



GUIA COMPLETO

COMO SE TORNAR UM
DESENVOLVEDOR

FULL-STACK



Introdução

Dominar algo nem sempre será uma tarefa simples, pelo contrário, irá exigir de você atitudes corajosas e ao mesmo tempo complexas. E se tratando de programação, irá exigir muito, mais muito tempo de estudos e prática, e para aqueles que tem o desejo de se tornar um Desenvolvedor Full-Stack certamente precisará de um guia completo como este, capaz de lhe conduzir por um caminho com obstáculos, dificuldades mas um caminho certo, que lhe levará ao nível de maestria como um desenvolvedor Full-Stack. Iremos lhe mostrar o real caminho para você dominar as tecnologias necessárias e assim poder construir aplicações completas com as melhores e mais atuais tecnologias do mercado.

Ser um desenvolvedor Full-Stack exigirá de você aprender muitas habilidades. E para iniciantes, muitas vezes não é fácil encontrar o caminho certo de aprendizado para já começar a ter os primeiros resultados. No começo pode ser que você se assuste com algumas tecnologias que terá que dominar, no entanto se você tem alguém que lhe conduza pela caminhada a historia muda bastante.

E essa é a proposta deste e-book, assim como descrito no titulo, iremos lhe mostrar o caminho para você dominar cada tecnologia do universo Full-Stack, desde as bases até módulos mais avançados, que serão suficientes para você começar hoje mesmo a estruturar seu ambiente de trabalho e começar escrever a linguagem das maquinas.



O Full-Stack é a definição perfeita para “desenvolvedor completo”, pois são as duas pernas que você precisa para caminhar. Uma perna seria o desenvolvedor Front-End e a outra perna seria o desenvolvedor Back-End.

A vantagem de ser um desenvolvedor completo é justamente o fato de você não precisar de outras partes para desenvolver suas aplicações.

O desenvolvedor Front-End geralmente cuida da parte da aplicação que o usuário vê estampada na tela, e o Back-End é a parte da aplicação que trabalha com a lógica, interação de banco de dados, autenticação de usuário, configuração do servidor etc.

Você se tornando um desenvolvedor Full Stack, não significa que você terá que dominar tudo que é necessário do Front-End e do Back-End, mas isso significa que você poderá trabalhar em ambos os lados e compreender tudo que está acontecendo de forma holística.

E se de alguma forma você foi despertado para se tornar um Desenvolvedor Full-Stack nesse e-book é realmente o caminho completo para você começar hoje mesmo a dominar esse universo fantástico.

Tudo pronto? Apertem os cintos developers, sua jornada pelo caminho Full-Stack começa AGORA!



Índice

Introdução	2
------------	---

PARTE 1: O Desenvolvedor Front-End

Capítulo 1 – Lançando as Bases	6
--------------------------------	---

Capítulo 2 – HTML, CSS e JavaScript / JQuery	8
--	---

Capítulo 3 – Pré-processadores SASS e LESS	28
--	----

Capítulo 4 – Responsive Web Design	32
------------------------------------	----

Capítulo 5 – AngularJS	34
------------------------	----

Capítulo 6 – Vue.js	38
---------------------	----

Capítulo 7 – React	42
--------------------	----

PARTE 2: O Desenvolvedor Back-End

Capítulo 8 – PHP	45
------------------	----

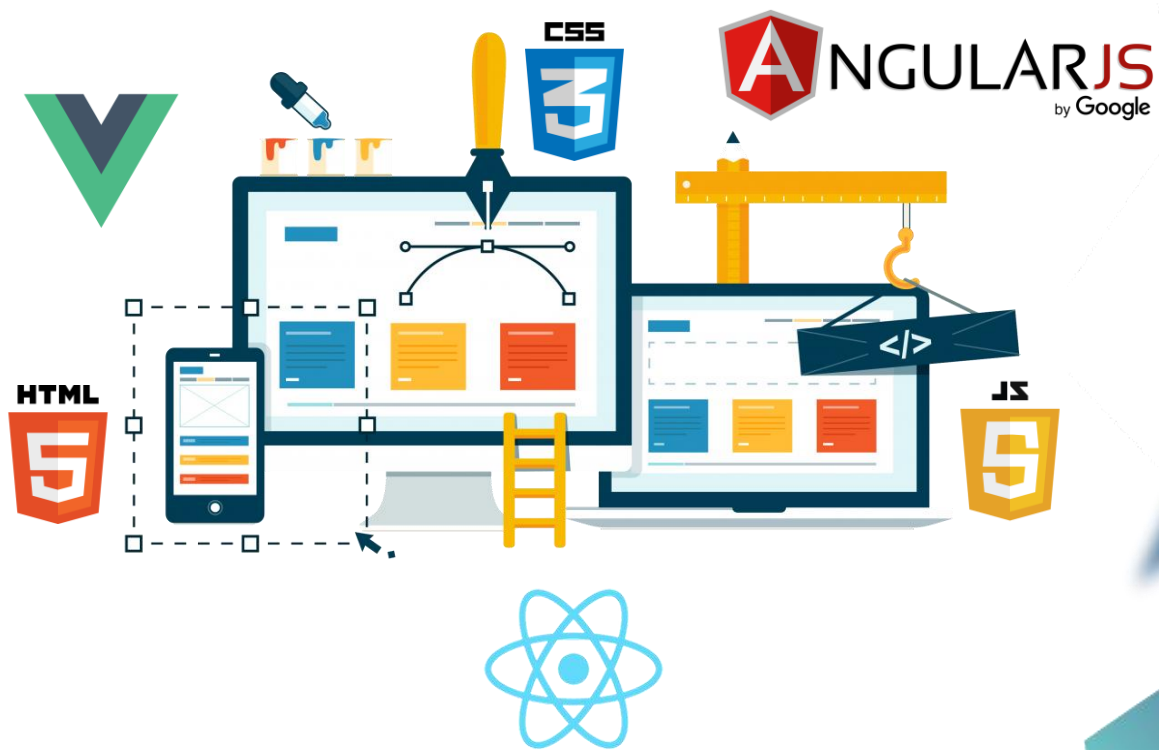
Capítulo 9 – MySQL	50
--------------------	----

Capítulo 10 – Node.js	52
-----------------------	----

Conclusão	56
-----------	----

PARTE 1

DESENVOLVEDOR FRONT-END



Capítulo 1 – Lançando as bases

Vou passar um conceito rápido e bem antigo mas que fará toda a diferença na compreensão do que vem a seguir sobre Front-End, que é o conceito de como nós organizamos um código.

Existem 3 camadas no desenvolvimento web: Informação, formatação e comportamento.

A camada de informação: é normalmente controlada pelo HTML. Ela exibe e dá significado à informação que o usuário consome. É nela que você marca o que é um título, um parágrafo etc.. Dando significado e estruturando cada pedaço de informação publicada.

A camada de formatação: é normalmente controlada pelo CSS. É nesta camada que você controla como a informação será formatada, transformando o layout feito pelo designer em código.

O CSS é o responsável por dar forma à informação marcada com HTML, para que ela seja bem-vista em qualquer dispositivo.

A camada de comportamento: Era totalmente controlada pelo JavaScript, no entanto o CSS agora está tendo algumas responsabilidades nesse terreno. Nesta camada é onde controlamos qual será o comportamento da informação.



Quando uma modal se abre ou um slider de imagens funciona, é o JavaScript que está entrando em ação. São explicações bem básicas de cada camada, vamos entender agora cada uma delas.

Capítulo 2 – HTML, CSS e JavaScript / JQuery

HTML5

Tudo que você for criar na web começa e termina com HTML, e com certeza para você ser um desenvolvedor Full-Stack completo você precisa domina-lo bem.

Impossível você desenvolver qualquer tipo de aplicação web sem dominar o funcionamento do HTML, pois ele estará presente desde o início do projeto e será o responsável por muitas partes cruciais durante toda a sua elaboração.

Se tratando de projetos maiores e que demandem uma equipe de desenvolvedores, você precisa redobrar sua atenção quanto a semântica do seu HTML. Se a semântica não estiver bem-feita, o site terá dificuldade de aparecer nos mecanismos de buscas, irá ter problemas com a acessibilidade, e com o tempo até a manutenção será prejudicada.



Estrutura padrão do documento HTML.

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="utf-8"/>
    <title>Entendo a estrutura e semântica do HTML5</title>
  </head>
  <body>
    <p>Olá mundo!</p>
  </body>
</html>
```

Doctype e o elemento HTML.

O Document Type Definition (DTD, ou Doctype) é uma instrução que diz ao navegador qual é a especificação do código que está sendo utilizada no documento, e deve ser declarado antes da tag <html>.

Já na versão antiga do HTML, a declaração do Doctype era mais extensa e difícil de decorar, havendo a necessidade de referenciar para o navegador o arquivo DTD com as definições daquela especificação.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```



No HTML5 a inserção do Doctype foi simplificada, e a responsabilidade de buscar as definições da especificação fica por conta do próprio navegador:

```
<!DOCTYPE html>
```

Após a declaração do Doctype, iniciamos o código HTML. Na árvore de elementos do código, a tag principal é a `<html>`, que comporta todos os outros elementos filhos.

```
<html lang="pt-br">
```

É na tag `<html>` que nós declaramos o idioma principal do documento. Na tag `<html>` também podemos inserir atributos como o `xmlns` (XML Namespace), como quando usamos em nossa página elementos da FBML (Facebook Markup Language):

```
<html lang="pt-br" xmlns:fb="http://www.facebook.com/2008/fbml">
```

Metadados

Os Metadados são um conjunto de informações sobre a página e do conteúdo nele publicado. Essas informações são usadas pelos navegadores e user-agents em geral, sendo invisíveis para os usuários. Todos os Metadados ficam contidos na tag `<head>`:



Exemplo:

```
<head>
  <meta charset="utf-8">
  <title>Entendo a estrutura e semântica do HTML5</title>
  <meta name="keywords" content="HTML5, CSS3, frontend, agni">
  <meta name="description" content="Se você quer se aventurar com o HTML5,
entenda aqui a sua estrutura básica, a semântica das principais marcações novas
e algumas ferramentas.">
  <meta name="author" content="Edu Agni">
  <meta name="robots" content="index,follow">
  <link rel="alternate" type="application/rss+xml" title="Feed de notícias"
href="/feed">
  <link rel="stylesheet" type="text/css" href="/sidecode/style.css">
  <script src="/sidecode/scripts.js"></script>
</head>
```

Informações como Title e Description por exemplo, são usadas pelos mecanismos de busca para tornar o seu site localizável nos mecanismos de buscas.



CSS3

CSS (Cascading Style Sheets) permite que você crie excelentes páginas Web, mas como funciona na prática? Esta parte do e-book explica o que é CSS, como o navegador transforma HTML em um Document Object Model (DOM), como o CSS é aplicado a partes do DOM, alguns exemplos de sintaxe muito básicos e qual código é usado para incluir nosso CSS em nossa página Web.

O que é CSS?

CSS é uma linguagem para especificar como os documentos são apresentados aos usuários - como eles são estilizados e disponibilizados.

Um documento geralmente é um arquivo de texto estruturado usando uma linguagem de marcação - HTML é a linguagem de marcação mais comum, mas você também encontrará outras linguagens de marcação, como SVG ou XML .

Como o CSS afeta o HTML?

Os navegadores Web aplicam regras CSS a um documento para afetar o modo como são exibidos. Uma regra CSS é formada por:



- Um conjunto de propriedades, que possuem valores definidos para atualizar como o conteúdo HTML é exibido, por exemplo, eu quero que a largura do meu elemento seja 50% de seu elemento pai e que seu plano de fundo seja vermelho.
- Um seletor, que seleciona o(s) elemento(s) para o qual deseja aplicar os valores de propriedade atualizados. Por exemplo, quero aplicar minha regra de CSS a todos os parágrafos do meu documento HTML.

Um conjunto de regras CSS contidas em uma folha de estilos determina como uma página Web deve ser exibida.

Exemplo:

As descrições acima talvez não tenham feito sentido, então vamos garantir que as coisas fiquem claras apresentando um exemplo rápido. Primeiro de tudo, vamos pegar um documento HTML simples, contendo uma `<h1>` e um `<p>` (observe que uma folha de estilo é aplicada ao HTML usando um `<link>`):

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8">
    <title>My CSS experiment</title>
    <link rel="stylesheet" href="style.css">
  </head>
```



```
<body>
  <h1>Hello World!</h1>
  <p>This is my first CSS example</p>
</body>
</html>
```

Agora vamos dar uma olhada em um exemplo CSS muito simples contendo duas regras:

```
h1 {
  color: blue;
  background-color: yellow;
  border: 1px solid black;
}

p {
  color: red;
}
```

A primeira regra começa com um seletor `h1`, o que significa que ela aplicará seus valores de propriedade ao elemento `<h1>`. Ele contém três propriedades e seus valores (cada propriedade / valor é chamado de declaração):



O primeiro define a cor do texto para azul.

O segundo define a cor de fundo para amarelo.

O terceiro coloca uma borda ao redor do cabeçalho com 1 pixel de largura, sólido (não pontilhado ou tracejado etc.) e a cor preta.

A segunda regra começa com um seletor p, o que significa que ela aplicará seus valores de propriedade ao elemento <p>. Ele contém uma declaração, que define a cor do texto para vermelho.

Em um navegador Web, o código acima produziria o seguinte:



JavaScript

Como o nosso foco aqui nesse e-book não é apenas o Front-End, vou já aqui adiantar algumas coisas que abrangem a vida do desenvolvedor Full-Stack, pois o JavaScript diferente do HTML e CSS pode ser utilizado tanto para o desenvolvimento Front-End como para o Back-End também.

Segundo, Stoyan Stefanov, em seu livro Pattern JavaScript ele diz:

“Com o JavaScript você pode programar em uma variedade cada vez maior de plataformas. Você pode escrever código no lado do servidor (Node.js), aplicações desktop (que funcionam em todos os sistemas operacionais) e extensões de aplicação como (Firefox ou Photoshop), aplicações para dispositivos móveis e scripts de linha de comando. O JavaScript é uma linguagem incomum. Ela não possui classes, e funções são usadas como objetos de primeira classe em várias tarefas. Curiosamente linguagens como Java e PHP começaram a adicionar funcionalidades como closures e funções anônimas, que os desenvolvedores JavaScript vêm utilizando corriqueiramente há algum tempo.”

Pois bem e para sermos um Full-Stack JavaScript Developer? Bom, o JavaScript que a maioria conhece serve apenas para manipular DOM(Document Object Model) ou validar campos de formulários, certo? Errado! Hoje podemos fazer de tudo com JavaScript, inclusive roda-lo no servidor como uma linguagem Server-Side, ou seja, agora temos o JavaScript do lado do Servidor (NodeJs) e do lado do Cliente da qual podemos adotar alguns frameworks ou bibliotecas, como Angular, Ember, Backbone, React, Meteor ou até mesmo o jQuery.



Ferramentas essas que fazem o “ciclo de vida” dos produtos serem extremamente produtivos em relação à escalabilidade e agilidade no desenvolvimento Client-Side, pensando em manutenibilidade e outros fatores. Outro ponto que acho extremamente relevante abordar é que uma das comunidades mais ativas é a do Node.JS, +70000 módulos no NPM (Node Package Manager).

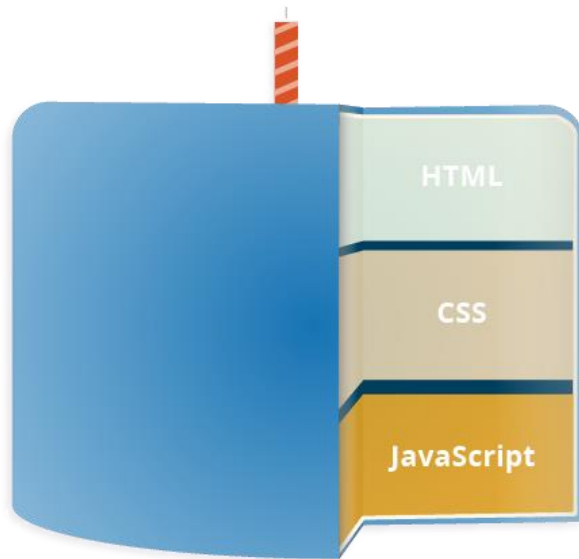
Aqui estamos apenas fazendo uma breve abordagem, pois iremos ver com mais detalhes sobre o nodeJS e outras bibliotecas JavaScript extremamente importantes para nossa carreira Full-Stack.

Mas levando em conta que você talvez esteja iniciando precisamos fazer uma abordagem um pouco mais simples do JavaScript.

Definição

JavaScript é uma linguagem de programação que permite a você implementar itens complexos em páginas web — toda vez que uma página da web faz mais do que simplesmente mostrar a você uma informação estática — mostrando conteúdo que se atualiza em um intervalo de tempo, mapas interativos ou gráficos 2D/3D animados, etc. — você pode apostar que o JavaScript provavelmente está envolvido. É a terceira camada do bolo das tecnologias padrões da web, duas das quais (HTML e CSS) nós falamos com muito mais detalhes em outras partes da Área de Aprendizado.





- HTML é a linguagem de marcação que nós usamos para estruturar e dar significado para o nosso conteúdo web, por exemplo, definindo parágrafos, cabeçalhos, tabelas de conteúdo, ou inserindo imagens e vídeos na página.
- CSS é uma linguagem de regras de estilo que nós usamos para aplicar estilo ao nosso conteúdo HTML, por exemplo, definindo cores de fundo e fontes, e posicionando nosso conteúdo em múltiplas colunas.
- JavaScript é uma linguagem de programação que permite a você criar conteúdo que se atualiza dinamicamente, controlar multimídias, imagens animadas, e tudo o que há de mais interessante. É maravilhoso o que você pode efetuar com algumas linhas de código JavaScript.



As três camadas ficam muito bem uma em cima da outra. Vamos exemplificar com um simples bloco de texto. Nós podemos marcá-lo usando HTML para dar estrutura e propósito:

```
<p>Jogador 1: Chris</p>
```

Nós podemos adicionar um pouco de CSS na mistura, para deixar nosso parágrafo um pouco mais atraente:

```
p {  
  font-family: 'helvetica neue', helvetica, sans-serif;  
  letter-spacing: 1px;  
  text-transform: uppercase;  
  text-align: center;  
  border: 2px solid rgba(0,0,200,0.6);  
  background: rgba(0,0,200,0.3);  
  color: rgba(0,0,200,0.6);  
  box-shadow: 1px 1px 2px rgba(0,0,200,0.4);  
  border-radius: 10px;  
  padding: 3px 10px;  
  display: inline-block;  
  cursor: pointer;  
}
```

JOGADOR 1: CHRIS

[CLIQUE AQUI PARA CONHECER O PACOTE FULL-STACK!](#)



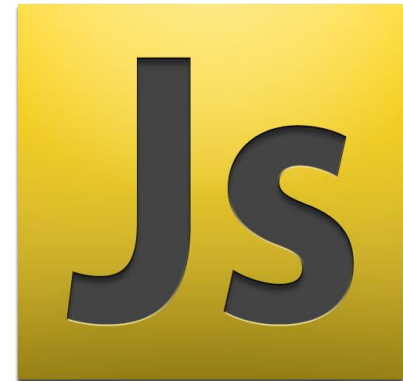
E finalmente, nós podemos adicionar JavaScript para implementar um comportamento dinâmico:

```
var para = document.querySelector('p');  
  
para.addEventListener('click', atualizarNome);  
  
function atualizarNome() {  
  var nome = prompt('Insira um novo nome');  
  para.textContent = 'Jogador 1: ' + nome;  
}
```

Experimente clicar no botão que nós criamos para ver o que acontece.

Sim, mas como adicionamos JavaScript a nossa página?

O JavaScript é inserido na sua página de uma maneira similar ao CSS. Enquanto o CSS usa o elemento `<link>` para aplicar folhas de estilo externas e o elemento `<style>` para aplicar folhas de estilo internas, o JavaScript só precisa de um amigo no mundo do HTML — o elemento `<script>`. Vamos aprender como funciona.



JavaScript interno.

Antes de tudo, faça uma cópia local do nosso arquivo de exemplo aplicando-javascript.html. Salve-o em alguma pasta, de uma forma sensata.

Abra o arquivo no seu navegador web e no seu editor de texto. Você verá que o HTML cria uma simples página web contendo um botão clicável.

Agora, vá até o seu editor de texto e adicione o código a seguir antes da tag de fechamento `</body>`:

```
<script>
```

```
// O JavaScript fica aqui
```

```
</script>
```

Agora nós vamos adicionar um pouco de JavaScript dentro do nosso elemento `<script>` para que a página faça algo mais interessante — adicione o seguinte código abaixo da linha `// O JavaScript fica aqui`:

```
function criarParagrafo() {  
  var para = document.createElement('p');  
  para.textContent = 'Você clicou no botão!';  
  document.body.appendChild(para);  
}
```



```
var botoes = document.querySelectorAll('button');

for(var i = 0; i < botoes.length ; i++) {
  botoes[i].addEventListener('click', criarParagrafo);
}
```

Salve seu arquivo e recarregue a página — agora você deveria ver que quando você clique no botão, um novo parágrafo é gerado e colocado logo abaixo.

JavaScript externo.

Isso funciona muito bem, mas e se nós quiséssemos colocar nosso JavaScript em um arquivo externo? Vamos explorar isso agora.

Primeiro, crie um novo arquivo na mesma pasta que está o arquivo HTML de exemplo. Chame-o de script.js — tenha certeza de que o nome do arquivo tem a extensão .js, pois é assim que ele será reconhecido como JavaScript.

Agora, copie todo o código que está dentro do elemento <script> e cole dentro do arquivo .js. Salve o arquivo.

Agora substitua o atual elemento <script> pelo seguinte código:

```
<script src="script.js"></script>
```

Salve e atualize seu navegador, e você deverá ver a mesma coisa! Funciona igualmente, mas agora nós temos o JavaScript em um arquivo externo. Isso é geralmente uma coisa boa em termos de organização de código, e faz com que seja possível reutilizar o código em múltiplos arquivos HTML. Além disso, o HTML fica mais legível sem grandes pedaços de script no meio dele.



Manipuladores de JavaScript inline.

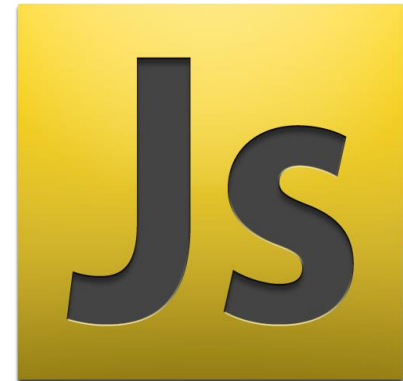
Note que às vezes você vai encontrar código JavaScript escrito dentro do HTML. Isso deve ser algo como:

```
function criarParagrafo() {  
  var para = document.createElement('p');  
  para.textContent = 'Você clicou no botão!';  
  document.body.appendChild(para);  
}  
<button onclick="criarParagrafo()">Click me!</button>
```

Contudo, por favor, não faça isso. É uma má prática poluir seu HTML com JavaScript, e isso é ineficiente — você teria que incluir o atributo `onclick="criarParagrafo()"` em todo botão que você quisesse aplicar JavaScript.

Usando uma estrutura feita de puro JavaScript permite a você selecionar todos os botões usando uma instrução. O código que nós usamos acima para servir a esse propósito se parece com isso:

```
var botoes = document.querySelectorAll('button');  
  
for(var i = 0; i < botoes.length ; i++) {  
  botoes[i].addEventListener('click', criarParagrafo);  
}
```



Isso talvez parece ser mais do que o atributo onclick, mas isso vai funcionar para todos os botões, não importa quantos tem na página, e quantos forem adicionados ou removidos. O JavaScript não precisará ser mudado.

Comentários

Assim como HTML e CSS, é possível escrever comentários dentro do seu código JavaScript que serão ignorados pelo navegador, e existirão simplesmente para prover instruções aos seus colegas desenvolvedores sobre como o código funciona (e para você, se você tiver que voltar ao seu código depois de 6 meses e não se lembrar do que fez). Comentários são muito úteis, e você deveria usá-los frequentemente, principalmente quando seus códigos forem muito grandes. Há dois tipos:

Um comentário de uma linha é escrito depois de duas barras. Por exemplo:

```
// Eu sou um comentário
```

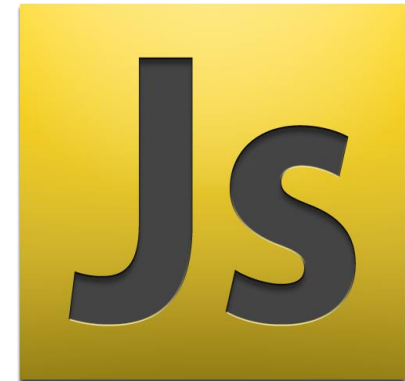
Um comentário de múltiplas linhas é escrito entre os caracteres `/*` e `*/`. Por exemplo:

```
/*
```

```
Eu também sou  
um comentário
```

```
*/
```

Então, por exemplo, você poderia fazer anotações na nossa última demonstração de código JavaScript, da seguinte forma:



// Função: Cria um novo parágrafo e o insere no fim do arquivo HTML.

```
function criarParagrafo() {  
  var para = document.createElement('p');  
  para.textContent = 'Você clicou no botão!';  
  document.body.appendChild(para);  
}
```

```
/*  
  1. Captura referências de todos os botões na página e armazena isso em um  
  array.  
  2. Vai até todos os botões e adiciona um event listener click a cada um deles.  
  
  Quando cada botão é clicado, a função criarParagrafo() será executada.  
*/
```

```
var botoes = document.querySelectorAll('button');
```

```
for(var i = 0; i < botoes.length; i++) {  
  botoes[i].addEventListener('click', criarParagrafo);  
}
```

O JavaScript talvez pareça um pouco assustador agora, mas não se preocupe — logo, logo com pesquisa e mais pratica, tudo vai começar a fazer sentido.



JQuery

O JQuery não é uma nova linguagem de programação como alguns imaginam, na verdade ele é uma compilação do JavaScript. Para utilizá-la, basta referenciar o arquivo .js do jQuery em sua página e você já pode usar esta linguagem a todo vapor.

Nós recomendamos que todo o código escrito também esteja em um arquivo externo; isso diminui o tamanho final de sua página HTML e também assegura que o código não ficará exposto, tornando as coisas mais fáceis, caso você queira mudar um código que apareça em todas as suas páginas.

Iniciando o JQuery

Para se usar o jQuery é necessário um conhecimento básico em JavaScript e HTML. Portanto, como já vimos um pouco de HTML e JavaScript é certo que vocês possuem um conhecimento básico dessas linguagens. A partir daí, podemos começar, mas por onde? Primeiro faça o download da versão mais recente da biblioteca jQuery (copie e cole a url <https://code.jquery.com/jquery-3.3.1.min.js> em seu navegador). Crie um arquivo HTML, faça a referência à biblioteca jQuery (usando o mesmo comando para se fazer referência à um arquivo .js qualquer).

Jquery iniciado com sucesso.



Agora crie um arquivo .js em branco, nele iremos desenvolver todas as lições. Faça a referência a ele em sua página HTML.

```
<script language="javascript" type="text/javascript" src="seu_arquivo.js">  
</script>
```

Pronto, agora estamos prontos para entrar no mundo do JQuery!

Seu primeiro script

Primeiramente temos que ver como funcionará um script jQuery. A sintaxe é a seguinte – pode ser digitada no documento .js que você criou ou entre as tags <script> </script> caso você opte por não usar um documento separado.

```
$(document).ready(function () {  
  //insira seu código aqui  
});
```



Capítulo 3 – Pré-processadores SASS e LESS

Pré-processadores

CSS por si só pode ser divertido, mas as folhas de estilo estão ficando maiores, mais complexas e mais difíceis de manter. É aqui que um pré-processador pode ajudar. Os pré-processadores permitem que você use recursos que não existem no CSS, como variáveis, aninhamento, mixins, herança e outras coisas bacanas que tornam a sua escrita CSS bem mais divertida.

Basicamente é como se você programasse em uma linguagem de programação compilada. Você escreve na sintaxe definida (Less/Sass/etc) e o pré processador compila para CSS e então você acrescenta o arquivo CSS a sua página normalmente.



SASS

Uma vez que você comece a trabalhar com o Sass, ele vai pegar o arquivo Sass pré-processado e salvá-lo como um arquivo CSS normal que você pode usar no seu site.

A maneira mais direta de fazer isso acontecer é no seu terminal. Uma vez que o Sass esteja instalado, você pode compilar seu Sass para CSS usando comandos sass. Você precisará informar ao Sass qual arquivo construir e para onde enviar o CSS. Por exemplo, a execução `sass input.scss output.css` do terminal levaria um único arquivo Sass `input.scss` e compilaria esse arquivo para `output.css`.

Por que SASS?

O site do Sass chama-se de 'CSS com superpoderes' e dá essa definição:

Sass é a linguagem de extensão CSS de nível profissional mais madura, estável e poderosa do mundo.

E uma das características que mais impressionam nele é a capacidade que ele tem de trabalhar com variáveis e aninhamento.

Documentação oficial: <https://sass-lang.com/guide>



LESS

Segundo o [Wikipédia](#), LESS é uma linguagem de folha de estilos dinâmica desenhada por Alexis Sellier. Ela foi influenciada pelo Sass e influenciou a nova sintaxe "SCSS" do Sass, que adaptou sua sintaxe de formação de blocos do tipo CSS.

LESS é de código aberto. Sua primeira versão foi escrita em Ruby, no entanto em versões posteriores, o uso de in Ruby foi depreciado e substituído por JavaScript.

A sintaxe indentada de LESS é uma metalinguagem aninhada, uma vez que um código válido em CSS também é válido em LESS e tem a mesma semântica. LESS fornece os seguintes mecanismos: variáveis, aninhamento, mixins, operadores e funções; a principal diferença entre LESS e outros pré-compiladores de CSS é que LESS permite a compilação em tempo real pelo navegador por meio de LESS.js

LESS pode ser executado tanto no lado do cliente quanto no lado do servidor ou pode ser compilado em CSS plano.



LESS

O que é o LESS?

Como já foi dito a respeito do SASS, não muito diferente, o LESS é um pré-processador de CSS, ou seja, você escreve seu CSS utilizando as funcionalidades do LESS e ele transforma seu código em CSS para que o navegador 'entenda'.

Utilizando o LESS direto no HTML.

```
<!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">
  <!-- Adicione seu código LESS assim como faria com o css. Por convenção
  a extensão do arquivo LESS. -->
  <link rel="stylesheet/less" type="text/css" href="styles.less">

  <!-- Adicione o seu javascript LESS. -->
  <script src="less.js"></script>

  <title>Desenvolvedor Full-Stack - LESS</title>
</head>
<body>
|
</body>
</html>
```



Documentação Oficial: <http://lesscss.org/>

[CLIQUE AQUI PARA CONHECER O PACOTE FULL-STACK!](#)

Capítulo 4 – Responsive Web Design

O que é Responsive Web Design

Com o crescimento da variedade de dispositivos onde os websites são visualizados (laptops, tablets, netbooks, celulares, desktops com tela pequena, iMacs com telas gigantescas, segundo monitor etc.), seria enlouquecedor desenhar múltiplas versões de um mesmo site que suprissem cada uma dessas variações de tamanho de tela e cada uma das resoluções de tela disponíveis no mercado.

O Responsive Web Design é uma das soluções técnicas para esse problema: programar um site de forma que os elementos que o compõem se adaptem automaticamente à largura de tela do dispositivo no qual ele está sendo visualizado.



Check-list básico do Design Responsivo

- Adaptar o layout da página de acordo com a resolução em que está sendo visualizada.
- Redimensionar as imagens automaticamente para que caibam na tela e para que não sobrecarreguem a transferência de dados em um celular, por exemplo.
- Simplificar elementos da tela para dispositivos móveis, onde o usuário normalmente tem menos tempo e menos atenção durante a navegação.
- Ocultar elementos desnecessários nos dispositivos menores.
- Adaptar tamanho de botões e links para interfaces touch onde o ponteiro do mouse é substituído pelo dedo do usuário.
- Utilizar de forma inteligente recursos mobile como geolocalização e mudança na orientação do aparelho (horizontal ou vertical).



Capítulo 5 – AngularJS

O que é AngularJS?

O AngularJS é um framework JavaScript muito poderoso. Ele é usado em projetos de Aplicativo de Página Única (SPA). Ele estende o HTML DOM com atributos adicionais e o torna mais responsivo às ações do usuário. O AngularJS é open source, totalmente gratuito e usado por milhares de desenvolvedores em todo o mundo. Está licenciado sob a versão 2.0 da licença Apache.

Foi originalmente desenvolvido em 2009 por Misko Hevery e Adam Abrons. Agora é mantido pelo Google. Sua última versão é 1.4.3.

Definição de AngularJS como colocado pela sua documentação oficial é a seguinte:

O AngularJS é um framework para aplicativos web dinâmicos. Ele permite que você use HTML como sua linguagem de modelo e permite estender a sintaxe do HTML para expressar os componentes da sua aplicação de forma clara e sucinta. A vinculação de dados e a injeção de dependência do Angular eliminam grande parte do código que você tem atualmente para escrever. E tudo acontece dentro do navegador, tornando-o um parceiro ideal com qualquer tecnologia de servidor.



Principais características

- O AngularJS é um poderoso framework de desenvolvimento baseada em JavaScript para criar o RICH Internet Application (RIA).
- O AngularJS fornece opções de desenvolvedores para escrever aplicativos do lado do cliente (usando JavaScript) em um modo MVC (Model View Controller) limpo.
- O aplicativo escrito em AngularJS é compatível com vários navegadores. O AngularJS manipula automaticamente o código JavaScript adequado para cada navegador.
- O AngularJS é open source, totalmente gratuito e usado por milhares de desenvolvedores em todo o mundo. Está licenciado sob a licença Apache versão 2.0.

No geral, o AngularJS é um framework para criar aplicativos Web de grande escala e alto desempenho, com uma fácil manutenção.

Principais recursos

Ligação de dados - É a sincronização automática de dados entre os componentes de modelo e exibição.



O Caminho para dominar as tecnologias mais poderosas do universo Full-Stack

- Escopo - Estes são objetos que se referem ao modelo. Eles atuam como uma cola entre o controlador e a *view*.
- Controlador - São funções JavaScript vinculadas a um escopo específico.
- Serviços - O AngularJS vem com vários serviços integrados, por exemplo, \$https: para fazer um XMLHttpRequests. Estes são objetos singleton que são instanciados apenas uma vez no aplicativo.
- Filtros - selecionam um subconjunto de itens de uma matriz e retornam uma nova matriz.
- Diretivas - Diretivas são marcadores em elementos DOM (como elementos, atributos, css e mais). Eles podem ser usados para criar tags HTML personalizadas que servem como novos widgets personalizados. AngularJS tem diretivas internas (ngBind, ngModel ...)
- Modelos - são a exibição renderizada com informações do controlador e modelo. Estes podem ser um único arquivo (como index.html) ou várias visualizações em uma página usando "parciais".
- Roteamento - É o conceito de alternar exibições.
- Model View Whatever - MVC é um padrão de design para dividir um aplicativo em diferentes partes (chamado Model, View e Controller), cada uma com responsabilidades distintas. O AngularJS não implementa o MVC no sentido tradicional, mas sim algo mais próximo do MVVM (Model-View-ViewModel).



[**CLIQUE AQUI PARA CONHECER O PACOTE FULL-STACK!**](#)

- Deep Linking - Deep linking permite que você codifique o estado do aplicativo no URL para que ele possa ser marcado. O aplicativo pode ser restaurado da URL para o mesmo estado.
- Injeção de Dependência - O AngularJS possui um subsistema de injeção de dependência integrado que ajuda o desenvolvedor, facilitando o desenvolvimento, o entendimento e o teste do aplicativo.

Os componentes do AngularJS

O framework AngularJS pode ser dividido em três partes principais -

- ng-app - Esta diretiva define e vincula um aplicativo AngularJS ao HTML.
- ng-model - Essa diretiva associa os valores dos dados do aplicativo AngularJS aos controles de entrada HTML.
- ng-bind - Esta diretiva vincula os dados do aplicativo AngularJS às tags HTML.



Capítulo 6 – Vue.js

O que é Vue.js?

O VueJS é um Framework JavaScript progressivo de software livre usado para desenvolver interfaces da Web interativas. É um dos mais famosos frameworks usados para simplificar o desenvolvimento web. O VueJS foca na camada de view. Ele pode ser facilmente integrado em grandes projetos para desenvolvimento de front-end sem quaisquer problemas.

A instalação do VueJS é muito fácil de começar. Qualquer desenvolvedor pode entender e criar interfaces interativas na Web em uma questão de tempo. O VueJS foi criado por Evan You, um ex-funcionário do Google. A primeira versão do VueJS foi lançada em fevereiro de 2014. Recentemente, ela atingiu 64.828 estrelas no GitHub, tornando-o muito popular.

Características

A seguir estão os recursos disponíveis com o VueJS.

Virtual DOM - O VueJS faz o uso do virtual DOM, que também é usado por outros frameworks como React, Ember, etc. As mudanças não são feitas no DOM, em vez disso uma réplica do DOM é criada e está presente na forma de frameworks de dados JavaScript.



Sempre que alguma alteração for feita, ela será feita nas estruturas de dados do JavaScript e a última será comparada com a estrutura de dados original. As alterações finais são atualizadas para o DOM real, que o usuário verá mudando. Isso é bom em termos de otimização, é menos dispendioso e as mudanças podem ser feitas em um ritmo mais rápido.

Ligação de dados - O recurso de ligação de dados ajuda a manipular ou designar valores a atributos HTML, alterar o estilo, designar classes com a ajuda da diretiva de ligação chamada v-bind disponível com o VueJS.

Componentes - Os componentes são um dos recursos importantes do VueJS que ajuda a criar elementos personalizados, que podem ser reutilizados em HTML.

Manipulação de eventos - v-on é o atributo adicionado aos elementos DOM para ouvir os eventos no VueJS.

Animação / Transição - O VueJS fornece várias maneiras de aplicar a transição a elementos HTML quando eles são adicionados / atualizados ou removidos do DOM. O VueJS possui um componente de transição integrado que precisa ser agrupado ao redor do elemento para efeito de transição. Podemos facilmente adicionar bibliotecas de animação de terceiros e também adicionar mais interatividade à interface.

Propriedades computadas - Esta é uma das características importantes do VueJS. Ajuda a ouvir as alterações feitas nos elementos da interface do usuário e executa os cálculos necessários.



Não há necessidade de codificação adicional para isso.

Modelos - O VueJS fornece modelos baseados em HTML que ligam o DOM aos dados da instância Vue. O Vue compila os modelos em funções virtuais do DOM Render. Podemos fazer uso do template das funções de renderização e para isso temos que substituir o template pela função render.

Diretivas - O VueJS possui diretivas internas, como v-if, v-else, v-show, v-on, v-bind e v-model, que são usadas para executar várias ações no frontend.

Observadores - Os observadores são aplicados aos dados que mudam. Por exemplo, forma elementos de entrada. Aqui, não precisamos adicionar nenhum evento adicional. O Watcher cuida de lidar com qualquer alteração de dados, tornando o código simples e rápido.

Roteamento - A navegação entre as páginas é realizada com a ajuda do roteador vue.

Peso leve - O script do VueJS é muito leve e o desempenho também é muito rápido.

Vue-CLI - O VueJS pode ser instalado na linha de comandos usando a interface de linha de comandos vue-cli. Isso ajuda a construir e compilar o projeto facilmente usando o vue-cli.



Configuração do ambiente

Existem várias maneiras de instalar o VueJS. Algumas das maneiras de como realizar a instalação são analisadas adiante.

Usando a tag `<script>` diretamente no arquivo HTML

```
<html>  
  <head>  
    <script type = "text/javascript" src = "vue.min.js"></script>  
  </head>  
  <body></body>  
</html>
```

Vá para o site inicial <https://vuejs.org/v2/guide/installation.html> do VueJS e baixe o vue.js conforme a necessidade. Existem duas versões para uso - versão de produção e versão de desenvolvimento. A versão de desenvolvimento não é minimizada, enquanto a versão de produção é minimizada, conforme mostrado na captura de tela na página seguinte. A versão de desenvolvimento ajudará com os avisos e o modo de depuração durante o desenvolvimento do projeto.

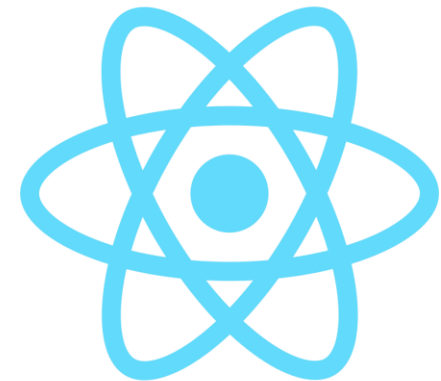


Capítulo 7 – React

O que é React?

React é uma biblioteca JavaScript usada para criar componentes de interface do usuário reutilizáveis. De acordo com a documentação oficial da React, segue a definição:

React é uma biblioteca para criar interfaces de usuários componíveis. Ele incentiva a criação de componentes de interface do usuário reutilizáveis, que apresentam dados que mudam com o tempo. Muitas pessoas usam o React como o V no MVC. React abstrai o DOM de você, oferecendo um modelo de programação mais simples e de melhor desempenho. O React também pode renderizar no servidor usando o Node e pode alimentar aplicativos nativos usando o React Native. O React implementa o fluxo de dados reativo unidirecional, o que reduz o clichê e é mais fácil de avaliar do que a vinculação de dados tradicional.



42

Recursos do React

JSX - JSX é extensão de sintaxe JavaScript. Não é necessário usar o JSX no desenvolvimento do React, mas é recomendado.

[**CLIQUE AQUI PARA CONHECER O PACOTE FULL-STACK!**](#)

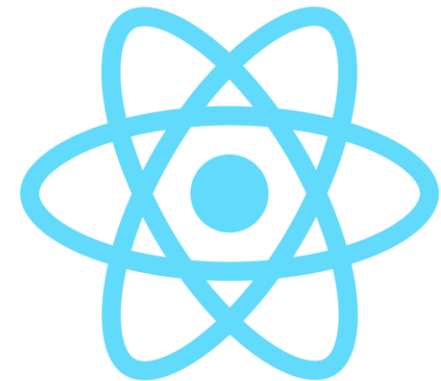
Componentes - React é tudo sobre componentes. Você precisa pensar em tudo como um componente. Isso ajudará você a manter o código ao trabalhar em projetos de escala maior.

O fluxo de dados unidirecional e o Flux - React implementam o fluxo de dados unidirecionais, o que facilita o raciocínio sobre o seu aplicativo. O fluxo é um padrão que ajuda a manter seus dados unidirecionais.

Licença - A React está licenciada sob o Facebook Inc. A documentação está licenciada sob CC BY 4.0.

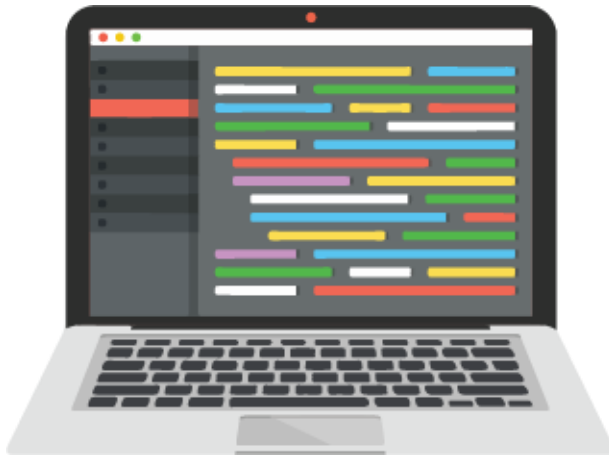
Vantagens do React.

- Usa o virtual DOM, que é um objeto JavaScript. Isso melhorará o desempenho dos aplicativos, pois o virtual DOM JavaScript é mais rápido que o DOM comum.
- Pode ser usado no lado do cliente e do servidor, bem como com outros frameworks.
- Os padrões de componentes e dados melhoram a legibilidade, o que ajuda a manter aplicativos maiores.



PARTE 2

DESENVOLVEDOR BACK- END



Capítulo 5 – PHP

Introdução ao PHP

De acordo com o “Secure.php.net”, O PHP (um acrônimo recursivo para PHP: Hypertext Preprocessor) é uma linguagem de script open source de uso geral, muito utilizada, e especialmente adequada para o desenvolvimento web e que pode ser embutida dentro do HTML.

E sabe porque ela é muito utilizada? Simplesmente porque além de ter uma sintaxe simples ela é de fácil aprendizado e isso faz com que o seu potencial cresça cada vez mais.

O PHP é totalmente processado no servidor, e isso faz dele uma linguagem Server-side.

Dessa forma suas aplicações não poderão ser copiadas por outros usuários. Todas as rotinas, funções e processos serão feitas no servidor e o usuário receberá apenas o resultado em seu navegador.



Fundamentos do PHP

Não podemos falar sobre os Fundamentos do PHP sem antes falar sobre como funcionam os servidores HTTP. Se escrevemos um simples arquivo .html, basta abrir ele no navegador e tudo estará funcionando, então já podemos ver a página assim como ela foi escrita. Mas quando se tem um servidor no processo as coisas já não são tão simples assim. O protocolo HTTP é utilizado para servir páginas na Web. É justamente por isso que todos os endereço na Web começam com o famoso http://.

Quando acessamos estes endereço na Internet, informamos que está sendo feita uma requisição no servidor. Ou seja, solicitamos que determinado conteúdo seja exibido.

Por exemplo, ao acessar <https://cursos.dankicode.com/php-jedai> estamos conectando via HTTP ao servidor cursos.dankicode.com e requisitando a URL "/php-jedai". Do outro lado, há um servidor HTTP aguardando por ser feitas novas requisições que é responsável por servir o que o usuário está pedindo.

Esse servidor que instalamos é um programa que fica responsável por processar todas as requisições. A questão que está em jogo é que esse servidor não precisa ser algo que simplesmente lê o arquivo HTML e envia os seus conteúdos para o cliente.

O servidor pode executar código e gerar HTML para o cliente na hora, de forma dinâmica. É esse processo de lógica dinâmica dentro do servidor que queremos fazer com o PHP.



Instalando PHP em casa.

O site oficial do PHP é o <http://php.net> e lá você irá encontrar todos os downloads e códigos fontes completos.

Windows

Para facilitar a instalação do PHP e dependências no Windows, existe um famoso pacote chamado WAMP da BitNami. Ele não só instala o PHP mas também o MySQL e o servidor Apache em um clique, além de várias outras dependências.

[Clique aqui, faça o download e execute o instalador agora mesmo.](#)

Depois de instalado, conseguimos acessar o binário do PHP pela linha de comando através de um menu. Vá em Iniciar -> BitNami Application Stack -> Use Application Stack.

Mac e Linux

Estes sistemas operacionais costumam vir com o PHP instalado.

Vamos agora para um exercício prático para entendermos de fato toda essa teoria.



Exercício – Executando o PHP

Crie um novo arquivo chamado hello-danki.php e coloque-o no diretório root do seu servidor web (DOCUMENT_ROOT) com o seguinte conteúdo:

```
<html>
<head>
<title>PHP Jedi</title>
</head>
<body>
<?php echo "<p>Olá Mundo</p>"; ?>
</body>
</html>
```

Abra o seu navegador e acesse o arquivo com a URL de seu servidor web, terminando com a referência ao arquivo /hello-danki.php. Quando o desenvolvimento for local esta URL será algo como `http://localhost/hello-danki.php` ou `http://127.0.0.1/hello-danki.php` mas isso depende da configuração do seu servidor web.



Se tudo foi configurado corretamente, este arquivo será interpretado pelo PHP e a seguinte mensagem será enviada ao seu navegador:

```
<html>
<head>
  <title>PHP Jedai</title>
</head>
<body>
<p>Olá Mundo</p>
</body>
</html>
```

O programa que você escreveu é realmente simples e você não precisa do PHP para criar uma página assim.

Tudo o que ela faz é apresentar: Olá Mundo utilizando a instrução echo. Note que o arquivo não precisa ser executável ou ter alguma ação especial.

O servidor web descobre que este arquivo precisa ser interpretado pelo PHP por causa da extensão “.php”, que o servidor é configurado para repassar ao PHP.

Pense nisso como um arquivo HTML normal que por acaso possui um conjunto de tags especiais disponíveis para você fazer muitas coisas interessantes.



Capítulo 6 – MySQL

Banco de dados MySQL

O MySQL é um sistema líder de gerenciamento de banco de dados de código aberto. É um sistema de gerenciamento de banco de dados multi-usuário e multithread. O MySQL é muito popular na web. É uma das partes da plataforma LAMP muito popular. Linux, Apache, MySQL e PHP. O banco de dados MySQL está disponível nas plataformas mais importantes do sistema operacional. Ele roda no BSD Unix, Linux, Windows ou Mac. Wikipédia, YouTube, Facebook usam o MySQL. Esses sites gerenciam milhões de consultas todos os dias. O MySQL vem em duas versões: sistema de servidor MySQL e sistema integrado MySQL. O software do servidor MySQL e as bibliotecas do cliente possuem licença dupla: GPL versão 2 e licença proprietária.

O desenvolvimento do MySQL começou em 1994 por uma empresa sueca MySQL AB. A Sun Microsystems adquiriu a MySQL AB em 2008. A Sun foi comprada pela Oracle em 2010.

MySQL, PostgreSQL, Firebird, SQLite, Derby e HSQLDB são os sistemas de banco de dados de código aberto mais conhecidos.

O MySQL é desenvolvido em C / C ++. Exceto para C / C ++, existem APIs para PHP, Python, Java, C #, Eiffel, Ruby, Tcl ou Perl.



MariaDB

MariaDB é uma bifurcação desenvolvida pela comunidade do MySQL, destinada a permanecer livre sob a GNU GPL.

É notável por ser liderado pelos desenvolvedores originais do MySQL, que o fechou devido a preocupações sobre sua aquisição pela Oracle. O MariaDB pretende manter alta compatibilidade com o MySQL, garantindo uma capacidade de substituição "drop-in" com equivalência binária da biblioteca e correspondência exata com as APIs e comandos do MySQL.

Vou deixar para você aqui a documentação do MySQL para você poder se aprofundar mais: <https://dev.mysql.com/doc/>



Capítulo 7 – Node.js

O que é Node.JS?

O Node.js é uma plataforma do lado do servidor (server-side) criada no [JavaScript Engine \(V8 Engine\) do Google Chrome](#). O Node.js foi desenvolvido por Ryan Dahl em 2009 e sua versão mais recente é v8.11.1. A definição de Node.js fornecida por sua documentação oficial é a seguinte:

O Node.js é uma plataforma criada no tempo de execução JavaScript do Chrome para criar facilmente aplicativos de rede rápidos e escalonáveis. O Node.js usa um modelo de I/O sem bloqueio orientado a eventos que o torna leve e eficiente, perfeito para aplicativos em tempo real que usam muitos dados e que são executados em dispositivos distribuídos.

O Node.js é um cross-platform runtime environment Open Source para o desenvolvimento de aplicativos do lado do servidor e de rede. As aplicações do Node.js são escritas em JavaScript e podem ser executadas no runtime do Node.js no OS X, Microsoft Windows e Linux.

O Node.js também fornece uma rica biblioteca de vários módulos JavaScript que simplifica o desenvolvimento de aplicativos da Web usando o Node.js em grande medida.

Node.js = Runtime Environment + JavaScript Library



Recursos do Node.JS

A seguir, alguns dos recursos importantes que tornam o Node.js a primeira escolha de desenvolvedores web.

Asynchronous and Event Driven (baseado em eventos) - Todas as APIs da biblioteca Node.js são assíncronas, isto é, sem bloqueio. Essencialmente, significa que um servidor baseado em Node.js nunca espera por uma API para retornar dados. O servidor passa para a próxima API depois de chamá-lo e um mecanismo de notificação de Eventos do Node.js ajuda o servidor a obter uma resposta da chamada da API anterior.

Muito rápido - Sendo construída no Mecanismo JavaScript V8 do Google Chrome, a biblioteca Node.js é muito rápida na execução de códigos.

Single Threaded, mas Altamente Escalável - O Node.js usa um único modelo encadeado com loop de eventos. O mecanismo de eventos ajuda o servidor a responder de maneira não-bloqueante e torna o servidor altamente escalável, ao contrário dos servidores tradicionais, que criam encadeamentos limitados para lidar com solicitações. O Node.js usa um único programa encadeado e o mesmo programa pode fornecer serviço para um número muito maior de solicitações do que os servidores tradicionais, como o Apache HTTP Server.



Sem armazenamento em buffer - os aplicativos Node.js nunca armazenam nenhum dado em buffer. Esses aplicativos simplesmente exibem os dados em partes.

Licença - Node.js é liberado sob a licença [MIT](#) .

Quem está utilizando o Node.JS ?

A seguir está o link no wiki do github contendo uma lista exaustiva de projetos, aplicativos e empresas que estão usando o Node.js. Esta lista inclui o eBay, a General Electric, a GoDaddy, a Microsoft, o PayPal, o Uber, o Wikipins, o Yahoo !, e o Yammer, para citar alguns.

[Projetos, aplicativos e empresas usando o node](#)

Onde usar o Node.js?

A seguir estão as áreas onde o Node.js está se mostrando um parceiro tecnológico perfeito.

- I/O Aplicativos vinculados
- Aplicativos de transmissão de dados
- Aplicativos Intensivos em Tempo Real de Dados (DIRT)



- Aplicativos baseados em APIs JSON
- Aplicativos de página única



Conclusão

Chegamos ao fim dessa incrível jornada pelas mais inovadoras e avançadas tecnologias do ambiente Full-Stack.

Esse é o caminho, e se você realmente perseverar em caminhar por ele, irá se tornar um desenvolvedor Full-Stack completo e requisitado, pois estará dominando as mais incríveis tecnologias de desenvolvimento web.

Se você já adquiriu o nosso Pacote Full-Stack, lá estaremos nos aprofundando ainda mais em cada tecnologia abordada aqui, com uma quantidade gigantesca de projetos práticos, desde o básico ao avançado para você de fato não ter desculpas quanto a se tornar um verdadeiro desenvolvedor Full-Stack

E caso você ainda não tenha adquirido o Pacote Full-Stack, deixo abaixo o link do do treinamento. Lá você verá em detalhes tudo que abordaremos dentro do pacote Full-Stack.

Este e-book não chega nem aos pés do nosso treinamento, na verdade o e-book é apenas para dar um vislumbre de tudo que nós iremos ver dentro do pacote Full-Stack.

SIM! QUERO SER UM(A) DESENVOLVEDOR(A) FULL-STACK!





Os melhores cursos para você

Torne-se um desenvolvedor completo e aprenda com quem está no mercado.



A Danki Code é uma empresa de desenvolvimento web e também atua na área de cursos online voltados também ao assunto. Sediada em Florianópolis, desenvolve soluções inovadoras para empresas e também forma profissionais capacitados e completos para a área de TI.

Atendimento ao cliente:

E-mail: contato@dankicode.com / Telefone/WhatsApp: (48)99901-3620

De segunda a domingo - 24h/dia