**Promises**

É um objeto que vai lidar com uma **operações assíncronas**, ou seja, para lidarmos com operações cujo tempo é indefinido e quero que a operação não pare a execução do script.

**Sintaxe de uma promise – primeira parte:**

***Let myPromise = new Promise(***

***function(myResolve, myReject){***

*// código que será executado*

***myResolve ()***

***myReject ()***

***}***

***)***

Depois que o código a ser executado chega a algum **resultado**, uma das funções de callback é chamada**: myResolve()** ou **myReject()**

* **myResolve()** => retorna um valor
* **myreject()** => retorna um objeto de erro

**Propriedades do Objeto Promise**

O objeto promise tem 2 propriedades importantes.

**Promise.state** => define o estado da operação, que pode ter 3 valores:

* **“pending”** => operação ainda está sendo executada;
* **“fulfilled”** => operação realizada com sucesso;
* **“rejected”** => operação falhou.

**Promise.result** => depedendo do estado da Promise, pode ser:

* Se estate for “pending” => result é **undefined**;
* Se estate for ‘fulfilled” => result é **algum valor**;
* Se estate for “rejected” => result será o **objeto error**

**Sintaxe para lidar com o resultado:**

O método then() aceita 2 parâmetros, que são duas funções callback, uma para sucesso e outra para falha. Then é opcional.

***myPromise.then(***

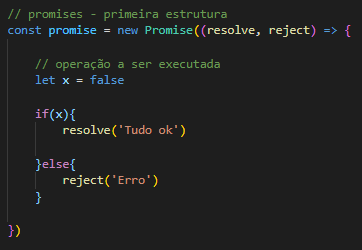
***function(valor){},***

***function(error){}***

***)***

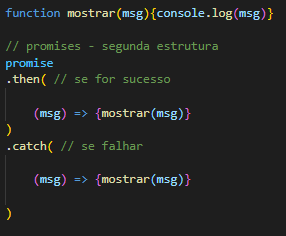
**Primeira parte:** uma operação(bloco de códigos será executado) dentro da arrow function que tem dois argumentos: duas funções de callback – resolve e reject.

Dependendo do resultado da operação, o callback resolve recebe um valor e reject recebe outro valor.



**Segunda parte:** as funções de callback resolve e reject retorna o resultado da operação, dependendo de como foi definido. Para pegar os valores retornados, precisamos criar duas estruturas:

**.then(**

**function(valor1){fazer alguma coisa com o valor1}**

**)**

**O que retornou do callback resolve() é o valor1**

**.catch(**

**function(valor2){ fazer alguma coisa com o valor2}**

**)**

**O que retornou do callback reject() é o valor 2.**

**Encadeamento de then**

É possível enviar o resultado de um resolve para outro **then** logo abaixo e assim por diante.

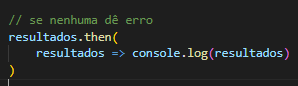
**Ex.:** enviando o valor retornado do primeiro resolve para o the logo abaixo

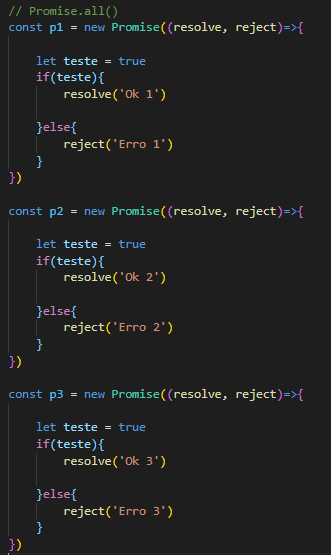




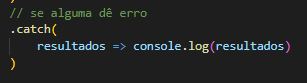
**Promise.all():** método do objeto promise que diz que um array de promises DEVE resolver em sucesso. Se alguma delas der erro, a promise do conjunto dará erro.

**Ex.:** um array com todos os valores dos resolves de cada promise será enviado para um array. Se algumas das promises dê erro, vai retorna o valor do reject. Caso haja mais de um erro, enviará o reject da primeira promise que der erro.







**j**

**Promise.race():** retorna o valor do reject ou resolve da promise que resolver ou falhar primeiro.

**Ex.:** perceba que cada promise tem um tempo de execução diferente. A p2 é a mais rápida a resolver, portanto será a primeira a terminar.

