ORIGAMID

JAVASCRIPT ASSÍNCRONO

ASYNC e AWAIT

async / await

A palavra chave `async` indica que a função possui partes assíncronas e que você pretende esperar a resolução da mesma antes de continuar. O await irá indicar a promise que devemos esperar. Faz parte do ES8.

```
async function puxarDados() {
  const dadosResponse = await fetch('./dados.json');
  const dadosJSON = await dadosResponse.json();

  document.body.innerText = dadosJSON.titulo;
}

puxarDados();
```

then / async

A diferença é uma sintaxe mais limpa.

```
function iniciarFetch() {
  fetch('./dados.json')
  .then(dadosResponse => dadosResponse.json())
  .then(dadosJSON => {
    document.body.innerText = dadosJSON.titulo;
  })
}
iniciarFetch();

async function iniciarAsync() {
  const dadosResponse = await fetch('./dados.json');
  const dadosJSON = await dadosResponse.json();
  document.body.innerText = dadosJSON.titulo;
}
iniciarAsync();
```

Try / Catch

Para lidarmos com erros nas promises, podemos utilizar o `try` e o `catch` na função.

```
async function puxarDados() {
  try {
    const dadosResponse = await fetch('./dados.json');
    const dadosJSON = await dadosResponse.json();
    document.body.innerText = dadosJSON.titulo;
  }
  catch(erro) {
    console.log(erro);
  }
}
puxarDados();
```

Iniciar Fetch ao Mesmo Tempo

Não precisamos esperar um fetch para começarmos outro. Porém precisamos esperar a resposta resolvida do fetch para transformarmos a response em `json`.

```
async function iniciarAsync() {
  const dadosResponse = fetch('./dados.json');
  const clientesResponse = fetch('./clientes.json');

  // ele espera o que está dentro da expressão () ocorrer primeiro
  const dadosJSON = await (await dadosResponse).json();
  const clientesJSON = await (await clientesResponse).json();
}
iniciarAsync();
```

Promise

O resultado da expressão à frente de await tem que ser uma promise. E o retorno do await será sempre o resultado desta promise.

```
async function asyncSemPromise() {
  // Console não irá esperar.
  await setTimeout(() => console.log('Depois de 1s'), 1000);
  console.log('acabou');
}
asyncSemPromise();

async function iniciarAsync() {
  await new Promise(resolve => {
    setTimeout(() => resolve(), 1000)
  });
  console.log('Depois de 1s');
}
iniciarAsync();
```