Rodrigo Branas – @rodrigobranas - http://www.agilecode.com.br

Promises

A linguagem JavaScript não tem suporte para threads e não é bloqueante. Isso quer dizer que eventos externos, que realizam operações de I/O, não devem bloquear o fluxo de execução.

Uma promise é basicamente um objeto que pode fornecer um resultado no futuro, podendo estar em 3 estados: fulfilled, rejected ou pending.

```
1.  let delayedSum = function (a, b) {
2.    setTimeout(function () {
3.      return a + b;
4.    }, 1000);
5.  };
```

Criando uma função com um retorno assíncrono

```
1.  let delayedSum = function (a, b, callback) {
2.    setTimeout(function () {
3.      callback(a + b);
4.    }, 1000);
5.  };
6.  delayedSum(2, 2, function (result) {
7.    console.log(result);
8.  });
```

Criando uma função com um retorno assíncrono em um callback

```
1.  let delayedSum = function (a, b) {
2.    return new Promise(function (resolve, reject) {
3.      if (!a || !b) return reject("Invalid input");
4.      setTimeout(function () {
5.        resolve(a + b);
6.      }, 1000);
7.    });
8.  };
9.  delayedSum(2,2).then(
10.   result => console.log(result)
11. ).catch(
12.   e => console.log(e)
13. );
```

Criando uma função com um retorno assíncrono em um callback