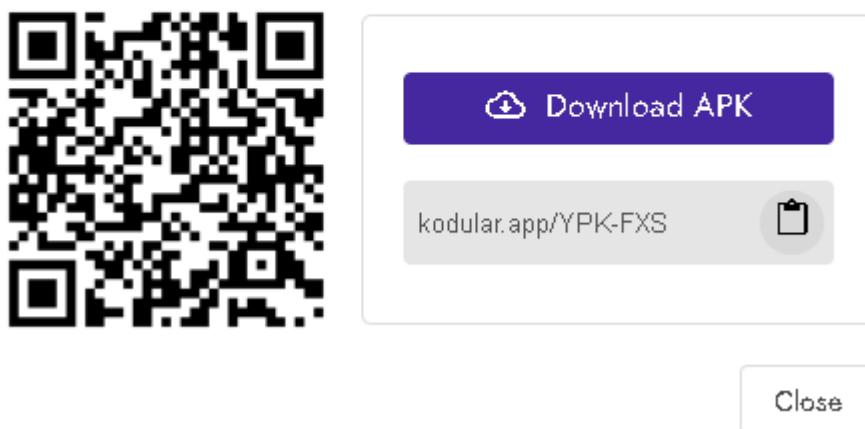
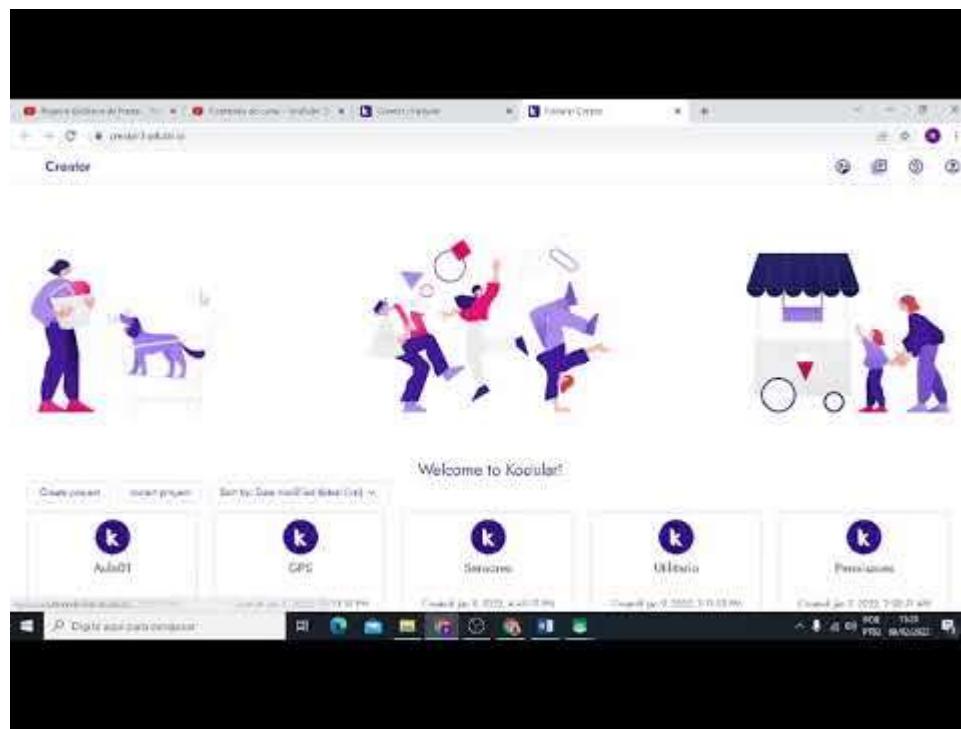


Figura 14 - Disponibilizando o link do apk.

Para que o aluno posso realmente compreender a forma correta para a instalação do aplicativo no seu dispositivo móvel, solicito que assista ao vídeo.

Agenda 09 – Instalando o aplicativo no celular, disponível em:
<https://youtu.be/YPcGEU2U2g4>



Ao término da instalação, o aluno poderá perceber que ao listar todos os aplicativos do dispositivo móvel, encontrará o ícone do novo aplicativo **Calculadora**. Conforme a ilustração da **Figura 15**. Concluindo que a instalação foi realizada com sucesso.



Figura 15 - Listando o aplicativo instalado.

Acesse os projetos finalizados através dos links:

[Exemplos apresentados agenda 09](#)

AGENDA 10

LAYOUT E
ESTRUTURA
DO PROJETO
NO KODULAR



GEEaD - Grupo de Estudos de Educação a Distância

Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO
EM DESENVOLVIMENTO DE SISTEMAS PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Atualização técnica:

Rogério Galdiano de Freitas

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim

Para que você consiga digitar um texto e enviar para seu amigo por meio de um App de troca de mensagens, vários **padrões** e **programações** foram estabelecidas para que o resultado seja alcançado por você, usuário, e seu amigo receba o texto digitado!

O aplicativo é constituído, basicamente, de uma tela e de uma classe responsável pelas ações realizadas por ela.

Nesta agenda, vamos aprender com nosso amigo de estudos, Marcelo, como estabelecer e criar o layout do aplicativo, desenvolvendo as telas e seus componentes. Vamos trabalhar com cores e definir os padrões de desenvolvimento através dos layouts.



Layout

O layout de uma aplicação é algo muito importante, é ele o responsável pela não satisfação do usuário com um determinado aplicativo. Não adianta o aplicativo conter as melhores codificações e desempenho, se o usuário não aprovar o seu layout e usabilidade.

O **Kodular** foi desenvolvido para melhorar a experiência do desenvolvedor durante o processo de criação de uma tela da aplicação. Ele necessita que sua criação seja fielmente apresentada nas telas dos mais diferentes dispositivos, assim como foi desenvolvido.

O Kodular sofreu algumas remodelações em seu “**Editor de Layout**” para que o desenvolvedor utilize o padrão *Layout*, e não pense que isso tornou o processo de construção mais difícil e demorado!

Aconteceu exatamente o inverso, o processo de desenvolvimento de interface ficou mais simples e menos complexo.



Figura 1 - Exemplo de Layout

Para trabalhar com o Layout é necessário compreender como ele funciona. Onde a interface, ou seja, a aparência do seu aplicativo é um fator determinante para o seu sucesso. Então é aconselhável planejar o design antes de elaborar toda a programação. Isto não é perca de tempo e sim um investimento futuro.

Atualmente, a plataforma de desenvolvimento do **Kodular**, permite a criação de formatos ainda mais sofisticados, com objetos já predefinidos que aceleram muito o cronograma de entrega em relação as outras plataformas de desenvolvimento. Portanto este material irá trabalhar especificamente com a categoria **Layout** do **Kodular**.

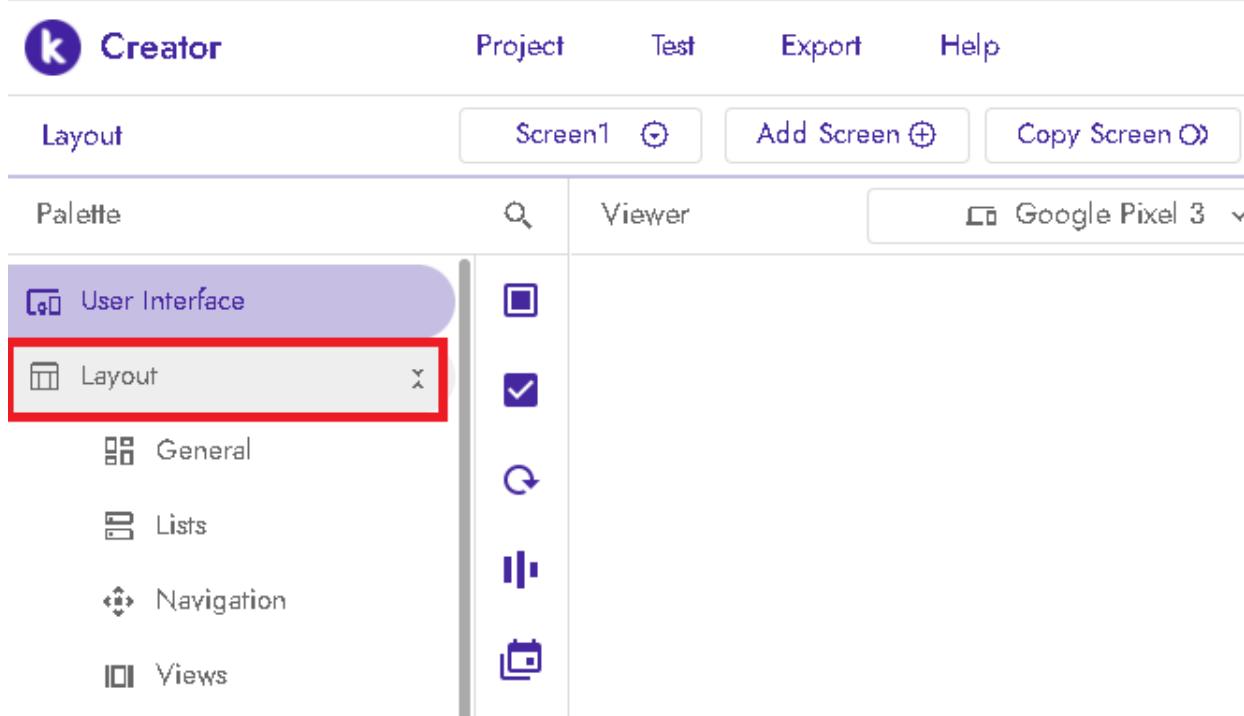


Figura 2 - Aba Layout, plataforma Kodular.

Para iniciarmos o assunto, a categoria **General** será fundamental, pois permite a diagramação principal que irá estruturar o layout do site. Através da tabela abaixo, descrevo um breve resumo de cada opção.

Componente	Função
Card View	Um componente visível que permite ao usuário agrupar outros componentes como um Cartão. Os cartões são painéis com uma elevação e um raio de borda definidos, destinados a chamar a atenção do usuário para o seu conteúdo.
Grid View	Um componente visível que agrupa outros componentes em uma grade bidimensional com rolagem.

Componente	Função
 Horizontal Arrangement	Um elemento de formatação no qual colocar componentes que devem ser exibidos da esquerda para a direita. Se você deseja que os componentes sejam exibidos uns sobre os outros, use VerticalArrangement.
 Horizontal Scroll Arrangement	Um elemento de formatação no qual colocar componentes que devem ser exibidos da esquerda para a direita. Com a barra de rolagem, caso exista muito itens.
 Space	Um componente visível que cria espaços entre os componentes.
 Swipe Refresh Layout	Um componente visível que agrupa outros componentes e permite que o usuário os atualize com um gesto de deslizar para baixo.
 Table Arrangement	Um elemento de formatação no qual colocar componentes que devem ser exibidos em formato tabular.
 Vertical Arrangement	Um elemento de formatação no qual colocar componentes que devem ser exibidos um abaixo do outro. Se você deseja que os componentes sejam exibidos próximos um do outro, use HorizontalArrangement.
 Vertical Scroll Arrangement	Um elemento de formatação no qual colocar componentes que devem ser exibidos um abaixo do outro. Com a barra de rolagem, caso exista muito itens.

Arquivos necessários para construção deste projeto.

[Faça o download das imagens.](#)

Primeiro Layout

O primeiro layout, utiliza os componentes: **Vertical Arrangement**, **Horizontal Arragement**, **Card View** e o **Image** (Categoria **User Interface**). A ideia principal de utilizar o componente Vertical é para agrupar todos as opções do Menu Principal, caso haja a necessidade de barra de rolagem estão todos em um mesmo grupo. A facilidade de gerenciamento é maior. Já por outro lado, o Horizontal é para permitir a inclusão de duas opções na mesma linha, onde o componente Card View, pode controlar cada opção como um botão.

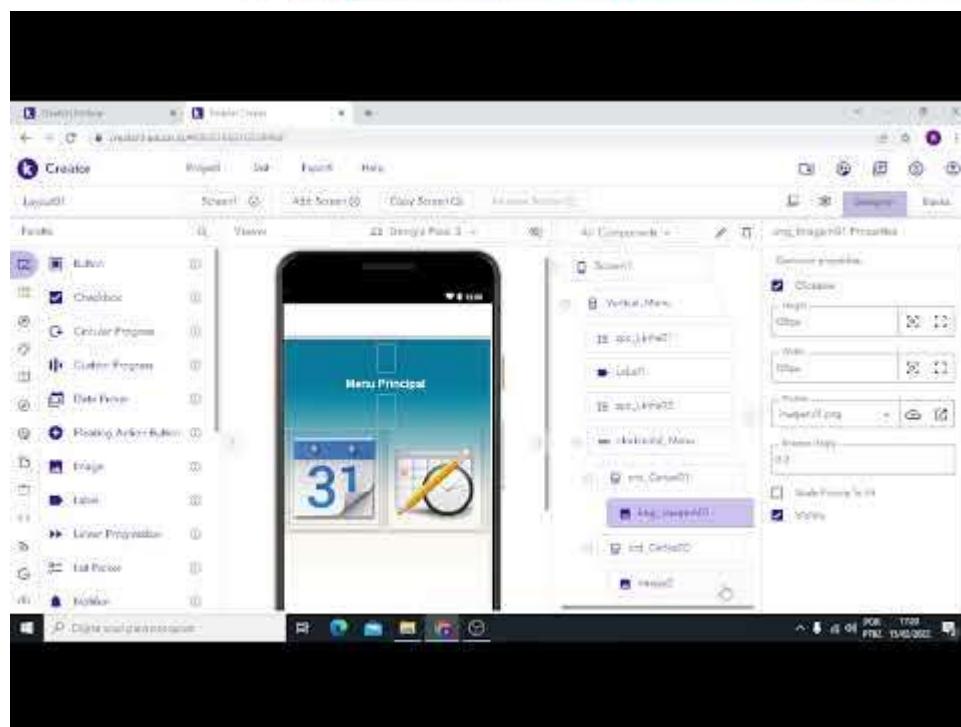


Figura 3 - Primeiro Layout

Vamos descrever o passo a passo, para a construção do primeiro layout. Para que as aulas se tornem mais interativas, estamos disponibilizando um vídeo referente a construção deste **Layout**, então assista ao vídeo:

Agenda 10 - Criando o projeto Layout01, disponível em:

<https://youtu.be/pYO9KyyZcgc>



Terminamos o projeto da Layout, agora vamos explicar a forma de transferir o seu aplicativo para o dispositivo móvel. A instalação de forma permanente do aplicativo, para isso precisamos gerar o arquivo .apk que nada mais que o arquivo de instalação para o sistema operacional Android.

- Clique no menu **Export**, opção **Android App (.apk)**.

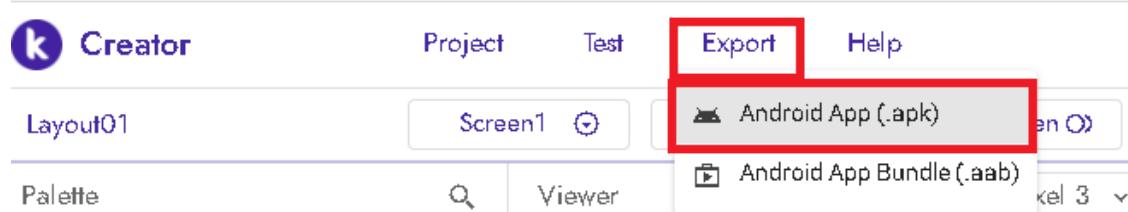


Figura 4 - Menu Export, opção Android app (.apk)

Aguarde alguns instantes, pois a plataforma está gerando o arquivo .apk com todos os recursos utilizados durante a construção do aplicativo.



Drawing the app icon

0%

Figura 5 - Gerando o arquivo .apk

Ao gerar o arquivo .apk para facilitar a plataforma Kodular permite o uso do aplicativo kodular companion para baixar e instalar o aplicativo no dispositivo móvel.

Android App for "Layout01"

Scan the QR code on your phone to install the app or download the APK file to your computer.

Note: This link is valid only for 10 minutes. It is recommended to export your app as an Android App Bundle for distribution via Google Play.

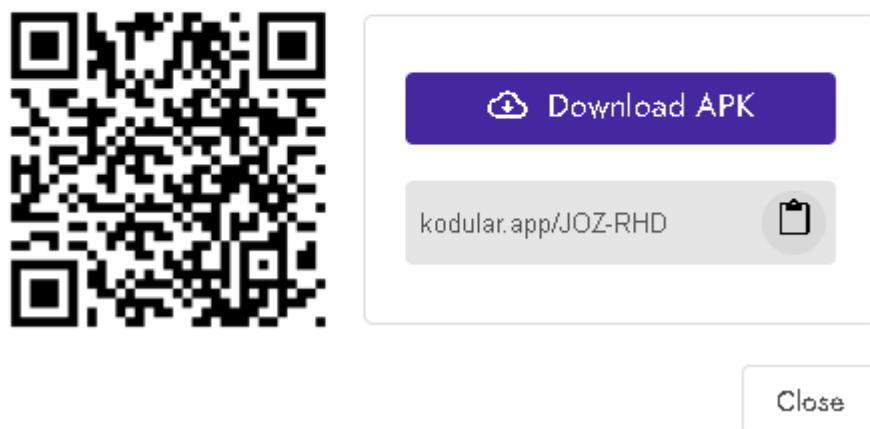


Figura 6 - Disponibilizando o link do apk

- No dispositivo móvel, abra o aplicativo Kodular Companion e peça para escanear o qrcode apresentada pela plataforma.

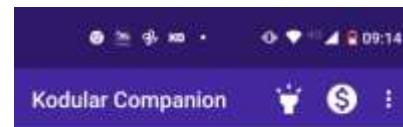


Figura 7 - Kodular Companion

Ao escaner o qrcode o sistema operacional Android apresentará uma informação, solicitando a confirmação para baixar e instalar o aplicativo, pois o mesmo não é oferta pela loja da Play Store, por questão de segurança.

- Clique no botão **OK** para confirmar o download e a instalação.

Arquivos disponíveis para download

Projeto Finalizado

Imagens utilizados no projeto: pasta Imagens

AGENDA 11

LAYOUT DO
PROJETO, TELA
DE SPLASH,
IMAGENS E
CORES



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Atualização técnica:

Rogério Galdiano de Freitas

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim



Vamos agora desenvolver o aplicativo “**Conversor de Medidas**” proposto pelo Gustavo. Esse aplicativo conta com alguns recursos básicos de customização da interface do usuário. Essas ferramentas são utilizadas para que a interface seja agradável para quem utiliza o sistema. Em um primeiro momento vamos elencar alguns pontos importantes nesse processo, começamos pelas cores utilizadas.

Dicas para o desenvolvimento da interface do usuário

A escolha das cores é fundamental para o desenvolvimento da interface do usuário. Todos nós já sabemos que a escolha e combinação de cores não deve ocorrer por acaso. É um processo que requer um estudo sobre o significado de cada cor. Você deve se lembrar das agendas de Design Digital, que estudou no módulo II! O vermelho, por exemplo, é considerado um tom quente, porque transmite energia e coragem, facilitando até mesmo uma ação do usuário. Já o verde, é considerado uma cor fria, por transmitir segurança.

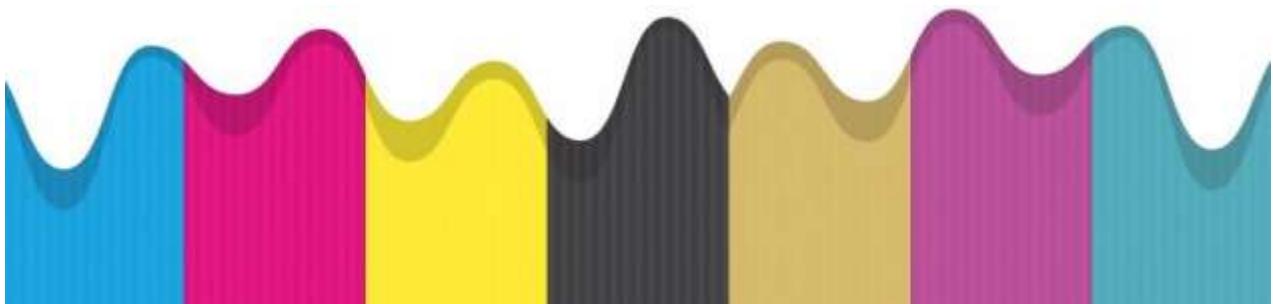


Figura 1 - Fonte: www.freepik.com

Utilizar algumas técnicas para escolher as cores certas é fundamental para que o projeto fique harmonioso e atrativo, contudo, a validação por parte do cliente é também muito importante. Imagine o desastre que seria, caso o desenvolvedor de um aplicativo da Coca-Cola utilizasse a cor azul, da Pepsi, sua principal concorrente!

Por isso, é muito importante levar em consideração as aulas de Design Digital, estudadas no módulo I, para saber como utilizar as cores e combinações a fim de trazer harmonia para os seus aplicativos. Ao final dessa agenda você encontra um vídeo sobre cores. Vale a pena conferir!

Outro ponto importante diz respeito a utilização de botões. É fundamental destacar com cores ou movimentos os botões responsáveis por desempenhar ações importantes em seu aplicativo.

Imagine que no projeto “Conversor de Medidas” o botão para chamar o conversor de “Km para Metros” fique com um *layout* que o deixe discreto perante os demais recursos. O usuário pode pensar que aquela função está desabilitada, por isso é importante fazer com que esses botões e componentes fiquem perceptíveis. A padronização do estilo utilizado nesses botões também é fundamental. Utilize simetria nos tamanhos e alinhamento, garantindo uma organização ao seu aplicativo.

A preocupação com ícones, logos e imagens também é importante! Eles devem ser usado apenas se for necessário para que não gere poluição visual ou até mesmo uma desordem no layout do aplicativo. Cuidado com imagens e fotos, elas devem respeitar e gerar uma harmonia com as cores utilizadas.



Figura 2 - Fonte: www.freepik.com

O acesso aos recursos do seu aplicativo deve ser fácil e intuitivo para o usuário. Imagine que para entrar na função do conversor de “Km para Metros” do app, o usuário tenha que clicar em quatro botões e deslizar duas telas. Isso se torna uma tarefa desgastante, gerando assim desestímulo para acessar as funções do aplicativo mobile.

Tela Splash

A tela de “*Splash*” é a tela de abertura de uma aplicação. Essa tela já foi muito mais explorada na programação de computadores, e atualmente é mais encontrada em aplicativos de jogos. Sua utilização no desenvolvimento mobile é garantir uma propaganda ao usuário para que ele fixe o nome do aplicativo ou até mesmo do desenvolvedor ou proprietário do sistema.

Uma outra função é que durante a exibição da tela de “*Splash*” o aplicativo pode carregar recursos para o seu funcionamento, transmitindo ao usuário sensação de que o sistema não “travou”.

É necessário atenção na utilização deste recurso: uma tela de “*Splash*” que demore muito pode levar a impaciência do usuário. Se você já passou por alguma situação como essa deve saber como é exaustivo ter que esperar um tempo muito longo para a execução dessa tela. Por isso, é importante saber dosar o tempo de execução da tela de *Splash*.



VOCÊ NO
COMANDO

Desenvolvimento da tela Splash

Chegou a hora de desenvolver o aplicativo “Conversor de Medidas”, para isso, crie na plataforma **Kodular** um novo projeto com o nome de “ConversorMedidas”.

Desenvolvimento do Conversor de Medidas

- Abra a plataforma de desenvolvimento do Kodular: <https://www.kodular.io/creator>
- Clique no botão **Create Project**

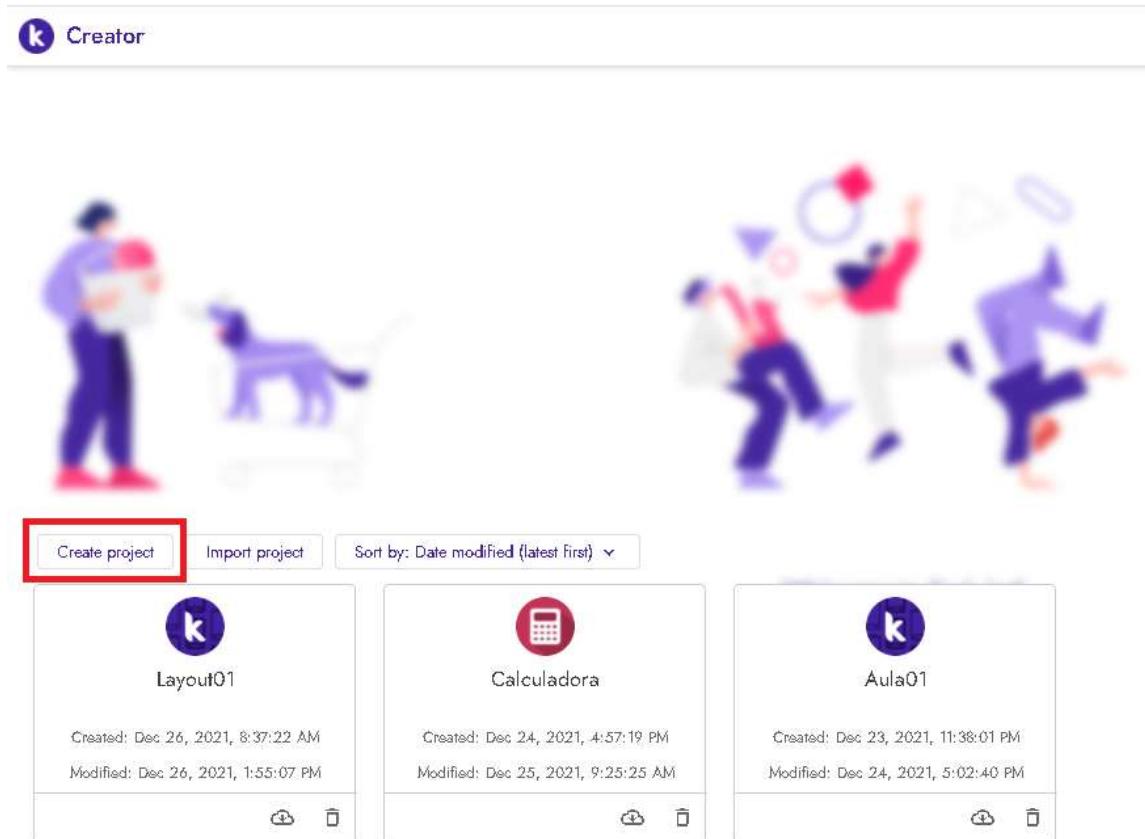


Figura 3- Criando o novo Projeto.

- Digite o nome **ConversorMedida** e clique no botão **Next**.



Figura 4 - Criando o projeto ConversorMedida

- Clique no botão **Finish** para finalizar a criação do novo projeto.

A plataforma de desenvolvimento **Kodular** possui alguns recursos que permitem ao desenvolvedor uma agilidade na construção do aplicativo, maior que outras plataformas. Durante as atualizações desta plataforma, a equipe de desenvolvimento já criou uma opção otimizada para criação de tela **Splash**. Com o mínimo de código possível. Então vamos aproveitar as novidades.

Inicialmente o aluno deverá montar uma imagem que será usada como tela de **Splash** no aplicativo em desenvolvimento. A sugestão será uma imagem e uma frase escrita. Uma propaganda simples que será apresentada todas as vezes que o aplicativo foi aberto.

- Clique no botão **Configurações**.



Figura 5 - Botão de Configurações.

- Ao acessar a janela de **Configurações**, o aluno deverá verificar se a opção **Splash Screen** está **selecionada**. Lembre de deixar esta opção marcada para este projeto.

Project settings X

- General**
- Theming
- Publishing
- Monetization
- API Keys

Your app's icon is displayed on your phone's launcher alongside the app name. Use a square or round .png image of size 512x512px for best results. [More information](#)

Splash Screen

If enabled, your app will show a splash screen while your app is first launched and is loading. A splash screen can be used to display custom branding or loading progress.

Splash Screen (highlighted)

Splash Image

If the splash screen property is enabled, this image will be used as the branding artwork while your app is loading

Accepted Shared File Types

These are the types of files that can be shared to your app from other apps. You can capture these shared files in Screen1.

Figura 6 - Configurando a tela de Splash.

Para facilitar a criação do nosso aplicativo, será disponibilizada uma imagem genérica para ser apresentada como tela de **Splash**.

*Figura 7 - Imagem de Splash.*

- Na opção **Splash Image**, o aluno deverá fazer o upload da imagem e depois selecioná-la como a imagem splash.

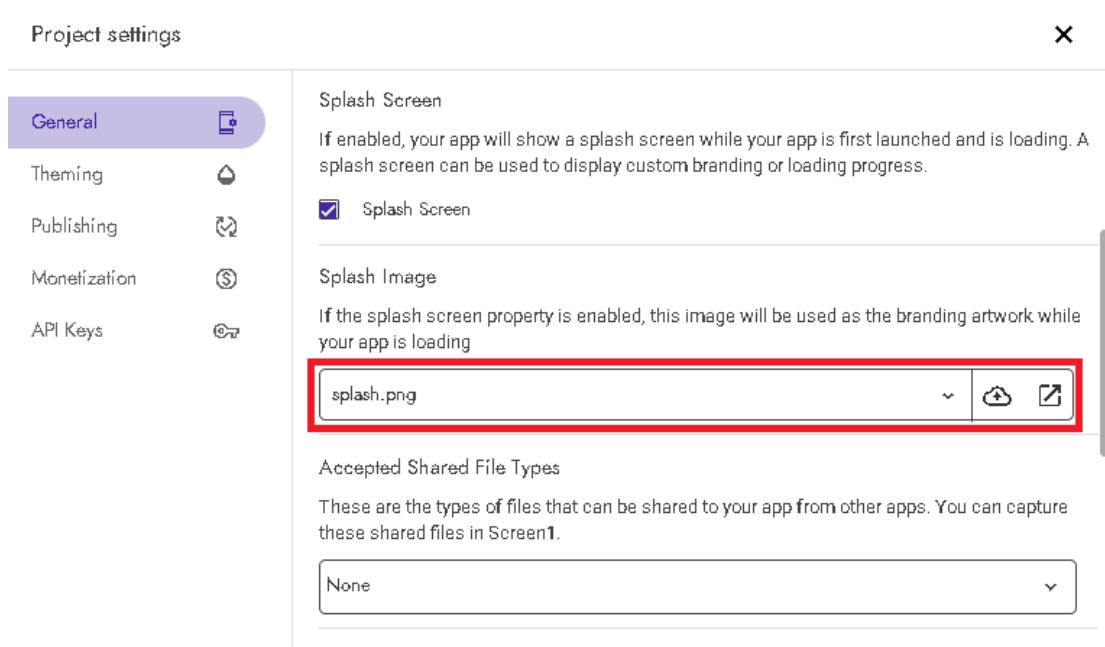


Figura 8- Inserindo a Imagem de Splash.

- Altere a opção **Icon** para imagem **icone.png**

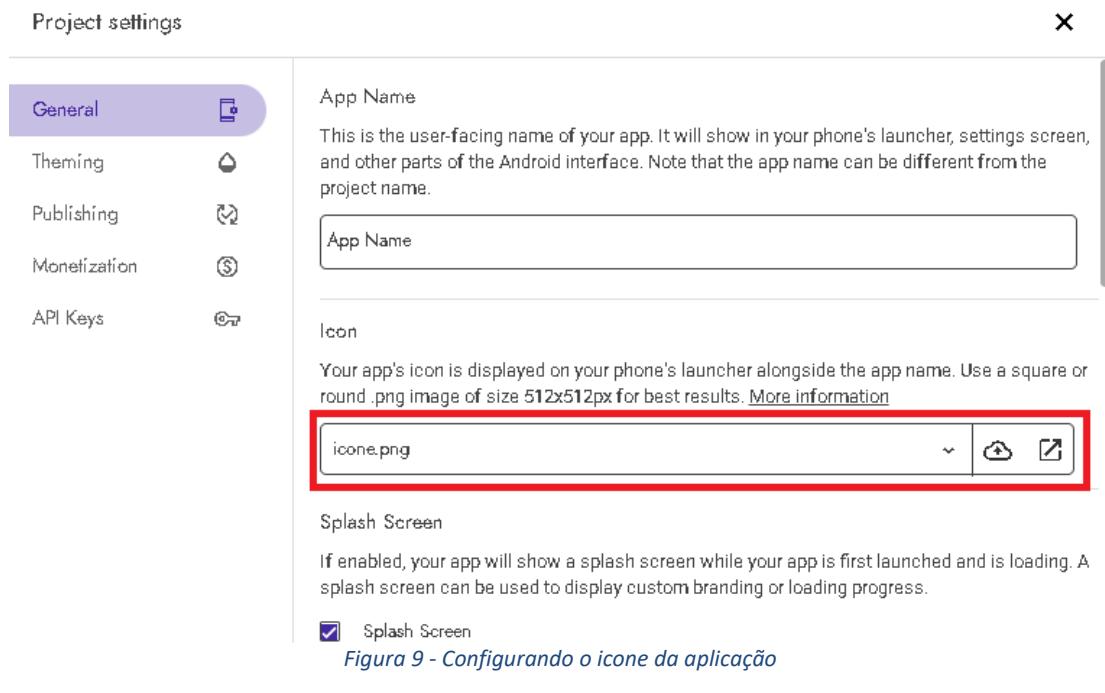


Figura 9 - Configurando o ícone da aplicação

- Feche a janela de **Configurações**.

- Altere as propriedades do objeto **SCREEN**

Propriedade	Valor	Função
Title	Conversor de Medidas	Definir o título da aplicação em desenvolvimento.
Align Vertical	Center	Alinhar todos os componentes ao centro.

- Inserir o componente **Space** da categoria **Layout**, categoria **General** na área de **VIEWER**.
- Altere as propriedades do objeto **Space**

Propriedade	Valor	Função
Height	40px	Definir a altura em px para representar o espaçamento.
Name	Espaco	Definir o nome do componente

- Clique no componente **Label** e insira na área de **VIEWER**.
- Altere as propriedades do objeto **Label**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Width	Fill Parent	Definir a largura total disponível para o componente.
Text	Converter Kilômetros em Metros	Definir o texto que será apresentado pelo Label .
Name	Lbl_Titulo	Definir o nome do componente.

Apos a inserção de todos os componentes e as respectivas alterações nas propriedades, o layout o aplicativo deverá ser idêntico a **Figura 10**.



Figura 10 - Visualizando o Layout.

- Insira o componente **TextBox** (categoria **User Interface**) na área **VIEWER**, para que o usuário possa digitar o valor em quilômetros que será convertido em metros.

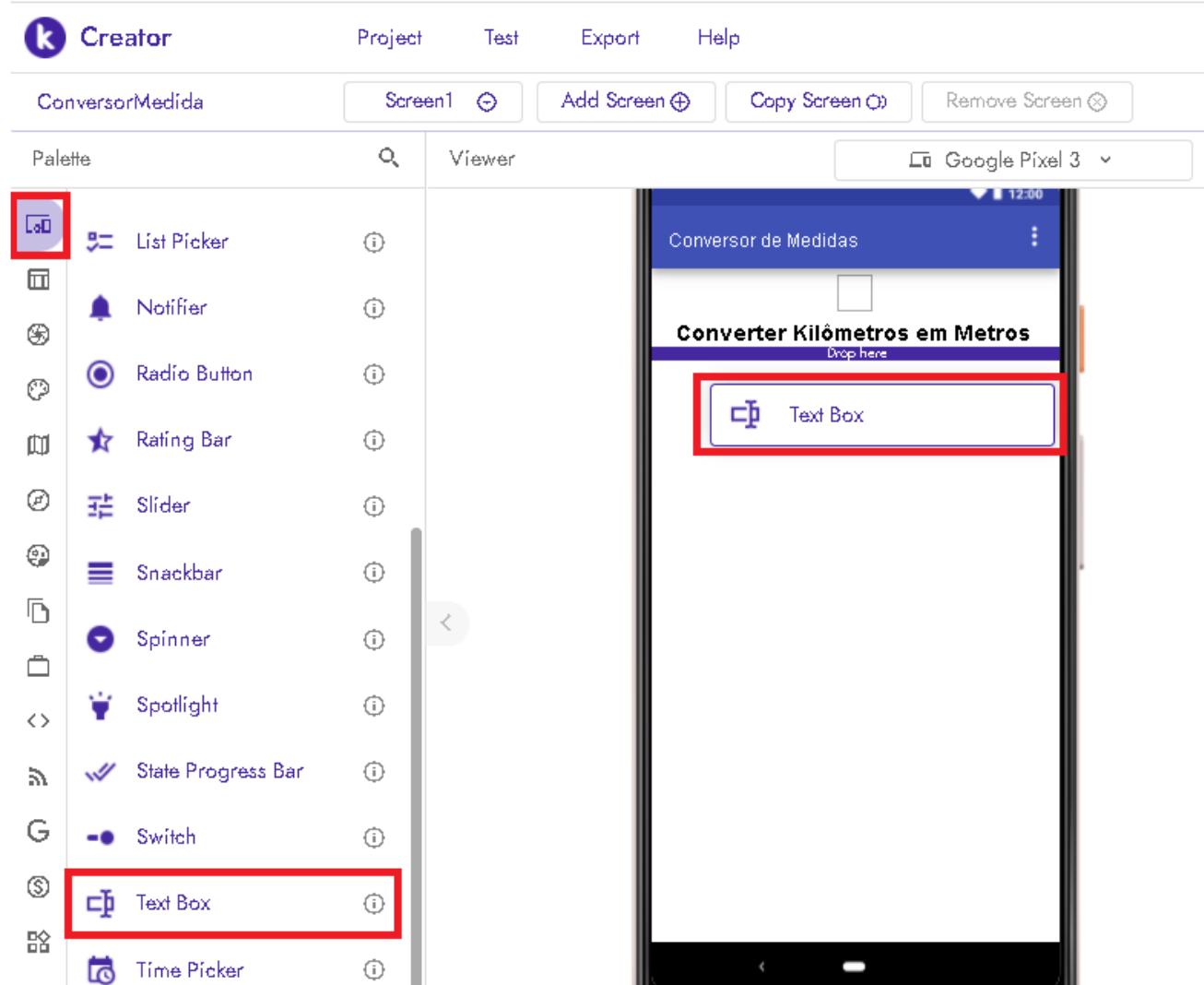


Figura 11- Inserindo componente TextBox.

- Altere as propriedades do componente **TextBox**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Width	Fill Parent	Definir a largura total disponível para o componente.
Text	Em Branco	Definir o texto que será apresentado pelo TextBox .
Hint	Em Branco	Definir sem dicas no campo.
Input Type	Number	Definir a entrada de somente dígitos

Propriedade	Valor	Função
		numéricos.
Name	txt_Km	Definir o nome do componente.

- Insira o componente **Button**

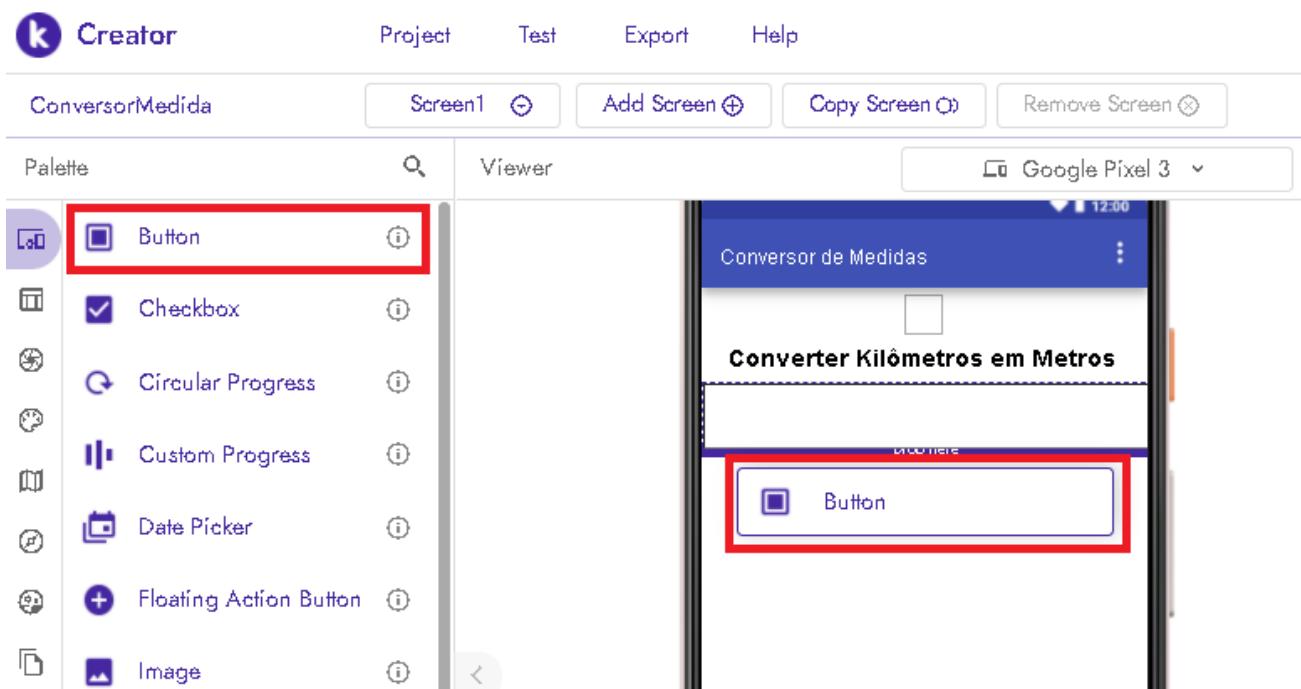


Figura 12 - Inserindo componente Button

- Altere as propriedades do componente **Button**

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Width	Fill Parent	Definir a largura total disponível para o componente.
Text	Converter	Definir o texto que será apresentado pelo Button .
Background Color	#03A9F3FF	Definir a cor de fundo do botão.
Name	btn_Converte	Definir o nome do componente.

- Clique no componente **Label** (Categoria **User Interface**) e insira na área de **VIEWER**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.

Propriedade	Valor	Função
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Text	Resultado	Definir o texto que será apresentado pelo Label .
Name	Lbl_Resultado	Definir o nome do componente.

Para conferir se todos os componentes foram inseridos de forma correta, a **Figura 13** abaixo ilustra o layout da aplicação.

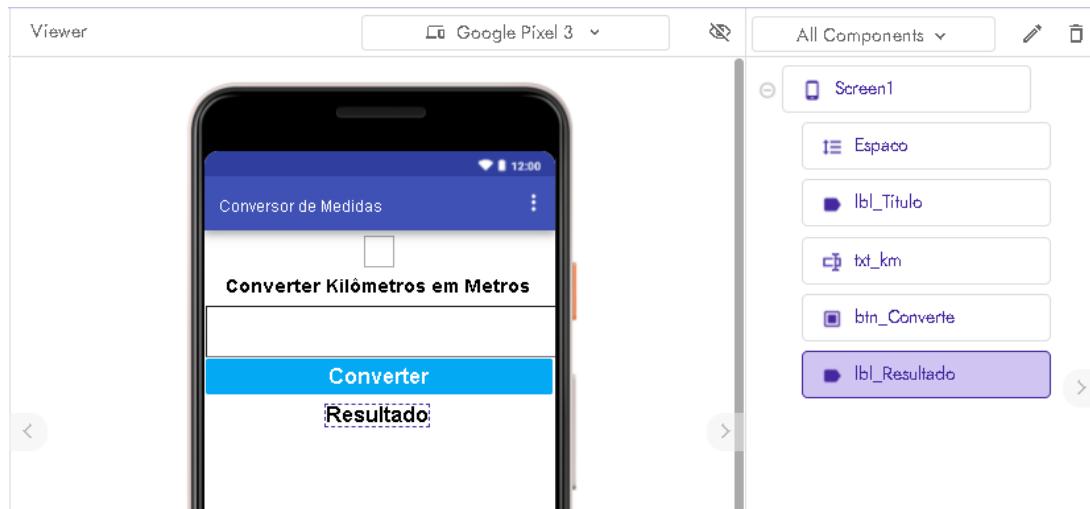


Figura 13 - Visualizando o Layout.

Agora chegou a hora da programação, vamos trabalhar com os blocos para que o resultado da aplicação seja feita de forma correta.

- Altere para a opção de **Blocks**.

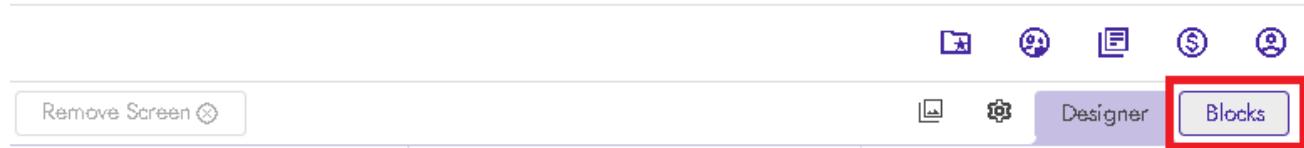


Figura 14 - Alterando a visão de blocks.

- Selecione o objeto **btn_Converte** e arraste a objeto **when btn_converte click**

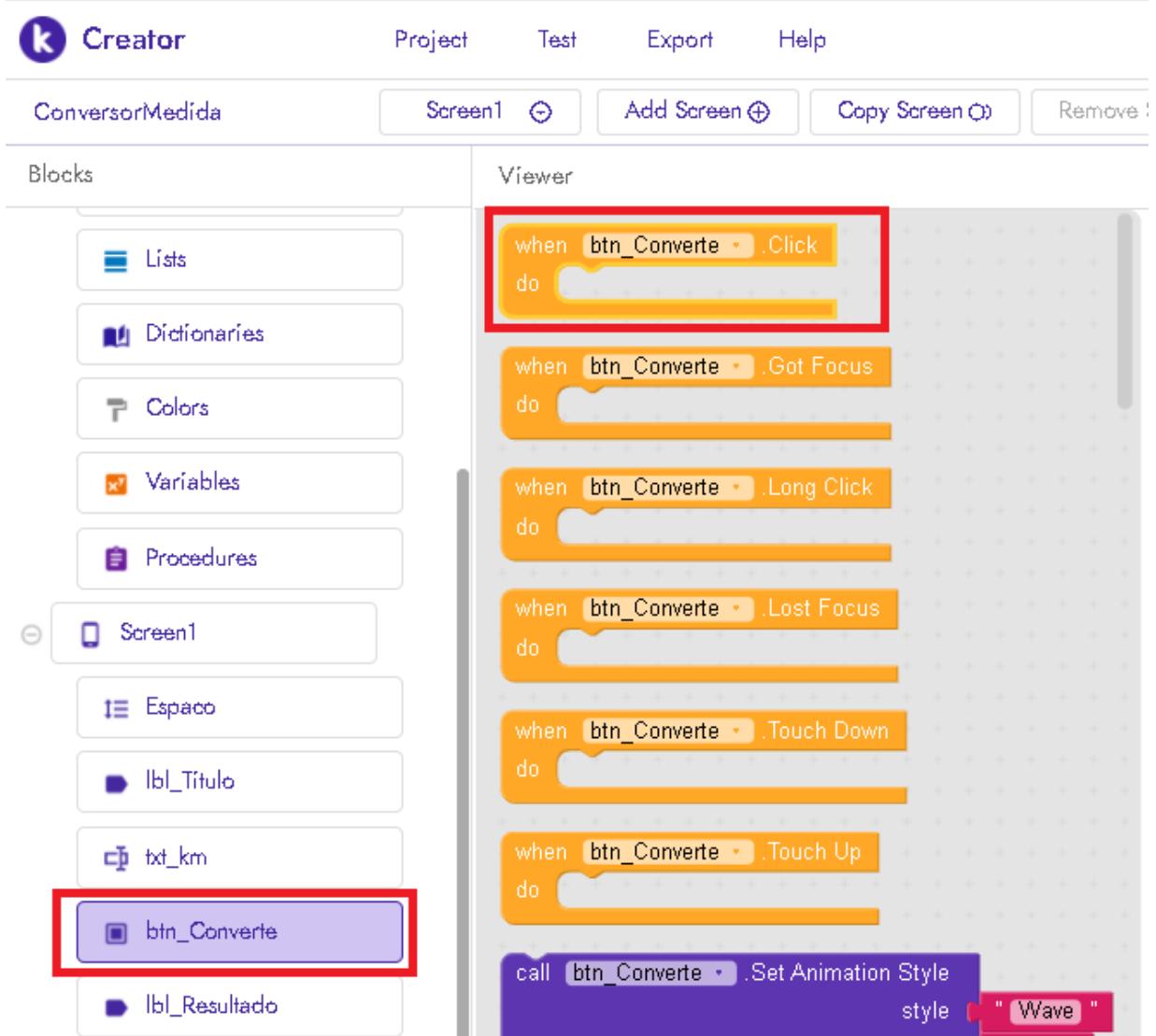


Figura 15 - Inserindo o evento click do botão.

Ao analisar a real situação da programação, podemos perceber que agora deveremos armazenar no componente **lbl_Resultado** a multiplicação do valor do componente **txt_km** pelo valor 1.000, pois a cada quilômetro possui 1.000 metros. Então não podemos esquecer que os operadores matemáticos estão no construtor **Math**.

- Clique no objeto **lbl_Resultado** e arraste a objeto **set lbl_resultado.txt to** para a área de bloco e conecte dentro do objeto **when btn_Converte .click**

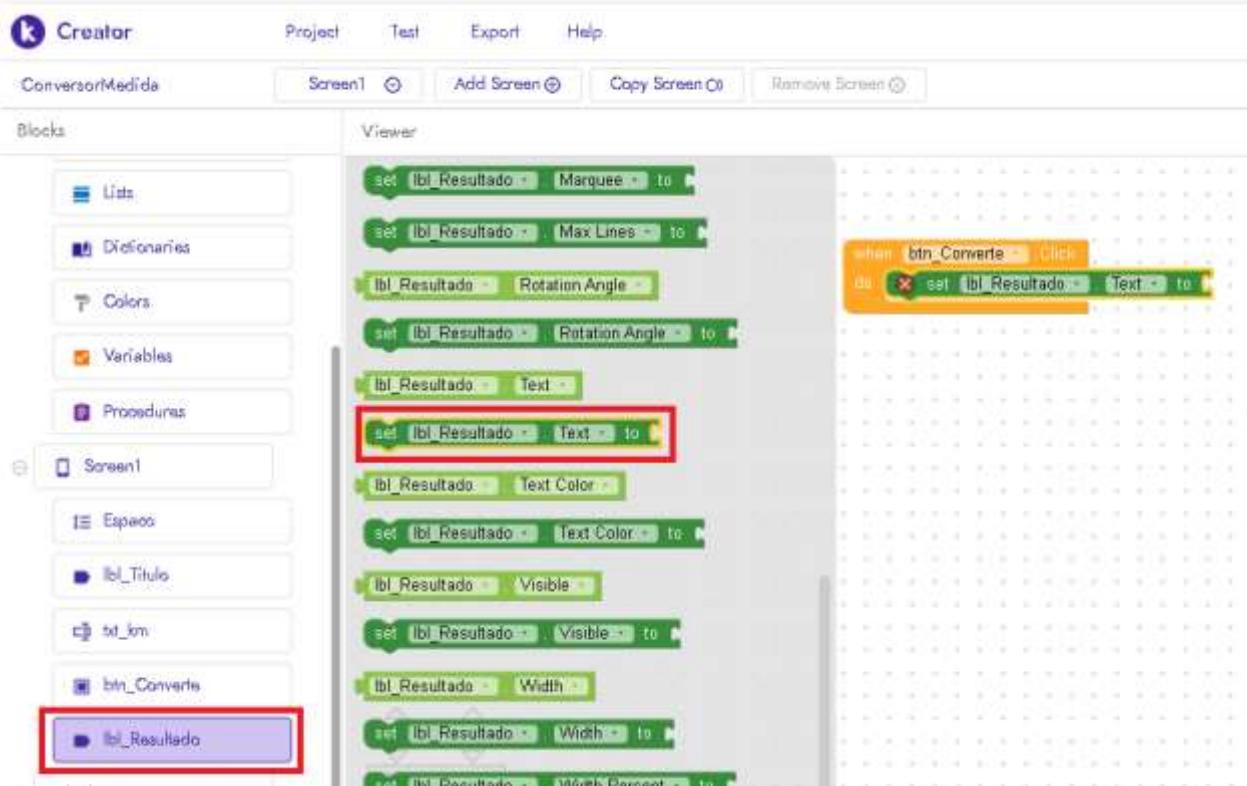


Figura 16 - Inserindo o valor digitado pelo usuário.

- Clique no construtor **Math** e arraste o objeto operador da Multiplicação e encaixe no objeto **set lbl_Resultado.text to**

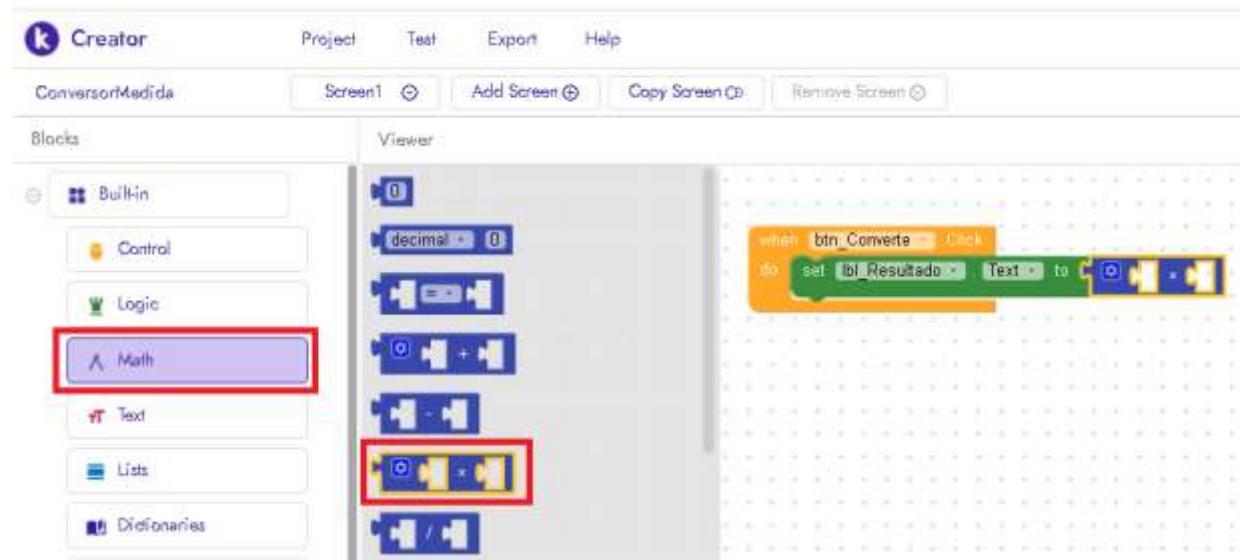


Figura 17 - Inserindo o operador matemático

- Clique no objeto **txt_km** e arraste o objeto **txt_km.text** para o primeiro operador da multiplicação.

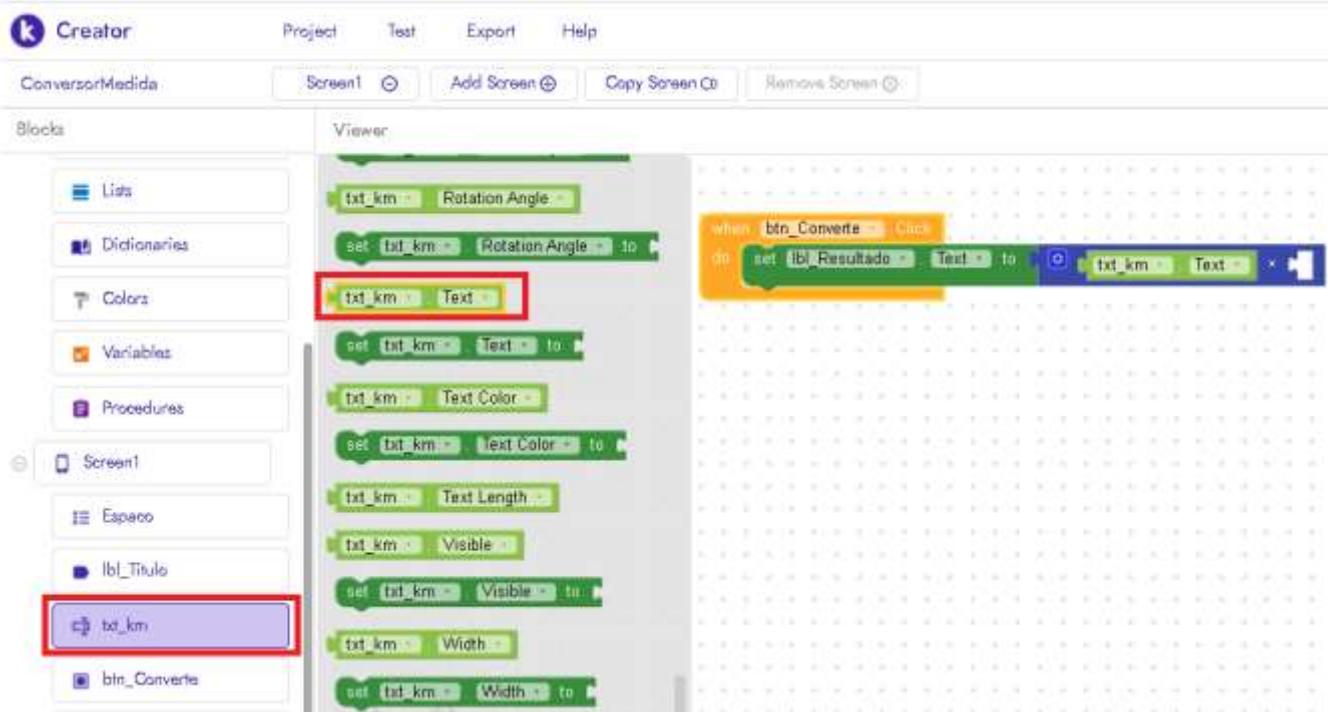


Figura 18 - Inserindo o primeiro valor da multiplicação.

- Clique no construtor **Math**, selecione o objeto **Number** e arraste-o para o segundo numero da multiplicação da fórmula desejada.

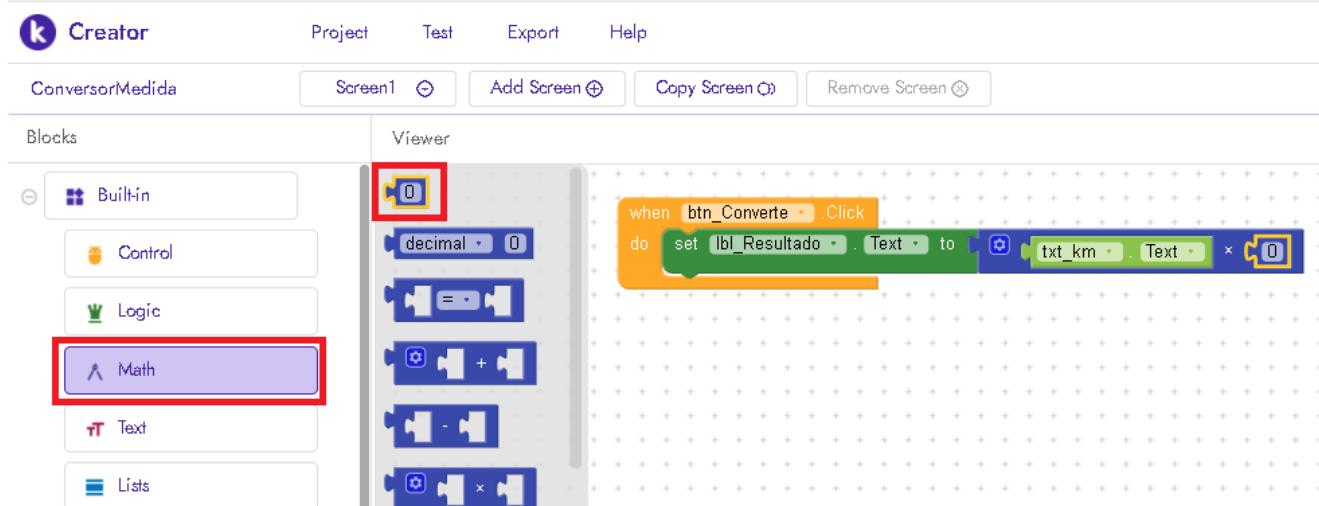


Figura 19 - Inserindo o segundo número.

- Altere o valor do **Number** de 0 para 1000.



Figura 20 - Alterando o valor do Number.

- Retorne a opção **Design** para acrescentar um novo componente.

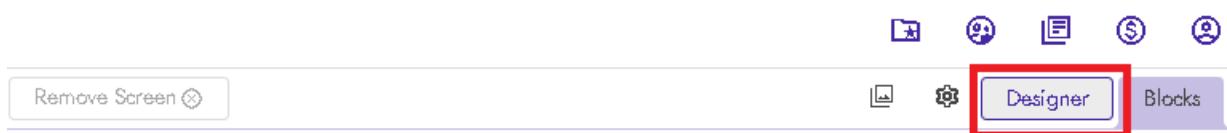


Figura 21 - Retornando a opção Design.

- Inserir um novo componente **Button**, abaixo de todos os outros componentes do layout.
- Altere as propriedades do objeto **Button**

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Width	Fill Parent	Definir a largura total disponível para o componente.
Text	Limpar	Definir o texto que será apresentado pelo Button .
Name	btn_Limpar	Definir o nome do componente.

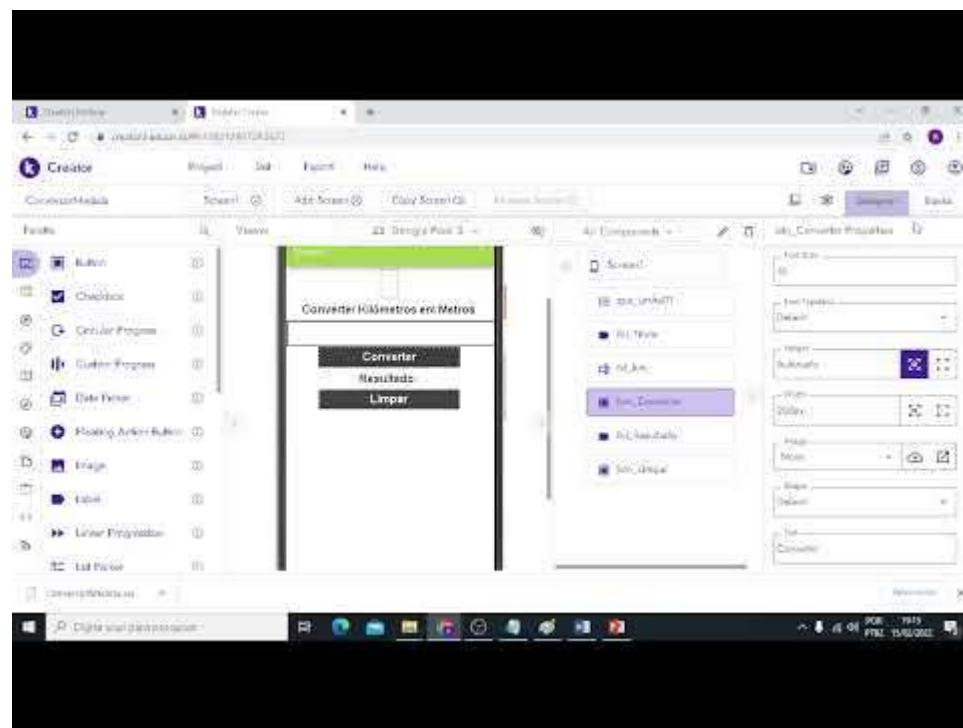
- Retorne a opção de **Blocks**, pois vamos programar mais um pouquinho.
- Construa a programação de blocos de acordo com a **Figura 23**.



Figura 22 - Inserindo o bloco do botão Limpar.

Caso o aluno tenha alguma dúvida durante a construção do projeto, deverá assistir ao vídeo:

Agenda 11 – Tela Splash, Imagens e Cores, disponível em:
<https://youtu.be/so8R-12mJbo>



Para finalizar o projeto, o aluno deverá exportar o **arquivo APK** para o dispositivo móvel e realizar a instalação através do aplicativo **Kodular Companion**.

- Clique no menu **Export**, na opção Android App (.apk).

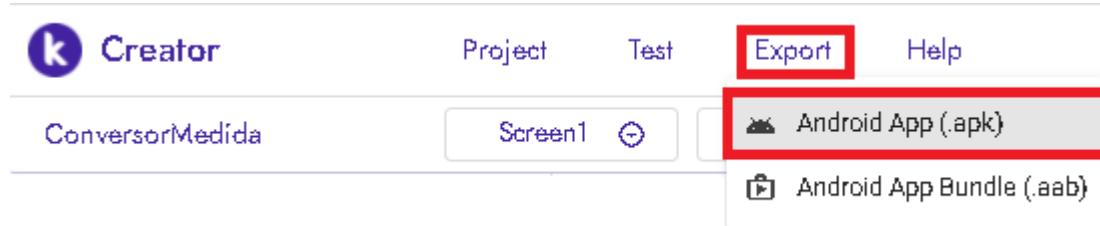


Figura 23 - Exportando o arquivo .apk

- Utilize o aplicativo **Kodular Companion** para escanear o qrcode e siga todos os passos para a instalação do aplicativo, de acordo com material anterior.

Android App for "ConversorMedida"

Scan the QR code on your phone to install the app or download the APK file to your computer.

Note: This link is valid only for 10 minutes. It is recommended to export your app as an Android App Bundle for distribution via Google Play.

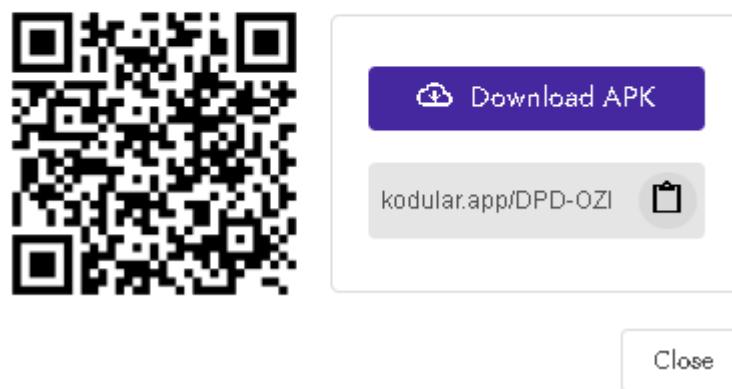


Figura 24 - Disponibilizando o Qrcode.

Arquivos disponíveis para download

Imagens utilizadas no projeto

AGENDA 12

**TRABALHANDO
COM LAYOUT
NO PROJETO**





Figura 1 - Simbolo Android

Você aprendeu até agora diversas formas de trabalhar com o Layout de um aplicativo, alterando cores, inserindo imagens. Agora vamos aprender que no desenvolvimento por meio do Kodular podemos utilizar a apresentação de mensagens. Inclusive podemos apresentar as próprias informações que foram digitadas no aplicativo. O mais interessante é que vamos integrar um pouco da linguagem HTML com a plataforma de desenvolvimento **Kodular**.

- Acesse a plataforma de desenvolvimento **Kodular**: <https://creator.kodular.io/>
- Crie um novo projeto com o nome de **Cadastro**
- Escolha um ícone para sua aplicação, sendo que a padronização do ícone deverá ser feita pelo botão **Configurações**.
- Altere as propriedades do componente **SCREEN**

Propriedade	Valor	Função
Title	Cadastro de Colaboradores	Definir o título da aplicação em desenvolvimento.

- Insira um componente **Space** da categoria **Layout**, opção **General**.
- Altere as propriedades do componente **Space**

Propriedade	Valor	Função
Height	40px	Definir a altura do componente
Name	Espaco	Definir o nome do componente

- Insira um componente **Label**, categoria **User Interface**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Text	Nome:	Definir o texto que será apresentado pelo Label .
Name	lbl_Nome	Definir o nome do componente.

- Insira um componente **TextBox**, categoria **User Interface**.
- Altere as propriedades do componente **TextBox**.

Propriedade	Valor	Função
Width	Fill Parent	Definir a largura do objeto como a largura disponível da tela.
Hint	Em branco	Definir sem dica de informação.
Text	Em branco	Definir o texto que será apresentado pelo TextBox .
Name	txt_Nome	Definir o nome do componente.

- Insira um componente **Label**, categoria **User Interface**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte.
Font Bold	✓ Marcado	Definir a opção de negrito na fonte
Text	Endereço	Definir o texto que será apresentado pelo Label .
Name	lbl_endereco	Definir o nome do componente.

- Insira um componente **TextBox**, categoria **User Interface**.
- Altere as propriedades do objeto **TextBox**.

Propriedade	Valor	Função
Width	Fill Parent	Definir a largura do objeto como a largura disponível da tela.
Hint	Em branco	Definir sem dica de informação.
Text		Definir o texto que será apresentado pelo TextBox .
Name	txt_Endereco	Definir o nome do componente.

- Insira um componente **Horizontal Arragement**, categoria **Layout**, opção **General**.
- Altere as propriedades do componente **Horizontal Arragement**.

Propriedade	Valor	Função
Align Horizontal	Center	Centraliza o conteúdo dentro do componente.
Width	Fill Parent	Definir a largura do objeto como a largura disponível da tela.
Name	Linha01	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, dentro do componente **Linha01**.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Height	50px	Definir a altura do componente button.
Width	100px	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Cadastrar	Definir o conteúdo a ser exibido do componente button.
Name	btn_Cadastrar	Definir o nome do componente.

- Insira um componente **Label** da aba **User Interface**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte
Heigth	Fill Parent	Definir a altura do componente, sendo o total disponível da tela.
Width	Fill Parent	Definir a largura do componente, sendo o total disponível da tela.
Text		Definir o conteúdo a ser exibido do componente Label.
Advance Properties → Html Format	✓ Marcado	Definir o uso da linguagem HTML, para a formatação do texto.
Name	lbl_Cadastro	Definir o nome do componente.

Ao terminar o layout desta aplicação, o aplicativo deverá estar de acordo com a **Figura 2**.

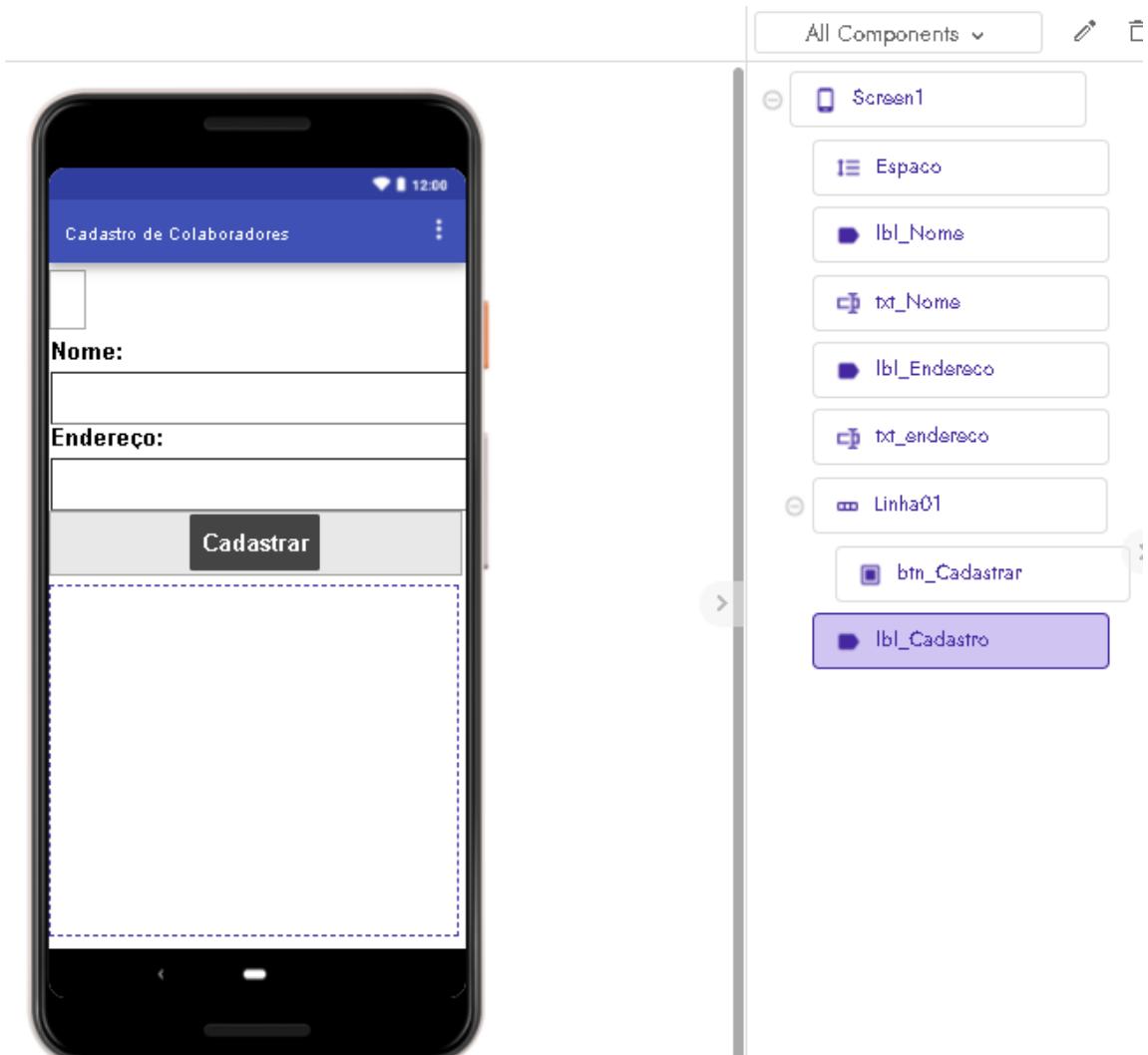
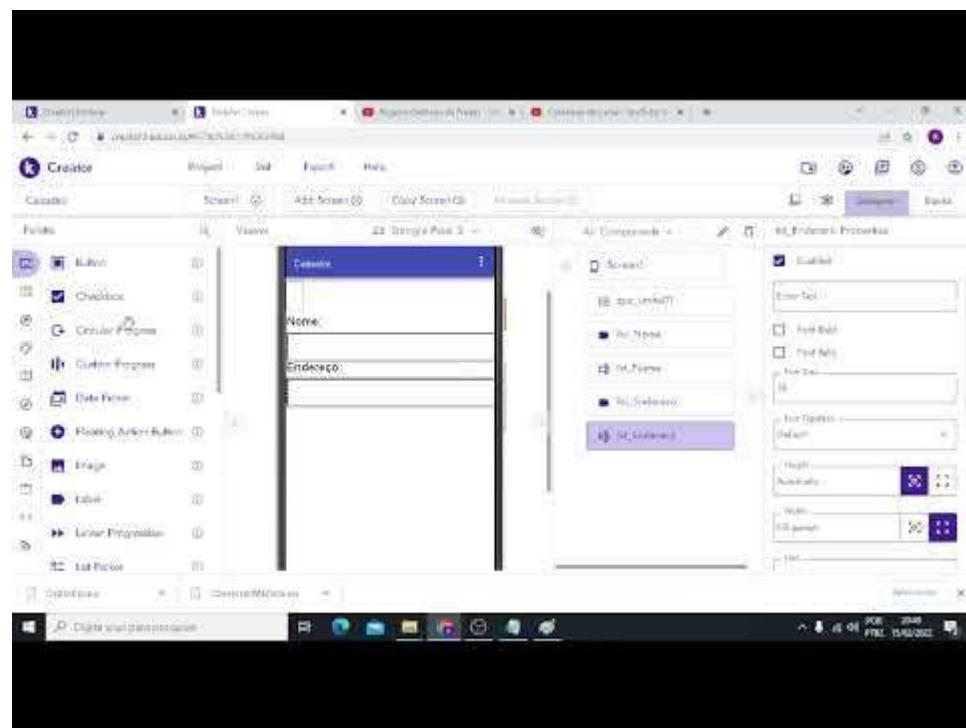


Figura 2- Visualizando o Layout

Caso o aluno tenha alguma dúvida em relação a criação do layout deste projeto, deverá assistir ao vídeo:

Agenda 12 – Projeto de Layout de Cadastro, disponível em:
<https://youtu.be/ycQoDnplj-U>



Chegou o momento de realizar a programação em blocos, pois o aplicativo deverá exibir as informações digitadas nos campos de cadastro. Sendo assim, será preciso fazer a programação.

- Altere a para o cenário de construção de **Blocks**
- Clique no objeto **btn_Cadastrar** e arraste o objeto **when btn_cadastrar.click do**

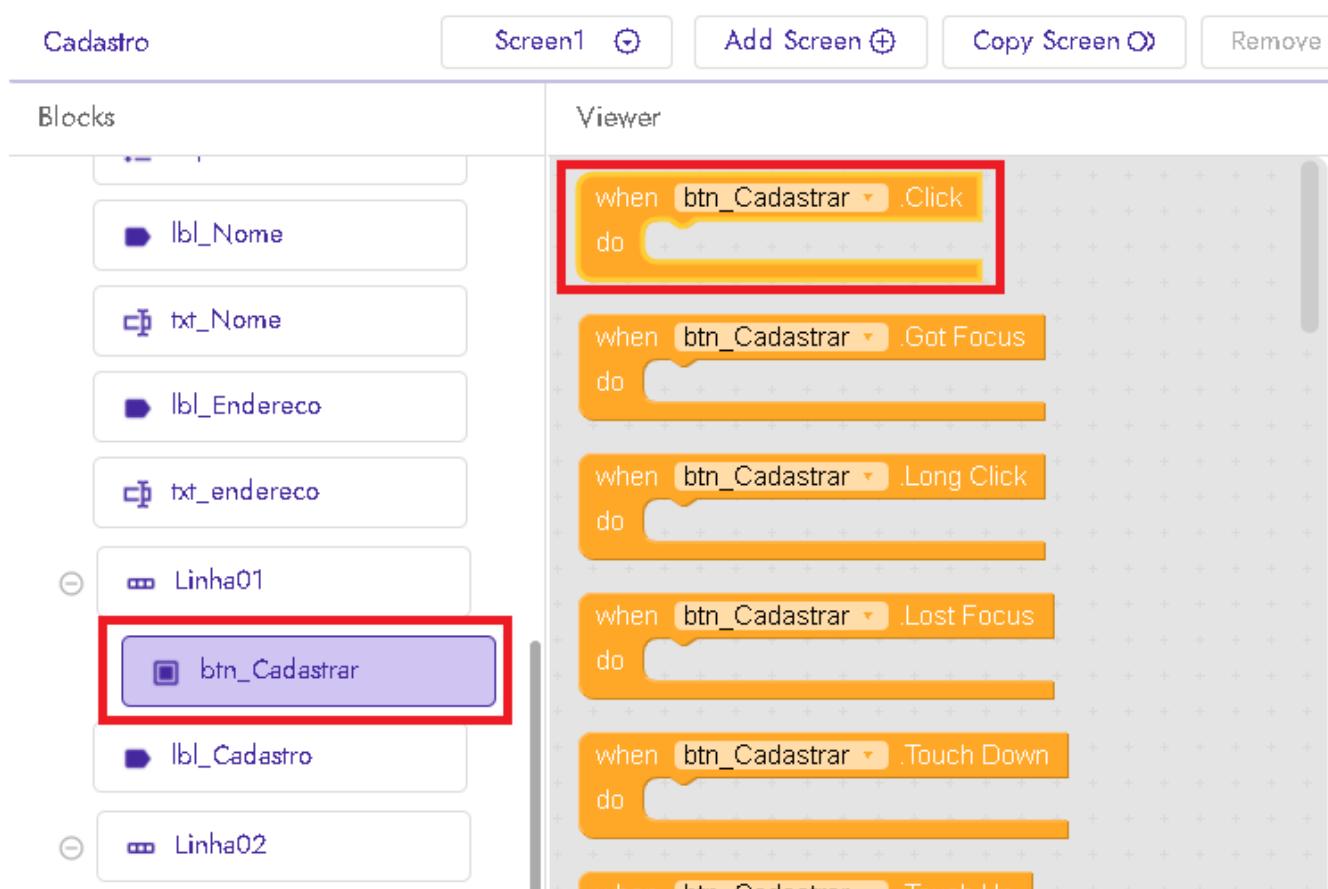


Figura 3 - Inserindo o evento click do btn_cadastrar

- Clique no objeto **lbl_cadastro** e arraste o objeto **set lbl_cadastro.text to**

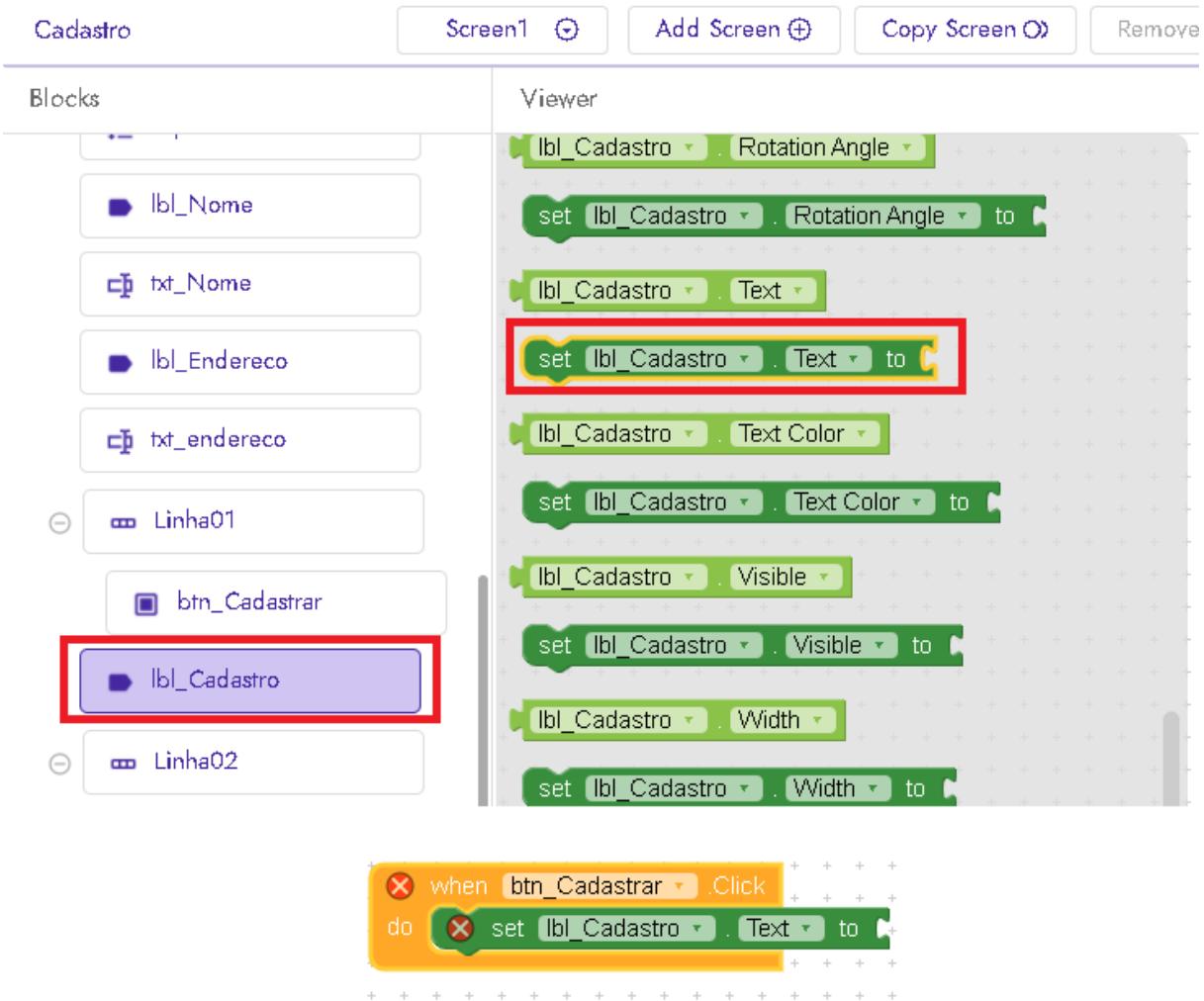


Figura 4 - Inserindo o objeto **lbl_cadastro**.

- Clique no construtor **Text** e insira o objeto **Join**

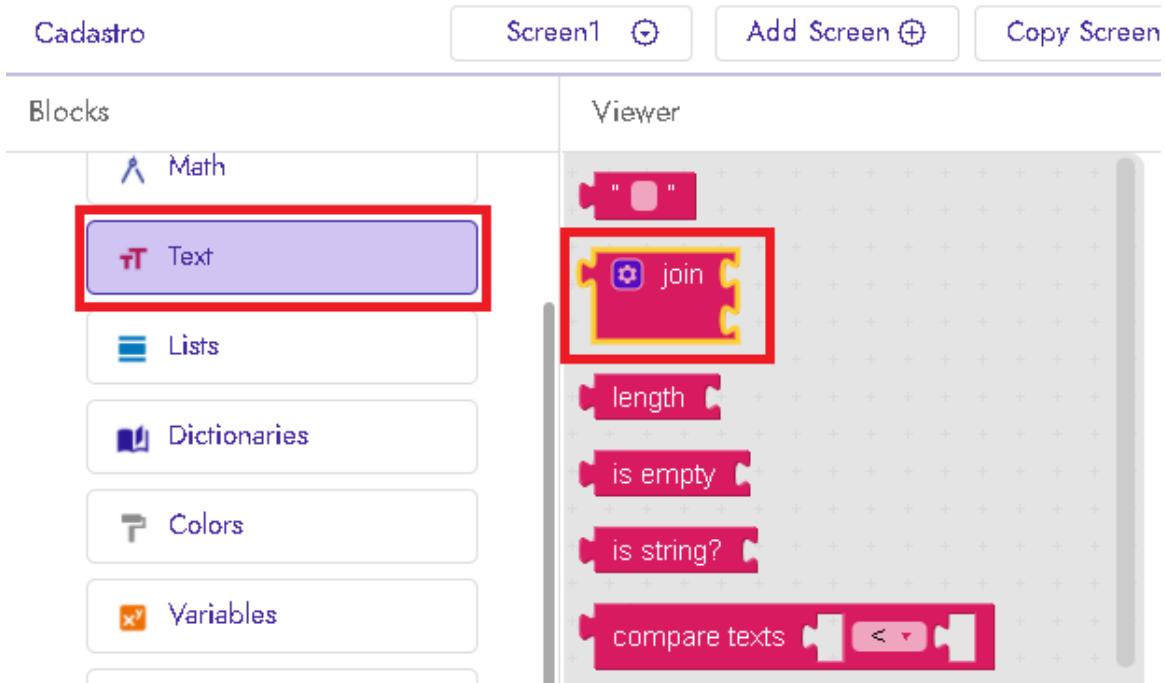


Figura 5 - Inserindo o objeto Join

- Aumentar o numero de string que o objeto Join irá concatenar. Clique no botão **Configuração** do objeto **Join**. Arraste a palavra **string** abaixo das duas outras string dentro do objeto **Join** ao lado direito da imagem.

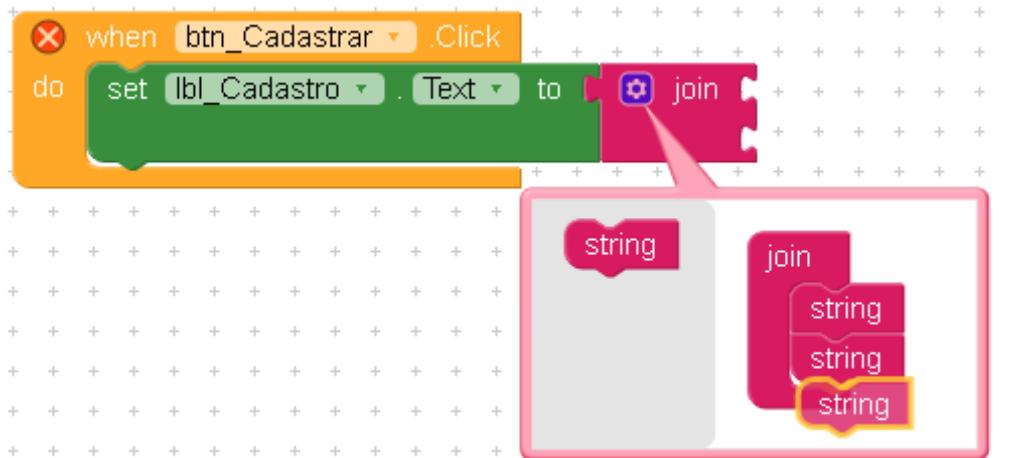


Figura 6 - Inserindo strings no objeto Join.

- O aluno deverá construir a programação de acordo com a **Figura 3**.

```

when btn_Cadastrar .Click
do set lbl_Cadastro .Text to join ( " <h1>Cadastro com sucesso! </h1> "
                                         " <b>Nome: </b> "
                                         txt_Nome .Text
                                         " <br> "
                                         " <b> Endereço: </b> "
                                         txt_endereco .Text )
  
```

Figura 7 - Código em Programação em Blocos.



Vamos agora criar um botão que permita ao usuário sair do aplicativo.

- Insira um componente **Horizontal Arragement** da aba **Layout**, opção **General**.
- Altere as propriedades do componente **Horizontal Arragement**.

Propriedade	Valor	Função
Align Horizontal	Right	Centraliza o conteúdo dentro do componente.
Width	Fill Parent	Definir a largura do objeto como a largura disponível da tela.
Name	Linha02	Definir o nome do componente.

- Insira um componente **Button** da aba **User Interface**, dentro do componente **Linha02**.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Font Size	18	Definir o tamanho da fonte
Text	Cadastrar	Definir o conteúdo a ser exibido do componente button.
Width	100px	Definir o tamanho da largura do componente.
Name	btn_Cadastrar	Definir o nome do componente.

Confira a inserção do botão **Sair** no layout do aplicativo.

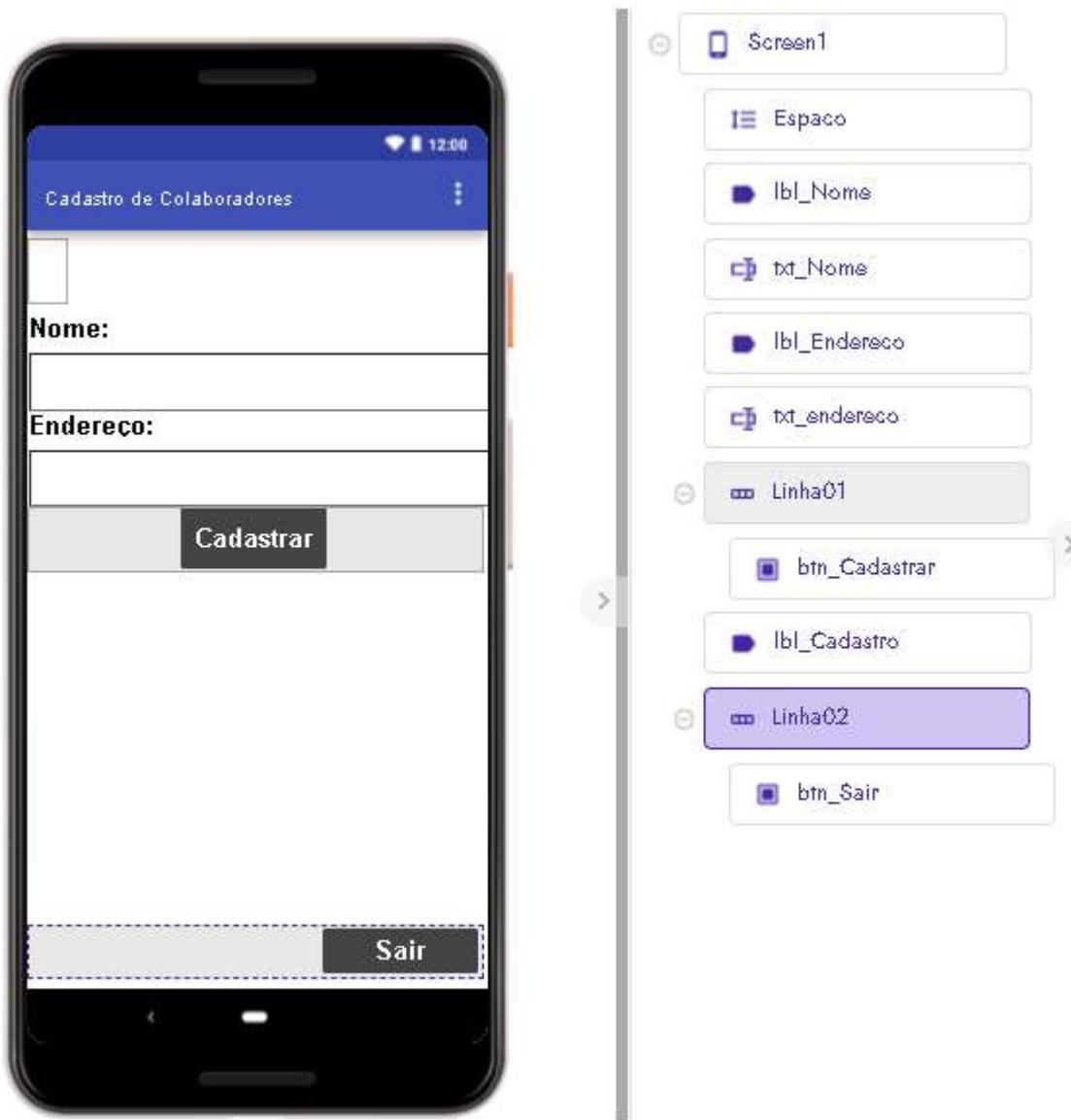


Figura 8 - Visualizando o Layout.

*Retorne para o cenário em programação em **Blocos**.*

- Clique no objeto **btn_Sair** e arraste o objeto **when btn_Sair.click**



Figura 9- Inserindo o evento click.

- Clique no construtor **Control**, arraste o objeto **close application** e encaixe dentro do objeto **when btn_sair.click**

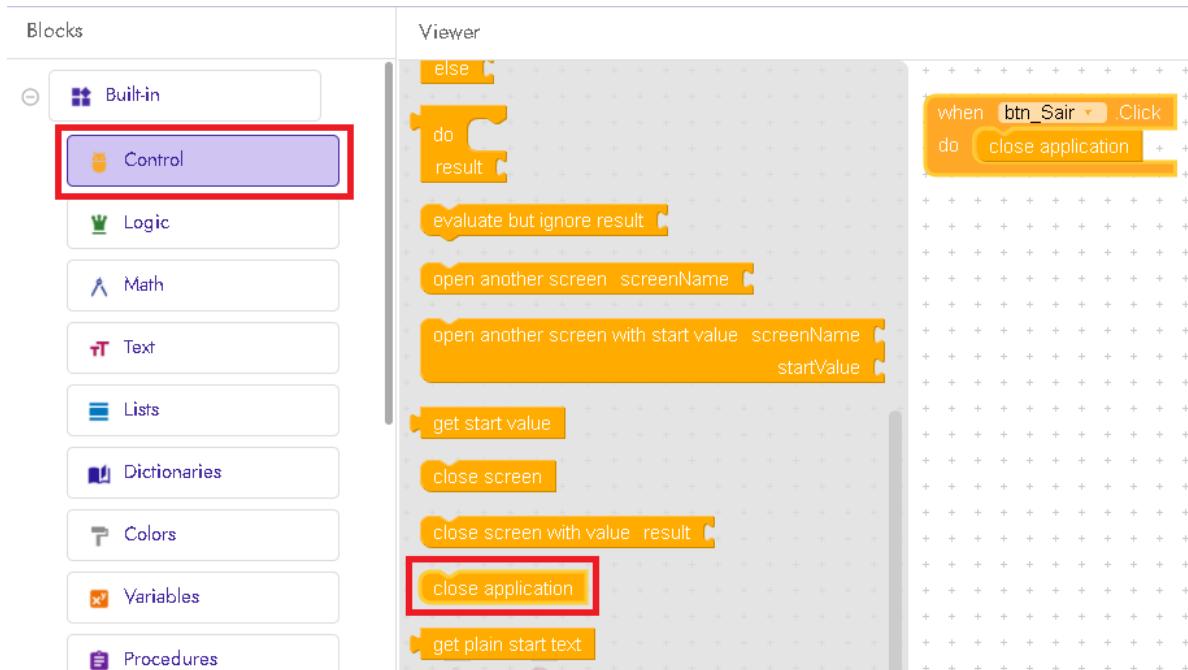
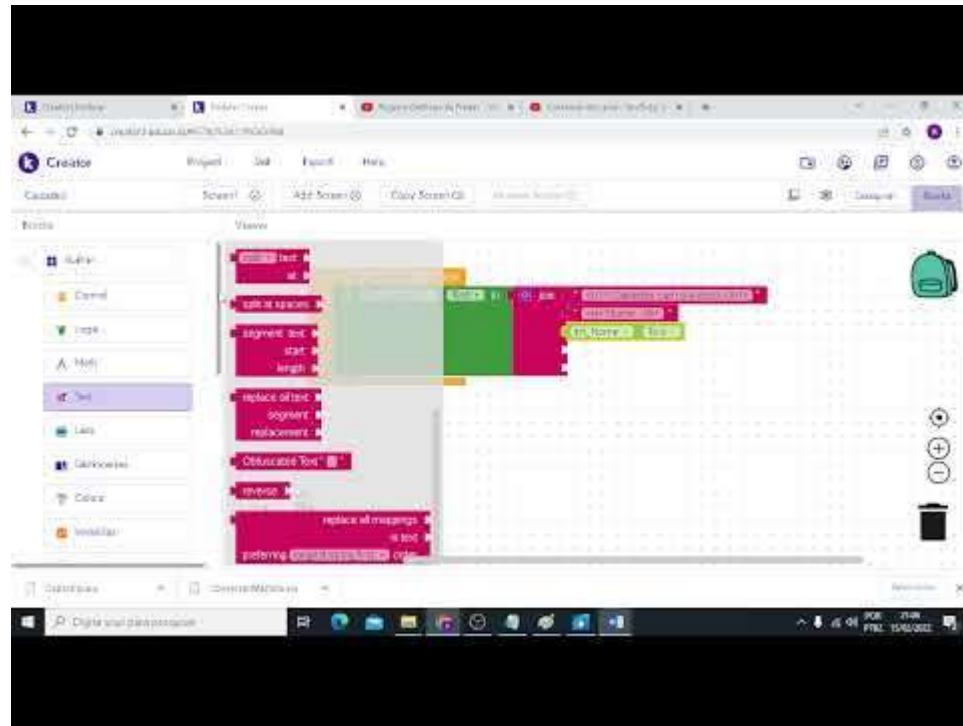


Figura 10 - Inserindo o código para sair da aplicação.

Para solucionar qualquer dúvida em relação a construção do blocos, disponibilizamos o video a seguir:

Agenda 12 – Construindo o bloco de programação, disponível em:
<https://youtu.be/rae5XciihEk>



Para finalizar o projeto, o aluno deverá exportar o **arquivo APK** para o dispositivo móvel e realizar a instalação através do aplicativo **Kodular Companion**.

- Clique no menu **Export**, na opção Android App (.apk).



Figura 11 - Menu Export, opção Android App (.apk)

- Utilize o aplicativo **Kodular Companion** para escanear o qrcode e siga todos os passos para a instalação do aplicativo, de acordo com material anterior.

Android App for "Cadastro"

Scan the QR code on your phone to install the app or download the APK file to your computer.

Note: This link is valid only for 10 minutes. It is recommended to export your app as an Android App Bundle for distribution via Google Play.

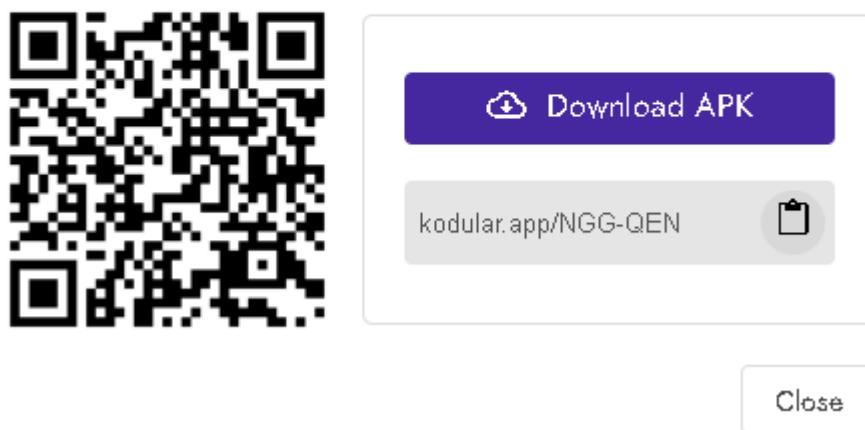


Figura 12 - Disponibilizando o Qrcode.

AGENDA 13

**TRABALHANDO COM
DESENVOLVIMENTO
DE JOGOS**

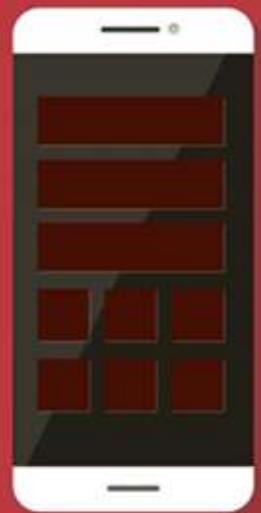




Figura 1 - Fonte: www.freepik.com

Segundo o site de documentação do Kodular, o RelativeLayout é um agrupamento que exibe componentes filhos em posições relativas. A posição de cada componente “filho” pode ser especificada como relativa a um elemento “irmão” ou vizinho, por exemplo, o Horizontal Arragement, pode assumir um valor em relação ao tamanho da tela do dispositivo móvel. Sendo assim, a cada dispositivo o mesmo poderá ter um tamanho diferente, de acordo com a tela que está disponível. Sendo que os componentes que estão dentro dele, também deverão sofrer as alterações pela função da herança, através do RelativeLayout.

O componente também pode ter posições relativas à própria área do RelativeLayout denominado “pai”, por exemplo, um Label está alinhado ao centro do RelativeLayout “pai”.

Para definir melhor a relação entre RelativeLayout, imagine o componente SCREEN sendo o componente pai, ou seja, se configurarmos o alinhamento horizontal ao centro, todos os objetos o qual foram colocados dentro do mesmo, serão alinhados ao centro também. É uma herança entre classe, que estabelecida através do processo de RelativeLayout.



Chegou a sua vez de colocar a “mão na massa”. Desenvolva um novoprojeto no Android Studio com o nome de “Jogodavelha”, como mostra a imagem 8.

Para inciarmos a construção deste novo projeto, será necessário a criação de um novo projeto na plataforma de desenvolvimento **Kodular**.

- Acesse a plataforma de Kodular : <https://creator.kodular.io/>
- Criar um novo projeto com o nome de **JogodaVelha**.
- Escolha um ícone para sua aplicação, sendo que a padronização do ícone deverá ser feita pelo botão **Configurações**.
- Altere as propriedades do componente **SCREEN**

Propriedade	Valor	Função
Align Horizontal	Center	Definir o alinhamento centralizado.
Title	Jogo da Velha	Definir o título da aplicação em desenvolvimento.

- Insira um componente **Button**, categoria **User Interface**.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#3F51B5FF	Definir a cor do fundo do componente button.
Height	50px	Definir a altura do componente button.
Width	100px	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Começar	Definir o conteúdo a ser exibido do componente button.
Name	btn_Comecar	Definir o nome do componente.

- Inserir um **Horizontal Arrangement** , categoria **Layout**, opção **General**.
- Altere as propriedades do componente **Horizontal Arrangement**.

Propriedade	Valor	Função
Align Horizontal	Center	Definir o alinhamento horizontal ao centro.
Align Vertical	Center	Definir o alinhamento vertical ao centro.
Height	30%	Definir a altura do componente Horizontal Arrangement.
Width	Fill Parent	Definir a largura do componente

Propriedade	Valor	Função
		button.
Background Color	#EEEEEEFF	Definir a cor de fundo do componente Horizontal Arrangement.
Name	Horizontal_Jogador	Definir o nome do componente.

- Insira um componente **Label**, categoria **User Interface**, dentro do componente **Horizontal_Vezes**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	25	Definir o tamanho da fonte
Text	É a vez do Jogador:	Definir o conteúdo a ser exibido do componente Label.
Name	lbl_frase	Definir o nome do componente.

- Insira um componente **Label**, categoria **User Interface**, a direita do componente **lbl_frase**.
- Altere as propriedades do componente **Label**.

Propriedade	Valor	Função
Font Size	25	Definir o tamanho da fonte
Text	0	Definir o conteúdo a ser exibido do componente Label.
Name	lbl_Jogador	Definir o nome do componente.

Ao inserir todos os componentes solicitados e alterados as suas respectivas propriedades, o layout do projeto deverá estar idêntico a **Figura 2**.

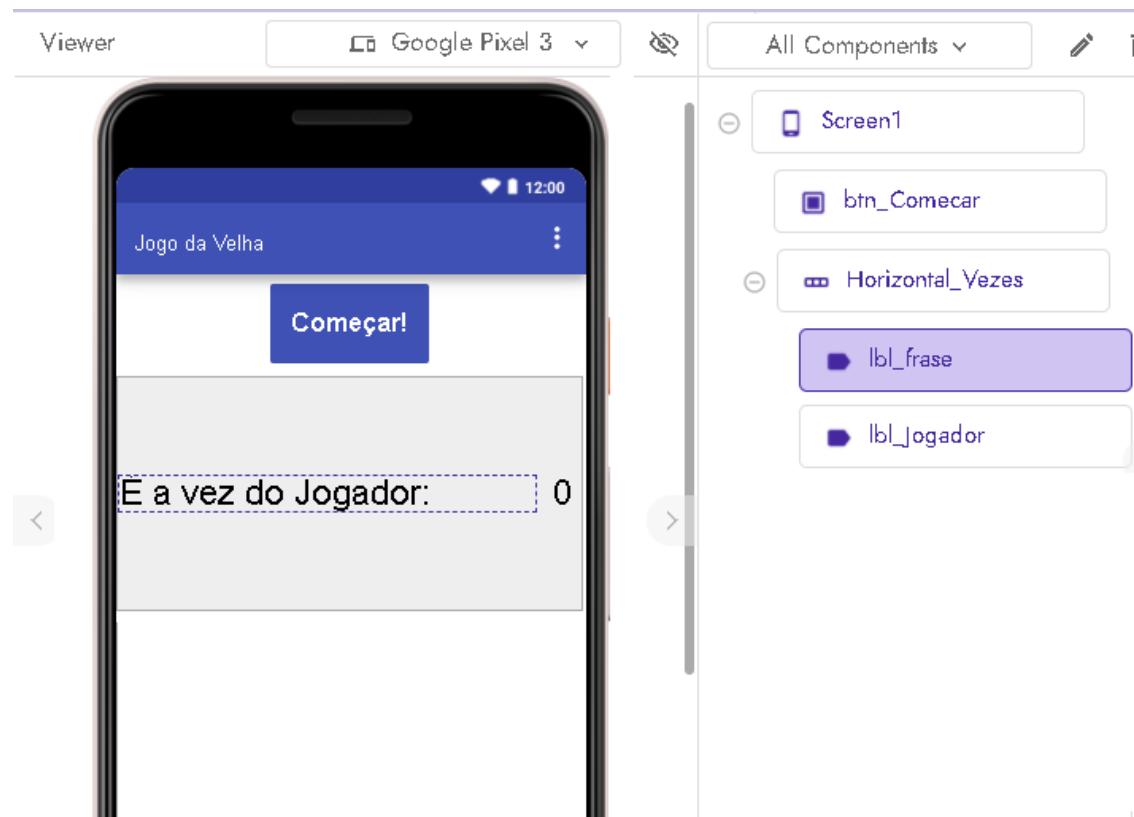


Figura 2 -Visualizando o Layout.

Para representar o tabuleiro, onde os jogadores deverão escolher os valores X ou 0, vamos utilizar o componente **Table Arragement**, que permite a definição de uma matriz, ou seja, várias linhas e colunas. Também podem ser representada por uma tabela, pois o conceito é o mesmo.

- Insira um componente **Table Arragement**, categoria **Layout**, opção **General**
- Altere as propriedades do componente **Table Arragement**.

Propriedade	Valor	Função
Columns	3	Definir a quantidade de colunas por linha
Height	Fill Parent	Definir a altura do componente
Width	Fill Parent	Definir a largura do componente
Rows	3	Definir a quantidade de linhas
Name	Table_Tabuleiro	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na primeira posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.

Propriedade	Valor	Função
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n1	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na segunda posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n2	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na terceira posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n3	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na quarta posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.

Propriedade	Valor	Função
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n4	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na quinta posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n5	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na sexta posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n6	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na sétima posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.

Propriedade	Valor	Função
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n7	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na oitava posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n8	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**, na nona posição do tabuleiro.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Background color	#444444FF	Definir a cor do fundo do componente button.
Height	120px	Definir a altura do componente button.
Width	120px	Definir a largura do componente button.
Font Size	25	Definir o tamanho da fonte
Text		Definir o conteúdo a ser exibido do componente button.
Text Alignment	Center	Definir o texto alinhado ao centro.
Name	btn_n9	Definir o nome do componente.

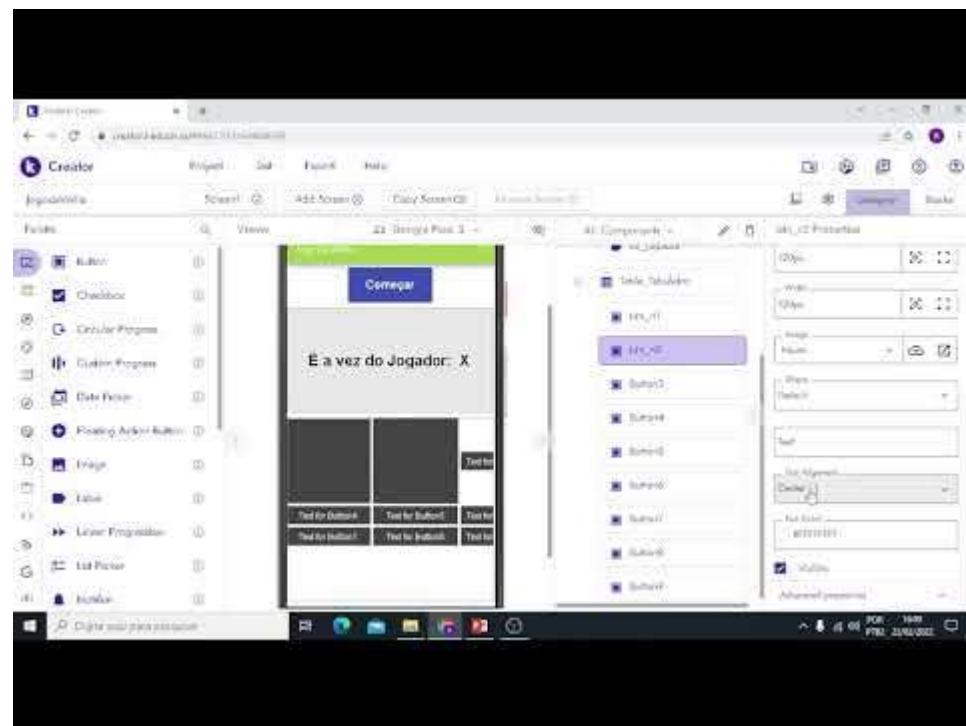
Ao término da inserção de todos os componentes e alterações em suas propriedades, o layout do projeto deverá conforme a **Figura 3**.



Figura 3 - Visualizando o Layout.

Caso o aluno tenha alguma dúvida em relação a construção do layout deste aplicativo, poderá assistir ao vídeo:

Agenda 13 – Trabalhando com Desenvolvimento de Jogos, disponível em:
<https://youtu.be/0Hvzz694dmI>



Ao pensar sobre o projeto, a conclusão que se tem, é que o aplicativo deverá alternar entre os jogadores, uma vez será o jogador X e na outra vez o jogador O. Portanto existe uma variação de valores e sendo assim, o uso de **varáveis** será de extrema importância.

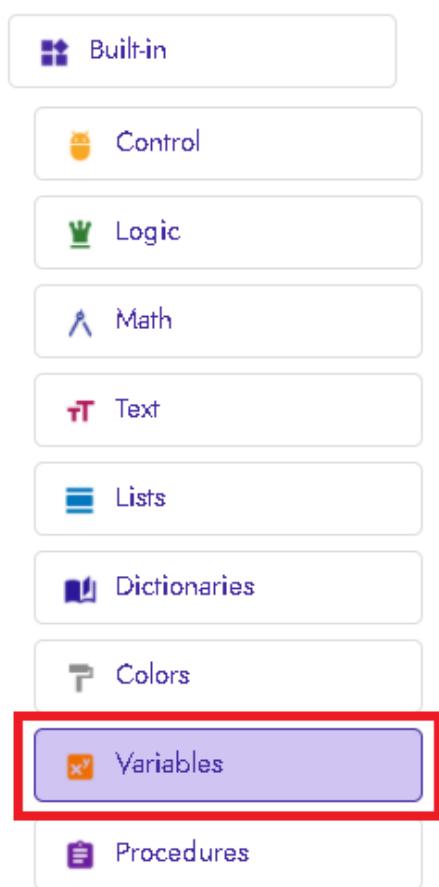
- Alterne para o cenário de desenvolvimento de **Blocks**.



Figura 4 - Alternando para a programação em blocos.

- Clique no construtor **Variables**, arraste o objeto **initialize global name to**

Blocks



Viewer



initialize global vez to " "

Figura 5- Criando uma variável.

- Defina o nome da variável como **vez** e acrescente um **text string** em branco como valor da variável.
- Altere o valor da variável vez para o texto **X**.
- Clique no construtor **Procedures** e arraste o objeto **to procedure do**

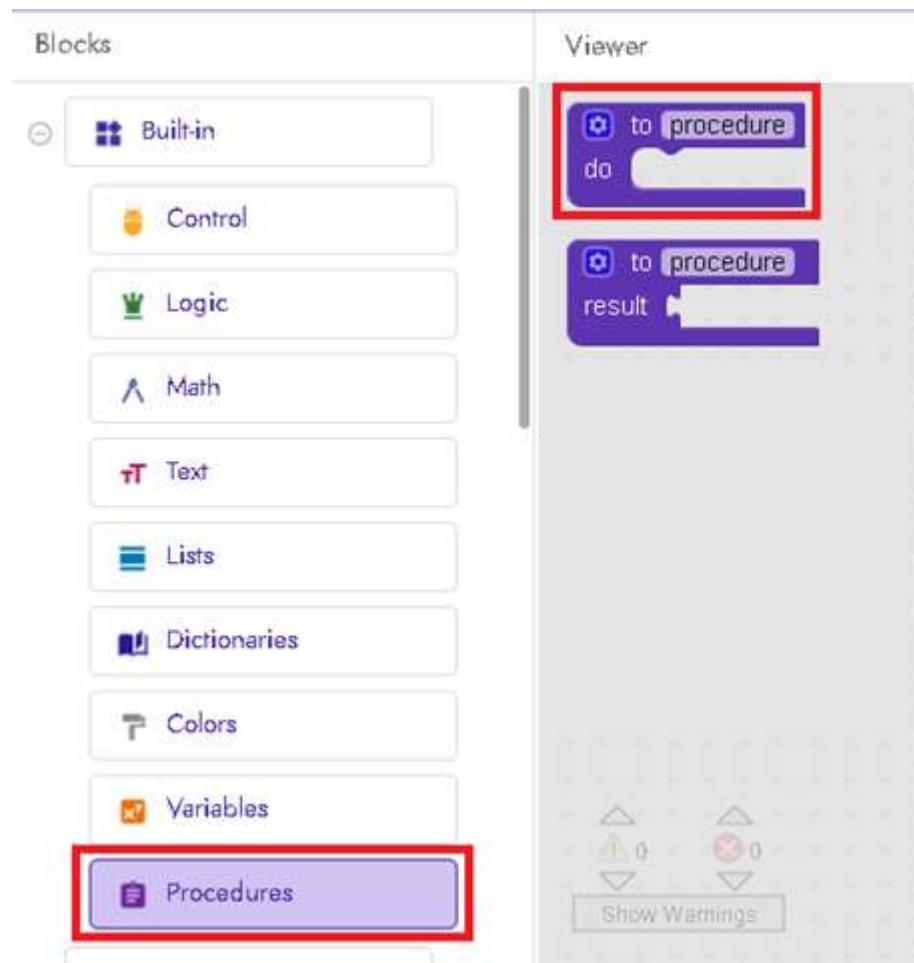


Figura 6 - Adicionando um procedimento.

- Defina o nome do procedimento como **Limpar**
- Adicione os campos conforme a **Figura 7**.

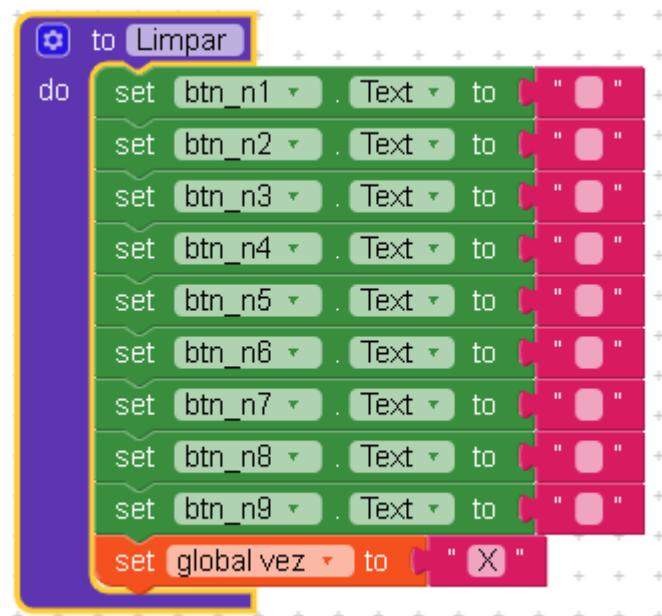


Figura 7 - Bloco do procedimento Limpar.

- Construa o código do evento click do botão **btn_Comecar**

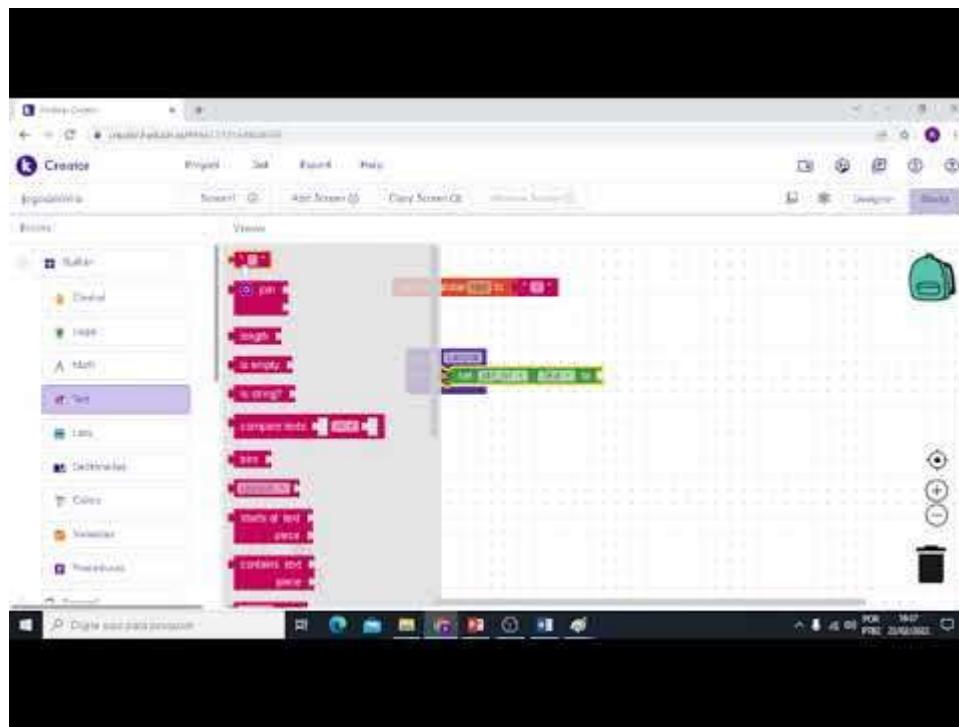


Figura 8 - Bloco de programação evento click

Para concluir a programação do botão Começar, será disponibilizado um vídeo para solucionar as dúvidas.

Agenda 13 – Criando a programação do botão Começar, disponível em:

https://youtu.be/jSWXxwjJ_uo



Para finalizar o projeto, o aluno deverá exportar o **arquivo APK** para o dispositivo móvel e realizar a instalação através do aplicativo **Kodular Companion**.

- Clique no menu **Export**, na opção **Android App (.apk)**.

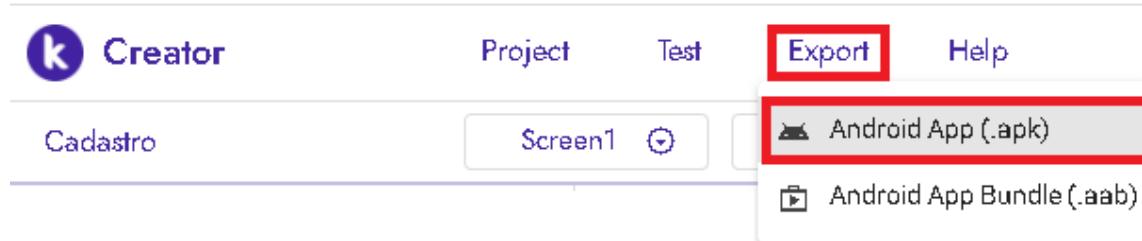


Figura 9 - Menu Export, opção Android App (.apk)

- Utilize o aplicativo **Kodular Companion** para escanear o qrcode e siga todos os passos para a instalação do aplicativo, de acordo com material anterior.

Android App for "Cadastro"

Scan the QR code on your phone to install the app or download the APK file to your computer.

Note: This link is valid only for 10 minutes. It is recommended to export your app as an Android App Bundle for distribution via Google Play.



Download APK

kodular.app/NGG-QEN



Close

Figura 10 - Disponibilizando o Qrcode.

AGENDA 14

**TÉCNICAS DE
PROGRAMAÇÃO
NO KODULAR**



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Atualização técnica:

Rogério Galdiano de Freitas

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim



Utilização de métodos

Em determinados momentos da programação de aplicativos, encontramos a necessidade de repetir inúmeras vezes o mesmo código ou trechos da programação em vários locais diferentes do nosso programa. Para evitar que o programador reescreva várias vezes os mesmos códigos em locais diferentes, utilizamos os procedimentos.

Um exemplo de método é a função de “trocaJogador”. Essa função é chamada todas as vezes que um jogador escolhe uma posição do tabuleiro, e passa a jogada para o seu adversário.



Figura 1 - Procedimento trocaJogador

Os procedimentos possuem uma visibilidade, um tipo de retorno e argumentos. A visibilidade “public” é responsável por deixar o método acessível para ser utilizado em qualquer parte do nosso app. Sendo que na plataforma Kodular todos os procedimentos serão públicos.

Cada procedimento pode gerar um valor após a sua execução, para isso é necessário informar qual é o tipo de retorno, como por exemplo, um valor inteiro. No caso do nosso procedimento “trocaJogador” não é necessário retornar nada após a sua execução.

Os argumentos são declarados no próprio procedimento. Os argumentos de um método têm a função de transmitir valores para as rotinas que serão executadas internamente no procedimento. Para a função de “trocaJogador” não foi necessário passar nenhum argumento para a sua execução. Mas no procedimento marca_posição utilizamos um argumento btn, que determina qual o botão que o usuário clicou.

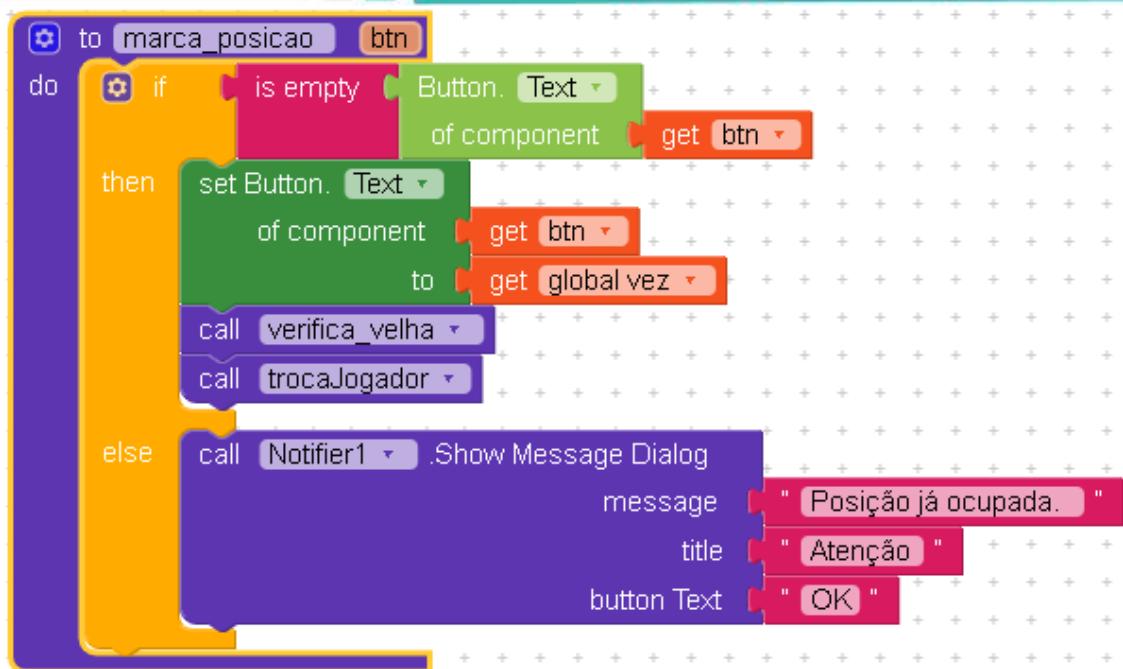


Figura 2 - Exemplo de procedimento com argumento.

Para utilizar um método basta apenas chamar o seu nome, como podemos observar no código que é executado quando o botão “**btn_Comecar**” é pressionado.

Durante a programação de aplicativos nos deparamos com a manipulação de dados na programação. Esses dados obedecem a um tipo e podem ser utilizados para armazenar valores em variáveis e/ou constantes, retornos de métodos, entre outras utilizações.

Encontramos alguns tipos que são utilizados com uma maior frequência no desenvolvimento mobile.

Variável

Variável é um local para armazenamento temporário na memória do dispositivo indicado por um nome previamente escolhido no momento da sua declaração. No ambiente de desenvolvimento Kodular não é necessário um tipo para a variável, A função principal da variável é armazenar valores temporários que serão utilizados nas rotinas do aplicativo, esses valores podem sofrer alterações durante a execução do aplicativo. Veja a seguir o código para gerar uma variável.

```
initialize global [vez] to [ " " ]
```

Figura 3 - Criando uma variável.

Vetor

Uma variável armazena apenas um valor para cada alocação. Para trabalhar com vários valores de um mesmo tipo relacionados há um processamento comum podemos utilizar um vetor. O vetor é indicado por um nome e um índice de acordo com a quantidade de divisões definidas no ato do seu desenvolvimento.

No exemplo a seguir encontramos uma variável para armazenar uma lista de valores.



Figura 4 - Criando uma lista de valores.

Na sequência declaramos um vetor para armazenar os nomes de três alunos de um grupo de estudos. É importante ressaltar que em um mesmo local vamos armazenar três nomes, separados internamente por um índice. O tamanho do vetor é definido de acordo com a necessidade da programação.



Figura 5 - Criando uma lista com três nomes.

O índice de um vetor sempre inicia no “0”. Em uma lista de 3 posições vamos encontrar índices 0, 1, e 2.

Estrutura de decisão

A estrutura de decisão é utilizada para que uma parte do código seja executada apenas se uma determinada condição “booleana” estiver sendo atendida. Caso contrário podemos determinar que o aplicativo execute uma outra parte do código. O código a seguir demonstra a utilização de uma estrutura de decisão “if e else” para alterar o símbolo do jogador. Caso o jogador atual seja o “X”, a condição será atendida e alteramos para o “0”. Caso o jogador não é o “X”, a condição não é atendida e executamos a alteração da variável para “X”, uma vez que ela atualmente só pode ser o “0”.



Figura 6 - Bloco de programação - trocaJogador.

Estrutura de repetição

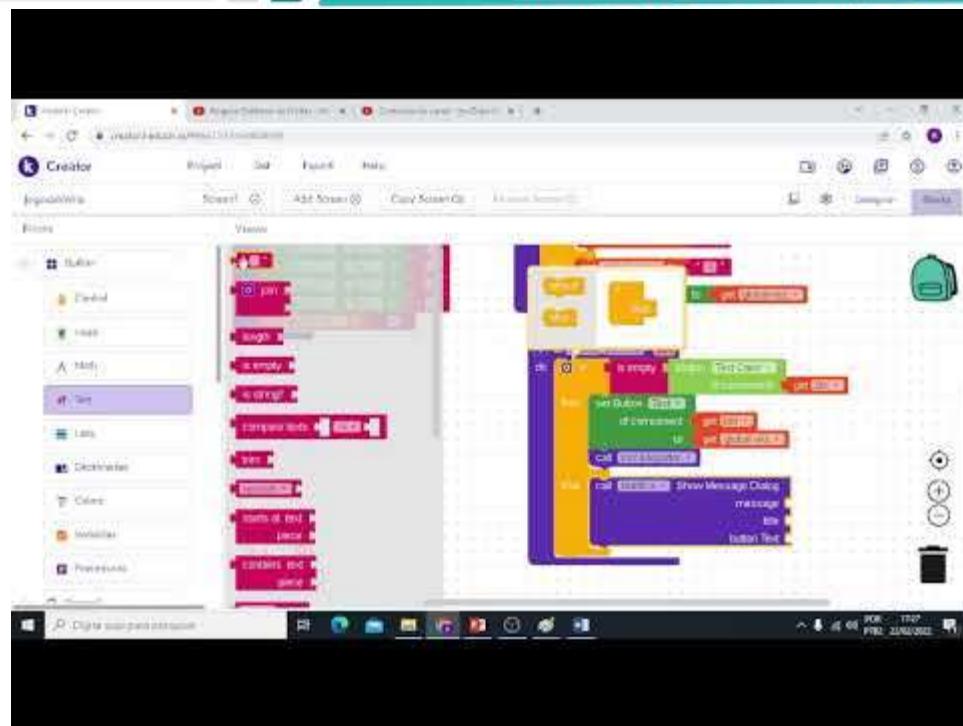
Estrutura de repetição é utilizada para que uma parte do código seja executada e repetida durante uma determinada condição. É uma estrutura muito utilizada na programação e uma das suas principais funções é gerar uma economia de código em certas rotinas.



Figura 7- Exemplo de estrutura de repetição.

Agora iremos criar dois procedimento que terão a função de alternar entre os jogadores e marcar o símbolo do jogador em cada botão clicado no tabuleiro. Portanto o aluno deverá assistir ao vídeo para compreender a programação dos procedimentos.

Agenda 14 - Procedimentos trocaJogador e marca_posicao ,disponível em :
<https://youtu.be/wdYw1HgOUZA>



- Clique no construtor **Procedure** e arraste o objeto **to procedure do**

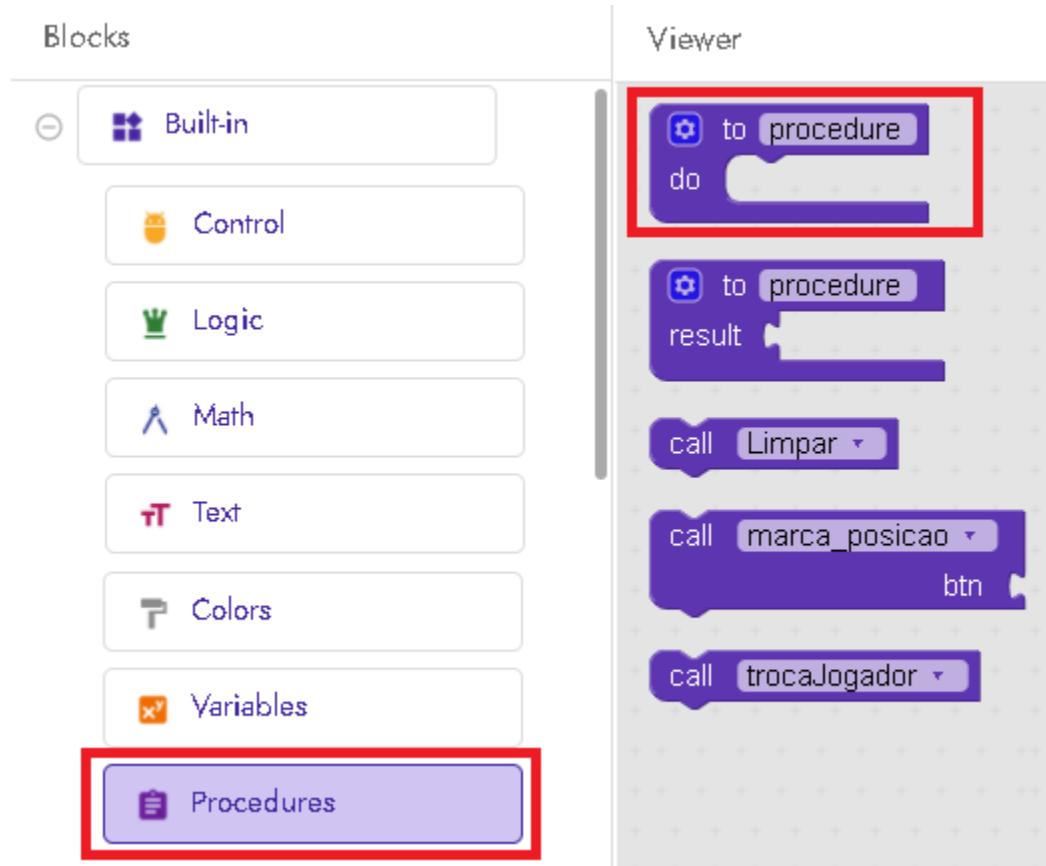


Figura 8 - Criando o procedimento verifica_velha

- Digite o **Verifica_velha** para o procedimento criado anteriormente.
- Clique no construtor **Control**, arraste o componente **IF** e encaixe no procedure **verifica_velha**.

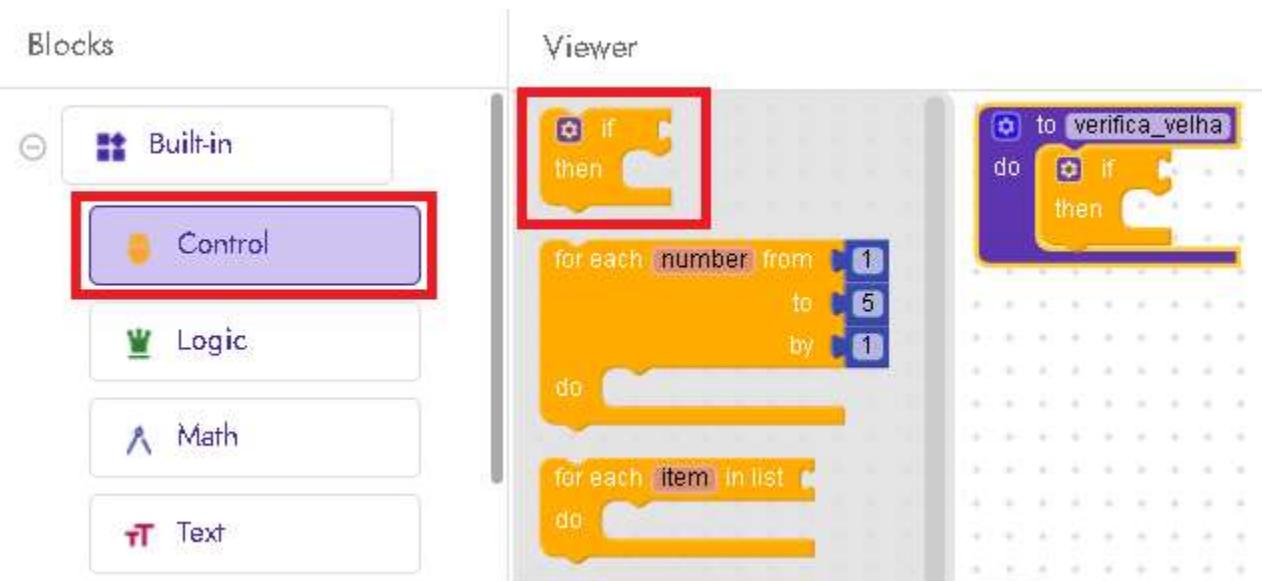


Figura 9 - Inserindo o componente IF.

- Arraste a condição **elseif** por 8 vezes, dentro da condição **IF**.

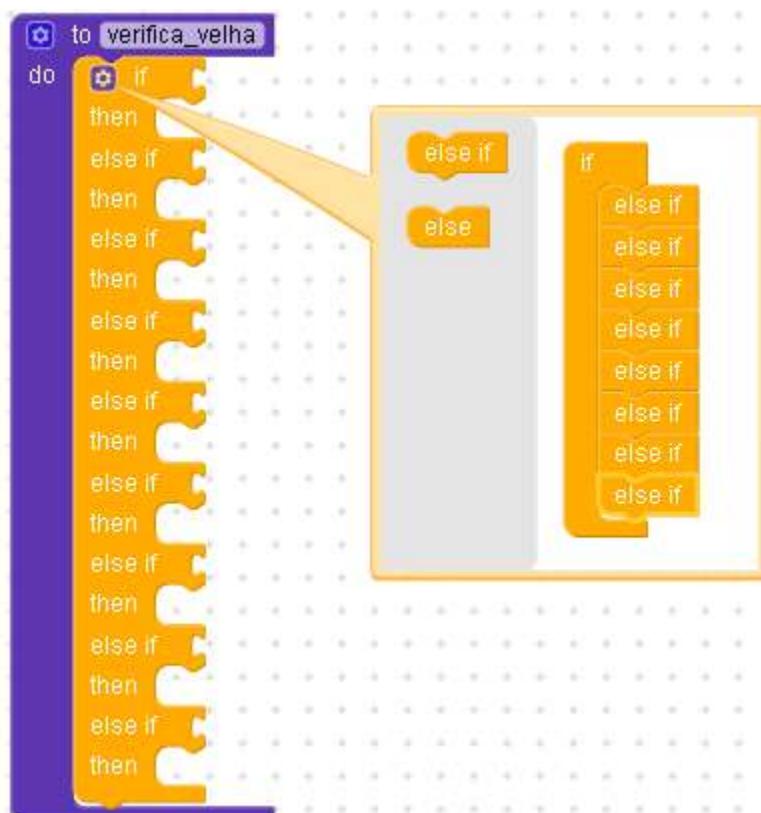


Figura 10 - Personalizando o procedimento verifica_velha.

- Clique no construtor **Logic**, arraste a opção **de teste de igualmente** e encaixe no bloco **IF**.



Figura 11 - Inserindo a condição no bloco de comando IF.

- Clique no construtor **Logic**, arraste novamente o componente de comparação e vamos encadear com a primeira estrutura de igualdade.



Figura 12- Inserindo o bloco de igualdade encadeada no bloco IF.

- O aluno deverá montar a primeira condição, como a **Figura 38**.

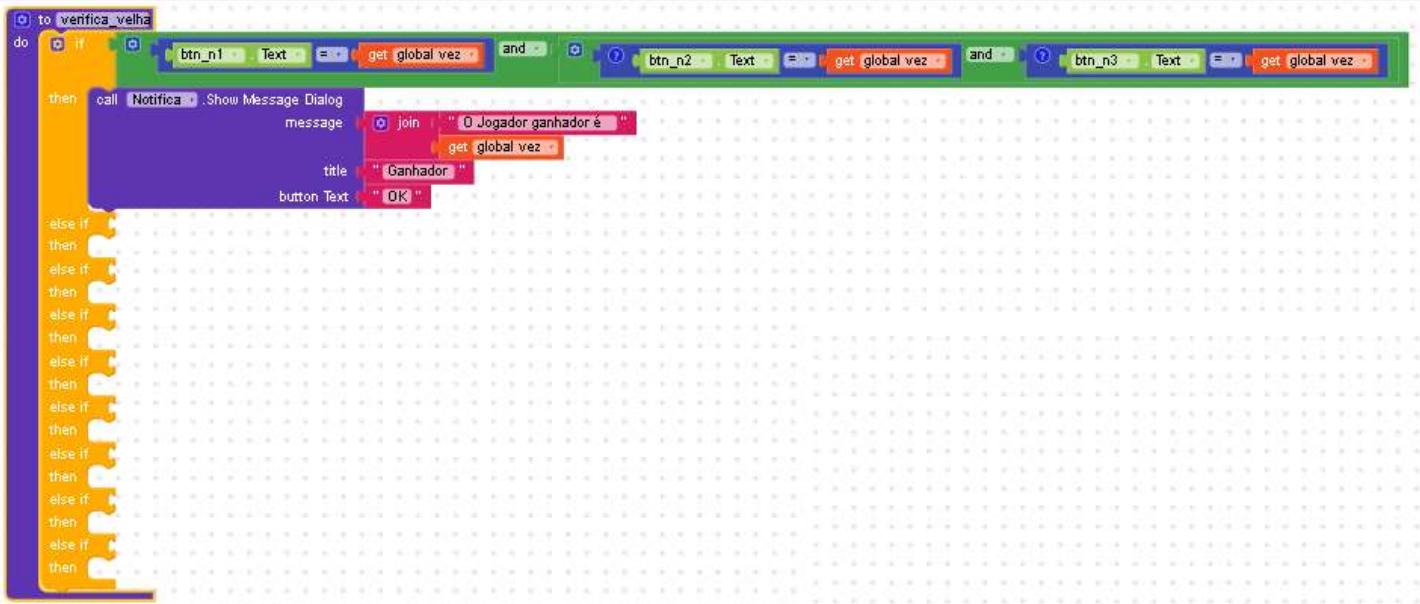


Figura 13 - Montagem do bloco do procedimento verifica_velha.

- Após a montagem do primeira condição, o aluno deverá construir todo o código que segue na **Figura 39**. Pois existe várias formas de ganhar o jogo, seja ela no sentido vertical, horizontal ou diagonal. Portanto a quantidade de código é um pouco grande, mas o resultado no final vale a pena.

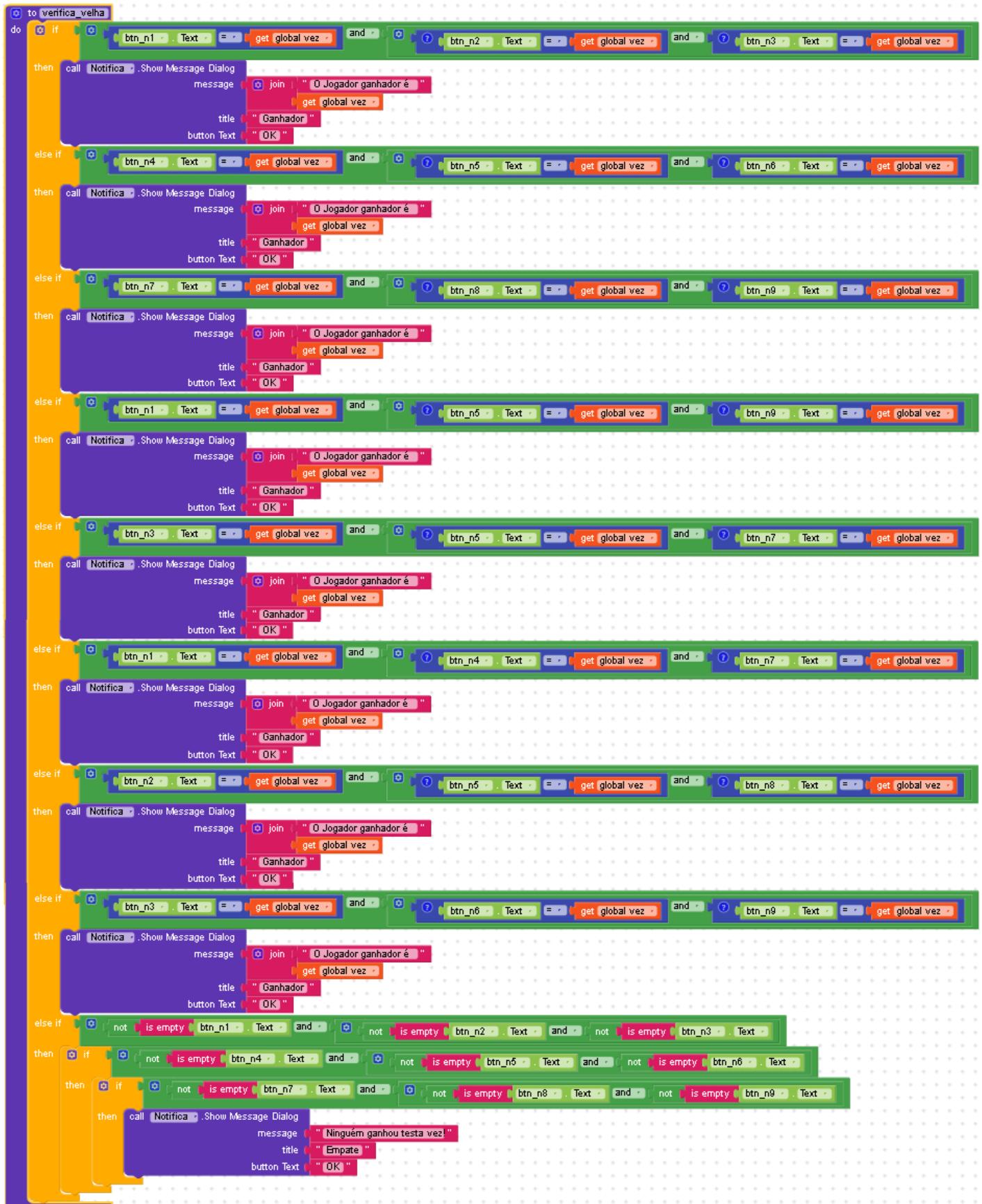


Figura 14 - Procedimento Verifica_Velha Completo.

Sendo assim, ao terminar o código do procedimento verifica_velha, o aluno deverá atualizar o código do procedimento **marca_posição**. Pois durante o jogo o aplicativo verifica se alguém ganhou a partida.

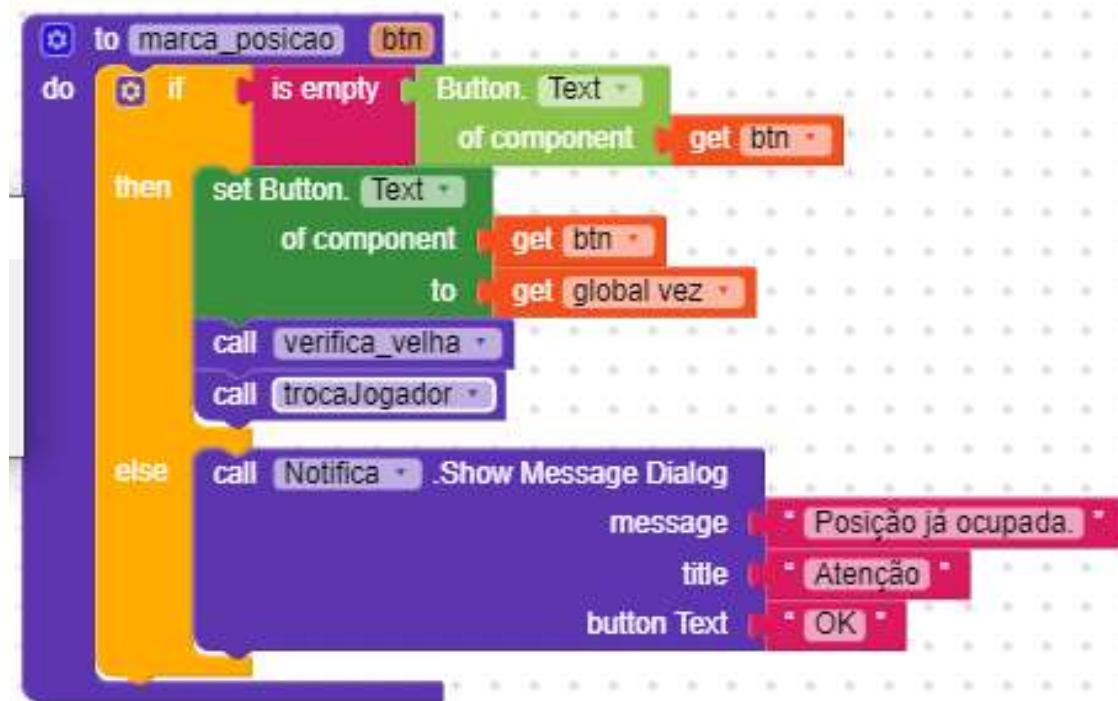


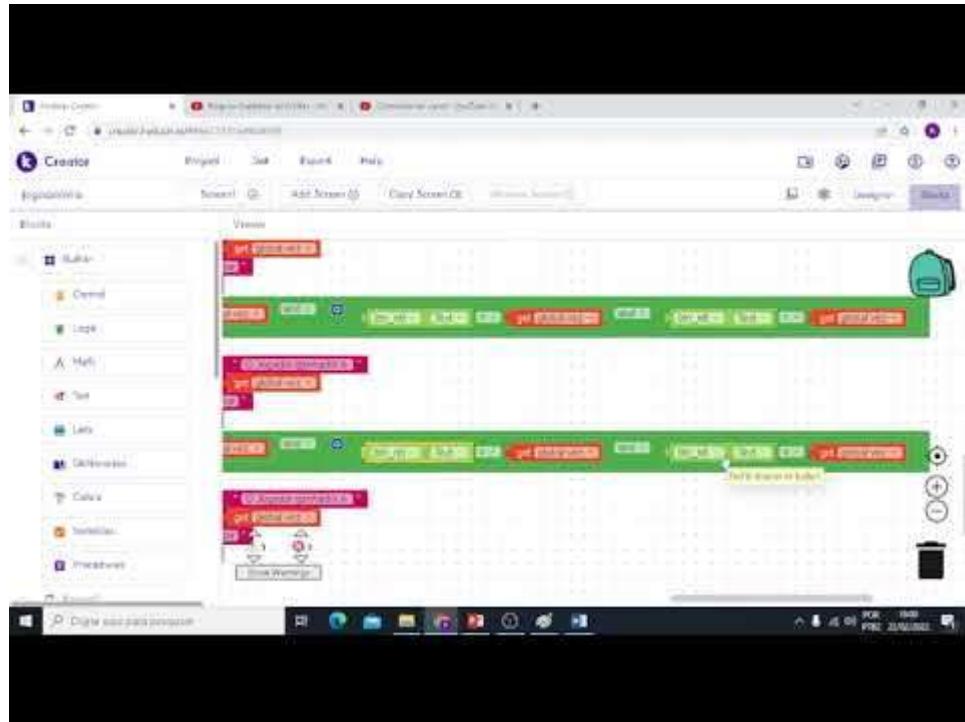
Figura 15 - Atualização procedimento marca_posicao.



Assista o vídeo a seguir para compreender algumas rotinas necessárias para o bom funcionamento do aplicativo jogo da velha.

Agenda 14 – Procedimento verifica_velha, disponível em:

<https://youtu.be/DifjY5Wa8pM>



Bons Estudos!

Confira o arquivo completo:

Jogo da Velha.aia

AGENDA 15

NOTIFICAÇÕES



GEEaD - Grupo de Estudos de Educação a Distância
Centro de Educação Tecnológica Paula Souza

GOVERNO DO ESTADO DE SÃO PAULO
EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS
PROGRAMAÇÃO MOBILE I

Expediente

Autor:

GUILHERME HENRIQUE GIROLLI

Atualização Técnica:

ROGÉRIO GALDIANO DE FREITAS

Revisão Técnica:

Eliana Cristina Nogueira Barion

Revisão Gramatical:

Juçara Maria Montenegro Simonsen Santos

Editoração e Diagramação:

Flávio Biazim



Uma notificação é uma mensagem criada por um aplicativo, em primeiro plano ou em execução em segundo plano, e serve para chamar a atenção do usuário do sistema operacional Android para uma ocorrência que foi gerada no aplicativo ou fora dele.

O sistema Android é responsável por gerar a notificação na área de notificação, que fica próximo ao relógio do dispositivo. E quando o usuário deseja visualizar maiores informações sobre a notificação, é necessário abrir a gaveta de notificações, onde ela é exibida de maneira completa.

A **Figura 1** mostra a área de notificação e a **Figura 2** mostra a gaveta de notificações.

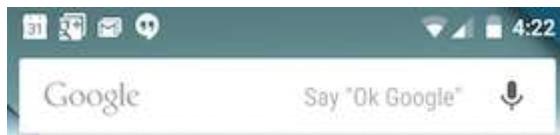


Figura 1 - Área de notificação. Disponível em:
<https://developer.android.com/guide/topics/ui/notifiers>

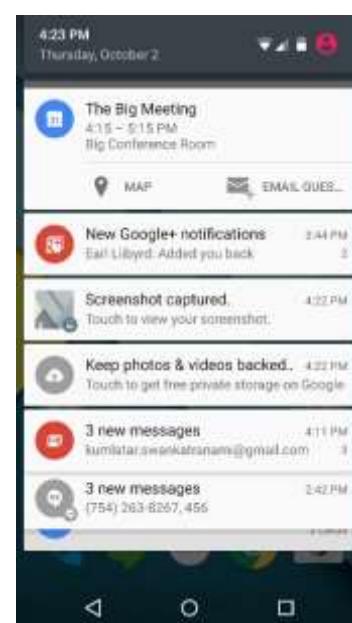


Figura 2 - Gaveta de notificações. Disponível em:
<https://developer.android.com/guide/topics/ui/notifiers/notifications?hl=pt-br>

Para criar uma notificação no sistema operacional Android é necessário trabalhar com algumas regras. Desta forma o **Kodular** oferece pacotes de classes que auxiliam o desenvolvimento das notificações.

O Componente **Notifier** foi desenvolvida para facilitar a criação das notificações. Ele permite o desenvolvimento de notificações expandidas e notificações simples de acordo com a necessidade do desenvolvedor e da versão do sistema operacional Android.

Ele oferece suporte para que o desenvolvedor construa os recursos mais simples de uma notificação, como o título, ícones e a mensagem. E oferece suporte aos recursos mais avançados como notificações expandidas, que possuem botões para inúmeras utilizações. A **Figura 3** mostra um exemplo de notificação expandida.

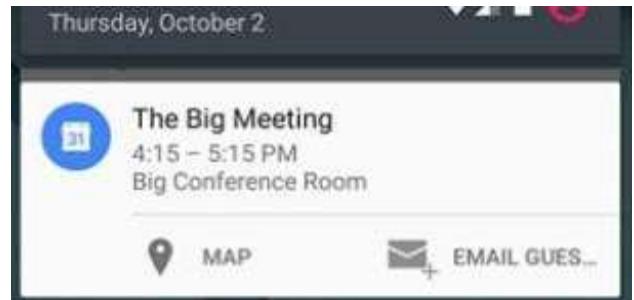


Figura 3 - Notificação expandida, com dois botões.

Vale ressaltar que as notificações que possuem botões, somente são exibidas em sua forma completa, ou seja, mostrando a mensagem e os botões nas versões superiores ao Android 4.1.

Aplicando a notificação no projeto

Vamos desenvolver um projeto para testar as notificações, é importante ressaltar que vamos os recursos da plataforma de desenvolvimento **Kodular**.

- Abra a plataforma de desenvolvimento do Kodular: <https://www.kodular.io/creator>
- Clique no botão **Create Project**

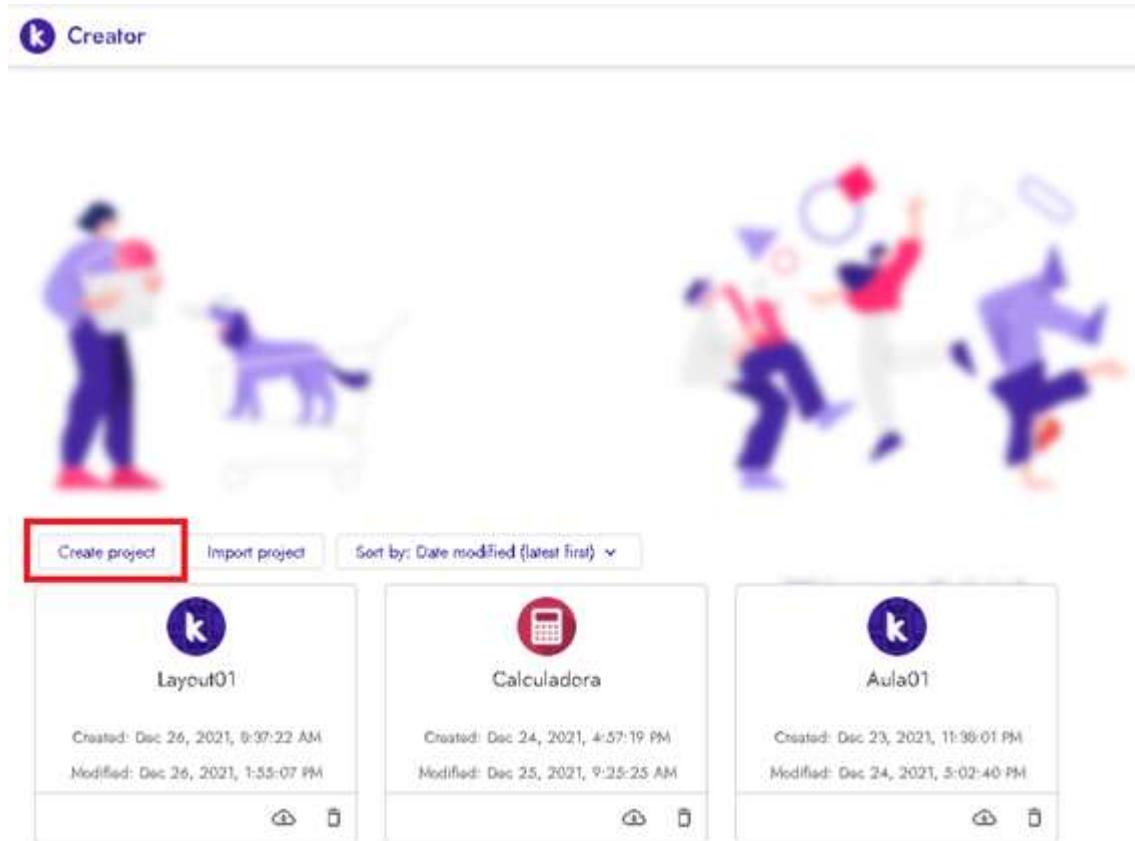


Figura 4 - Criando um novo projeto.

- Digite o nome **Notificacao** e clique no botão **Next**.



Figura 5 - Criando o projeto Notificacao.

- Clique no botão **Finish** para finalizar a criação do novo projeto.
- Altere as propriedades do objeto **SCREEN**

Propriedade	Valor	Função
Background Color	#000000FF	Definir a cor de fundo do aplicativo.
Title	Notificações	Definir o título da aplicação em desenvolvimento.
Align Horizontal	Center	Alinhar todos os componentes ao centro.

Através deste projeto, vamos explicar alguns exemplos sobre o uso de notificações em aplicativos. Sendo que a plataforma **Kodular** apresenta várias formas.

- Insira um componente **Button**, categoria **User Interface**.

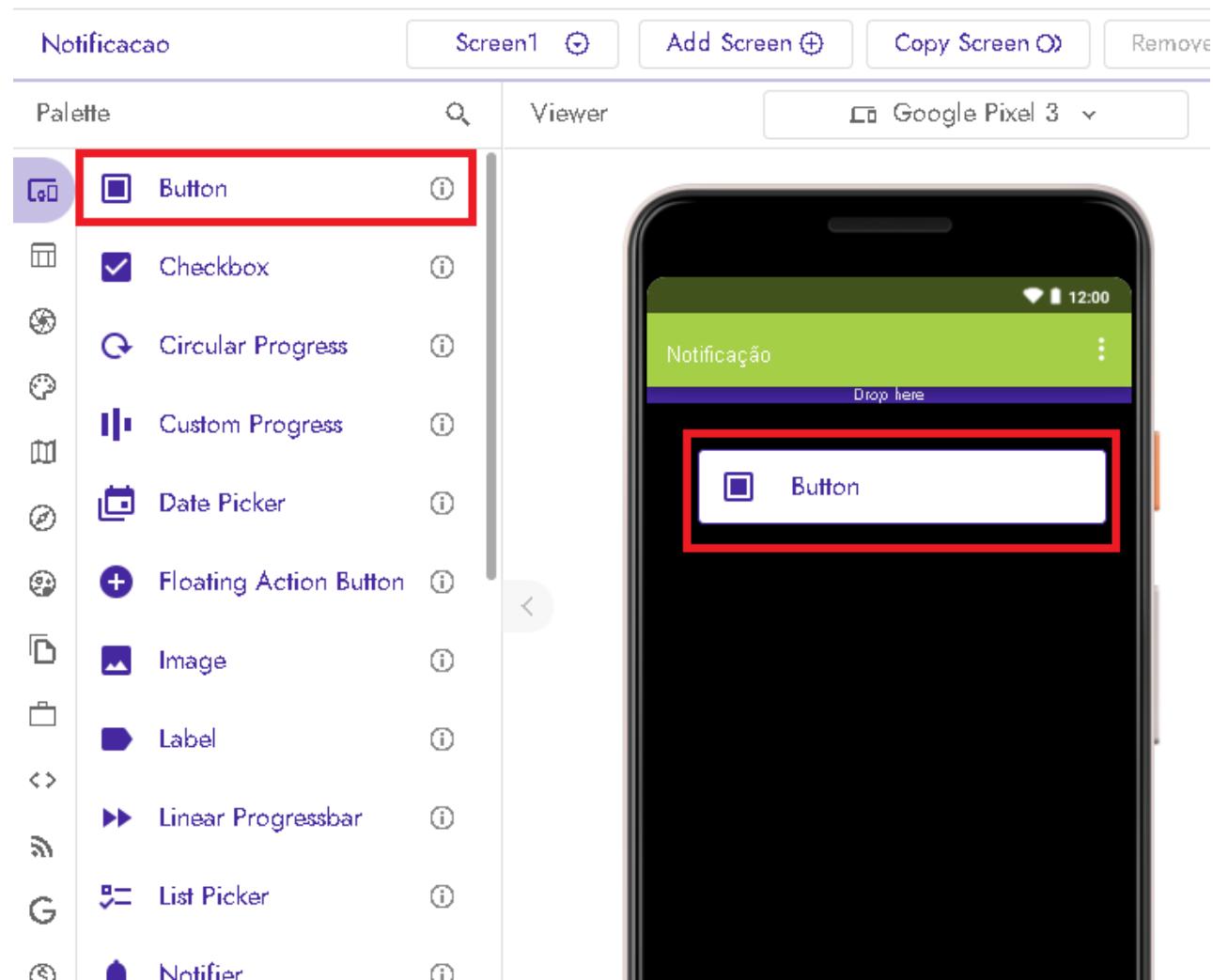


Figura 6 - Inserindo componente button no Projeto.

- Altere as propriedades do objeto **Button**.

Propriedade	Valor	Função
Width	Fill Parent	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte

Propriedade	Valor	Função
Text	Notificação 01	Definir o conteúdo a ser exibido do componente button.
Name	btn_notifica01	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**.
- Altere as propriedades do objeto **Button**.

Propriedade	Valor	Função
Width	Fill Parent	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Notificação 02	Definir o conteúdo a ser exibido do componente button.
Name	btn_notifica02	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**.
- Altere as propriedades do objeto **Button**.

Propriedade	Valor	Função
Width	100px	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Notificação 03	Definir o conteúdo a ser exibido do componente button.
Name	btn_notifica03	Definir o nome do componente.

- Clique na categoria **User Interface**, clique no componente **Notifier** e arraste para a área **VIEWER**.

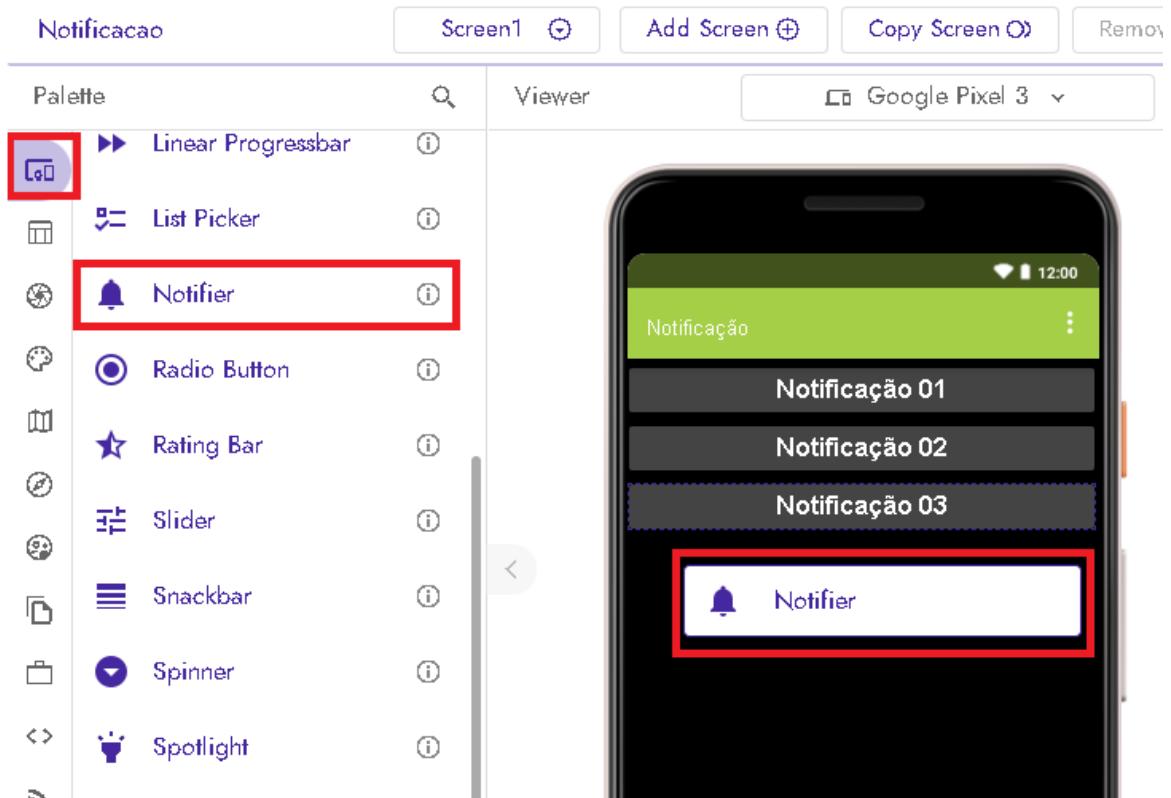


Figura 7 - Inserindo o componente Notifier

A função deste componente será apresentar notificações, ou seja, apresentar uma caixa de diálogo para exibir informações importantes.

- Altere as propriedades do componente **Notifier**.

Propriedade	Valor	Função
Name	Notifica	Definir o nome do componente.

- Altere para o cenário de construção de **Blocks**

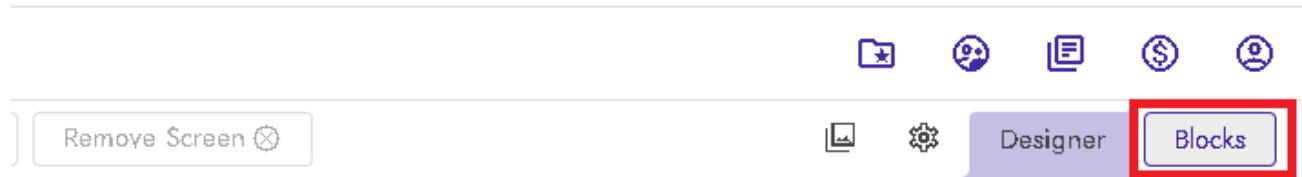


Figura 8 - Alternando para o cenário Blocks.

- Clique no objeto **btn_notifica01**, arraste o objeto **when btn_notifica01.click**

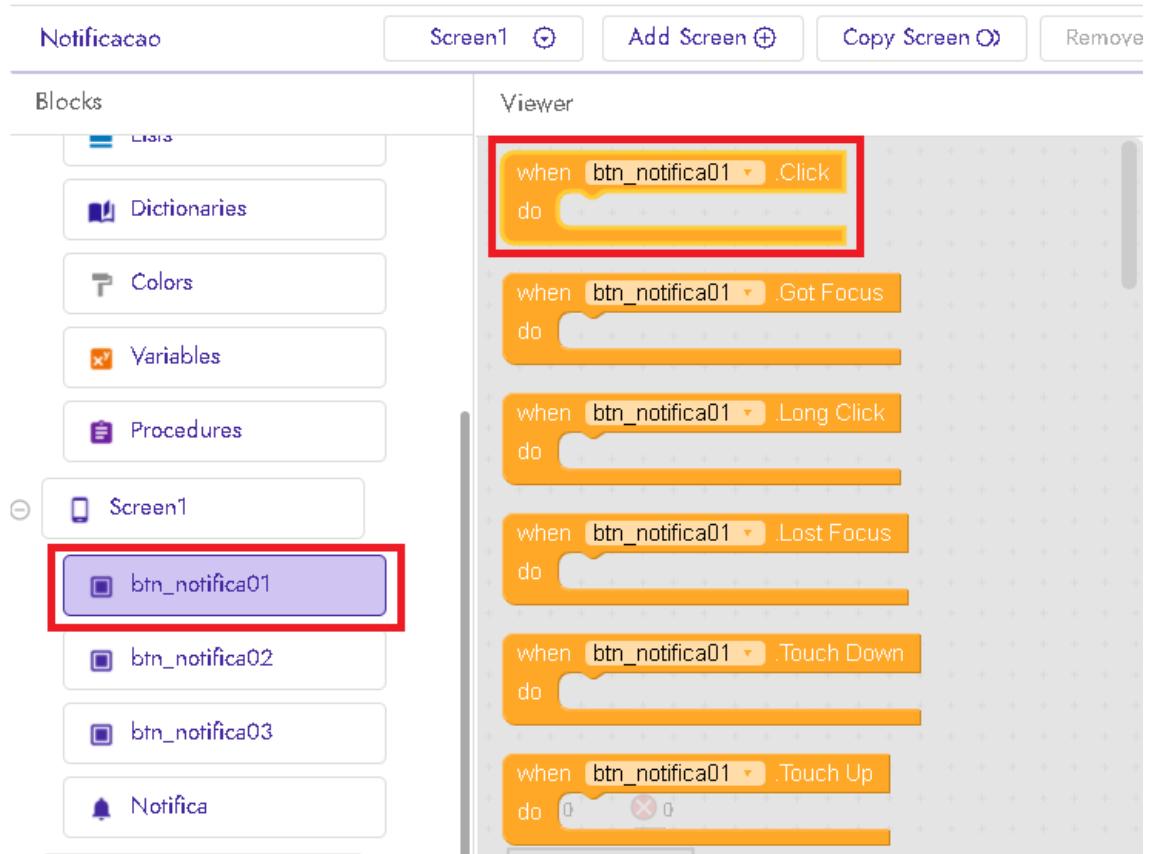


Figura 9 - Inserir o evento click do botão `btn_notifica01`.

- Clique no objeto **Notifica**, arraste o objeto **call notifica.Show Alert notice** para dentro do evento click do **btn_notifica01**.

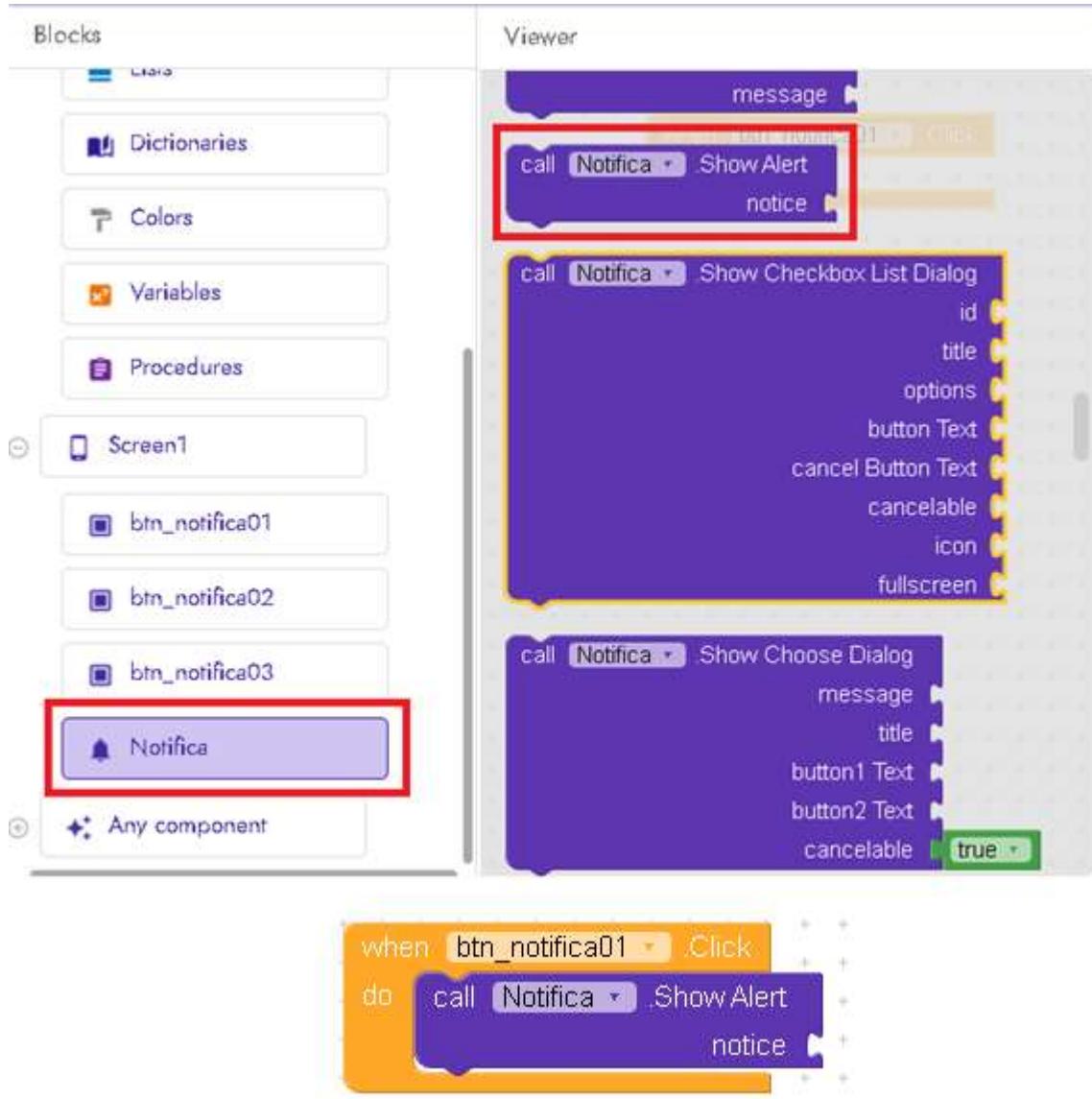


Figura 10 - Inserindo a notificação de alerta

- Clique no construtor **Text**, arraste o objeto **Text String (vazio)** e encaixe ao lado direito da notificação.

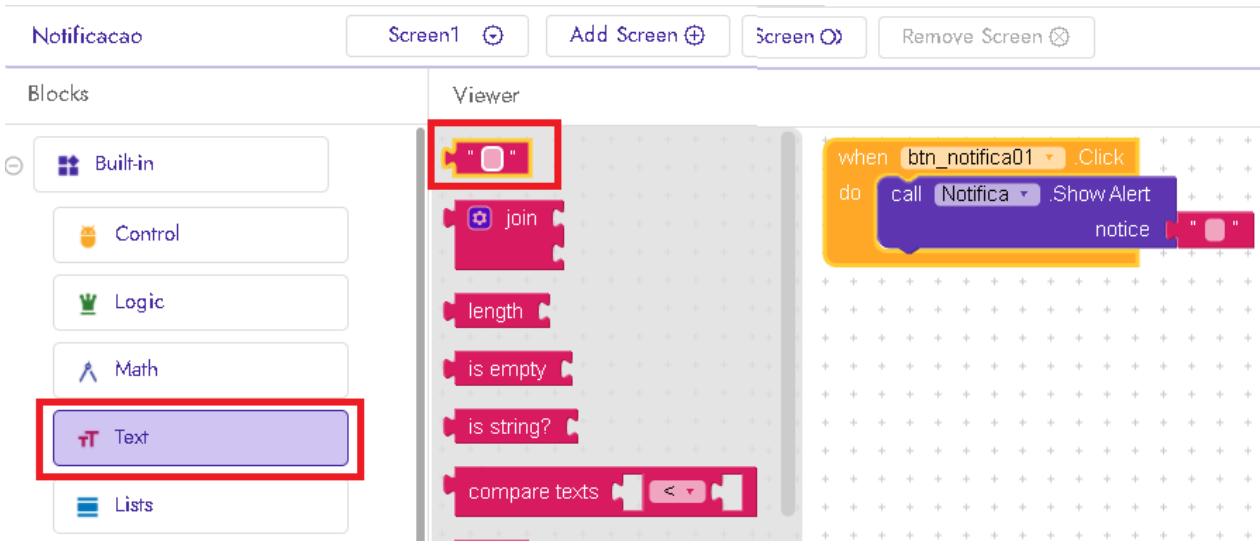


Figura 11 - Inserindo o componente String

- Altere o conteúdo do objeto **Text** para **Notificação de Alerta**



Figura 12 - Inserindo mensagem na notificação.

- Clique no menu **Test**, opção **Connect to companion**

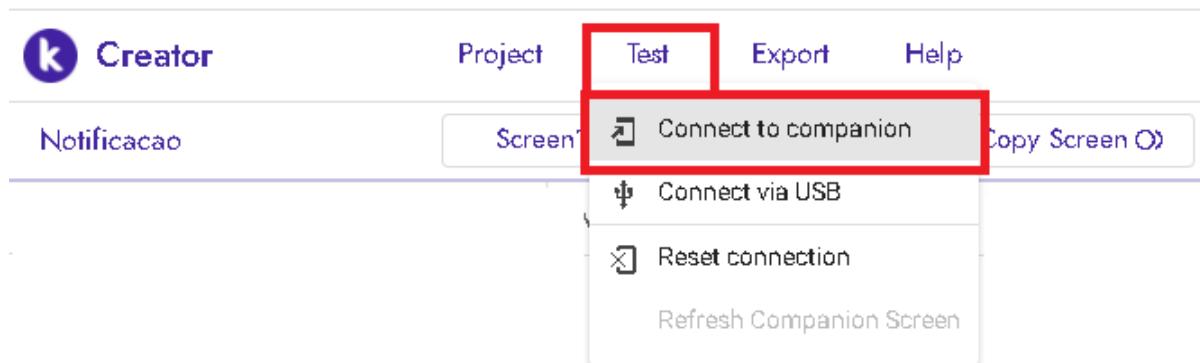


Figura 13 - Testando o Projeto.

Ao gerar o qrcode, o aluno deverá utilizar o aplicativo do dispositivo móvel **Kodular Companion** para testar o aplicativo. Sendo apresentado a primeira notificação ao clicar no primeiro botão.

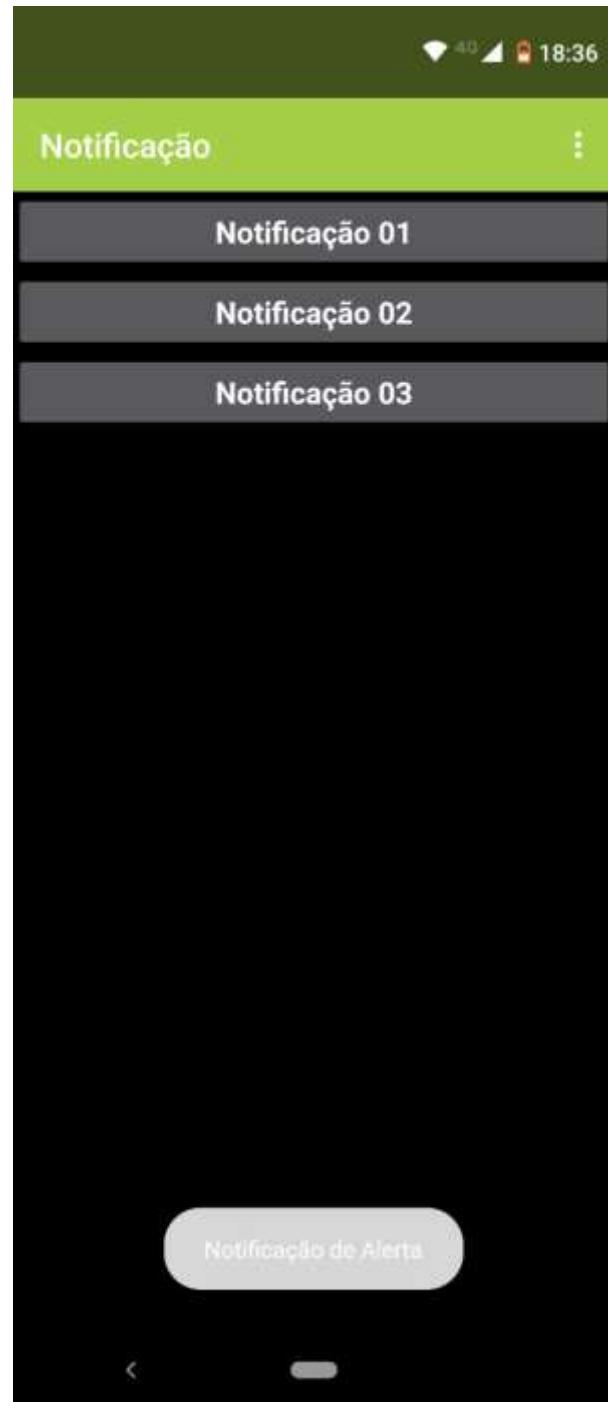


Figura 14 -Notificação Inserida com sucesso.

- Clique no objeto **btn_notifica02**, arraste o objeto **when btn_notifica02.click**
- Clique no objeto **Notifica**, arraste o objeto **call notifica.show message dialog**
- Clique no componente **Text**, arraste o objeto **String Text** e complete o bloco de notifcação de acorod coma **Figura 13**.

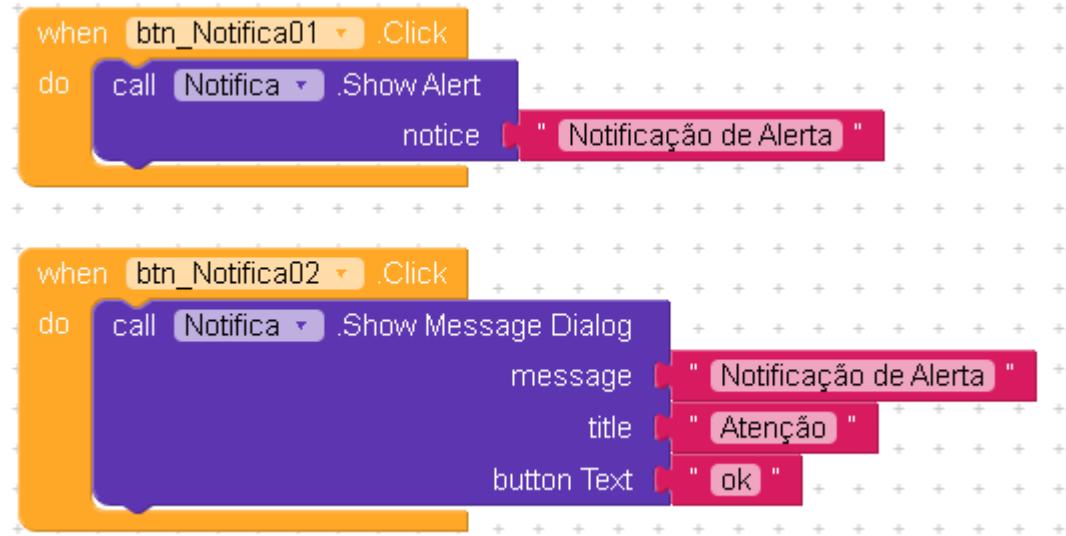


Figura 15 - Inserindo o bloco de código da segunda Notificação

- Clique no menu **Test**, opção **Connect to companion**

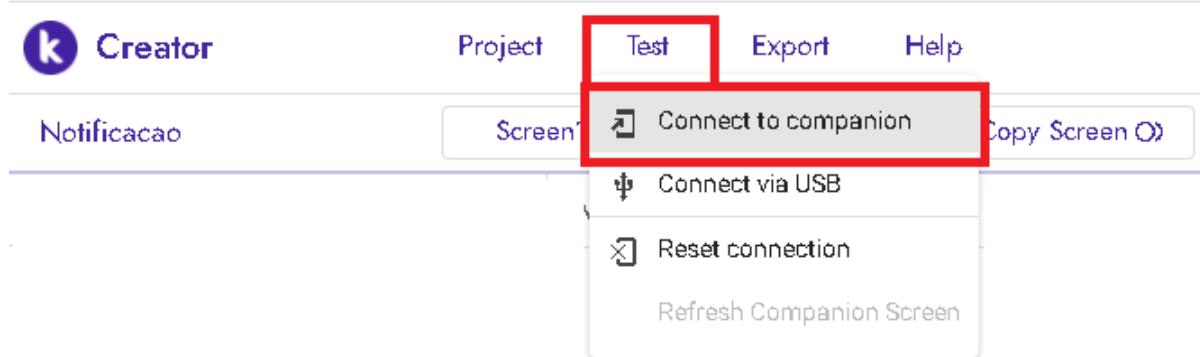


Figura 16 - Testando o projeto.

Ao gerar o qrcode, o aluno deverá utilizar o aplicativo do dispositivo móvel **Kodular Companion** para testar o aplicativo. Sendo apresentado a primeira notificação ao clicar no primeiro botão.

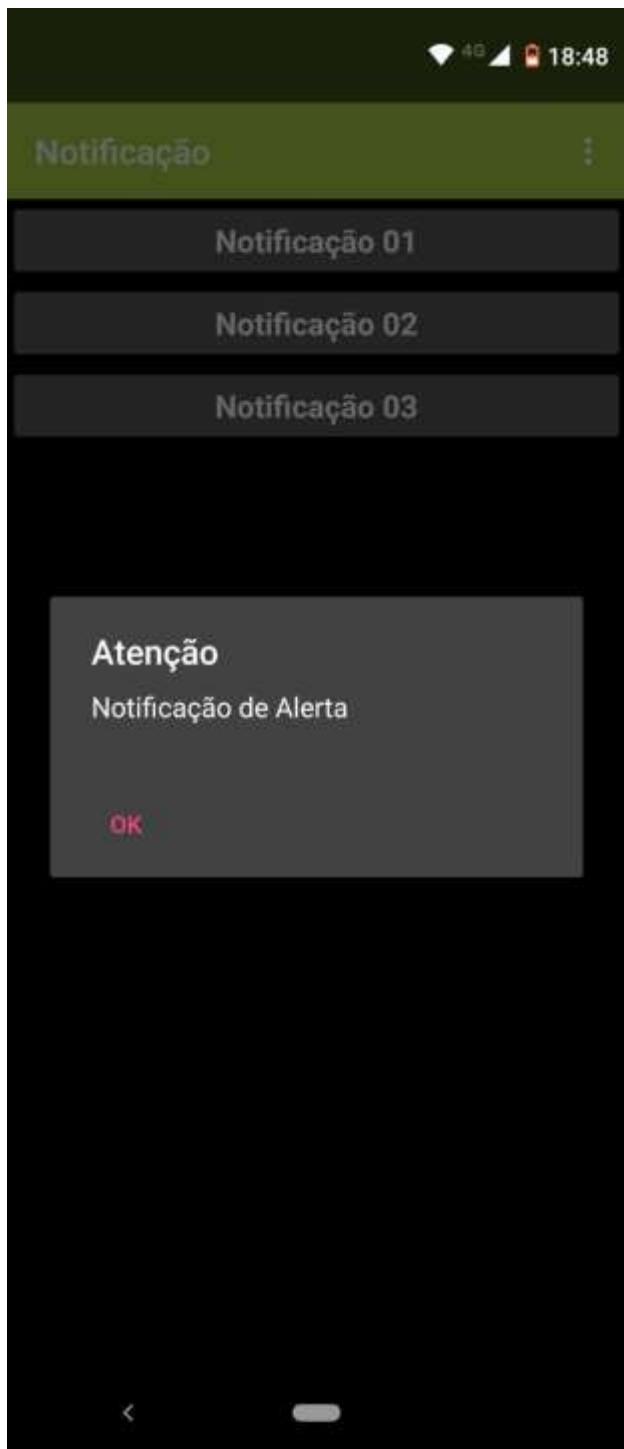


Figura 17 - Notificação Exibida com sucesso.

Neste momento, o aplicativo irá apresentar agora uma notificação e de acordo com a resposta do usuário a programação irá permitir continuar no aplicativo ou fechar. Portanto podemos fazer uma notificação com interação do usuário.

- Construa o bloco de programação da **Figura 17** para realizar a interação com o usuário.

```

when [btn_Notifica03].Click
do call Notifica .Show Choose Dialog
    message "Deseja sair do aplicativo?"
    title "Atenção"
    button1 Text "OK"
    button2 Text "Cancelar"
    cancelable false

when [Notifica].After Choosing
choice
do if get choice = "OK"
then close application

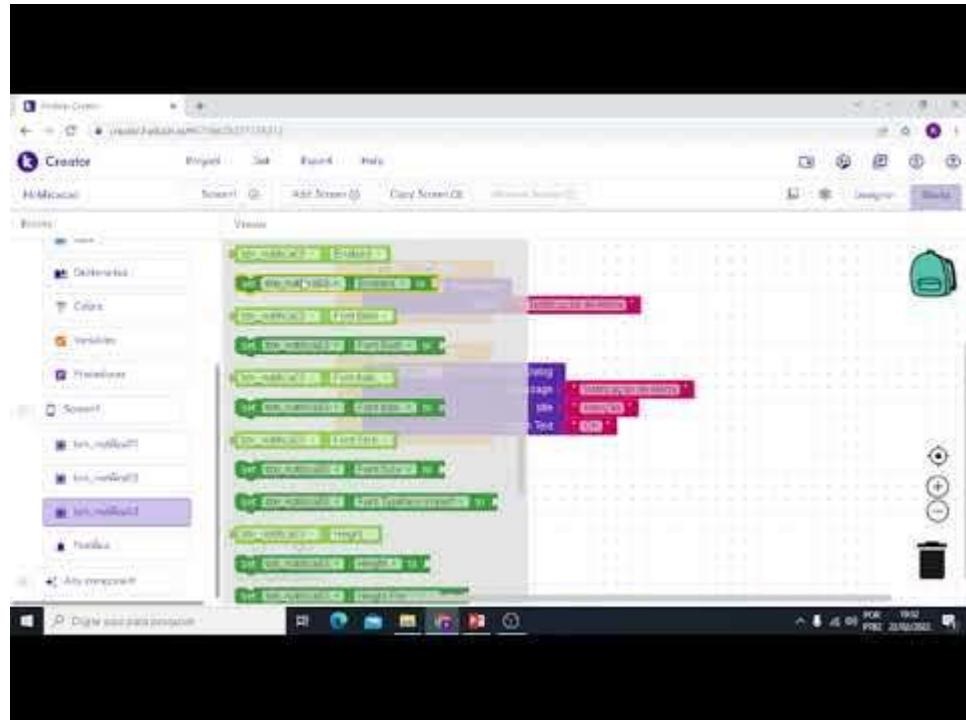
```

Figura 18 - Código de blocos da terceira notificação.

Caso o aluno encontre alguma dificuldade na construção deste aplicativo, o mesmo poderá assistir ao vídeo referente ao assunto.

Agenda 15 – Notificações, disponível em:

<https://youtu.be/2dTaRqC9vv0>



- Clique no menu **Test**, opção **Connect to companion**

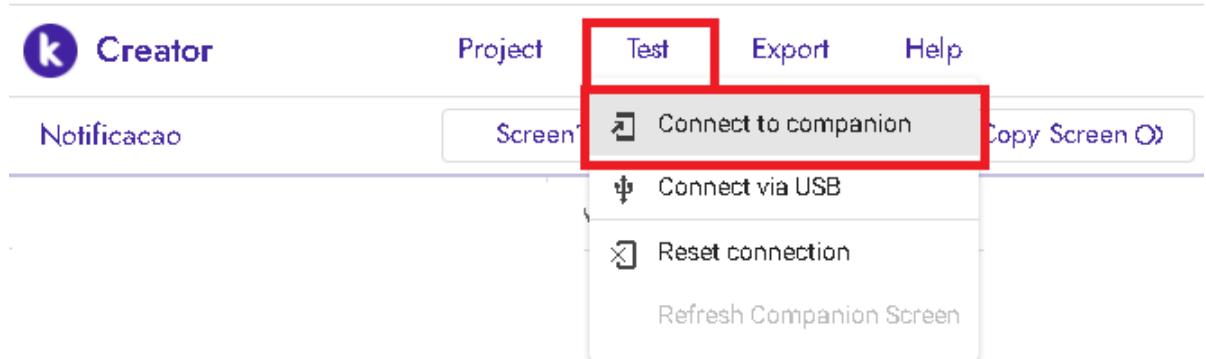


Figura 19 - Testando o projeto

Ao gerar o qrcode, o aluno deverá utilizar o aplicativo do dispositivo móvel **Kodular Companion** para testar o aplicativo. Sendo apresentado a primeira notificação ao clicar no primeiro botão.

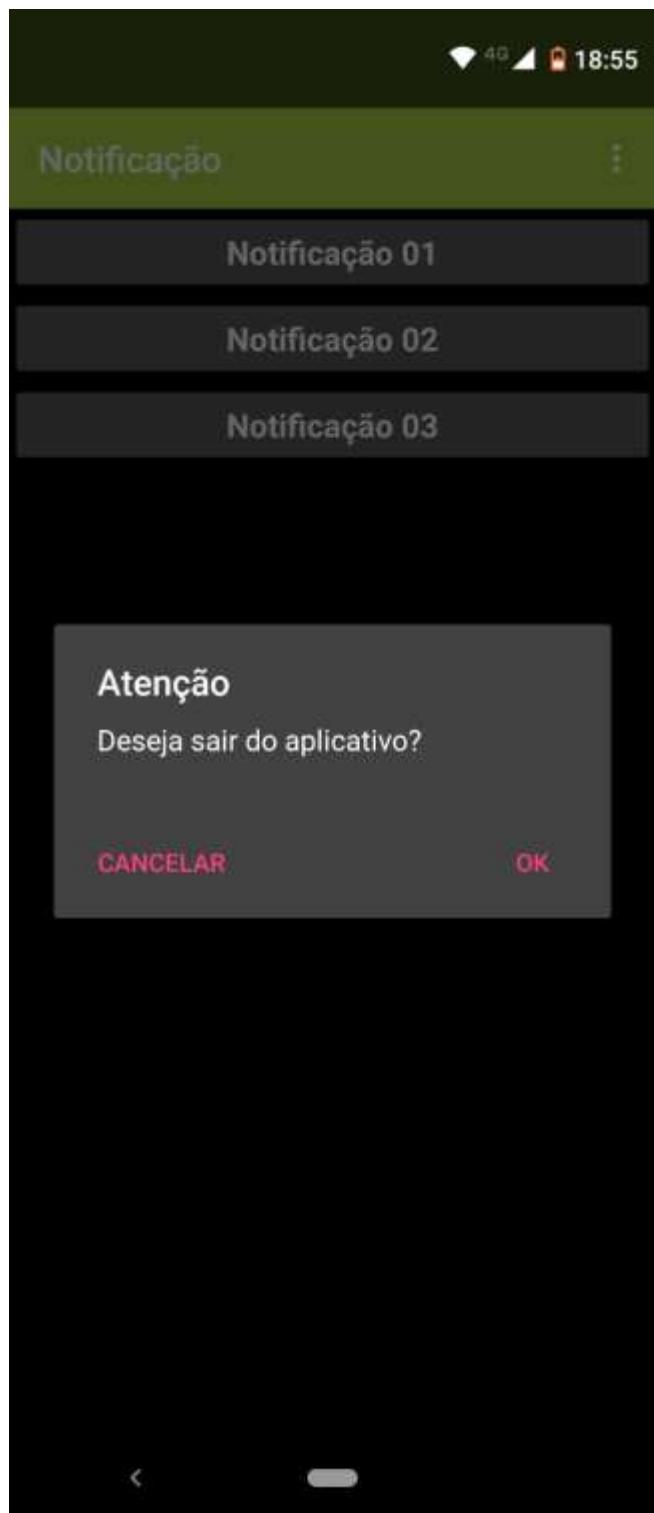


Figura 20 - Exibindo a terceira notificação.

Arquivos disponíveis para download

Projeto Finalizado:
[Notificacao.aia](#)

AGENDA 16

PERSONALIZANDO
OS BOTÕES



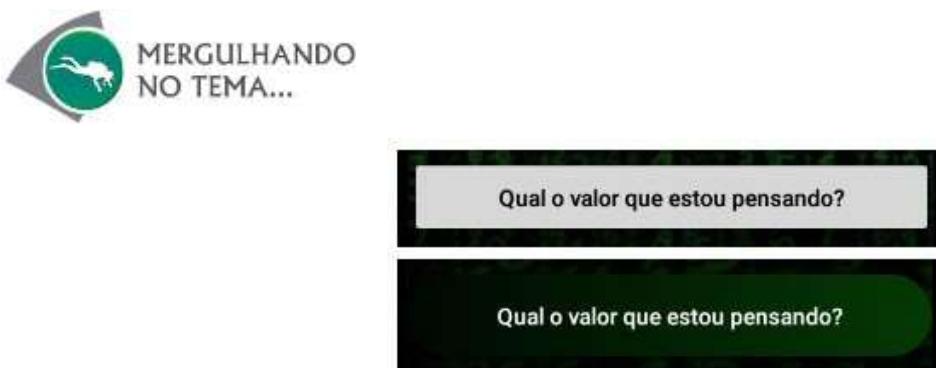


Figura 1 - Button convencional e Button Customizado

Para alterar o layout de um componente utilizado na tela do aplicativo, é necessário modificar as propriedades disponíveis para este componente. É por meio dessa configuração que alcançamos as modificações necessárias para deixar a interface do usuário mais atraente e moderna.

A **Figura 1** demonstra um componente “Button” em seu layout convencional e um mesmo componente que sofreu uma customização.

- Abra a plataforma de desenvolvimento do Kodular: <https://www.kodular.io/creator>
- Clique no botão **Create Project**.

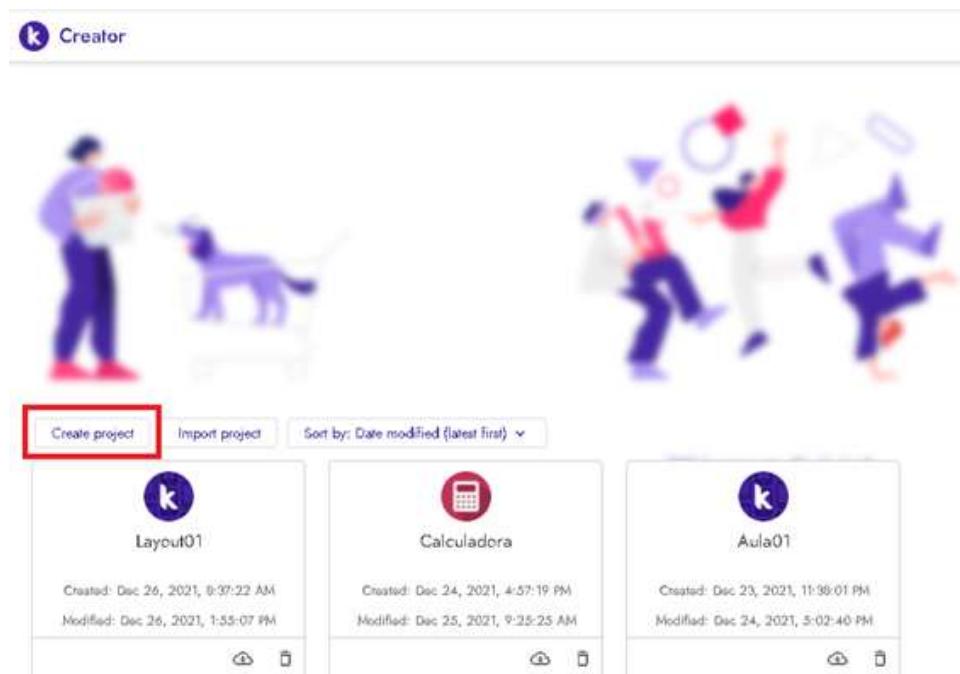


Figura 2 - Criando um novo projeto.

- Digite o nome **Botoes** e clique no botão **Next**.



Figura 3 - Criando o projeto Botões

- Clique no botão **Finish** para finalizar a criação do novo projeto.
- Altere as propriedades do objeto **SCREEN**

Propriedade	Valor	Função
Title	Botões Personalizados	Definir o título da aplicação em desenvolvimento.
Align Horizontal	Center	Alinhar todos os componentes ao centro.

- Insira um componente **Button**, categoria **User Interface**.

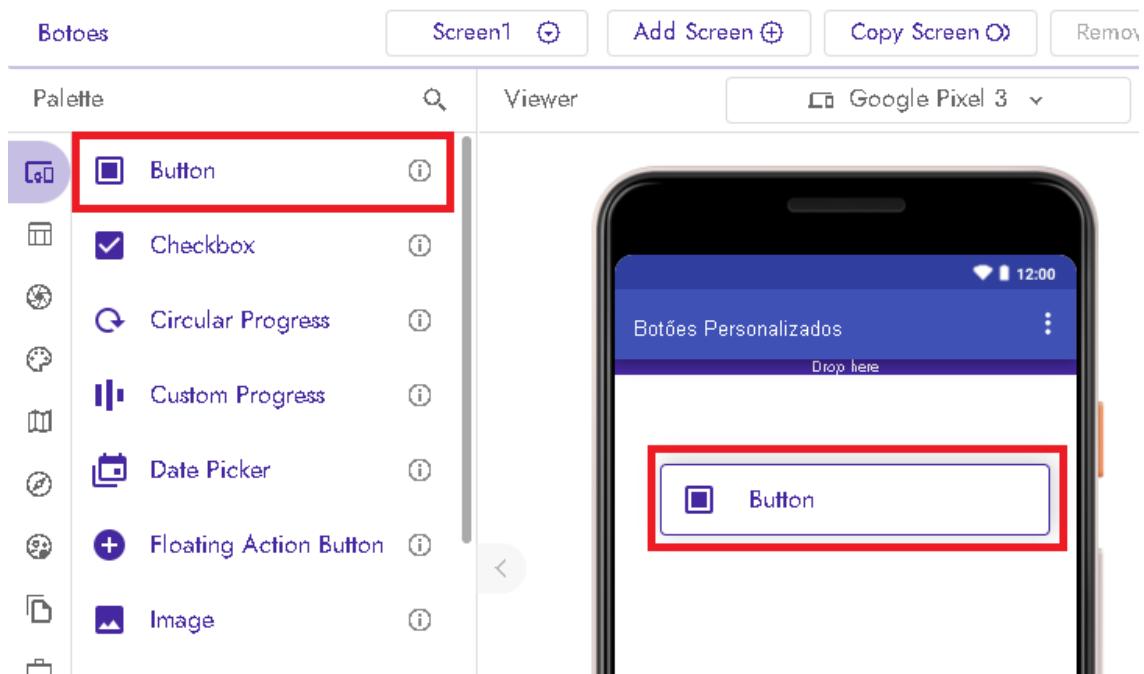


Figura 4 -- Inserindo componente button no Projeto

- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Width	200px	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Exemplo 01	Definir o conteúdo a ser exibido do componente button.
Shape	Rounded	Arredondar os campos do componente button.
Name	btn_Exemplo01	Definir o nome do componente.

- Insira um componente **Button**, categoria **User Interface**.
- Altere as propriedades do componente **Button**.

Propriedade	Valor	Função
Width	200px	Definir a largura do componente button.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Exemplo 01	Definir o conteúdo a ser exibido do componente button.
Shape	Oval	Arredondar os campos do componente button.
Name	btn_Exemplo01	Definir o nome do componente.

Após a inserção de todos os componentes e alterações em suas respectivas propriedades, o layout do aplicativo será idêntico a **Figura 5**.



Figura 5 - Visualizando o Layout

Vamos construir um botão personalizado com mais recursos, para deixar a interface de forma mais atraente.

- Clique na categoria **Layout**

A screenshot of a mobile application builder interface. On the left, a vertical palette titled "User Interface" contains categories: "Layout" (selected), "General" (highlighted with a red border), "Lists", "Navigation", "Views", and "Media". The "General" category is currently active. To the right is a "Viewer" window showing a preview of the smartphone screen from "Google Pixel 3". The preview displays the same "Botões Personalizados" interface as Figura 5, with "Exemplo 01" and "Exemplo 02" buttons. Navigation icons are visible along the bottom of the viewer window.

Figura 6 - Componente Layout

- Clique na categoria **General** e arraste o componente **Card View** para a área **VIEWER**

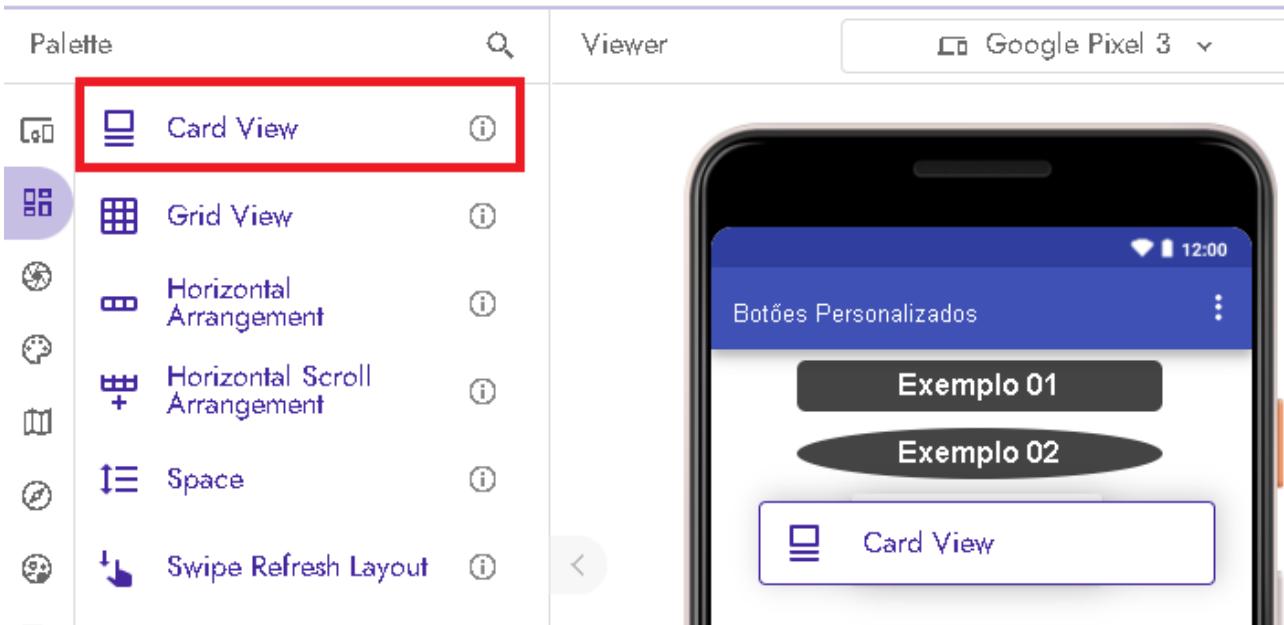


Figura 7 - Menu Layout, opção General, Card View

- Altere as propriedades do componente **Card View**.

Propriedade	Valor	Função
Name	Card_botao01	Define o nome do componente.

- Clique na opção **Horizontal Arragement** e arraste dentro do componente **Card View**

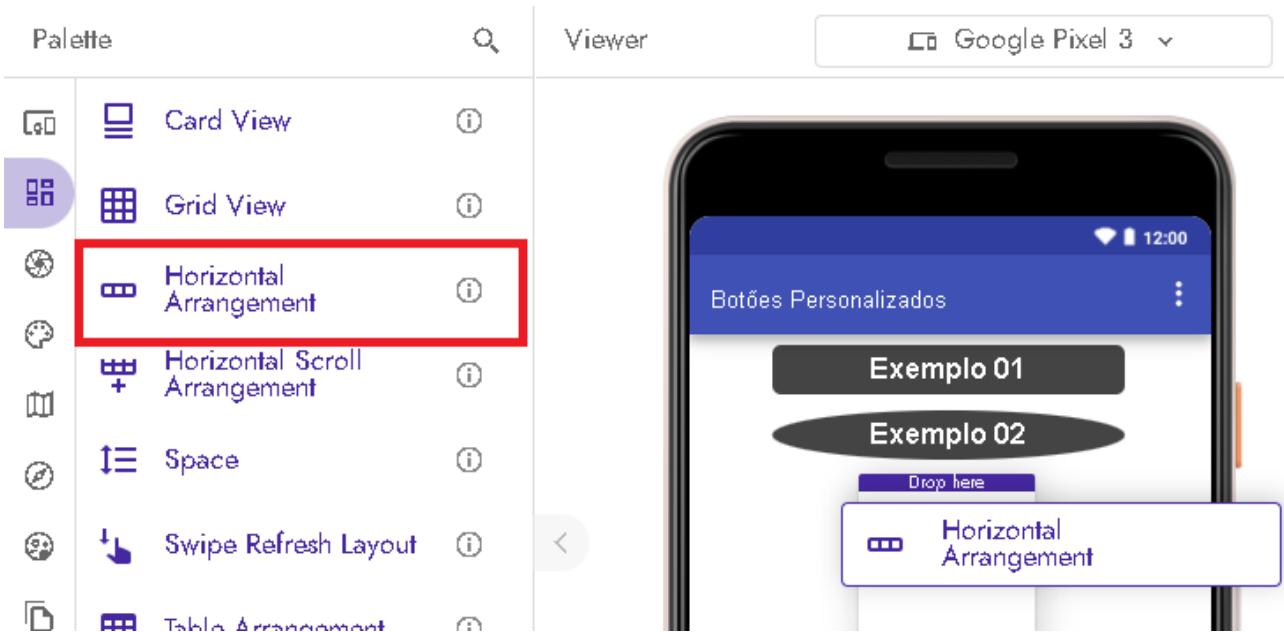


Figura 8 - Inserindo o componente Horizontal Arragement.

- Altere as propriedades do componente **Horizontal Arragement**.

Propriedade	Valor	Função
Align Horizontal	Center	Definir o alinhamento centralizado na horizontal.
Align Vertical	Center	Definir o alinhamento centralizado na vertical
Name	Linha01	Define o nome do componente.

- Clique na categoria **User Interface** e arraste o componente **Image** dentro do componente **Linha01**

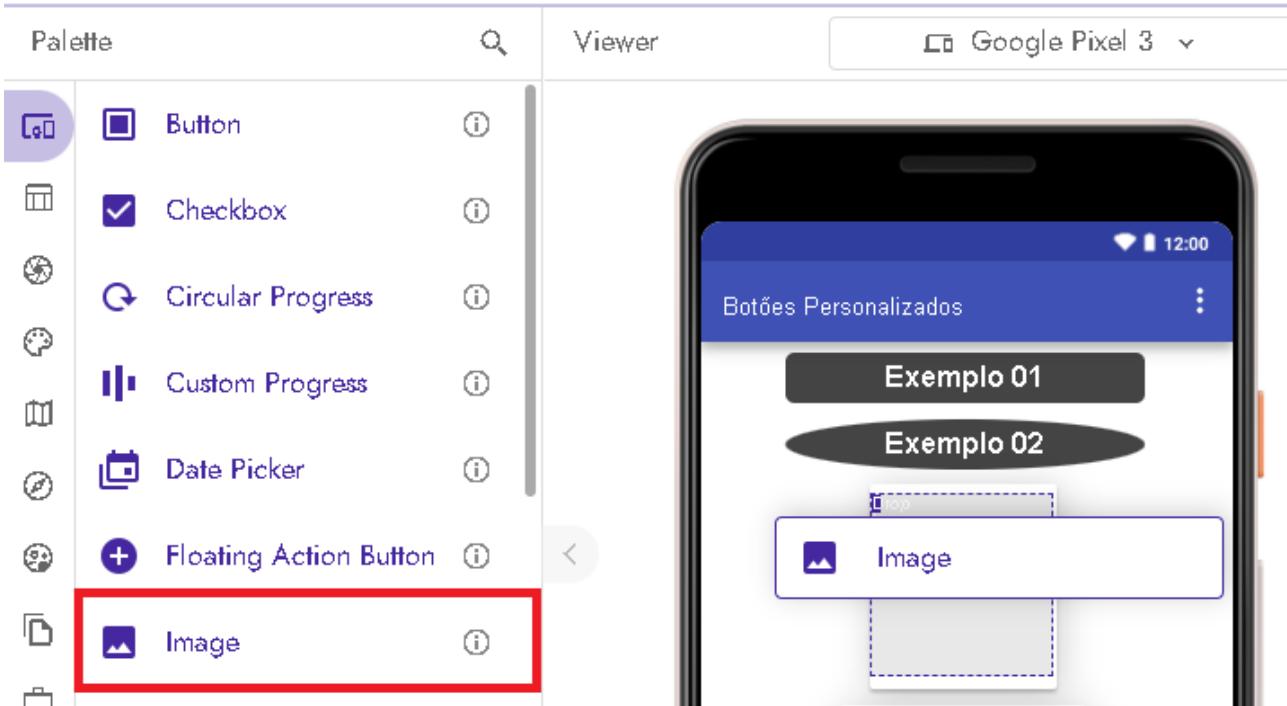


Figura 9 - Inserindo a Imagem no Horizontal Arragement.

- Altere as propriedades do componente **Image**.

Propriedade	Valor	Função
Clickable	✓ Marado	Definir que o componente deverá ser clicável.
Height	30px	Definir o alinhamento centralizado na horizontal.
Width	30px	Definir o alinhamento centralizado na vertical
Picture	Home.png	Faça o download de imagem aleatória da internet.
Name	img_btn01	Define o nome do componente.

- Clique na categoria **User Interface** e arraste o componente **Label** dentro do componente **Linha01** ao lado direito do componente **img_btn01**

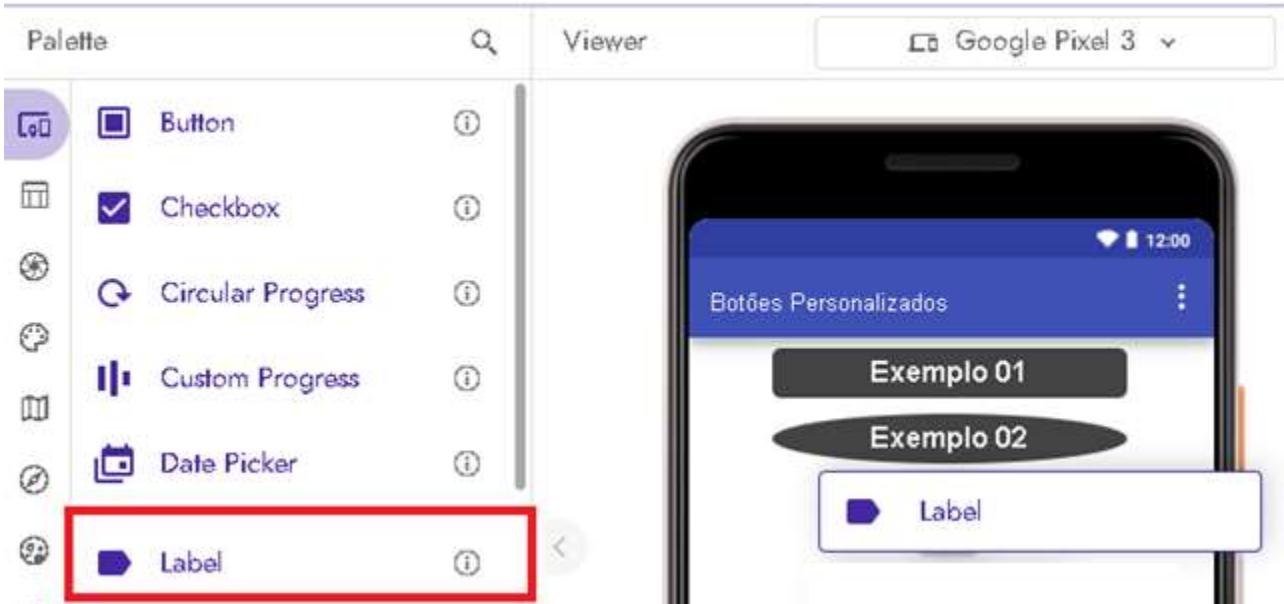


Figura 10 - Inserindo o componente Label no HorizontalArragement.

- Altere as propriedades do componente Label.

Propriedade	Valor	Função
Clickable	✓ Marado	Definir que o componente deverá ser clicável.
Font Bold	✓ Marcado	Definir a opção negrito
Font Size	18	Definir o tamanho da fonte
Text	Home Page	Definir o conteúdo a ser exibido do componente label.
Name	lbl_btn01	Definir o nome do componente.

Após a inserção de todos os componentes e alterados as respectivas propriedades, o layout do aplicativo deverá ser idêntico a **Figura 11**.

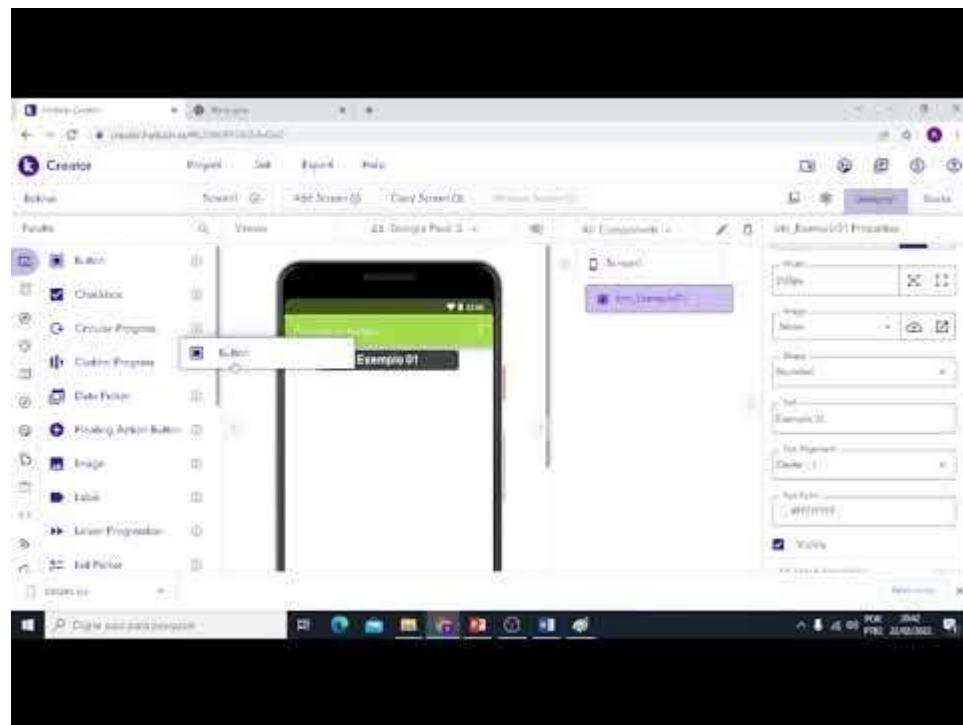


Figura 11 - Visualizando o Layout

Desta maneira o aluno consegui atingir o final do aplicativo, mas caso tenha dúvidas sobre a construção do layout para personalizar botões, poderá assistir ao vídeo.

Agenda 16- Personalizando botões, disponível em :

<https://youtu.be/XhX5xZoRJsA>





Vamos desenvolver a tela para o aplicativo da Karla. Para isso, criaremos um projeto no Kodular com o nome de “**JogoDescubraNumero**” . Seguindo o layout da **Figura 12**, o aluno deverá construir a aplicação e não se esqueça de renomear os componentes.

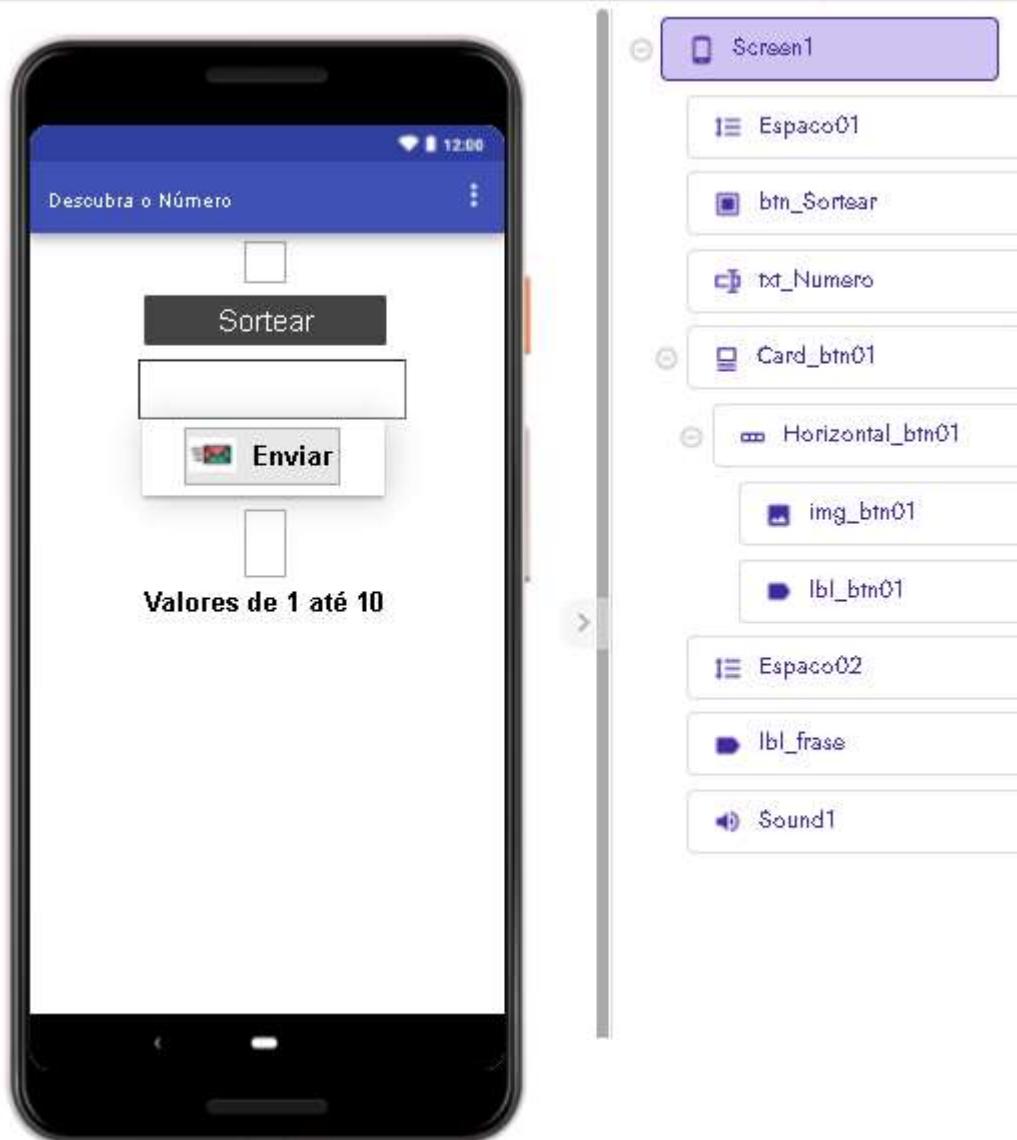


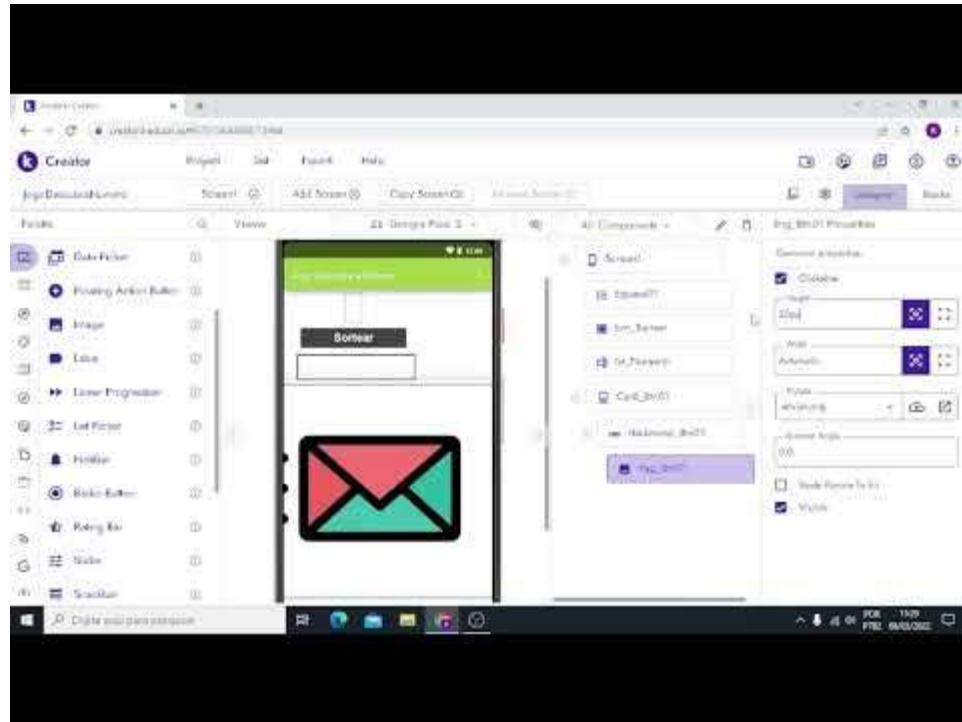
Figura 12 - Layout parcial do projeto

Chegamos ao fim do desenvolvimento da tela do aplicativo “**JogoDescubraNumero**” que Karla terá que construir para seu irmão. Guarde esse projeto! Ele será utilizado nas próximas atividades.

Caso o aluno tenha dificuldades na criação deste layout, acompanhe o video:

Agenda 16 – Criando a Interface da Atividade Online, disponível em:

<https://youtu.be/lIz3MWCa7B8>



Arquivos disponíveis para download

[Projeto Finalizados](#)

[Imagens utilizadas nos projetos](#)

[Som utilizados no projeto JogoDescubraNumero.aia](#)