### Aula 17: Removendo elementos

Nesta lição, vamos fazer alguns ajustes na funcionalidade atual da nossa loja.

**Mudança no tamanho do contêiner**

No momento, nosso **contêiner div** não redimensiona adequadamente para se ajustar a todos os nossos elementos. Precisamos corrigir isso alterando um pouco o estilo. Primeiro, vamos remover a propriedade de **altura** no arquivo **style.css** da **regra do contêiner**.

#container {

    background-color: #FFF;

    padding: 12px;

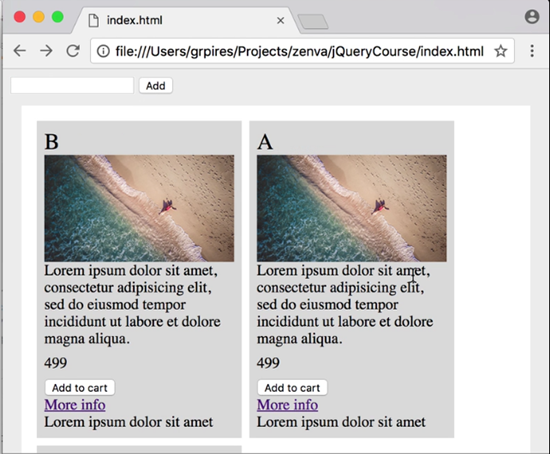
    margin: 12px;

}

Em seguida, vamos para o arquivo**index.html** e adicionar a seguinte linha abaixo da **tag de fechamento** do nosso **contêiner**.

<br style="clear: left;" />

Com essa linha, ajustamos os floats e a página para que o contêiner possa redimensionar corretamente para ajustar todos os nossos itens.



**Removendo Itens**

Na próxima parte, queremos adicionar uma função para **remover itens**. Abaixo do nosso botão “Add to Cart” em **index.html**, vamos adicionar um novo botão para essa função de remoção. Também queremos adicionar **classes** tanto para os botões “Remove” quanto para “Add to Cart”.

<button class="item-add">Add to cart</button>

<button class="item-remove">Remove</button>

Como alteramos a estrutura do HTML do **contêiner do item**, também precisamos garantir que alteramos a função de adicionar item em **main.js** para corresponder à nossa nova configuração HTML.

html += '<button class="item-add">Add to cart</button>';

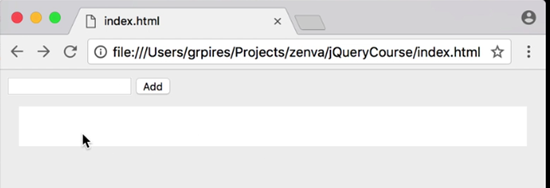
html += '<button class="item-remove">Remove</button>';

Agora podemos criar uma **nova função** no **documento pronto** para remover itens. No entanto, começaremos demonstrando a forma ***incorreta*** de fazer isso. Com a **classe “item-remove**” como **seletor**, podemos facilmente configurar um **ouvinte de evento de clique** nele. Como estamos no elemento botão, podemos usar **.parent** para subir para o div contêiner do **item na hierarquia**. Com **.remove**, removeremos o contêiner para eliminar o contêiner e todos os elementos dentro dele.

$('#container .item-remove').on('click', function() {

    $(this).parent().remove();

});



**Parece estar funcionando, então por que esse método está errado?** Se você tentar adicionar um novo item agora e imediatamente tentar remover esse novo item, você perceberá que o botão de remoção não funciona. Isso ocorre devido ao processo de carregamento da página. Quando a página é carregada, ela chama imediatamente a função que acabamos de criar e encontra todos os itens do **seletor** aos quais ela precisa ouvir. Como adicionamos novos itens após o carregamento da página, **a função nunca é chamada neles**. Em última análise, nossos novos botões de remoção não estão sendo ouvidos, pois eles nunca foram selecionados pelo seletor em primeiro lugar!

É por isso que a **delegação** **de eventos** é tão importante. Ao usar o **contêiner** como nosso **seletor** e a **classe “item-remove”** como o **delegado**, podemos corrigir isso. Dessa forma, quando a página é carregada, o ouvinte de eventos é configurado no contêiner. Todos os itens adicionados dentro do contêiner poderão ser ouvidos, pois o código instrui a página a ficar atenta a eles a qualquer momento.

$('#container').on('click', '.item-remove', function() {

    $(this).parent().remove();

});