**Para saber mais: Promises em JavaScript**

Para entender melhor as Promises, precisamos primeiro conhecer o conceito de programação assíncrona:

Na programação síncrona, o código é executado linha por linha, ou seja, uma linha de código não começará a ser executada até que a linha anterior tenha sido totalmente concluída. No entanto, na programação assíncrona, o código não espera que uma linha seja concluída antes de passar para a próxima. Isso é muito útil quando estamos realizando tarefas que podem demorar algum tempo, como carregar uma imagem ou buscar dados de um servidor.

As Promises em JavaScript são objetos que representam o estado de uma operação assíncrona. Uma Promise pode estar em um de três estados: pendente(estado inicial, nem cumprido nem rejeitado), resolvida/fulfilled (a operação assíncrona foi concluída com sucesso) ou rejeitada/rejected (a operação assíncrona falhou). Podemos criar uma Promise da seguinte forma:

**let** promise = **new** **Promise**((resolve, reject) => {

// código assíncrono aqui

**if** (/\* tudo ocorreu bem \*/) {

**resolve**("funcionou!");

} **else** {

**reject**("houve um erro");

}

});

No exemplo acima, temos uma função que recebe dois parâmetros: resolve e reject, que são funções que serão chamadas quando a Promise for resolvida ou rejeitada, respectivamente.

Depois que a Promise é retornada, podemos lidar com seu resultado utilizando o método then para o caso de sucesso (resolvida) e o método catch para o caso de erro (rejeitada):

promise.**then**((message) => {

console.**log**("Sucesso: " + message);

}).**catch**((message) => {

console.**error**("Erro: " + message);

});

É importante lembrar que a promise só pode ser resolvida ou rejeitada uma vez. Se nós resolvermos e depois rejeitarmos uma mesma promise, a rejeição será ignorada.

Além disso, pode ser um desafio lidar com várias Promises. Para isso, temos disponíveis as funções Promise.all e Promise.race:

* Promise.all: recebe um array de Promises e retorna uma nova **promise que** resolve quando todas as Promises do array foram resolvidas ou uma delas foi rejeitada.
* Promise.race: retorna uma **promise que** se resolve ou rejeita com a resolução ou rejeição da primeira promiseno array a se resolver.

As Promises são uma parte importante do JavaScript moderno e são usadas em muitas bibliotecas e frameworks populares, como React e Angular. Ao entender como as Promises funcionam, você terá mais facilidade para lidar com código assíncrono efetivamente.

Para se aprofundar mais, acesse a documentação de [Promises](https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Promise" \t "_blank).

Como dito na aula, se você quiser conhecer todos status HTTP que podem ser retornados em uma response, você pode acessar o [HTTP Cats](https://http.cat/), que apresenta a descrição de todos os status com fotos de gatinhos ��