Componentes no Angular

Todas as vezes que você precisa implementar um slider ou elementos de um framework que você gosta, invariavelmente você precisa copiar um grande pedaço de código HTML, CSS e Javascript para depois aplicar em seu projeto. Todos estes códigos estão sempre separados e sofrem, principalmente, efeitos do código externo do seu projeto ou de outros plugins. Muitas vezes você deve ter tido algum problema de conflito entre códigos do plugin e do seu projeto. Isso gera retrabalho e outros problemas.

Mesmo se você cria seus próprios plugins ou frameworks de CSS, você ainda precisa tomar uma série de cuidados para que o código não sofra nenhum conflito e que seja o mais flexível e reutilizável possível. Mas acredite, mesmo que você faça um trabalho perfeito, alguma coisa sempre foge do controle. Os Web Componentes podem ajudar nesses problemas e em muitos outros.

Web Components é uma suíte de diferentes tecnologias que permite a criação de elementos customizados reutilizáveis — com a funcionalidade separada do resto do seu código — e que podem ser utilizados em suas aplicações web.

Vantagens de Usar Componentes

- Reuso de código
- Agilidade do desenvolvimento
- Melhor legibilidade do código
- Aplicação web mais rápidas e fluidas
- Fácil manutenção

Quando usar Componentes?

A verdade é que não precisamos criar componentes para tudo. E também devemos criá-los sempre quando necessário melhores condições possíveis que um componente pode ser implementado.

Às vezes, é melhor ter um código duplicado do que uma estrutura de componente complexa, onde o esforço de deixá-lo na sua melhor forma poderia levar bastante tempo e ter um impacto alto no seu projeto e/ou negócio. Não podemos esquecer que, usualmente, componentes precisarão receber parâmetros de uso das classes, bem como, suas instâncias de serviços que, possivelmente, serão injetadas para consumo dos serviços. Imagine uma situação em que o componente só está sendo chamado em dois pontos

únicos na aplicação. Assim, os ajustes para performar um único componente não fazem sentido. Então, se pensarmos bem, será que faria sentido criar mais um componente ou ele servirá somente como enfeite de código?
Por outro lado, quando a utilização de um novo componente for necessária em mais de cinco ou mais pontos da aplicação, a sua criação será inevitável, ele fará que você poupe um bom tempo no desenvolvimento e na manutenção da aplicação. A sua criação também vai ter sentido quando o componente estiver inserido em um projeto com uma arquitetura bem definida nos quais poderão ser construídos com baixo encapsulamento e com um alto nível de abstração. Pensando neste contexto, será que faria sentido a criação de um novo componente ou ele seria mais um enfeite de código.

Componentes do Angular

Componentes no Angular são divididos em 3 partes essenciais: Template HTML, folha de estilo CSS e um arquivo Typescript com toda a lógica do componente. O HTML e o CSS podem ser tanto em arquivos separados e o arquivo TypeScript receber o caminho do arquivo HTML e CSS quanto podem ser escritos dentro do arquivo TypeScript. O primeiro exemplo de componente Angular que encontramos em um projeto recém criado, é o componente **app** encontrado na pasta **src/app/**
Além do componente, temos também um **módulo** (module). Um módulo no Angular é onde você agrupa as dependências, diretivas, pipes, entre outras coisas para formar a aplicação. Por enquanto, falaremos dos componentes.
Os componentes possuem a estrutura de nomenclatura: **nome_do_componente.**component.**extensão_do_arquivo.** Como pode ser observado na imagem acima. Todos os tipos de dados do Angular: o nome + o tipo de dado + extensão do arquivo, facilitando a compressão de cada estrutura de dados dentro dele.

Arquivo *app.componente.ts*

Sem dúvida esse é um dos arquivos importantes de um componente. É dentro dele em que toda lógica do componente é guardada. Esse arquivo é dividido entre o decorator **Component**, que guarda informações como o seletor do componente, url do arquivo HTML e urls de arquivos CSS, além de também poder receber o HTML e CSS dentro do próprio decorator. Além disso, também há uma classe que é exportada, que é o componente em si. A nomenclatura da classe do componente é **nome do componente + Component**.

Arquivo *app.component.html*

Nesse arquivo você encontra todo o template HTML. Ao contrário do arquivo *index.html* os templates de componentes do Angular não precisam ter toda a estrutura básica do HTML, apenas o conteúdo necessário para o componente.

Arquivo *app.component.css*

Esse arquivo é o responsável pela estilização do componente. Por padrão, esse arquivo vem vazio no componente app. No entanto, você pode adicionar código CSS sem nenhum problema.

Seletores

A propriedade *selector* de um componente funciona de uma maneira diferente. Essa propriedade recebe um tipo de seletor do CSS (seletor de elemento, classe, id, etc) que identificará o componente.
Ao usar somente o nome do componente no seletor, você estará usando um seletor de tag. Então, para acessar esse componente, coloque no HTML adicione uma tag com o nome que foi colocado na propriedade seletor. Se entre o nome do componente, for adicionado colchetes ([ ]), você estará usando um seletor de atributo. Então, para acessar o componente, use alguma tag original do HTML, como uma div e dentro dela, adicione um atributo com o nome que foi colocado. Se você usar o ponto (.) ou o jogo da velha (#) antes do nome componente, você estará usando um seletor de classe e de id, respectivamente. E para acessar o componente, adicione uma tag html original, como uma div, que terá a classe ou id respectiva do componente.

IMPORTANTE: Os componentes devem ser importados e registrados em **app.module.ts**, na propriedade *declarations* para poderem ser reconhecidos e usados na aplicação.

Gerar Componentes rapidamente

As vezes, o processo de criação de um componente de maneira manual pode ser um pouco demorado e chato. Por causa disso, o Angular CLI possui o comando ***generante***. Ele não só gera componentes, como também gera outras estruturas do Angular, como módulos, pipes, entre outros. Isso agiliza ainda mais o processo de criação e a dinamização da criação do projeto.

Para criarmos nosso componente pelo Angular CLI, use o comando ***ng generate component <nome-do-componente>***. Agora, indo na pasta **app/** você verá uma nova pasta com o nome do componente criado e com os arquivos gerados automaticamente.