ORIGAMID

OBJETOS

Native, Host e User

Native

Objetos nativos são aqueles definidos na especificação da linguagem e são implementados independente do host.

```
// Construtores de objetos nativos
Object
String
Array
Function
```

Host

Objetos do host são aqueles implementados pelo próprio ambiente. Por exemplo no browser possuímos objetos do DOM, como DomList, HTMLCollection e outros. Em Node.js os objetos do Host são diferentes, já que não estamos em um ambiente do browser.

```
// Objetos do browser
NodeList
HTMLCollection
Element
```

User

Objetos do user, são os objetos definidos pelo seu aplicativo. Ou seja, qualquer objeto que você criar ou que importar de alguma biblioteca externa.

```
const Pessoa = {
  nome: 'André';
}
```

Diferentes Versões

- ### Browsers diferentes

  Apesar de tentarem ao máximo manter um padrão, browsers diferentes possuem objetos com propriedades e métodos diferentes.

- ### Versões de browsers

  Sempre que o browser é atualizado, novos objetos, métodos e propriedades podem ser implementados.

- ### Host e Native Objects

  Por exemplo, browsers que não implementaram o ECMAScript 2015 (ES6), não possuem o método find de Array.

Versões de JavaScript

- ## ECMA

  Organização responsável por definir padrões para tecnologias. ECMAScript é o padrão de JavaScript.

- ## ECMAScript 2015 ou ES6

  ES é uma abreviação de ECMAScript, ES6 é a sexta versão do ECMAScript, que foi lançada em 2015. Por isso ECMAScript 2015 é igual a ES6. A partir da ES6, existe uma tendência anual de atualizações. ECMAScript 2015, 2016, 2017, 2018 e Next.

- ## Engine

  Existem diversas engines que implementam o ECMAScript como V8, SpiderMonkey, Chakra, JavaScriptCore e mais.

Bibliotecas

Bibliotecas como jQuery foram criadas para resolver o problema de inconsistências entre browsers e adicionar funcionalidades que não existiam nativamente. A padronização dos browsers e a implementação de soluções nativas, torna as mesmas obsoletas.

```
$('a').addClass('ativo');
$('a').hide();
$('a').show();
```

Verificar se Existe

O typeof retorna o tipo de dado. Caso esse dado não exista ou não tenha sido definido, ele irá retornar undefined. Ou seja, quando não for undefined quer dizer que existe.

```
if (typeof Array.from !== "undefined")
if (typeof NodeList !== "undefined");
```

API

Application Programming Interface, é uma interface de software criada para a interação com outros softwares. Ou seja, toda interação que fazemos com o browser utilizando Objetos, Métodos e Propriedades, estamos na verdade interagindo com a API do browser.

Exercícios

```
// Liste 5 objetos nativos

// Liste 5 objetos do browser

// Liste 2 Métodos, Propriedades ou Objetos
// presentes no Chrome que não existem no Firefox
```