Imersão Dev - Aula 4

Rafaella: Olá, Devs! Muito bom dia, boa tarde e boa noite. Depende do horário que você estiver vendo esse vídeo. Eu gostaria de te dar as boas-vindas ao nosso quarto dia de **Imersão Dev**.

Quanta coisa já vimos até agora, não é? Dá para imaginar a quantidade de conteúdo que você teve até agora? Em tão pouco tempo. E é por isso mesmo que você precisa aproveitar cada minuto dessa Imersão Dev. Se você tem dúvida, nos procure no Discord. Se você quiser aumentar o seu portfólio, faça os projetos que propomos e publique nas redes sociais.

Hoje teremos mais um projeto totalmente novo e com conceitos importantíssimos para qualquer desenvolvedor ou desenvolvedora que deseja entrar na área. Então pega o café, abra o CodePen e vamos para a quarta aula!

Paulo: Olá, estamos aqui nessa quarta aula da Imersão Dev. Continuamos nessa jornada, no mergulho, que você tá fazendo na carreira de desenvolvimento, na carreira de Dev. É muito legal ver os projetos que vocês estão construindo e ver você participar desse nosso universo que chamamos carinhosamente de **Aluraverso**.

Vou até aproveitar o começo dessa quarta aula para te deixar ainda mais próximo da nossa escola. Para você entender quem somos, como nós trabalhamos, como é muito além do Discord que você está participando, muito além do CodePen que estamos usando nessas aulas. Onde nós estamos.

Vou pedir para o Lucas colocar na tela canais do YouTube das pessoas que estão sempre muito próximas da Alura. Um dos canais é da própria Rafaella. Se você não conhece deve conhecer, muita gente veio pelo canal da Rafaella.

Vamos deixar aqui para você se inscrever. Tem o canal do Mario Souto, tem o canal do Guilherme Silveira, que é meu irmão e cofundador da Alura. Tem diversos professores e professoras, tem o Instagram do Guilherme Lima que está aqui com vocês.

Vamos deixar esses canais para vocês se inscreverem, assistirem, tem uns que são mais focados em assuntos de front-end, que é o HTML, CSS e JavaScript tem gente que vai falar mais de Python, tem gente vai falar mais de Ciência de Dados, gente vai falar mais de Inteligência Artificial, etc.

Essas pessoas fazem parte do universo da Alura e estão conosco, dentro da escola, dando aula e participando.

Talvez outros de vocês tenham chegado aqui por causa dos creators que trabalham muito próximo de nós e que eu conheço pessoalmente. Tem muita gente que vem por causa do Leon e da Nilce lá do Coisa de Nerd, eu sei, vocês fazem um barulho muito

grande. Tem um pessoal que veio por causa do Jovem Nerd, tem do Gaveta, do Manual do Mundo, do Ciência Todo Dia, do Átila, tem da Camila do Peixe Babel, entre outros.

Com alguns desses canais temos muita proximidade e escrevemos pautas de tecnologia junto com essas pessoas. Porque queremos realmente fomentar essa comunidade, fomentar esse universo que vai muito além da Alura, é muito mais amplo. É a comunidade de tecnologia do Brasil. Sabemos que isso pode mudar o cenário da sua carreira, de empregabilidade, de vagas de emprego que estão por aí e mudar a vida das pessoas.

De novo, nenhuma promessa, esse é o começo da sua carreira em Dev. Você está estudando com gente que é advogado, gente que é médico, tem músicos, tem engenheiros civis, tem gente que ainda está no ensino médio. Esses são seus primeiros passos. Sei que é muito animador, mas essa jornada é longa.

Esse é o primeiro mergulho, temos uma maratona, uma travessia enorme a ser feita. Então fica o convite para você se inscrever nesses canais, participar do nosso universo e entender melhor onde a Alura está e entender que a Alura é uma escola que vai além dos muros, eu sempre falo bastante disso. A escola tem que ir além do que é o nosso site, a nossa página da Imersão Dev, o nosso Discord.

Queremos estar presentes em todo lugar. Por isso, falamos para você postar no Instagram, postar no Linkedin, no Twitch. Nós queremos estar onde as pessoas estão aprendendo e isso é muito forte. Espero que você já esteja enxergando isso que realmente acontece aqui Alura.

Existem outras escolas que também têm uma abordagem assim, mas realmente enxergamos isso como o grande potencial de fazer a diferença no seu processo de aprendizado. Agora vou passar a palavra aos universitários, a quem realmente dá as cartas aqui: Rafaella e Guilherme. Para encararmos mais uma aula. Estou animado, porque sei que vai ter imagens, já sei qual é o exercício e o projeto, e gosto muito desse projeto.

Guilherme: Eu também estou super empolgado com esse projeto. E o Paulo já deu a dica, hoje vamos trabalhar com imagens. Nós já sabemos como trabalhar com `if` e com `else`, como trabalhar com variáveis, como passar informações do JavaScript para o HTML e também como pegar as informações do HTML; já vimos `parseFloat()`, `Math.random()` e um monte de funções que a própria biblioteca fornece.

Hoje vamos um pouco mais além. Estudaremos um conteúdo que é extremamente importante para nossa área de desenvolvimento, que é a questão das condicionais, a questão das interações. Vamos aprender isso nesta aula. Rafa, o que veremos hoje?

Rafaella: Acho que vou dar um spoiler do projeto. Faremos o Aluraflix. Uma página onde você poderá listar alguns filmes que você gosta e talvez queira indicar para alguém, ou fazer uma lista de desejos com filmes que você tem vontade de assistir.

Para fazer isso trabalharemos com listas, vamos listar bastante coisa, e vamos trabalhar com as estruturas de repetição que o Gui tinha comentado. Nesta aula, não vamos direto para o projeto, primeiro vamos entender como essas listas funcionam para depois podermos ir para o projeto de fato, aquele que já trazemos com HTML e CSS pronto.

Primeiro vamos criar uma nova pen, basta clicar em "pen" no menu lateral esquerdo. Esse não vamos precisar dar *fork* em nada, é mais para entender como vai funcionar. Então vamos lá. Nós queremos listar alguns filmes. Para isso eu queria saber, Gui e Paulo, quais são os filmes que vocês mais gostam? Pode começar pelo Gui.

Guilherme: Tem um filme que assisti recentemente e gostei muito. Um filme chamado "Yesterday". Achei bem bacana. Acontece uma coisa sinistra no mundo e só uma pessoa lembra dos Beatles. Só ele, ninguém mais lembra. Então ele começa a fazer sucesso com as músicas. O filme é uma comédia bem legal.

Rafaella: E você, Paulo? Qual filme você gosta?

Paulo: Eu vou colocar um que tem todo o contexto aqui da Alura. Ele tem ficção científica, alienígenas, línguas e ciência. É o filme "A Chegada", que foi lançado em 2016.

Rafaella: Eu também já estou pensando qual eu vou colocar. Gui, podemos começar criando uma variável para cada um desses filmes que foram citados, que são os filmes que vamos querer colocar na página. Então, no JavaScript, começamos criando essas variáveis.

Paulo: Para eu entender, dessa vez você só criou um novo arquivo, não fez o tal do *fork* de algo que já existia. Para podermos praticar com alguma liberdade.

Rafaella: Exatamente, já vamos para o projeto.

Paulo: Lembrando que é normal o visual do seu CodePen estar um pouco diferente do que estamos vendo aqui. Está vendo que muda o modelo de visualização? Você pode usar o melhor para você. Não precisa se preocupar se estiver diferente do nosso.

Isso vale para tudo na programação, sempre falo isso. Muitas vezes fazemos um exercício que a Rafa passou, um desafio que o Guilherme passou e o resultado é diferente do que mostraram no Discord, é diferente do que o Marco Bruno fez no Twitch é diferente de algo que eu já vi em um dos cursos da Alura.

Existem muitas formas de solucionar e às vezes não estar totalmente igual não é problema, muito pelo contrário, é aquele exercício de você praticar modos diferentes de

resolver o mesmo problema. Isso abre sua cabeça, porque você vai além do que passamos aqui, você pode tirar outras conclusões. Vai descobrindo o que pode e o que não pode.

Isso faz parte do seu trabalho como Dev, esse trabalho de investigação que, de novo, eu puxo as falas da Rafa lá no início.

Rafaella: Perfeito.

Guilherme: Uma coisa legal que podemos fazer nesse projeto, emendando no que o Paulo comentou, é dar um nome para esse projeto. Você tem alguma coisa em mente?

Rafaella: Verdade. Podemos colocar o nome do projeto de "Aprendendo sobre listas".

Guilherme: Eu já vou também clicar no "Settings > Behavior" e deixar "não" nas opções de salvamento automático e de auto-update e deixar "sim" para a formatação. E no "Pen Details", vou escrever no campo de tags: "imersaodev, alura, lacos" e, em seguida, clicar no botão "Salvar".

Já temos o nome, as tags e definimos o comportamento da nossa IDE.

Rafaella: Vamos agora criar uma variável para cada um dos três filmes no JavaScript. O terceiro filme vai ser "Escola do Rock", que é o filme que eu mais gosto.

```
var filme1 = "Yesterday"
var filme2 = "A chegada"
var filme2 = "Escola de Rock"
```

Rafaella: Então, já temos as nossas três variáveis de filmes e queremos imprimi-las na tela. Vamos mexer com imagens lá no projeto que faremos, mas é só para vocês aprenderem uma forma diferente de imprimir na tela. Nas outras vezes nós usamos aquele `getElementById()`, e já tínhamos no HTML uma tag específica onde queríamos imprimir as informações. Então tínhamos, por exemplo a tag `<div>` ou `<h2>`, e elas já tinham aquele id que nós pegávamos e puxávamos para o JavaScript, para imprimir nele.

Dessa vez, não temos nenhuma tag, não tem nada. Então não vamos dar um `getElementById()` nesse momento. No projeto, mais adiante, talvez usemos isso, mas agora vamos ter que escrever do zero e para isso também existe outro `document.` que é diferente do `getElementById()`, mas ele vai escrever na tela diretamente. É o comando `document.write()`, "*write*" de escrever mesmo, e coloca entre parentes o que queremos que ele escreva na tela.

Lembrando que como queremos criar novos elementos, porque ainda não tem nada na tela, temos que colocar a tag HTML que queremos escrever dentro desse

`document.write()`. Vamos utilizar a tag `<p>` que é uma tag de parágrafo, ela é usada para quando queremos escrever alguma frase, por exemplo. Nesse caso vamos escrever o nome de cada filme. Já pode colocar ela acompanhada da tag de fechamento, que é aquela barra que indica o fechamento.

```
document.write("<p>" "</p>");
```

Colocaremos nossos filmes dentro dessa tag. Vamos fechar com as aspas e fazer aquela concatenação que já vimos nas aulas anteriores, em que colocamos sinal de mais e vai somar com a variável que queremos imprimir. Será uma linha para cada variável, pois queremos imprimir um filme em cada linha:

```
document.write("<p>" + filme1 + "</p>");
document.write("<p>" + filme2 + "</p>");
document.write("<p>" + filme3 + "</p>");
```

Paulo: Esse `document.write()` é um pouco diferente. Porque para colocar algo na tela estávamos usando ou o `console.log()`, que aparece no console, ou aquele innerHTML. Esse é um pouco diferente, não é?

Rafaella: Isso. Antes pegávamos a tag que já existia. Como não tem tag nenhuma no HTML, estamos escrevendo.

E, olha só, salvamos e já executamos o código. Apareceu o texto com os nomes dos filmes na página. Se você quiser, pode tentar trocar a tag `<p>` de parágrafo por `<h1>`, por exemplo, ou `<h2>`. Qualquer outra tag para você ver como vai funcionando, como você realmente está colocando o HTML por meio do JavaScript.

Paulo: Ótima ideia, Rafaella. Deixe um comentário aí em cima desse código falando "experimente o h1, o strong" e as pessoas podem pesquisar outras tags para testar também.

Guilherme: Rafa, eu tenho uma dúvida. Essa tag que usamos, é como se tivéssemos criado no HTML, certo?

Rafaella: Isso.

Guilherme: Nós conseguimos estilizá-la, pelo menos um pouco aqui no CSS?

Rafaella: Sim. Para estilizar nós podemos criar, no CSS, o `p {}` e dentro dessas chaves colocamos tudo o que queremos estilizar. O que você quer colocar, Gui?

Guilherme: Seria legal deixar centralizado.

Rafaella: Então podemos colocar um `text-align:center;`.

Guilherme: Dá para alterar o tamanho da fonte?

Rafaella: Sim. Coloca `font-size:42px;` para vermos se esse tamanho fica legal.

Guilherme: Salvei, vamos rodar. Ficou bacana.

Rafaella: Nós criamos uma variável para cada filme, mas se você parar para pensar, no dia a dia, naquelas plataformas tipo a Netflix, Amazon, entre outras. Eles não conseguem ficar criando uma variável, de fato, dentro do programa para cada filme que querem colocar no catálogo. Porque isso gera um código gigantesco. E também para quando você quiser adicionar uma coisa fica muito complicado. Você tem que ir lá no código, criar uma variável e colocar o valor dela.

Então, para isso, nós utilizamos as listas. As listas entram de uma forma um pouco mais prática para colocarmos várias coisas que serão listadas. Coisas que têm praticamente o mesmo valor, todas são filmes, todas são imagens, as imagens colocaremos mais adiante.

Gui, primeiro podemos dar uma olhada na documentação dessas listas, que são chamadas também de arrays.

Paulo: Eu gosto desse nome "listas" que a Rafa colocou, mas você vai ver muitas pessoas falando array, vetor, sequência. Que não necessariamente são exatamente as mesmas coisas em determinadas linguagens.

Mas a lista é algo mais fácil de entender. Queremos uma lista de palavras e não só uma palavra. Porque ali como a Rafa e o Guilherme nos mostraram, toda vez que tem um filme novo deve ter uma nova variável. Será que não tem uma forma de tratar isso de uma maneira mais homogênea?

Guilherme: Além disso, Paulo. Tem um outro problema que é o seguinte: aqui nessa linha estamos guardando o título do `filme1`, aí teríamos a imagem do `filme1`, depois a legenda do `filme1`, o próprio arquivo do filme e assim por diante. Ficaria muito complicado manter isso.

Claro que no mundo real eles pensam em outras abordagens para solucionar isso, mas no nosso caso queremos guardar uma lista de filmes. Criar uma variável para cada um deles não funciona. Vamos acessar a documentação, com a Rafa comentou. Pesquisei por "Array JavaScript" no Google e entrei na documentação do próprio Mozilla.

Ele fala que o objeto array do JavaScript é um objeto global usado nas construções de arrays semelhante a listas que estamos acostumados. Então, a ideia principal é que teremos uma variável e daremos um nome para essa nossa lista. Aí colocaremos o

sinal de igual - e para identificarmos o que é uma lista e o que é uma variável utilizaremos os colchetes (`[]`). Eu gosto de frisar. Não é chaves e não é parênteses, é colchetes.

Porque no começo isso era muito confuso para mim. Então, sempre que falamos de listas, estamos falando dos colchetes. E, no exemplo da documentação, dentro da lista de frutas estão a maçã e a banana, `var frutas = ['Maçã', 'Banana'];`.

Quando criamos uma lista, existem funções e métodos para que essa lista tenha algumas propriedades que são bem legais. Podemos perguntar, por exemplo, o tamanho, quantos elementos tem dentro dessa lista. Veremos isso mais adiante com calma, mas nessa página da documentação você vai encontrar uma série de definições sobre a listas e como trabalhamos com listas no JavaScript.

Paulo: Eu acho muito legal o Guilherme e a Rafa trazerem de novo a documentação do Mozilla que é o criador do Firefox. Vale lembrar que sempre que se tem essa documentação mais clássica, a leitura é um pouco dura. Mesmo sendo em português, você vai ver que os exemplos são muito abstratos e já vai direto para termos um pouco mais complexos. Não é bem uma explicação detalhada, não é um tutorial. É realmente uma referência para usarmos. Acho interessante trazer isso, para tentar colocar um pouco mais da vida real de uma pessoa desenvolvedora e você entender como é.

Aos poucos vocês vão entender mais sobre isso. É grande a quantidade de sites, programas e sistemas que usamos. Nós começamos com o CodePen, agora acessa a documentação, depois vamos usar outra coisa. Para mostrar com calma o que usamos no dia a dia.

Porque, realmente, é muita coisa. E pode ter aquela coisa de alguém achar que não está entendendo. De novo, é normal e faz parte dessa carreira.

Rafaella: Já vimos mais ou menos como é a sintaxe para criar uma lista aqui no JavaScript. Então vamos fazer, vamos criar a nossa variável. Pode deixar esse código aí, vamos fazer no início do arquivo. Vamos criar uma `var listaFilmes`. Para colocarmos cada um dos elementos dentro desta lista, vamos usar os colchetes e dentro deles vamos separar os elementos usando a vírgula.

```
var listaFilmes = ["Yesterday, A Chegada, Escola de Rock"]
```

Rafaella: Pronto. Agora não precisamos mais dessas variáveis `filme1`, `filme2` e `filme3`. E quando formos imprimir usando o `document.write()`, teremos que falar de outra forma. Estávamos chamando cada uma das variáveis pelo nome delas, mas na lista fazemos de uma forma um pouco diferente. Quando temos elementos dentro dessa lista, conseguimos chamar cada um deles individualmente chamando apenas o índice dele.

Vamos deixar duas linhas como comentário e deixar só uma linha com o `document.write()`. Primeiro vamos ver o que aparece quando chamamos a lista inteira.

```
document.write("<p>" + listaFilmes + "</p>");
```

Rafaella: Salvamos, executamos e olha só. Os títulos dos filmes apareceram, mas está tudo junto e sem espaço. Não é isso que nós queremos. Nós queremos listar cada um dos filmes em uma linha. Queremos colocar uma tag de parágrafo para cada um deles.

Para isso, como falei, nós conseguimos chamar cada item da lista pelo índice dela. O que é o índice da lista? É a posição em que o elemento está dentro da lista. Por exemplo, "Yesterday" é o primeiro elemento da lista, "A Chegada" é o segundo elemento da lista e "Escola de Rock" é o terceiro elemento da lista.

"Então, Rafaela, para chamar o item 1 da lista, que vai ser o 'Yesterday', chamamos pelo índice 1"? Não. Uma coisa que é padrão de arrays e listas é que o primeiro elemento sempre vai ter índice 0. Então mesmo que você queira chamar a primeira coisa que você colocou, ela vai ser zero. Não tem jeito. Temos que aceitar e entender que funciona dessa forma.

O segundo elemento vai ser o índice 1 e assim por diante. O terceiro andamento vai ter o índice 2 e vamos chamando cada um deles dessa forma. Então, Gui, vamos começar a imprimir cada um desses elementos pelo índice deles?

Guilherme: Vamos.

Rafaella: Você pode descomentar essas duas linhas do `document.write()` e lá onde está `listaFilmes` colocamos `listaFilmes[0]`. Esse colchete vai indicar que eu quero o item que está no índice escolhido, no caso vai ser o 0, a primeira posição dessa lista. No segundo item colocaremos `listaFilmes[1]` e no terceiro item `listaFilmes[2]`.

```
var listaFilmes = ["Yesterday, A Chegada, Escola de Rock"];

document.write("<p>" + listaFilmes[0] + "</p>");
document.write("<p>" + listaFilmes[1] + "</p>");
document.write("<p>" + listaFilmes[2] + "</p>");
```

Rafaella: Vamos salvar e ver o que vai aparecer. Olha só, imprimiu corretamente.

Paulo: Então, Rafa, lá em cima usamos colchetes para criar, no lado direito da atribuição de uma variável. Aí embaixo, da primeira vez, que colocamos só `listaFilmes`, ele imprimiu o conteúdo inteiro da variável, toda a lista.

Para acessar determinado elemento, o enésimo elemento, esse termo aparece bastante em computação, não é? Para acessar o primeiro eu uso o colchetes passando o índice da primeira posição que é 0 e assim por diante.

Rafaella: Exatamente.

Paulo: E se eu tentar acessar o quarto elemento, que seria o índice 3, o que acontece?

Rafaella: Vamos ver.

Guilherme: Adicionei uma linha chamando o `document.write()` de `listaFilmes[3]`. Salvei e vou executar:

Yesterday

A Chegada

Escola de Rock

undefined

Rafaella: Apareceu um "undefined" (indefinido).

Paulo: E não foi um erro. É algo que ele aceita, mas fala: "Olha, não tem nada aqui". Mas se tivesse um quarto filme na lista ele teria aparecido. Podemos colocar um "Harry Potter" em homenagem à Rafaella. Vamos ver se aparece?

Guilherme: Inseri "Harry Potter" na `listaFilmes`. Salvei, executei. E apareceu corretamente. Só estou com uma dúvida agora. Rafa, é possível incluir algo na lista mesmo que eu já tenha a lista feita?

Rafaella: É possível.

Guilherme: Vou tirar o Harry Potter da lista e fazemos esse exemplo com ele, pode ser?

Rafaella: Pode. Se você quiser abrir a documentação para dar uma olhada, existem alguns comandos de arrays que são próprios para brincarmos com eles. Uma das coisas que podemos fazer é de fato ter um comando para colocar elementos novos, assim como tem para tirar e tem para fazer outras coisas. Para adicionar na lista tem esse exemplo de `.push` na documentação. Esse "*push*" é de empurrar, do inglês. Você vai "empurrar" algum valor para dentro do array.

```
var adicionar = frutas.push('Laranja');
```

```
// ['Maçã', 'Banana', 'Laranja']
```

Rafaella: Vamos fazer então, `listaFilmes.push()` e colocamos dentro dos parênteses a informação que queremos adicionar, que é "Harry Potter".

Guilherme: Vou colocar "Harry Potter 2" só para ter certeza que vai mudar.

```
var listaFilmes = ["Yesterday, A Chegada, Escola de Rock"]

listaFilmes.push("Harry Potter 2");

document.write("<p>" + listaFilmes[0] + "</p>");
document.write("<p>" + listaFilmes[1] + "</p>");
document.write("<p>" + listaFilmes[2] + "</p>");
document.write("<p>" + listaFilmes[3] + "</p>");
```

Rafaella: Perfeito. Vamos salvar e ver o que vai acontecer.

Yesterday

A Chegada

Escola de Rock

Harry Potter 2

Rafaella: Ele entrou no quarto lugar do array e está no índice número 3. Então você pode adicionar vários filmes usando esse `push()`. Pode, inclusive, criar a lista apenas com os colchetes em branco e ir adicionando os filmes com o `push()`. E eles vão entrando na ordem que escrevermos. O primeiro a darmos `push()` vai entrar no índice 0, o segundo no índice 1 e assim por diante.

Já estamos diminuindo nossas linhas de código. Ao invés de criar várias variáveis criamos uma lista. Isso já diminuiu bastante o nosso trabalho. Mas existe ainda outra coisa que está incomodando um pouco, é esse `document.write()`. Será que não tem como imprimir todos esses elementos, um em cada linha, um em cada parágrafo, de uma forma mais fácil, para não precisar ficar escrevendo a toda hora?

Guilherme: Existe. E agora eu quero que você preste bastante atenção, pois essa é uma parte muito importante da nossa aula.

Paulo: O Guilherme falou "muito importante", mas o que ele quer dizer é que assusta um pouco o código que vamos escrever. Acho que é isso que ele quer dizer.

Guilherme: É isso aí.

Paulo: Tem momentos em que você se pergunta "de onde saiu isso?", "de onde saiu esse *push*?". E tem muitos outros desses termos na documentação, o que será que cada um faz?

Uma variável desse tipo array, lista, tem uma série de ".alguma coisa". São funções, que também chamamos de "método". São funções que fazem alguma coisa com aquela informação.

E agora o Guilherme e a Rafaella estão preparando o terreno para trazer um mecanismo novo. Assim como teve o `if` que já tem uma sintaxe meio diferente. Vai aparecer um que tem uma cara um pouco mais diferente, e isso faz parte do nosso dia a dia quando aprendemos uma nova linguagem de programação. Principalmente quando aprendemos a nossa primeira linguagem.

Tem coisas que você vai achar que não fazem sentido. Mas é como aprender um novo idioma, se você traduzir literalmente pode não fazer sentido. É o mesmo mecanismo. Então, sem sustos, respirem fundo e vamos com o Guilherme e com a Rafa.

Guilherme: Antes de dar continuidade, vamos observar que da linha 9 até a linha 12 a única coisa que muda são os números do índice dentro dos colchetes.

```
document.write("<p>" + listaFilmes[0] + "</p>");
document.write("<p>" + listaFilmes[1] + "</p>");
document.write("<p>" + listaFilmes[2] + "</p>");
document.write("<p>" + listaFilmes[3] + "</p>");
```

Guilherme: Lá no começo da documentação aparece algo interessante no exemplo "Criando um Array". Aparece o `.length`, com essa propriedade conseguimos descobrir o tamanho de elementos que a nossa lista possui.

Então, se o programa soubesse o tamanho da nossa lista, poderíamos pedir assim: "Eu quero que você repita essa linha primeiro com o índice 0, depois repete com o índice 1, depois com o índice 2, depois repete com o índice 3. Quando a nossa lista acabar, quando não tiver mais elementos, você não precisa repetir mais".

A sacada é justamente essa. O código que escreveremos vai fazer isso. Ele possui uma estrutura um pouco diferente chamada *"for"* que vamos aprender agora. Quando eu escrevo só o `for`, repare que a cor da fonte ficou vermelha. É uma palavra reservada do JavaScript.

Então, temos o `for`. Semelhante ao `if`, onde abrimos e fechamos parênteses para colocar a nossa condição, no `for`, vamos fazer o mesmo: abrir e fechar parênteses,

`for ()`. Todo o código relacionado a ele, nós colocaremos entre chaves, semelhante ao que fizemos na nossa função.

```
for () {

}
```

Inicialmente, a nossa estrutura é essa: nós temos a palavra `for` e, dentro dos parênteses, vamos colocar uma determinada condição e ela verificará se será repetido ou não, se existem mais filmes a mostrar ou não, com base no tamanho da lista e, depois, vamos colocar uma linha só do `document.write` para que a mágica aconteça. Primeiro, quando criamos um `for`, precisamos de um ponto de partida, que nós colocaremos dentro dos parênteses.

Esse ponto de partida está relacionado, por exemplo, a se queremos varrer nossa lista toda e o nosso índice inicial é o zero. Assim como no JavaScritpt, no Python e na maioria das linguagens, o primeiro elemento da lista sempre possui o índice zero. Então, passaremos o índice que queremos trabalhar. Para isso, criaremos uma nova variável var, que chamaremos de `indice`, que será igual a zero e, por fim, colocaremos ponto e vírgula.

```
for (var indice = 0;) {

}
```

O nosso `for` precisa ter um ponto de partida, um valor inicial, por isso, colocamos a nossa condição. O ponto e vírgula é muito importante, pois passaremos ao nosso segundo argumento ou segundo parâmetro do `for`. Nós queremos que o `for`, essa estrutura de repetição, seja executada até quando? Isto é, precisamos saber quantas vezes o `for` será executado.

Desejamos que a linha `document.write` tenha um limite de repetição, não que seja exibido do índice zero ao um bilhão. Não teríamos memória suficiente para tudo isso. Sendo assim, no segundo parâmetro, passaremos nossa condição que é: se o índice for menor que o tamanho de elementos da lista - que o tamanho dos índices que temos na lista - a estrutura deve parar.

Ou seja, enquanto tivermos filmes, índices, então, mostra. Por exemplo, temos o índice 0, temos elemento (1), então, mostra. Temos o índice 1, temos elemento (2), então, mostra. Temos o índice 2, temos elemento (3), então mostra.

Paulo: Então, neste caso, é menor que 3, `for (var indice = 0; indice < 3) {`

Guilherme: Sim, porque nós já sabemos que temos o valor dos elementos (1, 2, 3), relacionados aos índices (0, 1, 2), e o "Harry Potter 2", aliás, vou até adicionar o índice 3.

```
//adicionando novos elementos - índice 3.
listaFilmes.push("Harry Potter 2");
```

Rafaella: Eu gostaria de comentar uma coisa. Se o índice for maior que o 3, é para ele parar, certo? Mas, precisamos lembrar que o índice 3 entra nisso. Portanto, ele deveria ser menor que o índice 4.

Paulo: Isso porque temos um quarto elemento agora. Você tem razão. Não desejamos que pare no 2. Considerando o "Harry Potter 2", ele vai no 0,1,2, e tem o três, quer dizer que vai até o 4. Ou, podemos escrever menor ou igual a 3, usando uma outra forma que não é tão usual. Normalmente, escrevemos como a Rafa falou. Essa é uma dúvida que, no começo, é comum aparecer, "até onde vai?", "preciso de um a mais, ou a menos?".

Enfim, começar pelo zero, pode gerar certa confusão. Mas, depois, pegamos o hábito. No caso de hoje, eu me enganei também.

Guilherme: Deixo o 4 mesmo?

```
for (var indice = 0; indice < 4;) {

}
```

Rafaella: Sim.

Guilherme: Agora passaremos o último argumento. Nós começaremos copiando a linha que queremos: `document.write("<p>" + listaFilmes[0] + "</p>");`. E podemos comentar esta e as demais linhas anteriores ao `for`. Continuando, desejamos que a sequência apresentada se repita, logo, na primeira vez, o valor do índice será zero. Pela lógica, 0 é menor que 4? Sim, então, execute o código.

```
for (var indice = 0; indice < 4;) {
  document.writer("<p>" + listaFilmes[0] + "</p>");

}
```

O código pegará o `document.write` na *tag* `"<p>"` e colocará a `listaFilmes` no índice zero. Mas, depois disso, como ele saberá que, agora, deve passar para o próximo índice, o 1? Nós precisamos somar o índice fazendo `indice++`.

Paulo: Calma, Guilherme. Vamos fazer `indice = indice +1`, ou seja, `for (var indice = 0; indice < 4; indice = indice +1) {`.

Guilherme: É a mesma ideia, nós podemos refatorar depois. No lugar do zero, vamos usar o próprio índice, isto é, `document.writer("<p>" + listaFilmes[indice] + "</p>");`

Paulo: Gostaria de recapitular algumas coisas antes de você executar. Assim como o `if` é uma estrutura de decisão de lógica e existem outras estruturas com outras funções, o `for` é uma **estrutura de repetição**.

Diferente do `if`, que também abre e fecha chaves e executa apenas uma vez ou nenhuma, dependendo se é verdadeiro ou falso, o `for` levará a peça do meio, `indice < 4`, para saber se ele executa ou não, e mais, se ele continua executando. Ele é um laço, um *loop*, diferente do `if` que é só uma vez.

Enfim, precisamos ter em mente que o guia principal é o do meio. O que está à esquerda, `var indice = 0`, só é executado uma única vez. O `indice = indice + 1`, pode nos causar confusão, já que, numa primeira leitura, podemos pensar que, na primeira linha, o índice já valerá 1, mas não vai.

O `for` é um tanto estranho, porque vem de outras linguagens de programação muito antigas e ele executa `var indice = 0`, verifica se ele é menor que 4, e não executa o `indice = indice +1`. Primeiro, ele vai para o "miolo" do laço para, depois, voltar ao `indice = indice +1` e para a verificação, buscando saber se deve ou não executar mais uma iteração. Quando temos um laço, chamamos de **iteração**.

Realmente, o `for` é uma estrutura diferente. Até agora, estávamos entendo tudo em uma ordem específica, if, parênteses, acontece determinada coisa. O `for` tem outros mecanismos, por exemplo, ponto e vírgula dentro dos parênteses, que é algo bem diferente. É normal levarmos um tempo para assimilar.

Guilherme: Isso mesmo. Vou até fazer um comentário em cima desses valores, `// valor inicial. condicao. expressão final`. O primeiro, é o valor inicial, o segundo, é a nossa condição e, para finalizar, podemos dizer, com base no seu comentário, Paulo, que temos a nossa expressão final. Significa que `indice = indice + 1` não será executado no início, é como se ele fosse colocado no final. A cada iteração, ele realizará o `indice = indice + 1`.

Então, o `for` é dividido nessas três estruturas. Nós temos o valor inicial, com o qual daremos o "pontapé" para a nossa repetição. Depois, vem a nossa condição: vamos executar até quando? Enquanto o índice for menor que 4, queremos executar. Depois que executamos o miolo, queremos passar para o próximo índice. Vamos rodar primeiro

com `listaFilmes` e, depois, só mostrando o índice, para que seja possível ver a mágica acontecendo.

Salvei e vou executar. Feito isso, aparecem todos os filmes: Yesterday; A chegada; Escola de Rock; Harry Potter 2. Repare que, os outros códigos, nós havíamos comentando, nós não temos mais.

Rafaella: Nós podemos apagar esses textos comentados, não?

Paulo: Agora já podemos. Ótimo, Rafa! Assim as pessoas podem olhar e comparar que o resultado está sendo o mesmo, sem a necessidade copiar e colar infinitamente.

Guilherme: Exatamente! Vou retirar então a parte comentada dos *document.writes*:

```
// experimente o h1, strong
// document.write("<p>" + listaFilmes[0] + "</p>");
// document.write("<p>" + listaFilmes[1] + "</p>");
// document.write("<p>" + listaFilmes[2] + "</p>");
// document.write("<p>" + listaFilmes[3] + "</p>");
```

Paulo: Pode rodar apenas o índice, ao invés do `listaFilmes`. Depois colocamos as duas linhas, mas rode para verificarmos o que acontece. Você mexeu e eu não havia percebido, o `document.write` agora está só com índice, `document.write("<p>" + indice + "</p>");`, não tem mais `listaFilmes`.

Guilherme: Não tem, porque o que eu quero, de fato, são esses valores.

```
// valor inicial. condicao. expressão final
for (var indice = 0; indice < 4; indice = indice +1) {
  document.write("<p>" + indice + "</p>");
}
```

Rafaella: É a variável da iteração que criamos dentro do `for`.

Paulo: Uma variável um tanto dispensável para o usuário final. Não é muito útil.

Rafaella: Sim, é apenas para usarmos agora.

Guilherme: Então, temos "0,1,2,3", indicando que ele parou corretamente. Agora, vou colocar os dois, Paulo.

Paulo: Isso, o índice e, depois, o `listaFilmes` na posição índice.

```
// valor inicial. condicao. expressão final
for (var indice = 0; indice < 4; indice = indice +1) {
  document.write("<p>" + indice + "</p>");
      document.write("<p>" + listaFilmes[indice] "</p>");

}
```

Guilherme: Tudo certo! Salvei, estou executando e, maravilha, temos: 0, Yesterday; 1, A chegada; 2, Escola de Rock; 3, Harry Potter 2. Ficou Harry Potter 2, no índice 3. Vou trocar para Harry potter 3.

Ficou bom, mas existe uma forma de conseguirmos melhorar o código da nossa estrutura de repetição, o `for`. Para começar, a expressão final `indice = indice +1`, no cotidiano, no mundo real, o programador sabe que esse trecho se trata da última coisa, a nossa variável de iteração, que precisaremos atribuir um valor no nosso índice.

Uma forma que temos o mesmo, isto é, o valor do índice que nós já temos e que é zero, depois +1, executamos, em seguida vem o índice, que é 1, e, um mais um é igual a 2, depois 3, enfim, podemos fazer tudo isso colocando `indice++`. Vamos executar, salvar e alcançaremos o mesmo resultado.

Paulo: À primeira vista parece estranho, por não haver o símbolo "igual", o que nos leva a questionar quanto vale o índice agora. Nós estamos acostumados a escrever que determinada variável vale determinado valor. Aqui não, nós estamos falando que "determinada variável, algo".

Rafaella: Soma com 1.

Paulo: Ele está indo de 1 em 1, o efeito é o mesmo.

Guilherme: Exatamente! Algo legal que eu gostaria de recomendar é, façam o teste depois, em casa, com outros valores. Usem, por exemplo, uma variável idade e coloquem `idade++` para ver esse valor, de fato, acontecendo. Seguindo, vou tirar o índice, `document.write("<p>" + indice + "</p>");`, e salvar, para exibirmos apenas os filmes, e teremos o resultado esperado: Yesterday, A chegada, Escola de Rock, Harry Potter 3.

Porém, há um ponto importante que eu gostaria de ressaltar. Em relação ao Harry Potter, nós sabemos que existem mais filmes. Por isso, colocaremos, também, `listaFilmes.push("Harry Potter 4");`. Vamos salvar, executar mais uma vez e ver o que acontecerá. Nós colocamos "Harry Potter 4", mas nada foi alterado, continuamos vendo "Harry Potter 3", por quê?

Nós temos um novo elemento na lista, o `// índice 4`, o nosso novo elemento, mas a nossa lista vai até o índice 3. Nós falamos: enquanto o índice for menor que 4, ou seja, quando for 4, ele é menor que 4? Não, ele já vai parar, não nos mostrará mais. Como fazer para que o valor em `indice < 4` seja dinâmico?

Ou seja, se inserirmos mais filmes ou removermos mais filmes, ele pegará exatamente o tamanho da nossa lista. Nós lemos isso na documentação, não demos tanta ênfase, mas passamos por isso.

```
Criando um Array

var frutas = [ 'Maçã', 'Banana' ];

console.log(frutas.length);
// 2
```

Se pegamos a lista e colocamos um ponto seguido da palavra **length**, que significa tamanho, nós pegamos o tamanho da lista, que é, justamente, o que queremos no nosso código. Então, se pegamos o `listaFilmes.length`, que eu sempre escrevo errado. Vou dar uma dica agora.

Quando eu ouvi, pensei, "por que não me falaram isso quando eu estava aprendendo programação", que é, na palavra "tamanho", em português, a letra "t" sempre vem antes de "nh" no final, portanto, é "leng" e o "t" primeiro. As vezes colocamos o "lgh" e depois o "t". Não! Em "tamanho" o "t sempre vem antes do "h". Apenas uma dica para vocês não se confundirem.

Poderíamos até criar um curso, Paulo, de dicas do mundo da programação, porque é algo que faz muita diferença. Se escrevemos "lenght", vai dar errado! O programa não vai entender, ele não reconhece essa palavra, portanto, a letra "t" deve vir sempre antes do "h".

```
// valor inicial. condicao. expressão final
for (var indice = 0; indice < listaFilmes.length; indice++) {
document.write("<p>" + listaFilmes[indice] + "</p>");

}
```

Essa alteração que nós fizemos é: enquanto houver índice e ele for menor que o tamanho da lista,

vou até fazer algo interessante, colocar, fora da nossa estrutura de repetição, em um console, só para visualizarmos, `console.log(listaFilmes.length)`, salvar, executar e, agora, perceberemos que aparecerá o Harry Potter 4. Executei.

Rafaella: Precisa abrir o console.

Guilherme: Vou abrir. Apareceu "Harry Potter 4" e vou abrir o console, que a Rafa lembrou e, nele, temos "5". Ficou legal! Mas também existe o Harry Potter 5. Na verdade, existem 7 filmes do Harry Potter. Então, vamos copiar a linha `listaFilmes.push("Harry Potter 4");` e substituiremos o 4 por 5, e, assim, verificaremos se ficará dinâmico.

Eu espero que o mostre o Harry Potter 5, e o índice, que será exibido no nosso console, precisa ser o número 6. Salvei e ele já executou antes mesmo de salvar. Colocamos Harry Potter 3, 4 e 5, e, no console, índice 6. Se inserirmos vários filmes do Harry Potter, que, se não me engano, vão até o 7, não é, Rafa?

Rafaella: Oito.

Paulo: Até o oito, porque o 7 é dividido em dois.

Guilherme: É verdade! A parte 7 é dividida em dois. Coloquei mais dois e pulei o número 6, que virou Harry Potter 7. O índice virou 7.

```
//indice 4
listaFilmes.push("Harry Potter 4");
listaFilmes.push("Harry Potter 5");
listaFilmes.push("Harry Potter 7");

console.log(listaFilmes.length);
```

Rafaella: Na verdade, não é o índice que está virando 7, é o tamanho, o número de elementos.

Guilherme: Isso! O número de elementos. E temos, agora o valor dinâmico.

Paulo: É possível ver no console.

Guilherme: Significa que se colocamos mais elementos, o for é capaz de repetir, criar o número de repetições necessárias. Só lembrando, nós temos o valor inicial, a nossa condição, usamos propriedades da nossa própria lista, o `.length` vem da nossa lista, alguém já programou isso, de forma que não precisamos ficar calculando quantos elementos tem e, no final, usamos uma estrutura com `++`, isto é, `indice++`, para ser a nossa expressão final.

Ficou legal, Rafa?

Rafaella: Ficou muito legal! Acho que, agora, podemos ir para o nosso projeto. Para ficar ainda mais interessante que só esses textos escritos.

Guilherme: Vamos tirar o Harry Potter 6 e deixar só até o Harry Potter 5. Façam os testes na casa de vocês. Agora, vamos entrar no link que está descrito, do projeto modelo, e fazer o Fork desse projeto.

Paulo: Então, nós temos um projeto. Nós estamos aprendendo um pouco sobre o funcionamento das listas e arrays, como inserir elementos, acessar o enésimo elemento, o elemento na primeira e segunda posição e como fazer, em especial, a estrutura de repetição, porque o `for` não serve apenas para passear em listas. Ele tem outras funções que aparecerão mais para frente.

Você acabou de aprender bastante das *arrays*, como usamos, acessamos, adicionamos e percorremos. Isto é, como percorremos uma lista e um *array*, como iteramos uma lista e um array. Esses verbos aparecerão com frequência.

Guilherme: Eu acessei o site do CodePen com a minha conta pessoal, e estou com o link para que você pegue o modelo do Aluraflix. Se observarmos, já existe uma estrutura no HTML e um CSS para termos esse visual bem bonito do Interestelar. Eu quero uma cópia desse projeto, desse *template*, para os meus arquivos, para a minha conta do CodePen.

Então, farei a "garfada", o Fork e aparecerá o nome da minha conta @guilimadev, já estou com uma cópia do código e podemos partir para o desenvolvimento. Qual é a ideia, Rafa?

Rafaella: Nós faremos uma plataforma, onde listaremos as imagens dos filmes que gostamos para mostrar para alguém, ou as imagens do filme que queremos ver na nossa lista de desejos, ou, por exemplo, "filmes de terror" e colocamos a lista dos filmes de terror que queremos ver. Enfim, você personaliza da forma que quiser. Nós também colocaremos os filmes que mais gostamos.

Para isso, pegaremos o conceito de lista mais uma vez, porém, faremos algo diferente, não vamos mais imprimir o nome de cada filme, como estávamos fazendo, pretendemos imprimir as imagens de cada um deles. Então, vamos criar a nossa variável que será var `listaFilmes` também e criaremos um *array,* uma lista com colchetes e cada coisa que colocaremos dentro, não será mais o nome, mas, a URL ou o link da imagem que deve aparecer na tela.

Como fazemos para pegar o link da imagem que aparece na tela? Nós não conseguimos simplesmente copiar a imagem, em si, e colar no CodePen. Por isso, precisamos utilizar o link delas, então, Gui, você consegue abrir o Google para

pesquisarmos as imagens? Então, podemos selecionar essa imagem do filme Yesterday em poster vertical mesmo.

Para copiar o link dessa imagem, apertaremos com o botão direito do mouse e aparecerá a opção "copiar endereço da imagem". Não é "copiar link", mas, sim, "copiar endereço da imagem", porque, apertando o "copiar link", entraremos, na verdade, no site que essa imagem está. Nós queremos apenas a imagem. Portanto, copiaremos o endereço da imagem e abriremos uma nova aba no navegador apenas para nos certificarmos de que copiamos a URL certa.

Quando usamos essa URL, esse link, apenas chamamos a imagem. Nós colaremos na nossa *string*, porque é um formato de texto no nosso CodePen também.

```
var listaFilmes =
["https://upload.wikimedia.org/wikipedia/pt/7/79/Yesterday_%282019%29
_poster.jpg"]
```

Já temos o nosso primeiro filme como primeiro elemento da lista. Agora, partiremos para o segundo. Vamos pesquisar "A Chegada". Podemos padronizar, eu gostei de pegar essas mais verticais para ficar mais padronizado. Acho que essa segunda talvez, "Arrival". Pode copiar o endereço dessa imagem também, Gui, e colar no segundo elemento.

```
var listaFilmes =
["https://upload.wikimedia.org/wikipedia/pt/7/79/Yesterday_%282019%29
_poster.jpg",
"https://1.bp.blogspot.com/-ImZPRqLsluE/WFK156_6pNI/AAAAAAAAYBY/0lEhN
RF5wfQdLfr6hpT57_Jt2eBrE9H5wCLcB/s1600/arrival-kartoun-desert.jpg"]
```

Guilherme: Colocando a vírgula, não é?

Rafaella: Vírgula entre os links e cada um deles entre aspas. Agora, "Escola de Rock".

Paulo: Lembrando que é bom tomar cuidado com essas URLs, com esses endereços, porque eles podem quebrar. Enfim, cuidado com a vírgula e as aspas, como as URLS são grandes e têm esses nomes complicados, as vezes nos perdemos um pouco.

Guilherme: Já escolhi a imagem da terceira posição.

```
var listaFilmes =
["https://upload.wikimedia.org/wikipedia/pt/7/79/Yesterday_%282019%29
_poster.jpg",
"https://1.bp.blogspot.com/-ImZPRqLsluE/WFK156_6pNI/AAAAAAAAYBY/0lEhN
```

```
RF5wfQdLfr6hpT57_Jt2eBrE9H5wCLcB/s1600/arrival-kartoun-desert.jpg",
"https://br.web.img3.acsta.net/c_310_420/medias/nmedia/18/91/90/98/20
169244.jpg"]
```

Rafaella: Você, em casa, pode escolher os filmes que quiser para adicionar. Nós estamos adicionando os que gostamos. O link pode acabar sendo gigante também, não se assuste se você colar algo gigante, mas, vai funcionar. Algo interessante é que todos esses links, todas essas URLs são de imagens e elas têm, ao final, o `.jpg`, que é, justamente, a extensão dessas imagens. É interessante verificar isso, perceber se está tudo correto.

Às vezes, quando não termina com `.jpg`, pode, realmente, direcionar para outra página e a imagem não funcionará. Agora, precisamos fazer algo que aprendemos no projeto inicial, e usamos apenas para ver a aula, que é imprimir essas imagens.

Guilherme: Para fazer isso, nós temos duas opções, quero a opinião de vocês. Nós podemos fazer três `documents.write` ou podemos usar o `for`, para um só, o que vocês acham melhor?

Rafaella: Podemos partir para o `for`. Nós já entendemos, mais ou menos, como ele funciona. Podemos começar a imprimir em *loop*.

Guilherme: Então, vou escrever `for` e vamos começar a criar nossa estutura. O `for` tem os parênteses, em que passaremos os três argumentos. Nós já sabemos que temos a inicialização, isto é, o valor inicial `for (var ) {`. E, ao invés de índice, posso usar "i"?

Rafaella: Pode, o resultado é o mesmo. Só mudamos o nome da variável.

Guilherme: Então, chamarei de "i", `for (var i = 0) {` , passarei para enquanto ele for menor que o tamanho do nosso `listaFilmes`, `for (var i = 0; i < listaFilmes.length;) {` e farei a expressão final, o i++, para que ele seja incrementado.

```
for (var i = 0; i < listaFilmes.length; i++) {

}
```

Paulo: É idêntico ao que fizemos antes com o índice, porém, usando "i". Como se trata de uma variável que só importa internamente, não importa tanto para outros lugares do nosso código, nós usamos um nome mais curto. Normalmente até escrevemos por extenso, mas, nesse caso, é bem usual aparecer "i"ou "j".

Rafaella: Como vamos colocar entre colchetes, quando formos chamar a nossa `listaFilmes`, fica mais compacto utilizar o "i", é bem comum usar "i" de iteração. O que queremos imprimir em *loop*, Gui?

Guilherme: A ideia é colocarmos um `document.write` e, dentro dele, uma *tag* do HTML. No outro projeto, usamos uma *tag* "p", neste, vamos criar uma *tag* "img", então, colocaremos o sinal de menor, e escreveremos `img.` Para essa *tag* específica funcionar, precisamos passar qual imagem queremos colocar. Nós temos uma propriedade, o `src`, em seguida, adicionaremos o sinal de igual e concatenaremos o caminho da imagem com o que temos no nosso `listaFilmes`.

```
for (var i = 0; i < listaFilmes.length; i++) {
  document.write("<img src=" + )

}
```

Rafaella: Gui, você pode abrir o HTML só para analisarmos um exemplo dessa *tag* "img"? Acho que, no nosso HTML, temos um exemplo dessa *tag*.

Guilherme: É verdade! O logo da Alura que aparece no canto direito da tela está utilizando essa mesma propriedade `<img src =` e o caminho. Se desejamos exibir uma imagem, precisamos dizer qual imagem. Nós passamos no `src` e conseguimos exibir.

Rafaella: Existem outras propriedades que estão aparecendo, o alt, por exemplo, class, mas, nesse momento, não precisamos. Vamos usar apenas qual o endereço da imagem que desejamos imprimir. Uma coisa interessante também nessa *tag*, é que ela não abre e fecha, como o "p" do código anterior, onde escrevíamos dentro da *tag* e tínhamos a fechadura dela. Não, a *tag* de imagem fecha no final do próprio "maior que" dela mesmo.

Isso significa que não teremos aquela segunda parte de fechar com outro "img/".

Guilherme: Nós até podemos já fazer isso, não é, Rafa? Vou concatenar com alguma coisa e depois só temos um ">" para fechar.

```
for (var i = 0; i < listaFilmes.length; i++) {
  document.write("<img src=" + + ">")

}
```

Agora, queremos percorrer a nossa `listaFilmes` com base no índice que estamos. Então, `listaFilmes`, abre e fecha colchetes e, neles, não colocarei 0 nem 1, quero que seja feito com base no nosso laço, na nossa estrutura de repetição. Passaremos o "i".

```
for (var i = 0; i < listaFilmes.length; i++) {
  document.write("<img src=" + listaFilmes[i] + ">")

}
```

Com base nisso, acho que já conseguimos ver alguma coisa acontecendo. Vamos testar? Salvei, "coração na garganta", executando e, vamos ver? Apareceram as três imagens!! Ficou muito bonito!! Qual é o nome quando jogamos para o lado e ele muda a posição das imagens para caber na tela?

Rafaella: Responsivo.

Guilherme: Responsivo! Coisa chique, não, pessoal?

Rafaella: Ele corresponde ao tamanho da tela. Se você abrir em uma tela pequena, ele vai para baixo. Se você abrir em uma tela maior, ele expande.

Guilherme: Se você está no final de semana e já terminou todas as aulas da imersão, você já tem três filmes legais para assistir. São recomendações minhas, da Rafa e do Paulo. Só um detalhe que vale a pena comentar. Na parte de cima da página, nós deixamos o nome Aluraflix e colocamos filmes, mas, Paulo, nós poderíamos criar um catálogo de HQs preferidas, acho que faz sentido, ou cursos da Alura preferidos. Seria bem interessante ver outros exemplos.

Paulo: Você pode colocar, ao invés de filmes, animes, capas de jogos do playstation 5 que você quer ter, trazer algo da sua comunidade, uma lista de projetos, sites que ajudam a periferia com alguma causa, e montar esse conjunto. É possível, até, ir além do que aprendemos e colocar links, mas ainda não estudamos isso. Seria necessário, ao menos, criar duas listas.

Enfim, até dá para fazer com o que estudamos, porém, fica mais complicado criar outro array e outra lista para ter os links e que eles sejam clicáveis, aprender a usar a tag "a". Eu também pedirei para Lucas apresentar algumas imagens de projetos que também chamamos de Aluraflix que as pessoas desenvolveram nas imersões mais avançadas da Alura, usando React, JavaScript, *frameworks* e outros.

Nestes casos, realmente temos o catálogo, podemos *scrollar*, como no Netflix, clicamos e abrimos coisas, cadastramos. De fato, é mais complicado, com servidor, mas, perceba como muitas pessoas criaram o próprio Aluraflix, não só na Imersão Dev, como em outras imersões, e levaram isso para algo muito pessoal. Isso vai trazer o seu portfolio.

É muito bom poder dizer que fez determinada coisa e colocou no ar. Na próxima semana, inclusive, aprenderemos a colocar no ar, para não ficar restrito ao CodePen, para que seja o seu “site.com.br”, por exemplo.

Então, comece a pensar nisso, não só nos seus filmes favoritos, mas em outras coisas que tenham imagens, em um catálogo, que pode ser de pessoas que você segue e admira como profissionais no Instagram, pode ser sites que você gosta de aprendizado, de conteúdo gratuito, canais no YouTube de programação gratuitos, ou seja, existem muitas coisas que você pode usar.

O que eu vou deixar, especificamente em relação à código, é que você pesquise algo em relação ao `for`. O `for` que o Guilherme e a Rafa escreveram cria uma variável que vale zero, verifica se ela é menor que determinado valor e soma mais um no final do laço. Ele é muito clássico.

Existem outras formas de percorrer também uma lista. Há, por exemplo, um `for` que não usa o "i", passa por elementos e as pessoas o conhecem como `.for` e o `&`, que é outra forma de percorrer. Mais uma vez, reforço que existem outras formas de resolução de um mesmo problema em programação.

Há até o `while`, que ainda não estudamos, mas que também é uma estrutura de laço que poderíamos usar para fazer o mesmo no nosso código, alterando apenas alguns detalhes. Então, fica o incentivo para você pesquisar outras formas de iterar uma coleção de objetos, nesse caso, numa *array*, uma lista.

Rafaella: Perfeito! Eu tenho um desafio também, e ele envolve um pouco do que estudamos nas aulas anteriores que é: nós podemos fazer uma condição, usando o `if`, para não adicionarmos filmes repetidos. Então, entrando com um link ou uma URL de um filme que você já havia adicionado antes, ele te dará uma mensagem "Você já adicionou esse filme" ou algo parecido. Colocar essa condição, essa verificação, seria bem interessante.

Guilherme: Muito legal! Já que todos estão passando um desafio, vou passar um desafio mais maluco ainda. Nós colocamos os filmes, as imagens dos filmes que temos, quando criamos a nossa lista. E se isso acontecesse na tela? Se tivéssemos um campo onde passássemos a imagem, apertássemos um botão e esse filme já aparece. Isso seria muito mais legal de acontecer, seria mais dinâmico.

Rafaella: Seria interessante! Muito bom! Então é isso, pessoal. Essa foi a nossa quarta aula da imersão Dev. Espero que vocês tenham gostado! Nós estudamos muitas coisas, por exemplo, como funciona o `for`, que é o laço de repetição. É um pouco complicado no começo, mas tenho certeza de que, com o tempo, vocês compreenderão o que é cada uma das coisas, já que elas não obedecem uma ordem exata.

Nos vemos amanhã! Espero que vocês também tenham tranquilidade para estudar. Se não conseguiram pegar uma aula inteira ou as anteriores, faça com calma, nós vamos até o final da imersão e você pode ir fazendo. Não se esqueça de colocar as *tags* nos projetos para que possamos ver, usem o Pen Details, Imersão Dev, Alura. Não deixe de postar no Discord também, nós queremos muito ver nos canais.

Além das redes sociais! Eu já estou vendo muitas pessoas postando no LinkedIn, tem muito legal compartilhar. É isso! Até amanhã! E bons estudos!

Guilherme: Tchau!!