



# Índice



# 1- Introdução



**1.1 Objetivo**

Este curso foi desenvolvido para o aprendizado de, pelo menos, o básico sobre criação e edição de um site utilizando o framework bootstrap com as linguagens HTML, PHP e auxílio do CSS.

Ao fim deste curso, o aluno poderá criar seus próprios layouts e implementar algumas funcionalidades através do PHP, dependendo da capacidade e criatividade do aluno.

**1.2 O que será desenvolvido**

O curso será baseado no desenvolvimento de um site que constantemente será atualizado ao github.

O site conterá o desenvolvimento, em PHP, das provas discursivas do vestibular de 2016 da UERJ, nas seguintes matérias: Matemática e Física. O desenvolvimento das questões ocorrerá de acordo com o gabarito da mesma.

**1.3 Materiais utilizados**

Falaremos, agora, os materiais utilizados para o desenvolvimento do nosso site e tentaremos resumir alguns conceitos, seguido do local onde tais materiais podem ser adquiridos.

**1.3.1 Sublime Text**

Editor utilizado para o desenvolvimento dos códigos HTML, PHP e CSS. Você poderá utilizar qualquer editor de sua preferência, como por exemplo, o Adobe DreamWeaver, o Notepad++ ou até mesmo o bloco de notas do Windows. Fica à sua preferencia.

Para adquirir o Sublime Text, basta fazer o download gratuito em: https://www.sublimetext.com/

**1.3.2 XAMPP**

O xampp cria um servidor web em nossa máquina fazendo com que possamos acessar nosso site através do seguinte endereço em nosso navegador: http://localhost/

Além disso, o xampp gera um servidor com o necessário para que possamos rodar nosso código PHP, o que não ocorreria caso salvemos nosso site na pasta “Meus Documentos”, por exemplo. Cria, também um servidor de banco de dados MySQL.

Para adquirir o XAMPP, basta fazer o download gratuito em: https://www.apachefriends.org/pt\_br/download.html

**1.3.3 ColorZilla**

Extensão do Google Chrome, também existente para o mozilla, que permite capturar a cor hexadecimal de algum site. Exemplo: Se você quer utilizar o tom vermelho do Youtube em seu site, basta ir à página que tem a cor e utilizar a extensão. Ela capturará o valor hexadecimal da cor e você poderá utilizar em seu arquivo CSS.

Para adquirir o ColorZilla, basta fazer o download gratuito da extensão na Chrome Store.

**1.3.4 Bootstrap 3**

Framework responsivo em CSS, iniciado por engenheiros do twitter. Atualmente está sendo desenvolvida sua quarta versão e seus códigos são abertos no github. Para saber mais: http://getbootstrap.com/

O download do bootstrap é gratuito e pode ser encontrado em: http://getbootstrap.com/getting-started/#download

**1.3.5 Flat UI Colors**

Paleta de cores flat para desenvolvimento do layout de seu site. Utilizaremos para montar a estrutura do nosso site com as cores de lá.

Para utilizá-la, basta acessar: http://flatuicolors.com/

**1.4 Antes de colocar as mãos na massa...**

No próximo capítulo ocorrerá o início do desenvolvimento do site. É importante que antes de inicia-lo você já tenha o seu editor, já tenha baixado o seu servidor, no nosso caso XAMPP, e que tenha feito o download do bootstrap 3. Esses são os pré-requisitos para que você possa iniciar o próximo capítulo.

# 2- Iniciando o projeto

**2.1 Preparando o nosso local**

Já com o XAMPP instalado, iremos criar o nosso localhost/Nome-Do-Nosso-Site, no nosso caso: http://localhost/Html-Bootstrap-PHP-CSS

Como faremos isso? Vá até o local de instalação do seu servidor. O padrão do xampp é C:\xampp. Dentro dele, entre na pasta chamada htdocs. Dentro dela iremos criar uma outra pasta com o nome que você desejar. Nesta pasta ficarão os arquivos do bootstrap, o seu CSS, as imagens do seu site e todo o código do seu site. Guarde bem o nome que você utilizar, pois seu endereço para testar o desenvolvimento do site será http://localhost/nome\_da\_sua\_pasta.

Agora, descompacte o arquivo do Bootstrap que você baixou para a pasta que você acabou de criar. O arquivo do bootstrap vem com as seguintes pastas: css, fonts, js.

Feito isto, já podemos iniciar nosso site. Abra seu editor e vamos navegar ao próximo tópico

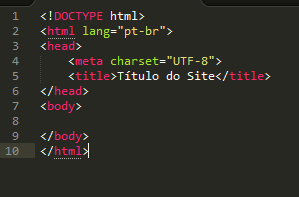
**2.2 Iniciando o site**

Este tópico será dividido em três. Os dois primeiros completamente teóricos, pois precisamos saber como é a estrutura básica de um documento HTML e de um site. Em seguida, a última seção trás a teoria que foi passada sendo aplicada na prática.

**2.2.1 Estrutura básica de um documento HTML**

Estamos na era do HTML5. Esta versão trouxe diversas melhorias ao HTML que existia em sua versão antecessora. Mas afinal, o que é HTML? O HTML é um código desenvolvido para a criação e edição de websites. É a linguagem que diz para o seu navegador o que deve ocorrer. É como se o site fosse uma folha com informações do que deve ocorrer. Essas informações são os códigos HTML, chamados de tags, que são lidas e interpretadas pelo seu navegador.

O HTML5, para seu correto funcionamento, necessita de uma estrutura padrão, que veremos a seguir:



Onde o que está rosado são as tags HTML. Sempre vêm entre os sinais de **<** e **>**.

Algumas tags podem ser usadas sozinhas enquanto outras necessitam que sejam abertas e fechadas. Um exemplo de tag que pode vir sozinha é a tag <**meta**>. Um exemplo de tag que precisa ser aberta e fechada é a tag <**title**> que foi aberta desta forma, guardou uma informação, que no caso é o tírulo do site e foi fechada com </**title**>. Todas as tags que devem ser fechadas são fechadas com **/** (barra).

**2.2.2 Estrutura básica de um site**

Um site, normalmente, é dividido em 3 partes: cabeçalho, onde normalmente fica uma logo e o menu; o corpo do site, que contém as informações do site propriamente dito e o rodapé, normalmente com informações adicionais como endereço e/ou contato.

**2.2.3 Iniciando o site**

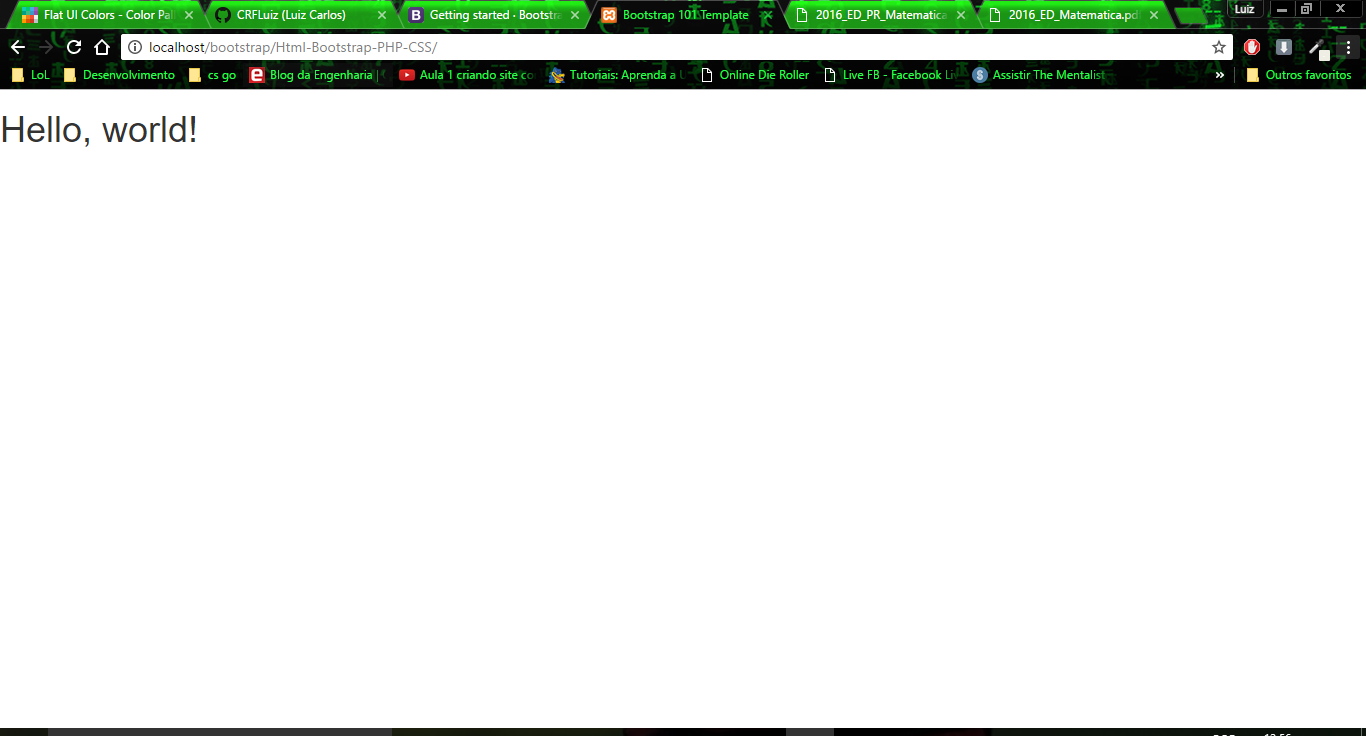
Como já foi falado, você já conhece a estrutura básica de um documento HTML. Porém, como iremos utilizar o bootstrap, teremos algumas alterações. A estrutura do HTML já editado para o bootstrap pode ser encontrado em: http://getbootstrap.com/getting-started/#template

Basta copiar todo aquele código e colar no seu editor. Você deve alterar o parâmetro lang da tag html para “pt-br”. Como mostra na figura a seguir:



**2.3 Analisando o site**

Assim que terminarmos de copiar e colar a estrutura básica do bootstrap, basta salvarmos como index.html dentro da pasta que você criou. Após isso, só abrir o xampp control panel e ativar o servidor **apache**. Pronto, nosso servidor localhost já está online e podemos executar qualquer código php.

Agora, basta abrirmos nosso navegador e abrir o link http://localhost/nome\_da\_pasta. Veremos o seguinte resultado: 

 Podemos observar que o título do site está como *Bootstrap 101 Template*. Com isso podemos perceber que o título do site fica dentro da tag **title**, que por sua vez deve estar dentro da tag **head**.

Exemplo:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Título do site</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<!--[if lt IE 9]>

<script src="https://oss.maxcdn.com/html5shiv/3.7.3/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<![endif]-->

</head>

<body>

<h1>Hello, world!</h1>

<script src="https://ajax.googleapis.com/ajax/libs/jquery/1.12.4/jquery.min.js"></script>

<!-- Include all compiled plugins (below), or include individual files as needed -->

<script src="js/bootstrap.min.js"></script>

</body>

</html>

Você deve ter percebido que eu tirei uma parte do código. Se vocês notarem, também, os códigos que foram removidos estavam entre **<!--** e **-->**. Perceba também que ele muda de cor, fica acinzentado. Isso acontece porque o que foi apagado estava sendo considerado comentário. Um exemplo é: *<!-- Este é um comentário -->*

Apesar de não ser obrigatório é uma parte importante do site, pois os comentários não são interpretados como código. Servem apenas para que os programadores saibam o que está acontecendo ali, onde foi comentado.

Outra coisa que podemos ver no site é uma grande mensagem escrito “Hello, world!”. Temos esta pequena frase no nosso código dentro de uma tag chamada **h1**, por sua vez dentro da tag **body**.

Vamos lá... Entendendo o que acontece, o **body** é o corpo do nosso site. Tudo o que queremos exibir no navegador, deverá estar dentro do body. A segunda tag é a **h1**. Existem 6 tags do tipo h e elas servem para dar estilos ao texto. A **h1** deixa o texto grande; a tag **h2** deixa o texto ainda grande, porém menor que a **h1** e assim sucessivamente até o tamanho mínimo (e bem pequeno) **h6**.



**2.4 Trabalhando o layout do site**

Para começarmos a trabalhar no layout do site, iremos aprender o que é o CSS. Para isso crie um novo arquivo chamado estilo.css, também dentro de sua pasta criada. A partir de agora, não indicaremos que os arquivos salvos ficarão na pasta do projeto. Cabe a você lembrar que todos os arquivos necessários para o site ficarão nesta pasta.

Dentro deste arquivo CSS colocaremos o seguinte código:

body

{

background-color: e74c3c;

color: white;

}

Em seguida, volte ao arquivo HTML e insira a seguinte tag, após a chamada do css do Bootstrap. Ficará assim:

<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="utf-8">

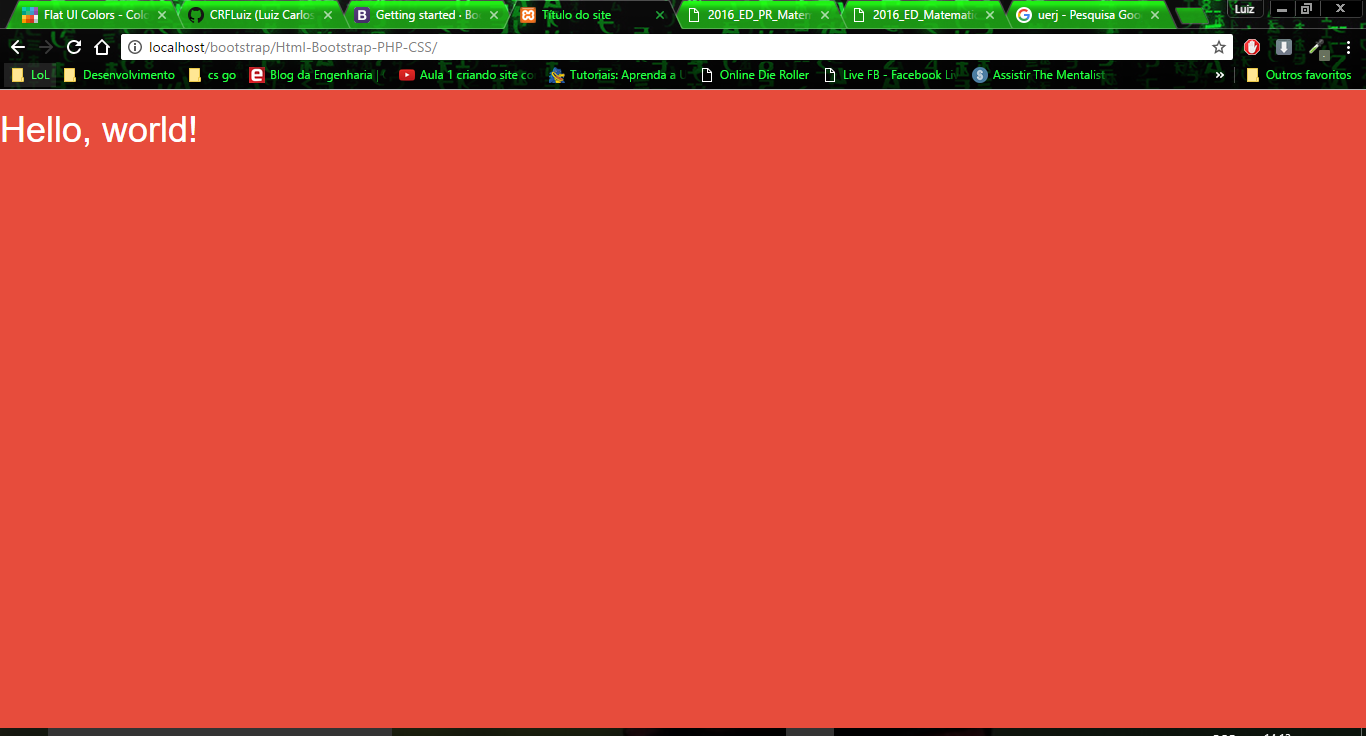
<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title>Título do site</title>

<link href="css/bootstrap.min.css" rel="stylesheet">

<link href="estilo.css " rel="stylesheet">

 Agora atualize seu site no navegador e veja o seguinte resultado:  Entendendo o que aconteceu: temos duas tags **link**. Nela temos os parâmetros **href** e **rel**. A tag link serve para chamarmos um arquivo externo que iremos usar no nosso site.

A primeira tag link está da seguinte forma: *<link href="css/bootstrap.min.css" rel="stylesheet">*. Isso significa que estamos utilizando um arquivo chamado bootstrap.min.css, que está dentro da pasta css, que por sua vez se encontra na raiz do diretório do seu projeto (a pasta criada). Em seguida, vemos um *rel=”stylesheet”*. Este parâmetro está indicando que o arquivo se trata de uma folha de estilo, em outras palavras, é um css.

Na segunda tag temos *<link href="estilo.css" rel="stylesheet">*. De acordo com o que aprendemos no exemplo anterior, estamos fazendo nosso site acessar uma folha de estilo chamada estilo.css. Como o arquivo se encontra dentro da pasta raiz, botamos apenas o nome do arquivo, sem a necessidade de mostrar o caminho, como foi feito anteriormente (o arquivo estava dentro da pasta css).

É importante que os arquivos css sejam chamados dentro da tag **head**. Outro ponto importante é que o nosso CSS seja a ultima tag **link** a ser chamada, pois o estilo do ultimo arquivo chamado sempre irá sobrepor um estilo que foi dado anteriormente a um mesmo elemento.

Enfim, é para isso que existe o CSS. Para darmos o estilo ao nosso site através de um arquivo externo. Arquivo, este, que poderemos modificar sempre que quisermos alterar o layout do site. Tornando tudo mais organizado e simples na hora de alterar.



**2.5 Entendendo melhor o CSS**

Para entendermos melhor como funciona o css, iremos criar o seguinte código no lugar do *Hello World*:

<body>

<div id=”azul”>

<h1>Hello, world!</h1>

</div>

<div class="texto-negrito texto-centralizado">

<h2>Hello, world!</h2>

</div>

<div id="texto-negrito ">

<h3>Hello, world!</h3>

</div>

</body>

O que foi feito? Foram criadas 3 **div**s. A primeira, além da tag **h1** com texto, recebeu uma **id** *azul*. A segunda, além da tag **h2** com texto, recebeu duas **classes:** *texto-negrito* e *texto-centralizado*. A terceira, além da tag h3 com texto, recebeu uma **classe** *texto-negrito*.

Mas para que servem a **id** e a **classe**? Podemos dizer que a **id** é o nome daquele elemento. Quando eu quiser me referir àquela primeira div, basta eu dizer que é a **id** que se chama *azul*. Dentro de uma página, não podem haver ids iguais. Por sua vez, uma mesma classe pode ser atribuída a vários elementos.

Exemplo: Eu tenho um site que contém uma div com o parâmetro *id=“azul”*, uma div com o parâmetro *class=”texto-negrito texto-centralizado”* e uma outra div com o parâmetro *class=”texto-negrito”* (O mesmo da imagem anterior). Criei um documento CSS com os seguintes códigos:



#azul

{

background-color: blue;

}

.texto-centralizado

{

text-align: center;

}

.texto-italico-serifado

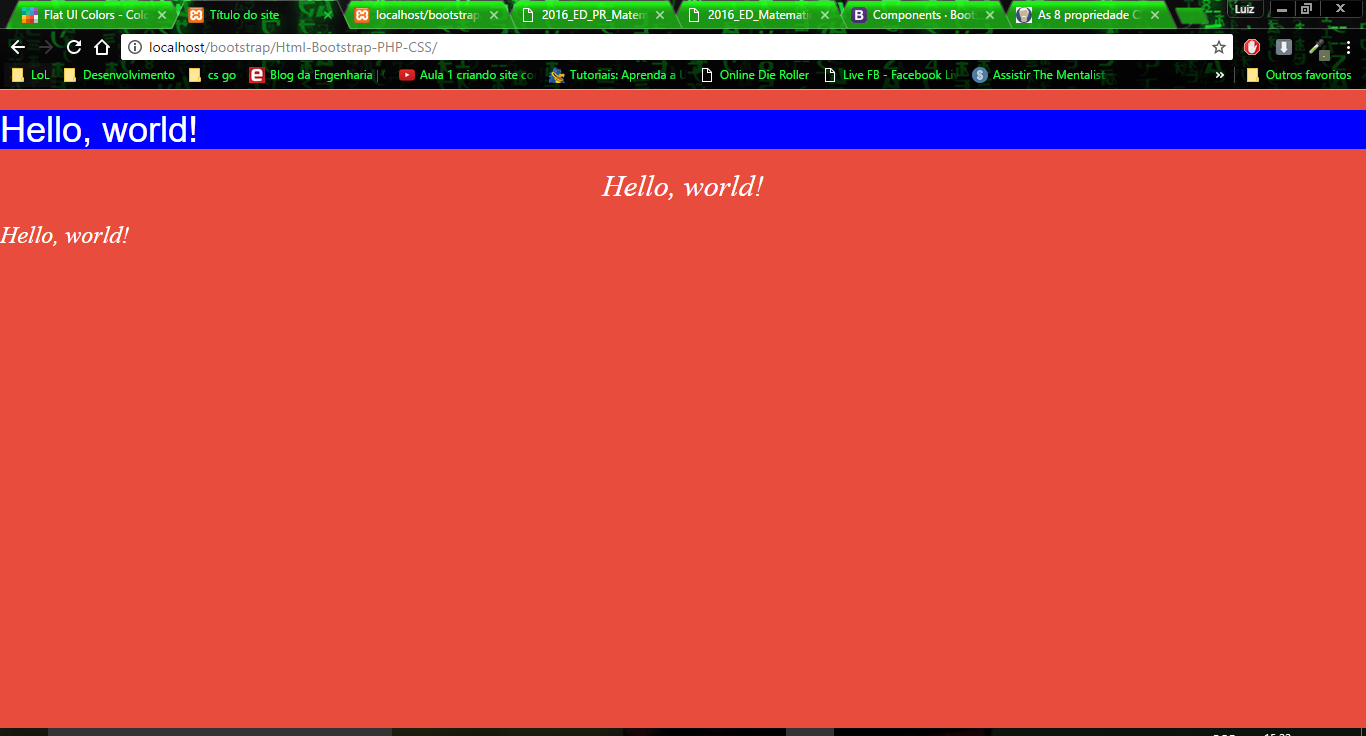
{

font-style: italic;

font-family: serif;

}

Após salvarmos tudo e entrar no nosso servidor para ver o site rodando, teremos algo assim:



Então, acabamos de ver que para estilizar um elemento de acordo com a id dele, precisamos da seguinte estrutura no CSS: *#nome\_da\_id{comandos;}*. Caso queiramos estilizar algo de acordo com a classe: *.nome\_da\_classe{comandos;}*. Se quisermos estilizar de acordo com a tag: *nome\_da\_tag{comandos;}*.

Todos os códigos devem ser seguidos de ponto e virgula em seu final. Neste exemplo, utilizamos 4 comandos diferentes do CSS. Vamos a eles:

 **2.5.1 background-color**

Como seu próprio nome já indica, este comando altera a cor do background do elemento que você selecionou, para uma cor que você predefinir.

Estrutura: *background-color: valor;*

Onde valor é o valor da cor e pode ser atribuído seu código hexadecimal, rgb, rgba ou simplesmente o nome da cor(em inglês).

Exemplos para a cor preta: *background-color: #000000;*

*background-color: rgb(0, 0, 0);*

*background-color: rgba(0, 0, 0, 1);*

*background-color: black;*

**2.5.2 text-align**

Esta propriedade diz qual deve ser o alinhamento do texto daquele elemento.

Estrutura: *text-align: valor;*

Onde valor é o alinhamento que você desejar.

Exemplos de alinhamento: *text-align: left;*

*text-align: center;*

*text-align: right;*

**2.5.3 font-style**

Esta propriedade diz qual deve ser o estilo de fonte do texto daquele elemento.

Estrutura: *font-style: valor;*

Onde valor é o alinhamento que você desejar.

Exemplos de estilo de fonte: *font-style: italic;*

*font-style: normal;*

*font-style: oblique;*

**2.5.4 font-family**

Esta propriedade especifica qual a família de fonte, o site deve seguir.

Estrutura: *font-family: valor;*

Onde valor é a família que desejar.

Exemplos de uso de família de fontes *font-family: Serif;*

*font-family: Sans-serif;*

*font-family: Monospace;*

 **2.6 Criando o cabeçalho**

A partir de agora iremos limpar toda a nossa tag **body**, mantendo apenas os scripts de javascript que estão nas últimas linhas do body. Iremos passar este script para dentro da tag **head**, deixando nosso body totalmente limpo. Ficaremos assim com o documento HTML desta maneira:



Agora iremos na documentação do bootstrap. Lá teremos o menu componentes, onde iremos clicar. Assim que a nova página abrir, teremos um menu lateral no canto direito. Procure por *Navbar* e clique (link direto: http://getbootstrap.com/components/#navbar). Desça um pouco até que apareça o primeiro bloco de código. Copie-o e jogue dentro do seu body, na primeira linha possível.

Este código é que irá criar o seu menu já responsivo. Assim que colar o código, salve seu arquivo e abra-o no navegador. Veja o menu já pronto em seu site. Este é o menu padrão do bootstrap, agora iremos por a mão na massa para alterá-lo da forma que queremos.

De acordo com o objetivo deste tutorial, não queremos tantas opções no menu e nem a barra de pesquisa. Vamos deixar, apenas, duas opções dropdown no canto direito, uma logotipo e mudar a cor do cabeçalho para azul.

**2.6.1 Dando a cor à barra de menu**

A

**2.6.1 Adicionando um logotipo**

A

**2.6.3 Retirando e alterando botões**

A

**2.6 Criando o corpo do site**

A partir

**2.8 Criando o rodapé**

A partir