escola britânica de artes criativas & tecnologia

Engenheiro Front-End

Ajax e Exceções

Durante este módulo construímos mais um projeto, fizemos uma landing page para um evento fictício: EBAC Tech Talks.

Você pode consultar o código escrito durante o módulo clicando aqui.

Exceções

Uma exceção ocorre quando existe um erro no código e interrompe a execução do programa.

O erro pode ser causado por uma falha na programação ou simplesmente um erro de digitação, experimente:

```
Const x = 10
X = 15
```

Este código causará um erro, pois estamos tentando modificar o valor de uma constante.

Exceções

O fluxo de execução do JavaScript é bloqueante, leve em consideração o código a seguir:

```
1 // console.log(“começo”);
2 // Const x = 10;
3 // x = 15;
4 // console.log(~fim~);
```

Por temos um código com bug, tentativa de alterar o valor de uma constante, nunca chegaremos ao passo 4, o que irá bloquear a execução do programa.

Exceções

Para que o fluxo de execução de um programa não seja interrompido precisamos aprender a tratar códigos passíveis de exceção, para isso possuímos os blocos try e catch.

No bloco **try (tentar)** inserimos o código que pode gerar uma exceção, utilizamos o verbo lançar para se referir a exceções.

Já no bloco catch fazemos a tratativa para a exceção.

Exceções

O exemplo anterior utilizando o **try...catch** ficaria:

```
1 // console.log("começo");
2 // try {
3 // const x = 10;
4 // x = 15;
5 // }
6 // catch(erro) {
7 // TRATAMENTO DA EXCEÇÃO
8 // }
9 // console.log("fim");
```

Exceções

Apesar de termos um erro no código, isso não irá interromper a execução do programa e podemos tratar o erro dentro do bloco catch.

O **bloco catch** é útil para darmos um feedback ao usuário quando algo inesperado acontecer. No bloco catch recebemos um argumento, que é o próprio erro, ele é um objeto e podemos acessar as propriedades: **name, message e stack**.

Propriedade name: retorna uma string com o nome do erro, como: **TypeError, SyntaxError ou Error**;

Propriedade message: retorna uma string com a mensagem do erro;

Propriedade stack: retorna uma string com o nome e mensagem do erro, além de alguns detalhes, como a linha em que o erro aconteceu.

Exceções

As exceções podem ser causadas por falhas no código, problemas com recursos externo e até mesmo podemos causá-las de forma propositalmente.

Imagine o cenário onde temos uma função no JavaScript que é responsável por fazer a transferência de saldo entre contas bancárias.

Para este cenário podemos ter uma função como essa:

```
function transferir(de, para, quantia) {
    de.saldo -= quantia; // retiramos a quantia
    para.saldo += quantia; // inserimos a quantia
}
```

Exceções

Temos um problema neste código, não verificamos se a pessoa que está fazendo a transferência tem dinheiro suficiente para isso, então inserimos este if:

```
if (de.saldo >= quantia) {
    de.saldo -= quantia;
    para.saldo += quantia;
}
```

Agora no else, caso o saldo do remetente seja insuficiente podemos provocar uma exceção:

```
if (de.saldo >= quantia) {
     de.saldo -= quantia;
     para.saldo += quantia;
} else {
    throw new Error("Saldo insuficiente");
}
```

Exceções

Utilizamos a palavra reservada **throw** para interromper o fluxo de execução de um programa e em seguida criamos o erro com **new Error,** passando como argumento a mensagem que queremos transmitir.

Chamamos isso de **lançar uma exceção.**

AJAX

AJAX é o acrônimo para **Asycronous JavaScript** and XML, apesar do XML no nome, essa tecnologia é compatível com outros formatos de arquivos.

O AJAX possibilita buscas e envios de dados para o servidor sem precisar recarregar a página. Isso foi uma grande revolução e teve uma enorme melhoria na experiência do usuário, afinal agora ele não precisaria esperar a página toda recarregar para que um formulário fosse enviado, por exemplo.

As requisições, o envio e a solicitação de dados na Internet é feita utilizando o HTTP **(Hypertext transfer protocol),** quando o navegador solicita uma página HTML para o servidor.

Essa solicitação, também chamada de **requisição**, é feita utilizando o protocolo HTTP.

AJAX

Uma chamada **HTTP** é composta por um verbo, cabeçalhos e pode conter um corpo (conjunto de informações).

Os principais verbos HTTP são:

GET: utilizado para solicitar um dado do servidor;

POST: quando desejamos enviar dados para o servidor;

PUT: utilizado para alterar todos os dados de uma informação;

PATCH: utilizado para alterar parcialmente os dados de uma informação;

DELETE: utilizado para deletar um recurso;

AJAX

Quando fazemos uma requisição HTTP obrigatoriamente teremos um código de status da requisição, informando se ela deu certo ou não. Também chamamos esses códigos de **"status code",** os status codes respeitam um intervalo:

100 – 199: indica um status de informação;
200 – 299: indica um status de sucesso;
300 – 399: indica um status de redirecionamento;
400 – 499: indica um status de erro por parte do cliente
500 – 599: indica um erro por parte do servidor;

Os status codes mais comuns estão em **2XX, 4xx e 5XX.**

Exemplos:

Quando solicitamos um recursos que não existe ao servidor recebemos um **status code 404**, que indica um erro por parte do cliente, do solicitante.