5. PROGRAMAÇÃO FRONT-END: HTML, CSS, JAVASCRIPT

PROGRAMAÇÃO FRONT-END

Um programador front-end é alguém que implementa web designs por meio de linguagens de programação como HTML, CSS e JavaScript.

Eles trabalham com o design e as perspectivas do site. Enquanto os desenvolvedores de back-end programam o que acontece nos bastidores, como bancos de dados. Se você acessar qualquer site, poderá ver o trabalho de um desenvolvedor front-end na navegação, nos layouts e também na aparência de um site diferente do seu telefone.

Os princípios de programação front-end

1. HTML/CSS

HyperText Markup Language (HTML) é a linguagem de marcação padrão usada para criar páginas da web. É o bloco de construção mais básico necessário para o desenvolvimento de sites. CSS (Cascading Style Sheets) é a linguagem usada para apresentar o documento que você cria com HTML.

HTML é usado para criar a base para sua página. Considerando que, CSS é usado para criar o layout da página, cor, fontes e estilo. Ambas as linguagens são absolutamente essenciais para se tornar um desenvolvedor front-end.

Os princípios de programação front-end

2. JavaScript/jQuery

Outra ferramenta importante para um desenvolvedor front-end é o JavaScript (JS). Se você está tentando implementar recursos interativos em seu site, como áudio e vídeo, jogos, habilidades de rolagem, animações de página, JS é a ferramenta que você precisa.

JavaScript consiste em bibliotecas como jQuery. É uma coleção de plugins e extensões que tornam mais rápido e fácil usar o JS em seu site. jQuery pega tarefas comuns que requerem várias linhas de código JS e as compacta em um formato que pode ser executado com uma única linha.

Os princípios de programação front-end

3. Estruturas

Estruturas CSS e JavaScript são coleções de arquivos CSS ou JS que executam tarefas diferentes, fornecendo funcionalidades comuns. Em vez de começar com um documento de texto vazio, você começa com um arquivo de código que já tem muito JavaScript presente nele.

Os frameworks têm seus pontos fortes e fracos, o que torna importante escolher o melhor framework para o tipo de site que você está construindo. Por exemplo, algumas estruturas JS são ótimas para criar interfaces de usuário complexas, enquanto outras são excelentes para exibir todo o conteúdo do seu site.

Os princípios de programação front-end

4. Design Responsivo

Usamos diferentes gadgets, como computadores, telefones e tablets, para visualizar páginas da web. As páginas da web se ajustam ao dispositivo que você está usando sem nenhum esforço extra de sua parte. Isso se deve ao design responsivo . Uma das principais funções de um desenvolvedor front-end é entender os princípios de design responsivo e como implementá-los no lado da codificação.

É uma parte intrínseca de frameworks CSS como o Bootstrap . Essas funções estão todas interconectadas e, à medida que você aprende uma, muitas vezes estará progredindo nas outras ao mesmo tempo.

Os princípios de programação front-end

5. Controle de Versão/Git

O controle de versão é o processo de rastreamento e controle de alterações em seu código-fonte para que você não precise começar do início se algo der errado. É uma ferramenta que você pode usar para rastrear as alterações feitas anteriormente para que você possa voltar a uma versão anterior do seu trabalho e descobrir o que deu errado sem destruir tudo.

Os princípios de programação front-end

6. Teste/Depuração

O teste é uma parte importante de qualquer projeto para manter os bugs afastados. Assim, um desenvolvedor front-end deve possuir a habilidade e capacidade de testar e depurar códigos. Existem diferentes métodos de teste para desenvolvimento web. O teste funcional analisa uma parte específica da funcionalidade do seu site e garante que ela faça tudo de acordo com o código. O teste de unidade é outro método que testa o menor pedaço de código e o examina individualmente para operação correta. O teste é uma grande parte do processo de desenvolvimento front-end e existem estruturas para ajudá-lo. Programas como Mocha e Jasmine são projetados para acelerar e simplificar seu processo de teste.

Os princípios de programação front-end

7. Ferramentas do desenvolvedor do navegador

Os navegadores da Web modernos vêm equipados com ferramentas de desenvolvedor para teste e depuração. Essas ferramentas permitem testar as páginas da web no próprio navegador e descobrir como a página está interpretando o código.

Os princípios de programação front-end

7. Ferramentas do desenvolvedor do navegador

As ferramentas do desenvolvedor do navegador geralmente consistem em um inspetor e um console JavaScript. O inspetor permite que você veja a aparência do HTML de tempo de execução em sua página, qual CSS está associado a cada elemento na página e também permite que você edite seu HTML e CSS e veja as alterações ao vivo à medida que ocorrem. O console JS permite visualizar quaisquer erros que ocorram enquanto o navegador tenta executar seu código JS.

Os princípios de programação front-end

8. Desempenho na Web

É importante certificar-se de que seu site funciona sem problemas, sem qualquer falha. O desempenho da Web define o tempo que leva para o site carregar. Se você estiver tendo problemas com tempos de desempenho, há etapas que você pode seguir para melhorá-los, como otimizar imagens e reduzir CSS e JavaScript.

Programas como Grunt e gulp podem ser usados para automatizar a otimização de imagens, minificação de CSS e JS e outras tarefas de desempenho da web. Isso ajuda a tornar seu site mais eficiente.

Os princípios de programação front-end

9. Pré-processamento de CSS

CSS Preprocessor é uma versão avançada do CSS. Isso é usado para aprimorar a classe primária de CSS para criar versões melhores de sites. Não é apenas uma linguagem para melhorar os elementos de estilo, mas ajuda os desenvolvedores a pular tarefas como escrever seletores CSS e strings de cores com frequência.

Existem três tipos de pré-processadores disponíveis, como Sass, LESS e Stylus . Você deve escrever um código que ajude o pré-processador e, por sua vez, converta em CSS que funcionará para o site.

CLEAN CODE

CLEAN CODE

Quem é programador certamente já se deparou com um código ruim, mal feito ou um título que não demonstrava efetivamente sua função. Esse cenário é mais comum do que se imagina e é justamente o que o Clean Code busca combater.

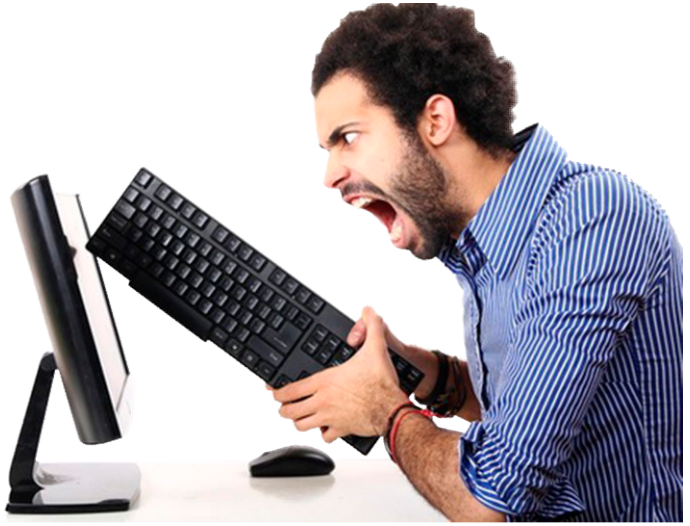
Clean Code, ou Código Limpo, é uma filosofia de desenvolvimento de softwares que consiste em aplicar técnicas simples que facilitam a escrita e a leitura de um código. Tornando-o, assim, de fácil compreensão.

Quando surgiu o termo Clean Code?

As técnicas do Clean Code apareceram pela primeira vez no livro “Clean Code: A Handbook of Agile Software Craftsmanship“, lançado em 2008. Ele foi escrito por Robert Cecil Martin, conhecido na comunidade como Uncle Bob. O autor atua na área de desenvolvimento desde 1970 e é um dos profissionais por trás do Manifesto Ágil, lançado em 2001.

Quando surgiu o termo Clean Code?

Com seus longos anos de experiência, ele conseguiu perceber que o gargalo principal no desenvolvimento de software estava justamente na manutenção. Ou seja, um código mal escrito desde a sua primeira versão pode funcionar mas vai gerar prejuízos enormes.



Quando surgiu o termo Clean Code?

O termo se disseminou pela comunidade de uma forma praticamente unânime. E um dos motivos pode ser justificado no seguinte dado: a proporção média de leitura e escrita de códigos fonte é de 10 para 1. Isso significa que as pessoas passam mais tempo tentando entender os códigos existentes do que efetivamente escrevendo novos códigos.

Quando surgiu o termo Clean Code?

Com essa informação, é possível perceber que, muito além de uma boa arquitetura e boas práticas computacionais, o código fonte – que é aquilo que faz o computador executar nossos comandos – precisa de princípios. E é justamente isso que o **Clean Code** traz.

Para que serve o Clean Code?

Um dos principais erros que os programadores cometem é acreditar que, uma vez que o código está pronto e funcionando, não precisa mais de revisão.

Porém, um sistema nunca está totalmente finalizado, sempre existe a necessidade de atualizações e novas funcionalidades. Além disso, o código envelhece e pode se tornar obsoleto. É nesse cenário que o Clean Code se encaixa.

Para que serve o Clean Code?

A ideia por trás dos seus princípios é tornar o desenvolvimento e a manutenção cada vez mais simples. Afinal, um código sendo remendado por muito tempo torna-se impossível de manter, fazendo com que seja mais vantajoso iniciar do zero do que prosseguir em uma versão ruim.

Logo, o código limpo evita gastos desnecessários com manutenção e torna o software preparado para novas atualizações e melhorias.

Conheça as 7 principais regras do Clean Code

1 – Nomes são muito importantes

A definição de nome é essencial para o bom entendimento de um código. Aqui, não importa o tipo de nome, seja ele:

Variável;
Função;
Parâmetro;
Classe;
Método.

Conheça as 7 principais regras do Clean Code

Ao definir um nome, é preciso ter em mente 2 pontos principais:

Ele deve ser preciso e passar logo de cara sua ideia central. Ou seja, deve ir direto ao ponto;

Não se deve ter medo de nomes grandes. Se a sua função ou parâmetro precisa de um nome extenso para demonstrar o que realmente representa, é o que deve ser feito.

Conheça as 7 principais regras do Clean Code

2 – Regra do escoteiro

Há um princípio do escotismo que diz que, uma vez que você sai da área em que está acampando, você deve deixá-la mais limpa do que quando a encontrou.

Trazendo a regra para o mundo da programação, a regra significa deixar o código mais limpo do que estava antes de mexer nele.

Conheça as 7 principais regras do Clean Code

3 – Seja o verdadeiro autor do código

O ser humano é acostumado a pensar de forma narrativa, portanto, o código funciona da mesma forma.

Logo, ele é uma história e, como os programadores são seus autores, precisam se preocupar na maneira com que ela será contada.

Conheça as 7 principais regras do Clean Code

3 – Seja o verdadeiro autor do código

Em resumo, para estruturar um código limpo, é necessário criar funções simples, claras e pequenas. Existem 2 regras para criar a narrativa via código:

- As funções precisam ser pequenas;
- Elas têm de ser ainda menores.

Não confunda com os termos “nome” e “função”. Como dissemos no primeiro princípio, nomes grandes não são um problema. Já as funções precisam ser as menores possíveis.

Conheça as 7 principais regras do Clean Code

4 – DRY (Don’t Repeat Yourself)

Esse princípio pode ser traduzido como “não repita a si mesmo”. Essa expressão foi descrita pela primeira vez em um livro chamado The Pragmatic Programmer e se aplica a diversas áreas de desenvolvimento, como:

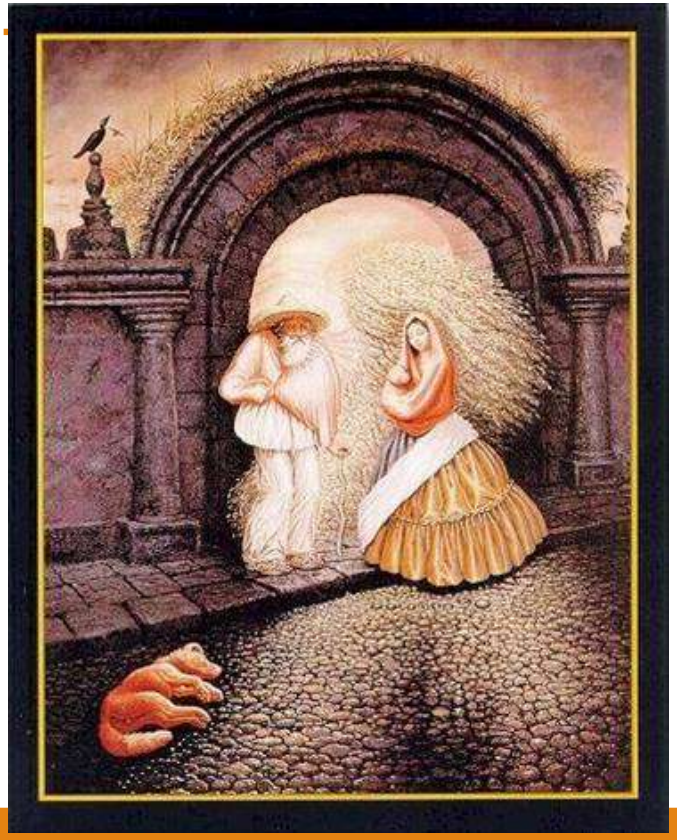
- Banco de Dados;
- Testes;
- Documentação;
- Codificação.

Conheça as 7 principais regras do Clean Code

4 – DRY (Don’t Repeat Yourself)

O DRY diz que cada pedaço do conhecimento de um sistema deve ter uma representação única e ser totalmente livre de ambiguidades.

Em outras palavras, define que não pode existir duas partes do programa que desempenhem a mesma função.



Conheça as 7 principais regras do Clean Code

5 – Comente apenas o necessário

Esse princípio afirma que comentários podem ser feitos, porém, se forem realmente necessários. Segundo Uncle Bob, os comentários mentem. E isso tem uma explicação lógica.

O que ocorre é que, enquanto os códigos são constantemente modificados, os comentários não. Eles são esquecidos e, portanto, deixam de retratar a funcionalidade real dos códigos.

Logo, se for para comentar, que seja somente o necessário e que seja revisado juntamente com o código que o acompanha.

Conheça as 7 principais regras do Clean Code

6 – Tratamento de erros

Tem uma frase do autor Michael Feathers, muito conhecido na área de desenvolvimento, que diz que as coisas podem dar errado, mas, quando isso ocorre, os programadores são os responsáveis por garantir que o código continuará fazendo o que precisa.

Ou seja: saber tratar as exceções de forma correta é um grande e importante passo para um programador em desenvolvimento.

Conheça as 7 principais regras do Clean Code

7 – Testes limpos

Testar, na área de programação, é uma etapa muito importante. Afinal, um código só é considerado limpo após ser validado através de testes – que também devem ser limpos. Por isso, ele deve seguir algumas regras, como:

Fast: O teste deve ser rápido, permitindo que seja realizado várias vezes e a todo momento;

Independent: Ele deve ser independente, a fim de evitar que cause efeito cascata quando da ocorrência de uma falha – o que dificulta a análise dos problemas;

Conheça as 7 principais regras do Clean Code

7 – Testes limpos

Repeatable: Deve permitir a repetição do teste diversas vezes e em ambientes diferentes;

Self-Validation: Os testes bem escritos retornam com as respostas true ou false, justamente para que a falha não seja subjetiva;

Timely: Os testes devem seguir à risca o critério de pontualidade. Além disso, o ideal é que sejam escritos antes do próprio código, pois evita que ele fique complexo demais para ser testado.

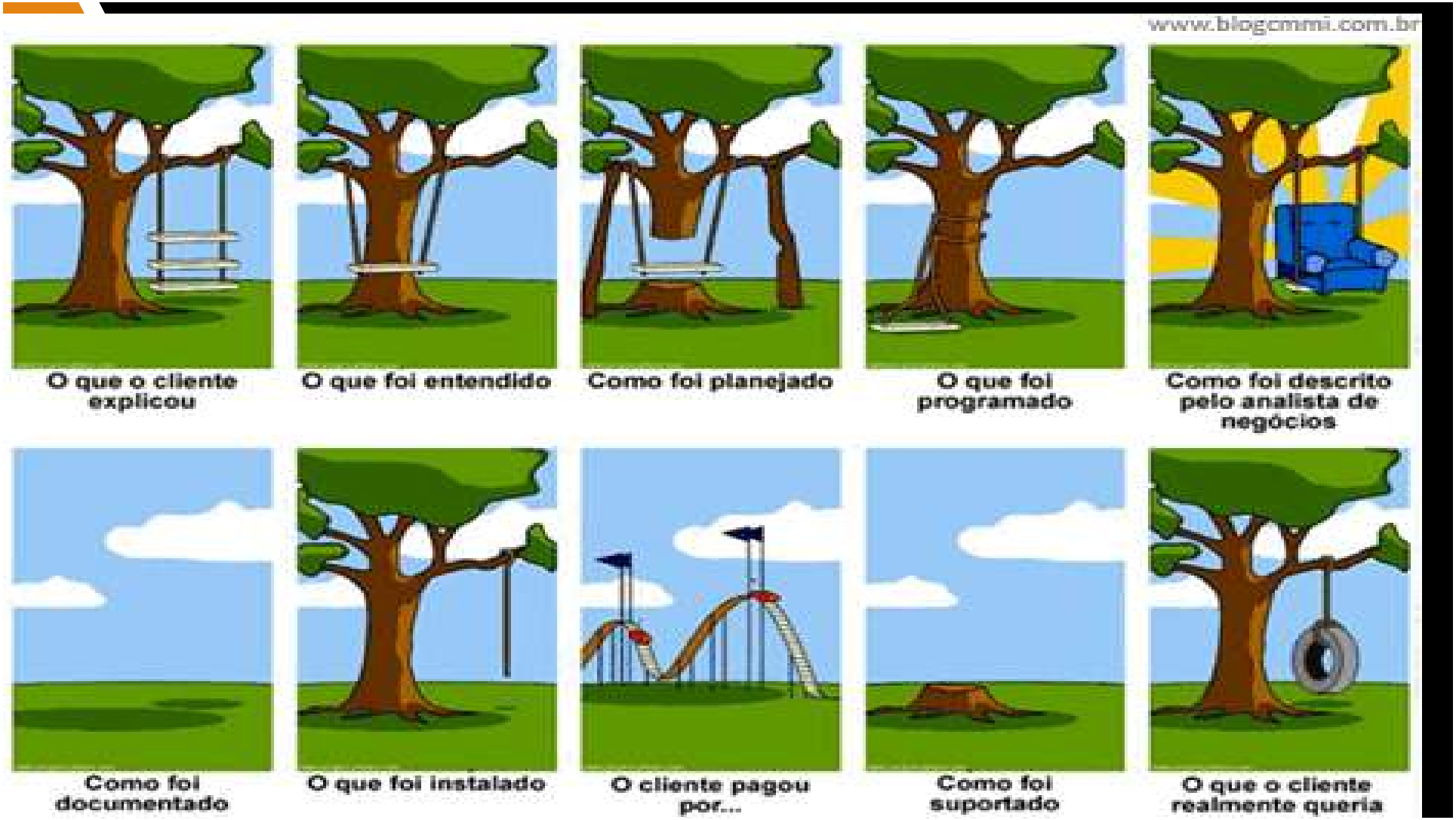
A Realidade é Dura

A Realidade é a que Fazemos

CONCLUSÃO

O Clean Code é um conceito que veio para ficar. Afinal, seus princípios solucionam com eficácia um dos principais problemas que grande parte dos projetos de sistemas enfrentam: a manutenção.

https://www.coquinhos.com/aprender-a-programar-um-robo/play/


www.blogcmmi.com.br
O que o cliente explicou
O que foi entendido
Como foi planejado
O que foi programado
Como foi descrito pelo analista de negócios
Como foi documentado
O que foi instalado
O cliente pagou por...
Como foi suportado
O que o cliente realmente queria