
HTML
CSS
JS
PHP
Sass
less
O CAMINHO PARA DOMINAR AS TECNOLOGIAS MAIS PODEROSAS DO UNIVERSO
FULL-STACK
node
ANGULARJS
by Google
MySQL

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 certeiro, 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.

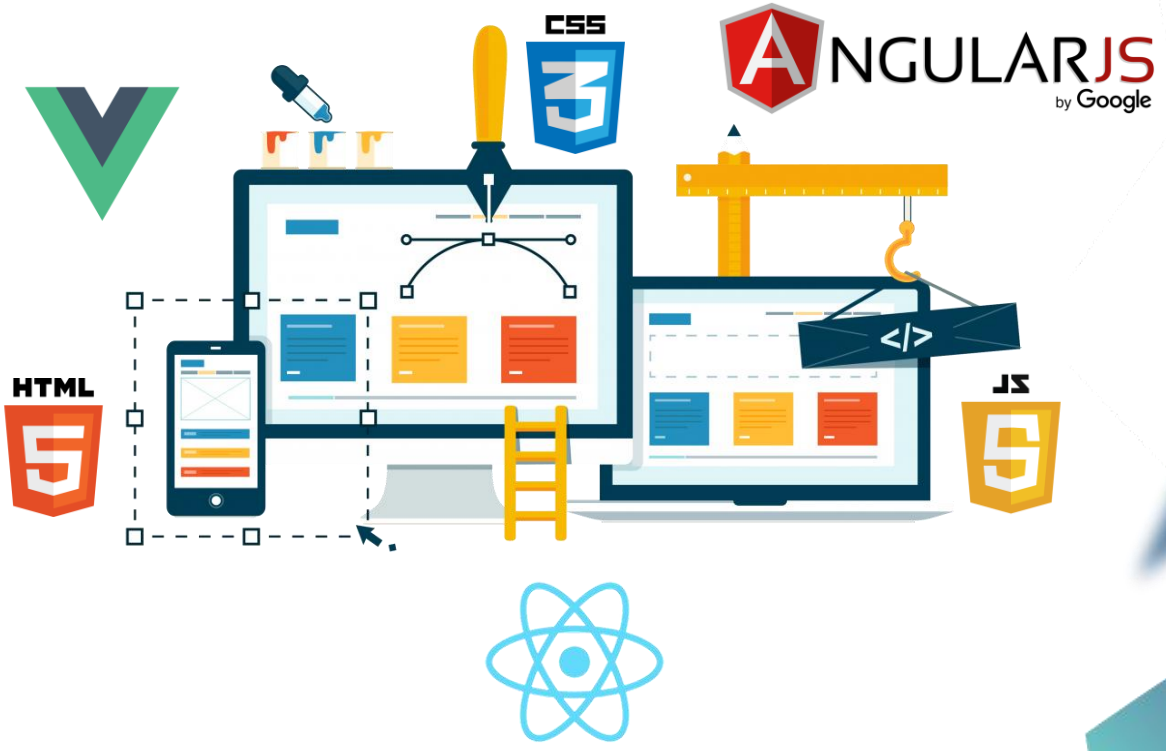
E se de alguma forma você foi despertado para se tornar um Desenvolvedor Full-Stack nesse e-book é realmente o caminho completo para você começar hoje mesmo a dominar esse universo fantástico.

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

Índice

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 da significado à informação que o usuário consome. É nela que você marca o que é um titulo, 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.

Mas antes de entrarmos nessa questão da semântica, vamos entender de fato as mudanças que ocorreram do HTML tradicional para o HTML5 que temos hoje.

A evolução do HTML.

A evolução para o HTML5 veio para transformar e facilitar a forma de escrever HTML, trazendo dois conceitos básicos e diferentes:

- É uma versão mais sofisticada, com novos elementos, atributos e comportamentos.
- Traz com ela um conjunto maior de tecnologias que permite o desenvolvimento de aplicações e sites diversificados e com um poder incrível. Esse conjunto tem um nome e se chama *HTML5 e friends* e muitas vezes abreviado apenas como HTML5.

Estrutura e semântica.

A semântica permite você descrever o seu conteúdo de forma mais precisa. E sem dúvida a semântica do HTML5 foi uma de suas principais evoluções.

Doctype e Metadados

Observando a estrutura fundamental do código do HTML5, vamos perceber que ela se mantem praticamente idêntica à versão anterior, com exceção do *Doctype* e *Metatag Charset.*

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 Defination (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>
```

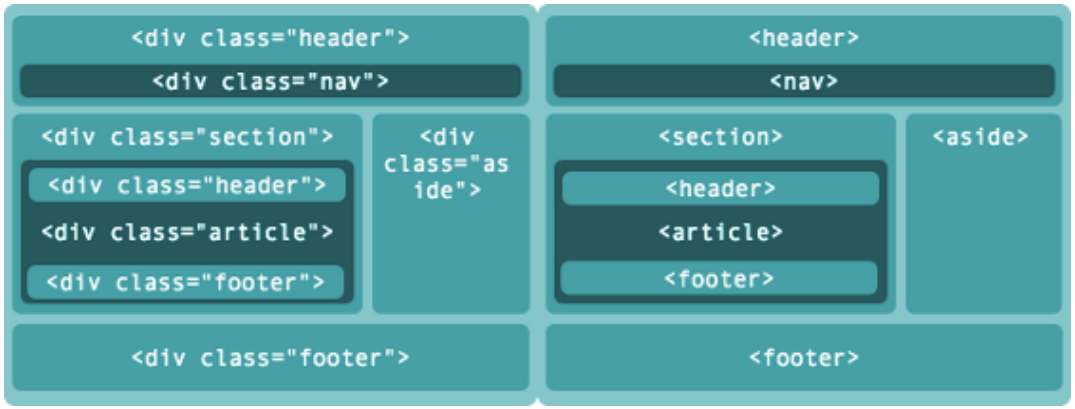
Informações como Title e Description por exemplo, são usadas pelos mecanismos de busca para tornar o seu site localizável nos mecanismos de buscas.

facebook

A semântica das novas marcações do HTML5

Nas versões passadas do HTML não havia tags com uma semântica adequada para cada uma das divisões do layout. Dessa forma, os desenvolvedores acabavam usando a tag para todas as situações, e criando seus próprios padrões de nomenclaturas através dos atributos id ou class.

No HTML5 foram criadas varias tags semânticas para indicar aos user-agents quais conteúdos estão sendo inseridos em cada uma das divisões da página, organizando e padronizando o desenvolvimento.



A semântica das novas marcações do HTML5

Nas versões passadas do HTML não havia tags com uma semântica adequada para cada uma das divisões do layout. Dessa forma, os desenvolvedores acabavam usando a tag para todas as situações, e criando seus próprios padrões de nomenclaturas através dos atributos id ou class.

Exemplo de cabeçalho com o elemento <header>

O novo elemento <header> é utilizado para definir o cabeçalho de uma página ou seção, e pode conter logo, títulos, menu de navegação, campo de busca, etc.

```
<header>
  <img alt="logo" src="/img/logo.png">
  <h1>Entendo a estrutura e semântica do HTML5</h1>
  <form action=www.enderecodosite.com.br/search id="searchform" method="get">
    <label for="s">Pesquisa</label>
    <input type="text" placeholder="Pesquisa" id="s">
    <input type="submit" value="Pesquisa" class="submit">
  </form>
</header>
```

Dentro do <header> também podemos inserir o elemento <hgroup>, que serve para agrupar dois ou mais elementos de títulos (h1 até h6), organizando-os hierarquicamente. O elemento <hgroup> foi descontinuado e não deve mais ser usado.

Navegando de forma Global com o elemento <nav>

O elemento <nav> serve para agrupar uma lista de links para outras partes do site, seja essa lista de navegação local ou global. Esses blocos de links podem estar em diferentes partes do layout, como no cabeçalho ou no rodapé.

```
<nav>
    <ul class="menu">
      <li><a href="inicio.html">Início</a></li>
      <li><a href="quem-somos.html">Quem somos</a></li>
      <li><a href="servicos.html">Serviços</a></li>
      <li><a href="contato.html">Contato</a></li>
    </ul>
</nav>
```

Seções diferentes com o elemento <section>

O elemento <section> é o menos especifico entre as novas tags. A diferença do <section> para um é que o primeiro serve para dividir o conteúdo em diferentes seções, que podem conter elementos como <header> ou <article>, enquanto o segundo divide qualquer conteúdo, sem uma finalidade específica.

Exemplo:

```
<section>
  <article>
    (...)
  </article>
</section>
```

Conteúdos relacionados com o elemento <aside>

O elemento <aside> serve para mostrar conteúdos que fazem referência ao conteúdo principal à sua volta, como informações, blocos de navegação ou até mesmo publicidade. Vejamos o exemplo abaixo:


```
<div id="main">
  <div id="primary">
    <article>
      <header>
        <h1>Entendendo o elemento Aside</h1>
```

```
<p>Publicado em <time pubdate datetime="2011-09-01">01 de Setembro de 2011</time></p>
        </header>
        (...)
        <aside>
          <h2>Curiosidades sobre o Aside</h2>
          (...)
        </aside>
        <footer>
                <p>Esse post foi publicado na categoria <a href="#" title="HTML5">HTML5</a>
        </footer>
    </article>
  </div>
```

```
<div id="secondary">
        <aside>
      <h3>Arquivos</h3>
<ul>
          <li><a href="#" title="agosto 2018">agosto 2011</a></li>
          <li><a href="#" title="julho 2018">julho 2011</a></li>
          <li><a href="#" title="junho 2018">junho 2011</a></li>
          <li><a href="#" title="maio 2018">maio 2011</a></li>
          <li><a href="#" title="abril 2018">abril 2011</a></li>
        </ul>
    </aside>
  </div>
</div>
```

Aqui temos duas situações onde o <aside> é empregado. No primeiro caso ele está dentro de um elemento <article>, e por isso podemos entender que ele traz conteúdos relacionados ao post ou artigo em questão.

No segundo caso o <aside> está fora do <article>, fazendo parte da barra lateral do site, trazendo links de navegação que tem relação com a página como um todo.

Rodapé utilizando o elemento <footer>.

O elemento <footer> representa o rodapé de um documento ou de uma seção específica do mesmo, podendo conter informações relacionadas ao autor e ao copyright, blocos de navegação ou links relacionados.

```
<body>
  <div id="page">
    <article>
      <header>
        <h1>Entendendo o elemento Footer</h1>
        <p>Publicado em <time pubdate datetime="2018-04-01">01 de abril de 2018</time></p>
      </header>
      (...)
      <footer>
        <p>Publicado na categoria <a href="#" title="HTML5">HTML5</a></p>
      </footer>
    </article>
```

```
<footer>
        <nav>
          <ul>
            <li><a href="inicio.html">Início</a></li>
            <li><a href="quem-somos.html">Quem somos</a></li>
            <li><a href="servicos.html">Serviços</a></li>
            <li><a href="contato.html">Contato</a></li>
          </ul>
        </nav>
        <small>Copyright © 2018 – Danki Code. Todos os direitos
reservados</small>
      </footer>
   </div>
</body>
```

No exemplo acima vemos o <footer> sendo empregado em duas situações: no primeiro caso ele está contido dentro do elemento <article>, representando o rodapé do post ou artigo em questão, e no segundo caso está ao final do código, representando o rodapé da página.

Depois de uma super fundamentação de HTML iremos abordar agora o CSS de uma forma bem dinâmica e prática, para que já comecemos a entender toda a engrenagem por traz de uma boa estilização.

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:

Claro que não é algo tão bonito, mas pelo menos da uma boa idéia de como o CSS funciona.

Como aplicar o CSS ao seu documento HTML?

Existem três maneiras diferentes de aplicar o CSS a um documento HTML que você normalmente verá. Aqui vamos rever brevemente cada um deles.

Folha de estilo externa.

Uma folha de estilo externa é quando você tem seu CSS escrito em um arquivo separado com uma extensão.css e faz referência a ele a partir de um <link> no HTML . O arquivo HTML fica algo como:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>Pacote Full-Stack</title>
  <link rel="stylesheet" href="style.css">
 </head>
```

```
 <body>
   <h1>Desenvolvedor Web Completo</h1>
   <p>Exemplo CSS para estilizar minhas páginas</p>
  </body>
</html>
```

E o arquivo CSS:

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

p {
  color: red;
}
```

Este método é indiscutivelmente o melhor, já que você pode usar uma folha de estilo para estilizar vários documentos, e só precisaria atualizar o CSS em um único lugar se as mudanças forem necessárias.

Folha de estilo interna.

Uma folha de estilo interna é onde você não tem um arquivo CSS externo, mas coloca seu CSS dentro de um elemento <style>, contido dentro do head do HTML . Então o HTML ficaria assim:

```
<!DOCTYPE html>
<html>
 <head>
  <meta charset="utf-8">
  <title>My CSS experiment</title>
  <style>
   h1 {
    color: blue;
    background-color: yellow;
    border: 1px solid black;
   }

   p {
    color: red;
   }
  </style>
 </head>
```

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

Estilo Inline:

Estilos inline são declarações CSS que afetam apenas um elemento, contido em um atributo style:

```
<!DOCTYPE html>
<html>
 <head>
   <meta charset="utf-8">
   <title>Pacote Full-Stack</title>
 </head>
 <body>
   <h1 style="color: blue;background-color: yellow;border: 1px solid black;">
Desenvolvedor Web Completo </h1>
   <p style="color:red;"> Exemplo CSS para estilizar minhas paginas </p>
 </body>
</html>
```

Mas por favor, não faça isso, a menos que você realmente precise! É muito ruim para manutenção do seu código (você pode ter que atualizar as mesmas informações várias vezes por documento), e também mistura suas informações de CSS com suas informações estruturais de HTML, tornando o CSS mais difícil de ler e entender. Manter seus códigos separados e puros torna um trabalho muito mais fácil para todos os que trabalham no código.

A única vez que você pode ter que recorrer ao uso de estilos embutidos é quando seu ambiente de trabalho é realmente restritivo (talvez seu CMS permita editar apenas o corpo do HTML).

Então aprendemos como funciona o CSS e esse é realmente o caminho para você iniciar qualquer estilização, agora vamos mergulhar um pouco mais fundo com o JavaScript que é um pouco mais complexo pois vamos realmente programar utilizando lógica de programação.

Apesar de o CSS já trazer algum movimento para o nosso Web Site, o JavaScript tem isso como prioridade, fazer do nosso site mais dinâmico e animado.

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.

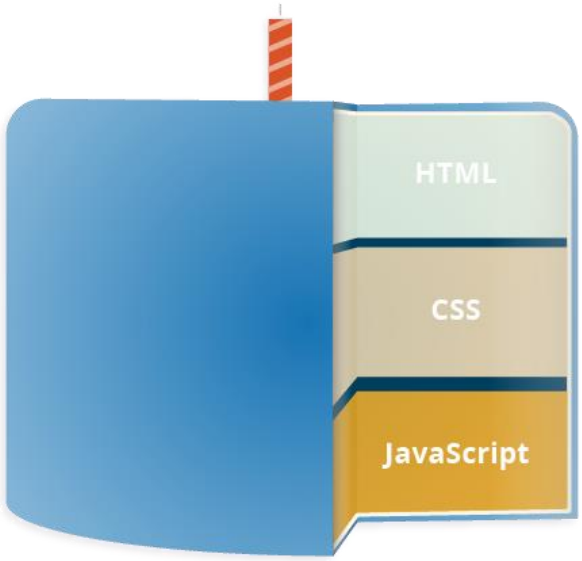
Ferramentas essas que fazem o "ciclo de vida" dos produtos serem extremamente produtivos em relação à escalabilidade e agilidade no desenvolvimento Client-Side, pensando em manutenibilidade e outros fatores. Outro ponto que acho extremamente relevante abordar é que uma das comunidades mais ativas é a 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

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.

facebook

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 usa-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).

Certifique-se de que a referência à biblioteca seja feita antes da referência ao seu arquivo .js, caso contrário o navegador não irá interpretar corretamente os códigos desenvolvidos.

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
});
```

Vamos analisar parte por parte deste código. Usa-se o cifrão ($) para referir-se a modificação de um elemento (podendo ser desde uma tag até uma classe e ID CSS). Usamos os parênteses após o cifrão para identificar qual elemento será modificado. Neste caso, nos referimos ao documento como um todo, pois os códigos que serão introduzidos irão alterar o conteúdo do documento.

A função .ready informa ao navegador que os comandos devem ser executados quando o documento estiver pronto (navegável). Adicionamos o parêntese após o .ready para informar ao navegador que os comandos enlaçados pelos parênteses deverão ser executados no momento em que o documento for navegável. O uso de function enlaça todos os comandos que deverão ser executados.

Use os colchetes e parênteses para fechar o enlace. Use ponto e virgula (;) para separar as linhas de comando e evitar erros de sintaxe. Esta linha inicial é de uso essencial, para não dizer obrigatório, no funcionamento do seu arquivo jQuery. Todos os comandos que usaremos no documento serão enlaçados por esta linha de comando. Agora que entendemos como o navegador irá interpretar os comandos, vamos construir nosso primeiro script; o famoso "olá mundo".

```
$(document).ready(function () {
alert("olá mundo");
});
```

Em suma, neste comando, estamos informando ao navegador que, assim que o documento estiver pronto, ele deve escrever na tela "olá mundo". Esse texto será exibido em uma janela de alerta.

Ótimo! Agora vamos tentar o mesmo comando só que de uma forma um pouco diferente:

Como já foi passado o uso do cifrão informa que elemento será alterado. Nesse caso a tag ("a").

Feito isto, construa um código HTML simples em seu arquivo HTML. Não se preocupe com formatação, apenas tenha certeza de ter feito as devidas referências aos documentos .js.

Crie o seguinte código em seu documento HTML:

```
<body>
<p><a href="#">clique aqui</a></p>
</body>
```

Salve o documento e abra-o em seu navegador, clique no link e veja o resultado. Agora vamos estudar o caso. Quando fizemos a referência à tag "a" e usamos o comando .click , informamos ao navegador que uma ação deveria ser executada; nesse caso o comando alert que exibirá o texto "olá mundo".

Adicionando CSS3

Usando alguns comandos em jQuery, é possível alterar a aparência dos documentos, podemos adicionar uma clasee que haja em nossos arquivos CSS, ou até mesmo adiciona o CSS de modo inline, ou seja como atributo style da tag. Para isso, usaremos o comando .addClass. Para testar esse comando, crie um arquivo css e referencie-o em sua página HTML, ou construa-o direto no arquivo HTML usando as tags <style> e </style>. Heis os dados que seu CSS deve conter:

```
.teste {
        border-width: 3px;
        border-style: dashed;
        border-color: red;
}
```

Agora vamos fazer com que, ao clicarmos no link, seja possível adicionar a classe ".teste" à tag <p>.

Exemplo:

```
$(document).ready(function () {
        $("a").click(function() {
            $("p").addClass("teste");
        });
 });
```

Agora, quando clicarmos no link, uma borda, de espessura media, pontilhada e vermelha, irá aparecer ao redor do texto. Também podemos adicionar uma formatação CSS direto à tag, usando o comando .css(). Dentro dos parênteses, coloque a formatação desejada.

Exemplo:

```
$(document).ready(function () {
        $("a").click(function() {
            $("p").css("border", "3px dashed red");
        });
 });
```

Modificando apenas parte do documento

Nesse capítulo, iremos aprender como modificar elementos específicos na página. Até agora, tudo o que aprendemos modificava todas as tags nas páginas; porém, com uma modificação na sintaxe nós podermos alterar apenas uma pequena parte do documento.

Os comandos novos que aprenderemos agora são .addClass e .removeClass, que como os próprios nomes dizem, servem para adicionar e remover uma classe CSS, respectivamente, da tag.

Lembra-se que, no começo desta apostila, foi dito que poderia se usar o cifrão ($) para indicar uma tag ou ID que seria alterado? Pois bem, chegou a hora de trabalhar com ID's. Para iniciarmos, crie um documento CSS com uma classe chamada "fundo". Seu conteúdo deve ser:

```
.fundo {
        background-color: red;
        color: white;
}
```

Agora, em seu documento HTML, crie duas listas com marcadores, ou não classificadas (
), com, no mínimo, dois itens; dê a uma das listas o ID lista_teste. Em seu documento .js, crie o seguinte comando:

```
$(document).ready(function () {
        $("#lista_teste").hover(function() {
            $(this).addClass("fundo");
          }, function() {
           $(this).removeClass("fundo");
          });
});
```

Com esse comando você irá modificar apenas os itens que contenham a ID "lista_teste", no momento em que você passar o mouse sobre eles. Agora, vamos aprofundar o código ainda mais. Ao invés de modificar a lista toda, que tal modificar apenas um único item dessa lista? Para isso, no seu documento .js, escreva o seguinte comando:

```
$(document).ready(function () {
        $("#lista_teste li:last").hover(function() {
            $(this).addClass("fundo");
         }, function() {
        $(this).removeClass("fundo");
        });
});
```

Com isso, apenas o último item da lista irá ser modificado quando você passar o mouse por cima dele, caso o mouse seja passado no resto da lista, nenhuma alteração será feita. Agora vamos entender o que nós escrevemos: Aprendemos um comando novo; o (this), ele faz a referência ao último elemento que foi alterado no script. Nesses casos, o ID "lista_teste" e o último item desta lista. Agora vamos criar um botão no documento HTML. Dê a ele o ID "botao1". Atualize seu código .js da seguinte forma:

```
$(document).ready(function () {
        $("#botao1").click(function() {
            $("#lista_teste").addClass("fundo");
        });
});
```

Dessa forma, toda vez que o usuário clicar no botão que possua o ID "botao1", a lista receberá a classe "fundo". Também é possível modificar apenas o último elemento da lista; basta adicionar "li:last" após o ID "lista_teste". O código ficará assim:

```
$(document).ready(function () {
        $("#botao1").click(function() {
            $("#lista_teste li:last").addClass("fundo");
        });
});
```

Agora vamos adicionar um botão para remover a classe "fundo". Crie em seu documento HTML outro botão e atribua a ele o ID "botão2", escreva o seguinte código em seu documento .js:

```
$(document).ready(function () {
        $("#botao1").click(function() {
            $("#lista_teste").addClass("fundo");
        });
        $("#botao2").click(function() {
            $("#lista_teste").removeClass("fundo");
        });
});
```

Para fazer isso apenas no último item da lista, basta adicionar "li:last" após o ID "lista_teste", nas duas vezes em que ela é referenciada no código, ficando dessa forma:

```
$(document).ready(function () {
        $("#botao1").click(function() {
            $("#lista_teste li:last").addClass("fundo");
        });
        $("#botao2").click(function() {
            $("#lista_teste li:last").removeClass("fundo");
        });
});
```

Essas funções .addClass e .removeClass podem ser usadas para alterar qualquer tag, porém tome muito cuidado para não usar junto ao evento .hover, pois em alguns casos, como por exemplo textos, pode ser que o parágrafo adicione e remova a classe CSS toda a vez que o ponteiro do mouse oscilar entre o texto e o espaço em branco.

Concluímos agora o capítulo 2 do nosso e-book, aprendemos algumas tecnologias mais utilizadas no front-end, e agora iremos continuar avançando e entender o universo dos pré-processadores CSS, compreendê-los e concluir se realmente vale ou não vale a pena utiliza-los.

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 pre-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.

facebook

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.

Variáveis no SASS são semelhantes às variáveis em qualquer outra linguagem de programação. Você define uma para guardar uma informação que deseja usar em todo o seu arquivo SASS.

A primeira coisa que penso aqui é como acompanhar as cores de um site. Se eu estiver reutilizando cores de texto, plano de fundo ou borda, não quero ter que fazer referência a uma lista para saber qual é qual. Ou se eu precisar mudar uma dessas cores, então eu tenho que encontrar em todos os lugares no arquivo CSS que eu usei. Com uma variável SASS eu configuro tudo em um lugar e é isso.

O aninhamento no SASS é outro recurso que ajuda a manter um arquivo mais organizado e fácil de ler. Se você tiver um site complexo e precisar definir uma propriedade CSS em um elemento profundamente aninhado, provavelmente terá alguns seletores descendente bastante desagradáveis para acessar esses elementos. O aninhamento acaba com isso. Em vez de seletores descendentes, o SASS usa o recuo para determinar a hierarquia de seletores. Isso torna o código mais limpo e mais fácil de manter.

Que tal irmos para prática e testarmos essa teoria na prática?

Instalando e configurando o SASS

Você pode instalar o SASS através da linha de comando, assim como a maioria das outras ferramentas. O SASS é instalado através de uma gem Ruby.

```
gem install sass
```

Depois que o SASS é instalado, você deve informar onde os arquivos SASS estão localizados e onde os arquivos CSS devem ser compilados. A sintaxe para isso é bem simples, você chama o comando sass e, em seguida, fornece um arquivo de entrada e um arquivo de saída.

sass style.sass: style.css

Outra opção para compilar é adicionar um sinalizador 'watch' para que o SASS observe um diretório ou um arquivo individual e sempre que o arquivo for alterado, o arquivo CSS será automaticamente compilado.

sass --watch style.sass: style.css

Quando você assistir aos arquivos, o terminal faz uma pausa para você. Quando você faz isso, salve-as como se estivessem usando sass ou node-sass.

```
// ♥ sass --watch style.sass:style.css
>>> Sass is watching for changes. Press Ctrl-C to stop.
>>> Change detected to: style.sass
      write style.css
      write style.css.map
```

Entrando em ação com o SASS

Agora que você sabe como configurar o SASS e executar o resto é muito fácil.

Apenas crie um arquivo .sass no diretório do seu projeto e escreva seu SASS lá. Quando você executa o processador (ou o arquivo está sendo observado para alterações), o código nesse arquivo será alterado para CSS.

Aqui está um exemplo que eu escrevi que cria uma barra de navegação realmente básica com um logotipo à esquerda e três links ao longo do resto da página.

```
<!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="style.css">
    <title>Desenvolvedor Full-Stack - SASS</title>
</head>
<body>
    <div id="navbar">
        <div id="logo"></div>
        <div id="links">
            <ul>
                <li>Sobre</li>
                <li>Blog</li>
                <li>Contato</li>
            </ul>
        </div>
    </div>
</body>
</html>
```

Observe que a tag <link> está apontando para o arquivo .css usual.

facebook

```
$logoColor: tomato
$divBorder: black 2px solid

#navbar
    display: flex
    flex-direction: row

    #logo
        background: $logoColor
        height: 100px
        width: 100px
        border-radius: 50%
        border: $divBorder

    #links
        width: 100%

        ul
            display: flex
            flex-direction: row
            justify-content: space-around

            li
                list-style: none
```

Código SASS

Eu criei várias seções aninhadas para esta página para que você possa ver essa parte do SASS sendo usada. Eu também incluí algumas variáveis no topo do arquivo. Aqui está o que o CSS parece depois de tudo isso.

```
#navbar {
  display: flex;
  flex-direction: row; }
  #navbar #logo {
    background: tomato;
    height: 100px;
    width: 100px;
    border-radius: 50%;
    border: black 2px solid; }
  #navbar #links {
    width: 100%; }
    #navbar #links ul {
      display: flex;
      flex-direction: row;
      justify-content: space-around; }
      #navbar #links ul li {
        list-style: none; }
```

Deu para entender como o SASS pode ser útil em seus projetos?

Então vamos passar para o LESS que é outro pre-processador bem interessante.

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>
```

Assim como no SASS, o LESS também trabalha com variáveis, onde você pode defini-las e reutiliza-las em todo o seu CSS. Desse forma se for mudar algum valor, precisará fazer isso me um único lugar.

Exemplo de Variáveis:

```
@corPrincipal: #5B83AD; /* Declaração da variável */

#header {
  color: @corPrincipal; /* Utilização da variável */
}
```

Reutilização de código

É possível declarar estilos no formato de classe CSS e reutiliza-los:

```
.bordaPadrao { /* Definindo estilo a ser reutilizado */
  border: 1px solid black;
}

#imagem{
  .bordaPadrao; /* Reutilizando 1 */
}
```

```
.quadroDestaque {
  color: red;
  .bordaPadrao; /* Reutilizando 2 */
}
```

Hierarquia

Com o LESS você pode definir uma hierarquia mais clara no css. Por exemplo observe o código abaixo:

CSS

```
#header {
  color: black;
}
#header .navigation {
  font-size: 12px;
}
#header .logo {
  width: 300px;
}
```

O código acima pode ser traduzido para a seguinte forma:

LESS

```
#header {
  color: black;
  .navigation {
    font-size: 12px;
  }
  .logo {
    width: 300px;
  }
}
```

Mas o mais importante é você olhar a documentação completa no site deles, pois aqui estamos dando apenas o start lhe apresentando o caminho, no entanto a jornada é longa se você quiser dominar tudo que estamos lhe passando nesse e-book.

Documentação Oficial: http://lesscss.org/

Tudo tranquilo até aqui? Vamos agora ver no capítulo 4 como trabalhar com Design Responsivo e descobrir as melhores tecnologias e práticas para progredir no desenvolvimento Web.

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).

O Google recomendou oficialmente o Responsive Web Design como seu método preferido para criar websites para dispositivos móveis. Se você tiver um site ou um blog, é hora de considerar a mudança para o design responsivo, em vez de manter websites separados para dispositivos móveis e tablets.

Por que você deve mudar seu site para RWD?

Seu site parece ótimo com uma tela da área de trabalho, mas essa afirmação pode não ser verdadeira quando seu site é visualizado em um smartphone ou tablet. Depois de tornar o design responsivo, o site ficará bem (e legível) em todas as telas.

Além da legibilidade, existem inúmeras vantagens para você utilizar o RWD, como:

- Economiza tempo e dinheiro, pois você não precisa manter sites separados para desktops e telefones celulares.
- Design responsivo é bom para o SEO do seu site (rankings de busca), pois cada página do seu site terá um único URL e, portanto, o juice do Google é preservado.
- Seus relatórios do Google Analytics exibirão uma imagem melhor do uso do seu site, já que os dados dos usuários de dispositivos móveis e computadores serão consolidados.

- O mesmo será válido para as estatísticas de compartilhamento social (curtidas no Facebook, tweets), pois as versões para dispositivos móveis e computadores das suas páginas web terão o mesmo URL.

- Os designs responsivos são mais fáceis de manter, pois não envolvem nenhum componente do lado do servidor. Você só precisa modificar o CSS subjacente de uma página para alterar sua aparência (ou layout) em um determinado dispositivo.

O que você precisa saber para começar a utilizar o Design Responsivo?

O design responsivo é puro HTML e CSS. Você cria regras em CSS que alteram o estilo com base no tamanho da tela do dispositivo do usuário.

Por exemplo, você pode escrever uma regra que diz que se o tamanho da tela de um usuário for menor que 320 pixels, não mostrar a barra lateral ou se o tamanho da tela for maior que 1920 pixels (desktop widescreen), aumentar o tamanho da fonte do texto para 15px.

Como verificar se determinado site tem o Design Responsivo?

Isso é simples. Abra esse site em qualquer navegador desktop e redimensione o navegador. Se o layout do site mudar conforme você redimensiona, o design é responsivo.

Primeiros passos com Design Responsivo

O primeiro passo para se começar a trabalhar com designs responsivos é fazer a transição de elementos fixos - de dimensões fixadas em px - para elementos fluídos - que se baseiam em 'porcentagens' e 'em'. Desta forma, a sua página consegue se expandir ou diminuir de acordo com a largura disponível, em vez de se fixar em algo similar a 960px de largura máxima.

A conversão de tamanhos fixos para fluídos é feita com base em uma fórmula bastante simples: "tamanho fixo"/ "contexto"= "tamanho fluído". Com ela você consegue converter dimensões em px para porcentagens (para larguras) ou em (para fontes), assim:

```
body {
/* Utilizando o tamanho de fonte padrão, geralmente 16px */
font: normal 100% Helvetica, Arial, sans-serif;
}

h1 {
font-size: 1.5em; /* 24px / 16px = 1.5em */
}
```

A princípio, esta definição terá o mesmo efeito que definir o tamanho da fonte do h1 como 24px.

Porém, em dispositivos que o tamanho padrão de fonte seja diferente, como em smartphones, font-size do elemento irá acompanhar esta mudança de tamanho: em um dispositivo que utilize 12px a fonte do elemento terá 18px de altura, e não 24px.

Esta mesma abordagem pode ser usada para definir as regras do seu grid - tornando-o um grid fluído – e o espaçamento entre elementos. Desta forma você consegue fazer os seus layouts acompanharem as diferenças de largura entre diversos dispositivos, e então você pode tratar as peculiaridades de cada contexto utilizando media queries, que é o que iremos ver agora.

Como funcionam as Media Queries?

Na minha opinião essa é a ferramenta mais poderosa quando se fala em Responsive Web Design, o Media Queries – que é uma regra específica para aplicar um bloco de CSS caso a regra seja atendida.

```
@media only screen
and (min-device-width : 320px)
and (max-device-width : 480px) {
/*
Esta regra será aplicada em
aparelhos com uma largura de 320px a 480px,
o que atende a maioria de smartphones.
*/
}
```

```
@media only screen
and (min-device-width : 768px)
and (max-device-width : 1024px)
and (orientation : landscape) {
/*
Além de verificar a largura do dispositivo,
você pode conferir a orientação, utilizando
"landscape" ou "portrait".
*/
```

Utilizando media queries, você pode definir estilos específicos para tamanhos diferentes, o que é melhor do que focar em um tipo de dispositivo específico. Diversos modelos diferentes de smartphones podem ser atendidos por um mesmo media query, o que também ajuda a atender novos modelos que irão surgir no futuro.

O site Media Queries (http://mediaqueri.es/) possui uma coletânea de sites que possuem diversos formatos para smartphones e tablets, uma ótima referência para se inspirar com outros trabalhos.

Bem, depois de estudarmos bastante sobre o universo Front-End, é hora de avançarmos para a parte 2 do nosso e-book e mergulharmos de forma ainda mais profunda no Back-End que é a peça que falta para completarmos nosso aprendizado como desenvolvedor Full-Stack.

Vamos lá?

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.

facebook

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 facil manuntenção.

Principais recursos

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

- 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).

- 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.

Configurando sua biblioteca AngularJS

Passo 1. Instale o Node.js® e o npm se eles ainda não estiverem na sua máquina.

Verifique se você está executando pelo menos node 6.9.x e npm 3.x.x executando node -v e npm -v em uma janela de terminal/console. Versões mais antigas produzem erros, mas versões mais recentes são boas.

Em seguida, instale o CLI Angular globalmente.

```
npm install -g @angular/cli
```

Etapa 2. Crie um novo projeto

Abra uma janela de terminal.

Gere um novo projeto e um aplicativo esqueleto executando os seguintes comandos:

```
ng new my-app
```

Paciência, por favor. Leva tempo para configurar um novo projeto; a maior parte é gasta instalando pacotes npm.

Etapa 3. Serve the application

Vá para o diretório do projeto e inicie o servidor.

```
cd my-app
ng serve --open
```

O comando ng serve inicia o servidor, observa seus arquivos e recria o aplicativo à medida que você faz alterações nesses arquivos.

Usando a opção --open(ou apenas -o) abrirá automaticamente o seu navegador http://localhost:4200/.

Passo 4: Edite seu primeiro componente Angular

A CLI criou o primeiro componente Angular para você. Este é o componente raiz e é nomeado app-root. Você pode encontrá-lo em ./src/app/app.component.ts.

Abra o arquivo do componente e altere a title propriedade de 'app' para 'Meu primeiro aplicativo Angular!'.

src/app/app.component.ts

```
export class AppComponent {
  title = 'Meu primeiro aplicativo Angular!';
}
```

O navegador é recarregado automaticamente com o título revisado. Isso é bom, mas poderia parecer melhor.

Abra src/app/app.component.csse dê ao componente algum estilo.

src / app / app.component.css

```
h1 {
  color: #369;
  font-family: Arial, Helvetica, sans-serif;
  font-size: 250%;
}
```

Welcome to Meu primeiro aplicativo Angular

A documentação completa juntamente com esse exemplo você encontra no:
https://angular.io/guide/quickstart

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.

facebook

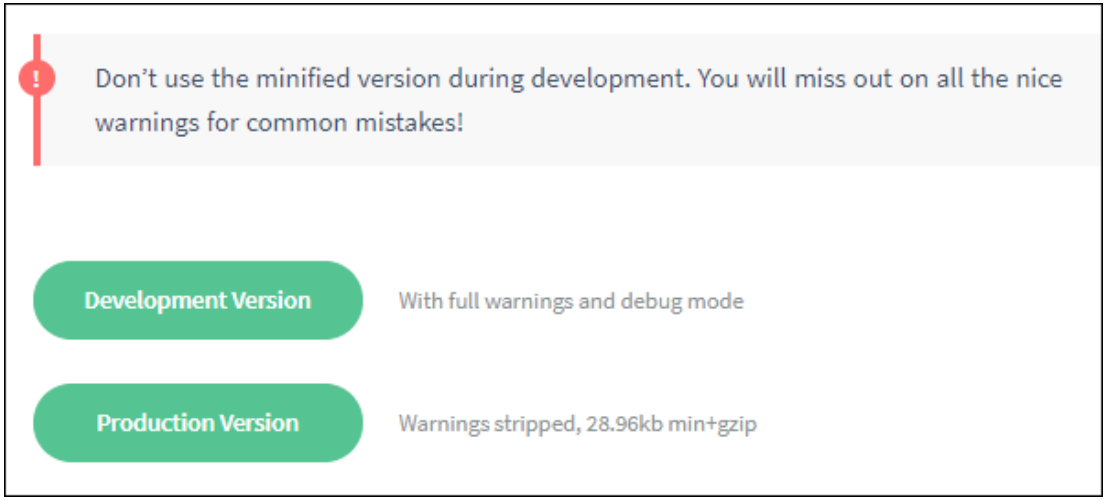
Configuração do ambiente

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.



Usando CDN

Também podemos começar a usar o arquivo VueJS da biblioteca CDN. O link https://unpkg.com/vue fornecerá a versão mais recente do VueJS. O VueJS também está disponível em jsDelivr (https://cdn.jsdelivr.net/npm/vue/dist/vue.js) e cdnjs (https://cdnjs.cloudflare.com/ajax/libs/vue/2.4.0/ vue.js).

Usando NPM

Para aplicações de grande escala com o VueJS, recomenda-se instalar usando o pacote npm. Ele vem com o Browserify e o Webpack, juntamente com outras ferramentas necessárias, que ajudam no desenvolvimento. A seguir, o comando para instalar usando o npm.

```
npm  install vue
```

facebook

Vamos agora para um exemplo simples de uma aplicação Vue.

```
<html>
  <head>
    <title>VueJs Introduction</title>
    <script type = "text/javascript" src = "js/vue.js"></script>
  </head>
  <body>
    <div id = "intro" style = "text-align:center;">
      <h1>{{ message }}</h1>
    </div>
    <script type = "text/javascript">
      var vue_det = new Vue({
        el: '#intro',
        data: {
          message: 'My first VueJS Task'
        }
      });
    </script>
  </body>
</html>
```

facebook

Saída

Este é o primeiro aplicativo que criamos usando o VueJS. Como visto no código da página anterior, incluímos o vue.js no início do arquivo .html.

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

Existe uma div que é adicionada no corpo que imprime "My first VueJS Task" no navegador.

```
<div id = "intro" style = "text-align:center;">
  <h1>{{ message }}</h1>
</div>
```

Nós também adicionamos uma mensagem em uma interpolação, ou seja, {{}} . Isso interage com o VueJS e imprime os dados no navegador. Para obter o valor da mensagem no DOM, estamos criando uma instância do vuejs da seguinte forma:


```
var vue_det = new Vue({
  el: '#intro',
  data: {
    message: 'My first VueJS Task'
  }
})
```

No trecho de código acima, estamos chamando a instância Vue, que pega o id do elemento DOM i.e. e1: '# intro', é o id do div. Há dados com a mensagem à qual é atribuído o valor 'My first VueJS Task' . O VueJS interage com o DOM e altera o valor no DOM {{message}} com 'My first VueJS Task' .

Depois dessa breve noção do Vue.js vamos para nossa última tecnologia desse e-book que é o React.

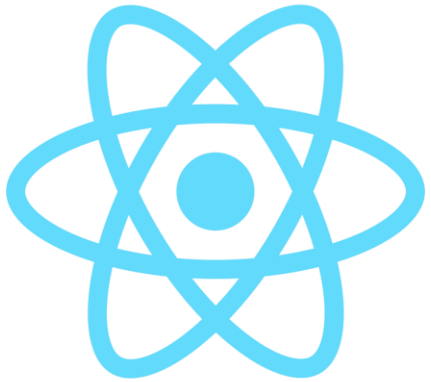
facebook

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.



Recursos do React

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

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.

facebook

Configuração do ambiente.

NodeJS e NPM

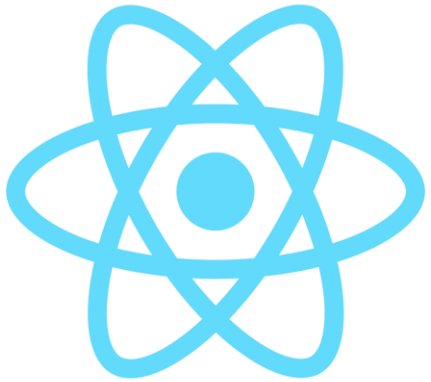
O NodeJS é a plataforma necessária para o desenvolvimento do ReactJS.

Etapa 1 - criar a pasta raiz

A pasta raiz será denominada reactApp e será colocada no Desktop. Depois que a pasta for criada, precisamos abri-la e criar o arquivo package.json vazio, executando npm init no prompt de comando e siga as instruções.

C:\Users\username\Desktop > mkdir reactApp

C:\Users\username\Desktop\reactApp > npm init



Etapa 2 - Instalar pacotes globais

Nós precisaremos instalar vários pacotes para esta configuração. Vamos precisar de alguns dos plugins do babel, então vamos primeiro instalar o babel executando o seguinte código na janela do prompt de comando.

C:\Users\username\Desktop\reactApp>npm install -g babel

C:\Users\username\Desktop\reactApp>npm install -g babel-cli

facebook

Etapa 3 - Adicionar dependências e plugins

Vamos usar o webpack bundler neste tutorial. Vamos instalar o webpack e o webpack-dev-server .

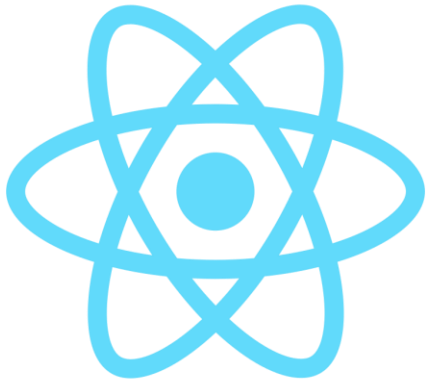
C:\Users\username\Desktop\reactApp > npm install webpack --save
C:\Users\username\Desktop\reactApp > npm install webpack-dev-server –save

Como queremos usar o React, precisamos instalá-lo primeiro. O comando --save adicionará esses pacotes ao arquivo package.json .

C:\Users\username\Desktop\reactApp>npm install react --save
C:\Users\username\Desktop\reactApp>npm install react-dom --save

Como já mencionado, precisaremos de alguns plugins do babel , então vamos instalar também.

C:\Users\username\Desktop\reactApp > npm install babel-core
C:\Users\username\Desktop\reactApp > npm install babel-loader
C:\Users\username\Desktop\reactApp > npm install babel-preset-react
C:\Users\username\Desktop\reactApp > npm install babel-preset-es2015



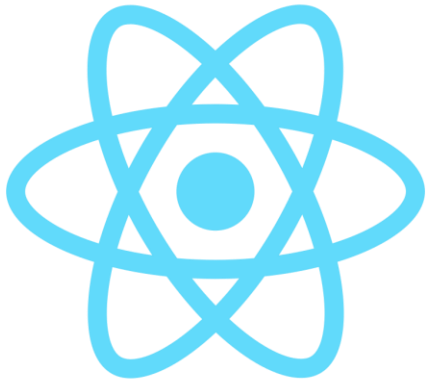
Etapa 4 - criar os arquivos

Vamos criar vários arquivos que precisamos. Pode ser adicionado manualmente ou usando o prompt de comando.

C:\Users\username\Desktop\reactApp > touch index.html
C:\Users\username\Desktop\reactApp > touch App.jsx
C:\Users\username\Desktop\reactApp > touch main.js
C:\Users\username\Desktop\reactApp > touch webpack.config.js

Forma alternativa para criar arquivos que precisamos

C:\Users\username\Desktop\reactApp>type nul > index.html
C:\Users\username\Desktop\reactApp>type nul > App.jsx
C:\Users\username\Desktop\reactApp>type nul > main.js
C:\Users\username\Desktop\reactApp>type nul > webpack.config.js



facebook

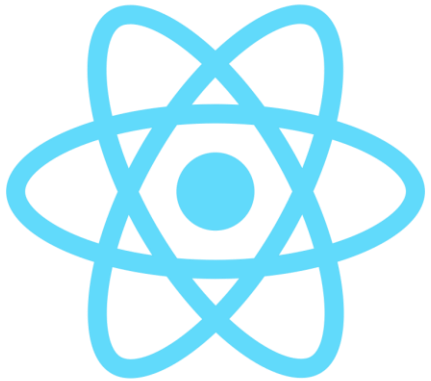
Etapa 5 - Definir Compilador, Servidor e Carregadores

Abra o arquivo webpack.config.js e adicione o seguinte código. Estamos criando ponto de entrada Webpack ser main.js. O caminho de saída é o local em que o aplicativo incluído será exibido. Também estamos configurando o servidor de desenvolvimento para a porta 8080 . Você pode escolher qualquer porta que quiser.

E, por último, estamos configurando os carregadores do babel para procurar por arquivos js, e usar es2015 e react as predefinições que instalamos antes.

webpack.config.js

```
var config = {
  entry: './main.js',
  output: {
    path:'/',
    filename: 'index.js',
  },
  devServer: {
    inline: true,
    port: 8080
  },
  module: {
    loaders: [
      {
```



```
test: /\.jsx?$/,
        exclude: /node_modules/,
        loader: 'babel-loader',
        query: {
          presets: ['es2015', 'react']
        }
      }
    ]
  }
}
module.exports = config;
```

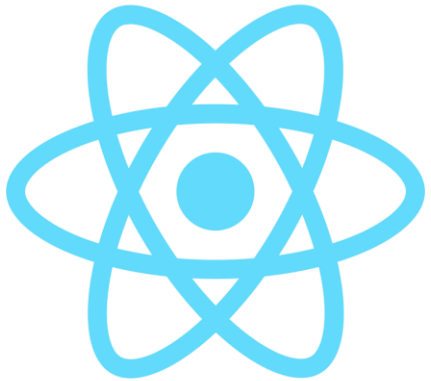
Abra o package.json e delete "test" "echo \" Erro: nenhum teste especificado \" "&& exit 1" dentro do objeto "scripts" . Estamos excluindo esta linha, pois não faremos nenhum teste neste tutorial. Vamos adicionar o comando start .

"start": "webpack-dev-server --hot"

Antes do passo acima, será necessário o webpack-dev-server . Para instalar o webpack-dev-server, use o seguinte comando.

C:\Users\username\Desktop\reactApp>npm install webpack-dev-server –g

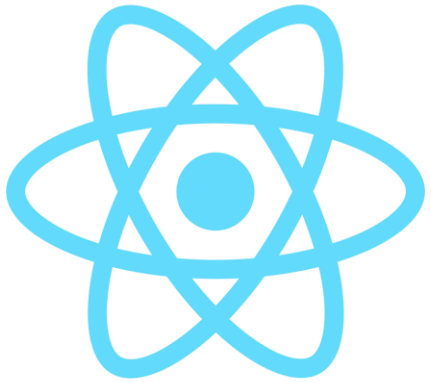
Agora, podemos usar o comando npm start para iniciar o servidor. O comando --hot irá adicionar o recarregamento ao vivo depois que algo for alterado dentro de nossos arquivos, portanto, não precisaremos atualizar o navegador toda vez que alterarmos nosso código.



```
<!DOCTYPE html>
<html lang = "en">

  <head>
    <meta charset = "UTF-8">
    <title>React App</title>
  </head>

  <body>
    <div id = "app"></div>
    <script src = "index.js"></script>
  </body>

</html>
```

Etapa 7 - App.jsx e main.js

Este é o primeiro componente React. Este componente irá renderizar o Hello World !!!



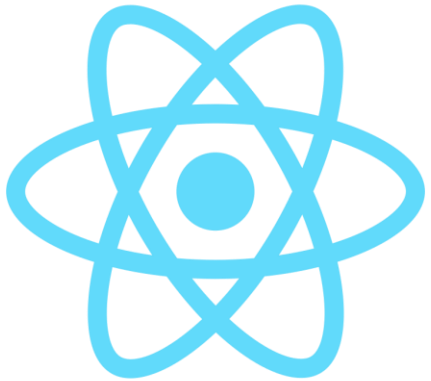
App.jsx

```
import React from 'react';

class App extends React.Component {
  render() {
    return (
      <div>
        Hello World!!!
      </div>
    );
  }
}
export default App;
```

Precisamos importar este componente e renderizá-lo em nosso elemento raiz App, para que possamos visualizá-lo no navegador.

main.js

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App.jsx';
```



ReactDOM.render(<App />, document.getElementById('app'));

Nota - Sempre que você quiser usar algo, você precisa importá-lo primeiro. Se você deseja tornar o componente utilizável em outras partes do aplicativo, é necessário exportá-lo após a criação e importá-lo no arquivo em que deseja usá-lo.

Etapa 8 - Executando o Servidor

A configuração está completa e podemos iniciar o servidor executando o seguinte comando.

C:\Users\username\Desktop\reactApp > npm start

Ele mostrará a porta que precisamos abrir no navegador. No nosso caso, é http://localhost:8080/ . Depois de abri-lo, veremos a seguinte saída.

PARTE 2

DESENVOLVEDOR BACK-END

MySQL

PHP 7

node.js

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.

facebook

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 Jedai</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.

Aprendendo a configurar seu ambiente PHP.

A capacidade de configurar seu próprio ambiente PHP é uma habilidade inestimável.

Não só permite que você sintonize sua configuração, mas também familiarize-se com a criação de extensões da fonte.

Abandone o Windows para o desenvolvimento – se o seu foco de trabalho principal for o Windows, instale o software de virtualização e execute uma máquina virtual Linux – O Windows case insensitivity, seus endereços de linha e todas as outras estranhezas incompatíveis com a maioria dos ambientes de servidor, apenas estão chamando por mais problemas, então é melhor desenvolver em um sistema que mais se parece com o ambiente em que você eventualmente está iniciando.

As máquinas virtuais também irão ajudar você a fazer testes – se algo der errado, você pode simplesmente limpar tudo e começar de novo ou fazer um retrocesso.

Você pode, literalmente, experimentar o tanto e da forma que você quiser, sem medo de mexer em qualquer coisa.

Fazer experimentos com suas próprias configurações também permitirá que você se familiarize com os diferentes servers out – quer usar o Apache do Nginx, não quer usar nenhum deles e ir com o Appserver , e assim por diante.

5 Frameworks PHP para você conhecer

Laravel

O Laravel é um dos framework PHP mais utilizados há algum tempo. Isto porque ele é bastante versátil e robusto, engloba uma série de ferramentas que já existiam antes de sua criação e que foram criadas pela comunidade em torno do PHP. O framework foi lançado em 2011 e, a cada ano, vem conquistando mais e mais popularidade.

Phalcon

O Framework PHP Phalcon iniciou sua jornada em 2012 e tem crescido de forma rápida no mercado de desenvolvimento web. Um dos seus diferenciais e que faz ele se destacar dos outros, é ter uma comunidade muito ativa, de forma que tem cada versão traduzida para dezenas de idiomas logo após serem lançadas.

Zend Framework

O Zend Framework nasceu em 2005, em uma época em que muitos Frameworks web estavam ficando famosos, como o Ruby on Rails. No caso, o Zend acabou alcançando fama rapidamente, já que o framework é apoiado por grandes empresas, como Google e o Microsoft.

CakePHP

Lançado em 2005, o CakePHP é um Framework que foca em simplificar o processo de desenvolvimento em PHP para programadores de qualquer nível. Possui um CRUD integrado ao banco de dados e Scaffold (sistema de organização de um projeto) muito intuitivo.

Symfony

O Framework Symfony foi lançado em 2005 e é um dos frameworks PHP que utilizam a arquitetura MVC, assim como o CakePHP e o Zend. Existem dois fatores muito interessantes sobre essa ferramenta:

- Foi criada para trabalhar em conjunto com metodologias ágeis de desenvolvimento. O que dá ainda mais ênfase para o desenvolvimento ágil, focado nas regras de negócio da aplicação e não em aplicações básicas.
- É um framework indicado para trabalhos mais pesados e de grande escala, já que a sua estrutura permite lidar com projetos deste porte.

facebook

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.

Definindo o banco de dados

Banco de dados relacional (*relational database*)

Um banco de dados relacional (*relational database*) é uma coleção de dados organizados em tabelas. Existem relações entre as tabelas. As tabelas são formalmente descritas. Elas consistem em linhas e colunas. SQL (Structured Query Language) é uma linguagem de banco de dados projetada para gerenciar dados em sistemas de gerenciamento de bancos de dados relacionais. Uma tabela é um conjunto de valores que é organizado usando um modelo de colunas verticais e linhas horizontais. As colunas são identificadas pelos seus nomes. Um esquema de um sistema de banco de dados é sua estrutura descrita em uma linguagem formal. Ele define as tabelas, os campos, relacionamentos, visualizações, índices, procedimentos, funções, filas, gatilhos e outros elementos.

Linha do banco de dados (*database row*)

Uma linha do banco de dados (*database row*) representa um único item de dados estruturado implicitamente em uma tabela. Também é chamado de tupla ou registro. Uma coluna é um conjunto de valores de dados de um tipo simples específico, um para cada linha da tabela. As colunas fornecem a estrutura de acordo com a qual as linhas são compostas. Um campo é um item único que existe na interseção entre uma linha e uma coluna. Uma chave primária identifica exclusivamente cada registro na tabela. Uma chave estrangeira é uma restrição referencial entre duas tabelas. A chave estrangeira (*foreign key*) identifica uma coluna ou um conjunto de colunas em uma tabela (referência) que se refere a uma coluna ou conjunto de colunas em outra tabela (referenciada).

Acionadores (*trigger*)

Um acionador (*trigger*) é um código processual que é executado automaticamente em resposta a determinados eventos em uma determinada tabela em um banco de dados. Uma *view* é um olhar específico dos dados em uma ou mais tabelas. Pode organizar dados em alguma ordem específica, destacar ou ocultar alguns dados. Uma *view* consiste em uma consulta armazenada acessível como uma tabela virtual composta do conjunto de resultados de uma consulta. Ao contrário das tabelas comuns, uma view não faz parte do esquema físico. É uma tabela dinâmica e virtual computada ou agrupada a partir de dados no banco de dados.

facebook

Transação (*transaction*)

Uma transação (*transaction*) é uma unidade atômica de operações de banco de dados em relação aos dados em um ou mais bancos de dados. Os efeitos de todas as instruções SQL em uma transação podem ser todos comprometidos com o banco de dados ou todos revertidos. Um conjunto de resultados SQL é um conjunto de linhas de um banco de dados, retornado pela instrução SELECT. Ele também contém meta-informações sobre a consulta, como os nomes das colunas e os tipos e tamanhos de cada coluna. Um índice é uma estrutura de dados que melhora a velocidade das operações de recuperação de dados em uma tabela de banco de dados.

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

Configuração do ambiente local.

Se você ainda estiver disposto a configurar seu ambiente para o Node.js, precisará dos dois softwares a seguir disponíveis em seu computador: (a) Editor de Texto e (b) Os binários instaláveis do Node.js.

Editor de texto

Isso será usado para digitar seu programa. Exemplos de alguns editores de texto: Sublime Text, Notepad, OS Edit command, Brief, Epsilon, EMACS e vim ou vi.

O nome e a versão do editor de texto podem variar em diferentes sistemas operacionais. Por exemplo, o Bloco de Notas será usado no Windows e o vim ou o vi poderão ser usados tanto no Windows quanto no Linux ou UNIX.

Os arquivos criados com o editor são chamados de arquivos de origem e contêm o código-fonte do programa. Os arquivos de origem dos programas Node.js geralmente são nomeados com a extensão " .js ".

Antes de iniciar sua programação, certifique-se de ter um editor de texto no lugar e você ter experiência suficiente para escrever um programa de computador, salvá-lo em um arquivo e, finalmente, executá-lo.

O Node.js Runtime

O código-fonte escrito no arquivo de origem é simplesmente javascript. O interpretador Node.js será usado para interpretar e executar seu código javascript.

A distribuição do Node.js é fornecida como um binário instalável para os sistemas operacionais SunOS, Linux, Mac OS X e Windows com as arquiteturas de processadores x86 de 32 bits (386) e 64 bits (amd64).

A seção a seguir o orienta sobre como instalar a distribuição binária do Node.js em vários sistemas operacionais.

Download do arquivo Node.js

Agora vamos baixar a versão mais recente do arquivo instalável do Node.js na sessão de downloads do site do Node.js. No momento que escrevo este e-book, seguem as versões disponíveis em diferentes sistemas operacionais.

OS	Nome do arquivo
Windows	node-v8.11.1-x64.msi
Linux	node-v8.11.1-linux-x86.tar.gz
Mac	node-v8.11.1-darwin-x86.tar.gz
SunOS	node-v8.11.1-sunos-x86.tar.gz

facebook

Instalação no UNIX / Linux / Mac OS X e SunOS

Com base em sua arquitetura de SO, faça o download e extraia o arquivo node-v8.11.1-osname.tar.gz para /tmp e, finalmente, mova os arquivos extraídos para o diretório /usr/local/nodejs. Por exemplo:

```
$ cd /tmp
$ wget http://nodejs.org/dist/v8.11.1/node-v8.11.1-linux-x64.tar.gz
$ tar xvfz node-v8.11.1-linux-x64.tar.gz
$ mkdir -p / usr / local / nodejs
$ mv node-v8.11.1-linux-x64 / * / usr / local / nodejs
```

Adicione /usr/local/nodejs/bin à variável de ambiente PATH.

OS	Output
Linux	export PATH=$PATH:/usr/local/nodejs/bin
Mac	export PATH=$PATH:/usr/local/nodejs/bin
FreeBSD	export PATH=$PATH:/usr/local/nodejs/bin

Instalação no Windows

Use o arquivo MSI e siga os prompts para instalar o Node.js.

Por padrão, o instalador usa a distribuição Node.js em C:\Arquivos de Programas\nodejs. O instalador deve configurar o diretório C:\Arquivos de Programas\nodejs\bin na variável de ambiente PATH da janela. Reinicie todos os prompts de comando abertos para que a alteração entre em vigor.

Verificar a instalação: Executando um arquivo

Crie um arquivo js chamado main.js em sua máquina (Windows ou Linux) com o seguinte código.

```
/* Olá, Mundo! Programando em node.js */
console.log("Olá, Mundo!")
```

Agora execute o arquivo main.js usando o interpretador Node.js para ver o resultado:

```
$ node main.js
```

Se tudo estiver bem com sua instalação, isso deve produzir o seguinte resultado:

```
Olá, Mundo!
```

Tudo certo? Node.js instalado, agora vamos para o próximo capitulo falar sobre outra tecnologia tão poderosa quanto o node, que é o AngularJS.

facebook

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!