

JSON

Prof. Rodrigo Macedo

Escopo do Curso

- Conceituação Geral Javascript: Tipos de dados e estruturas.
- Surgimento JSON.
- Objetos JSON.
- JSON x XML.
- Questões de concursos



Javascript

- Linguagem de script que trabalha para desenvolvimento web.
- Feita em 1995 para o Netscape 2.0 sob o nome de Mocha depois LiveScript e por fim, Javascript (segundo a especificação ECMA Script).
- O idealizador da linguagem foi Brendan Eich.
- JSON = Javascript Object Notation.
- Linguagem fracamente tipada.
- Tipagem dinâmica.



Exemplo código Javascript

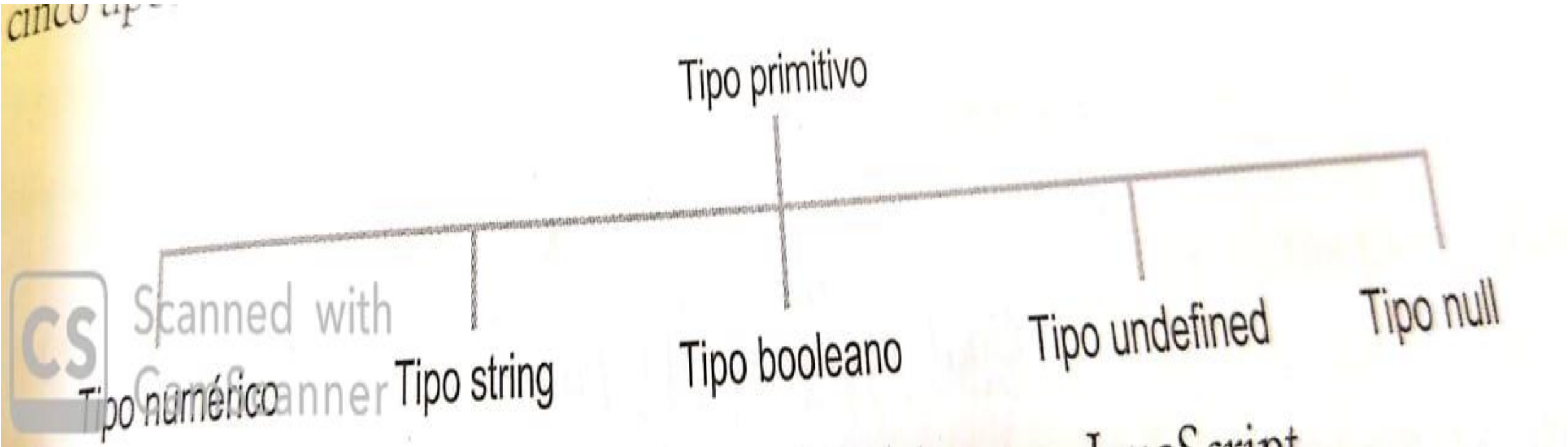
```
1 var welcomeMessage = "Hello World";  
2 //Linhas com '/' são usadas para comentários  
3 console.log( welcomeMessage );    //exibe Hello World no con  
4 console.log("A");    //exibe o caractere A  
5 console.log( 2+5 );    //exibe o número 7  
6  
7 console.log("goodbye" + " " + "all");    //exibe goodbye a
```



Scanned with
CamScanner

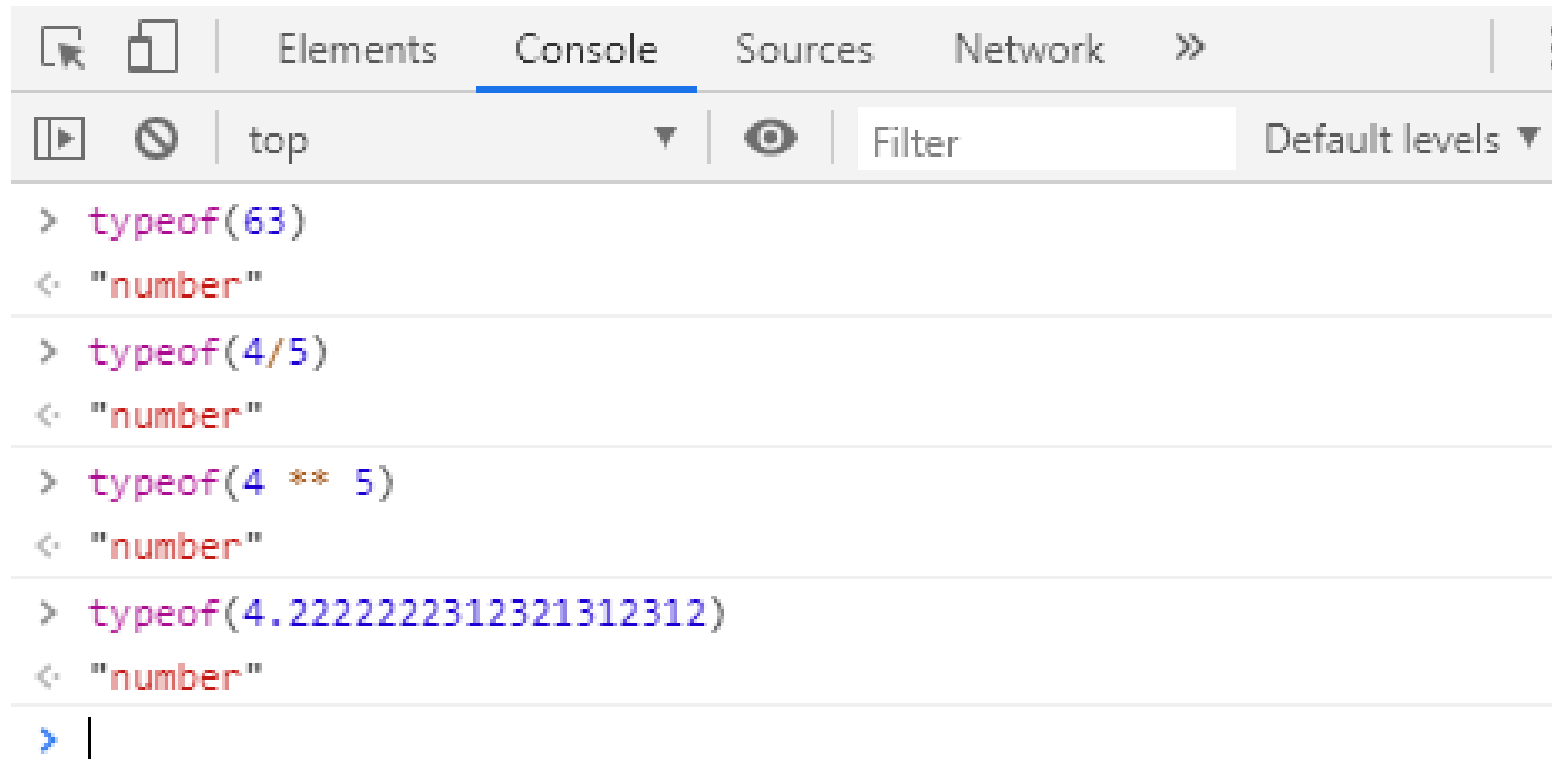
Tipos Primitivos

- Representam as possibilidades de tipificação nos dados em Javascript.



Numérico

- Representa todos os valores possíveis numéricos: frações, números inteiros, exponenciais, dentre outros.



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays four JavaScript commands and their corresponding outputs, all of which return the string "number".

```
> typeof(63)
< "number"

> typeof(4/5)
< "number"

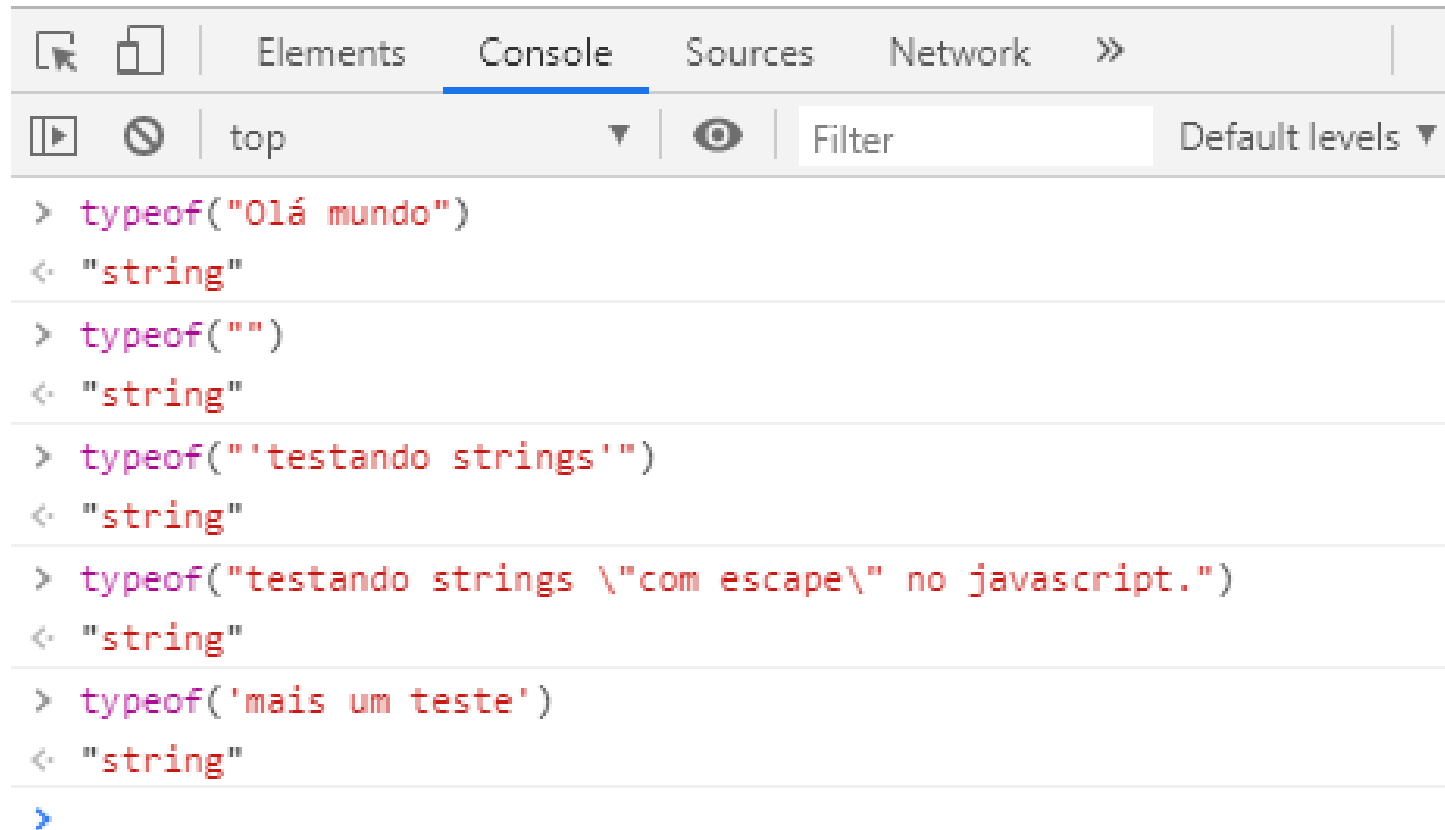
> typeof(4 ** 5)
< "number"

> typeof(4.2222222312321312312)
< "number"

> |
```

Strings

- Corresponde a uma representação finita que inclui zero ou mais caracteres Unicode.



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays a series of JavaScript commands and their outputs, demonstrating that various string literals are all of type 'string'.

```
> typeof("Olá mundo")
< "string"

> typeof("")
< "string"

> typeof("'testando strings'")
< "string"

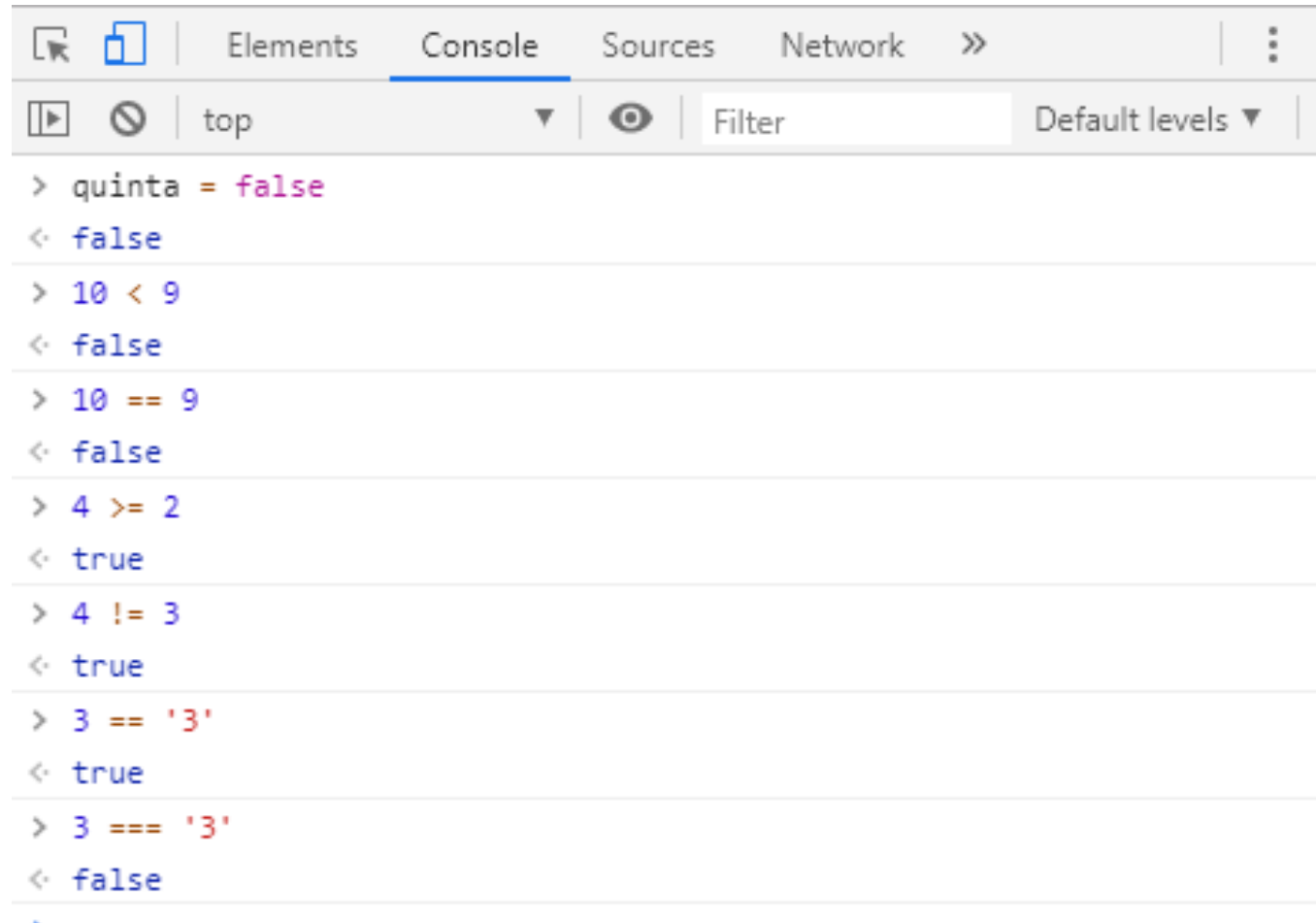
> typeof("testando strings \"com escape\" no javascript.")
< "string"

> typeof('mais um teste')
< "string"

>
```

Booleano

- Representa um valor lógico, constituindo-se de dois valores possíveis: verdadeiro e falso.



```
> quinta = false
< false

> 10 < 9
< false

> 10 == 9
< false

> 4 >= 2
< true

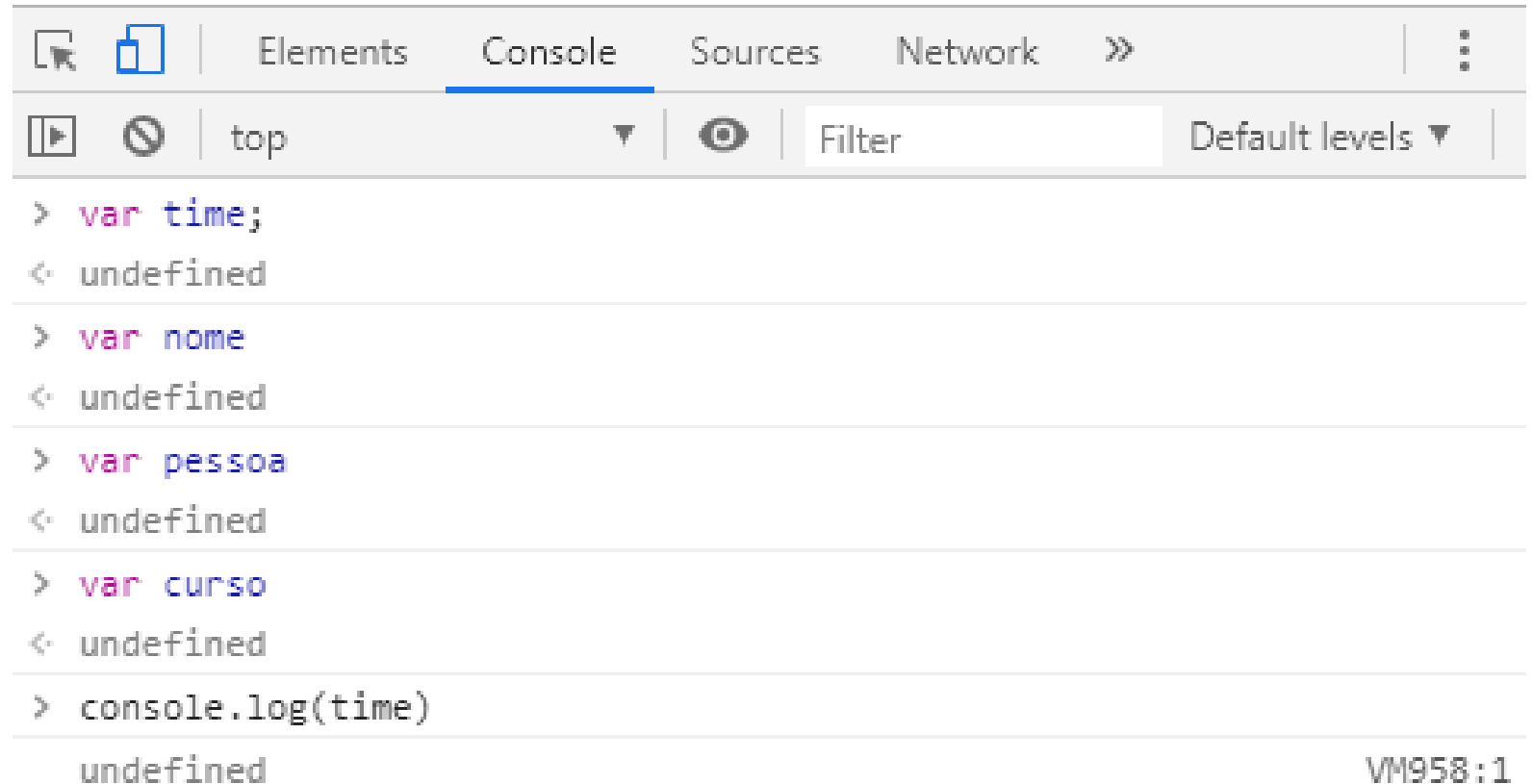
> 4 != 3
< true

> 3 == '3'
< true

> 3 === '3'
< false
```


Undefined

- É usado para representar o caso em que um identificador ainda não tem nenhum valor atribuído.



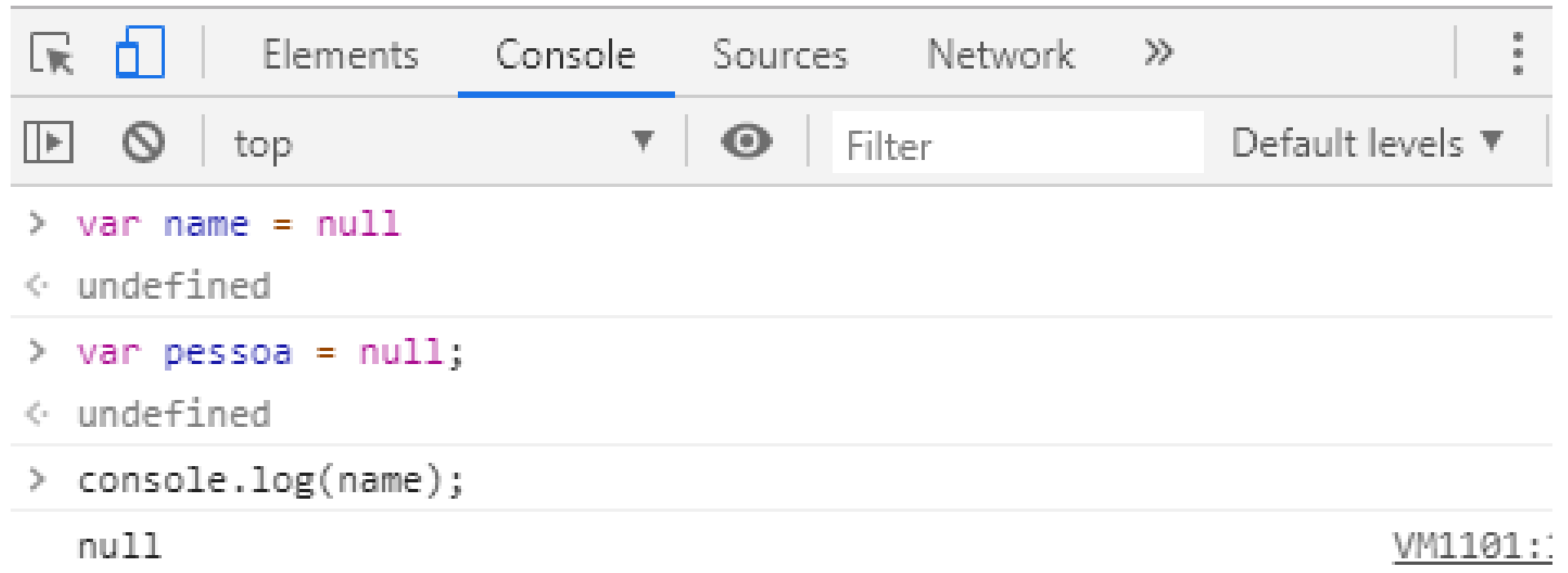
The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays the following sequence of commands and their results:

```
> var time;  
< undefined  
  
> var nome  
< undefined  
  
> var pessoa  
< undefined  
  
> var curso  
< undefined  
  
> console.log(time)  
undefined
```

The bottom right corner of the console shows the text VM958:1.

Null

- Representa a ausência intencional de um valor. É usado explicitamente para fazer referência vazia ou inexistente.



The screenshot shows a web browser's developer console with the 'Console' tab selected. The console displays three lines of JavaScript code and their corresponding outputs. The first line is `> var name = null`, which outputs `< undefined`. The second line is `> var pessoa = null;`, which also outputs `< undefined`. The third line is `> console.log(name);`, which outputs `null`. The console interface includes a toolbar with icons for opening the console, disabling it, and a 'top' button. There is also a 'Filter' input field and a 'Default levels' dropdown menu. The bottom right corner of the console shows the file path `VM1101:1`.

```
> var name = null
< undefined

> var pessoa = null;
< undefined

> console.log(name);
null
```

VM1101:1

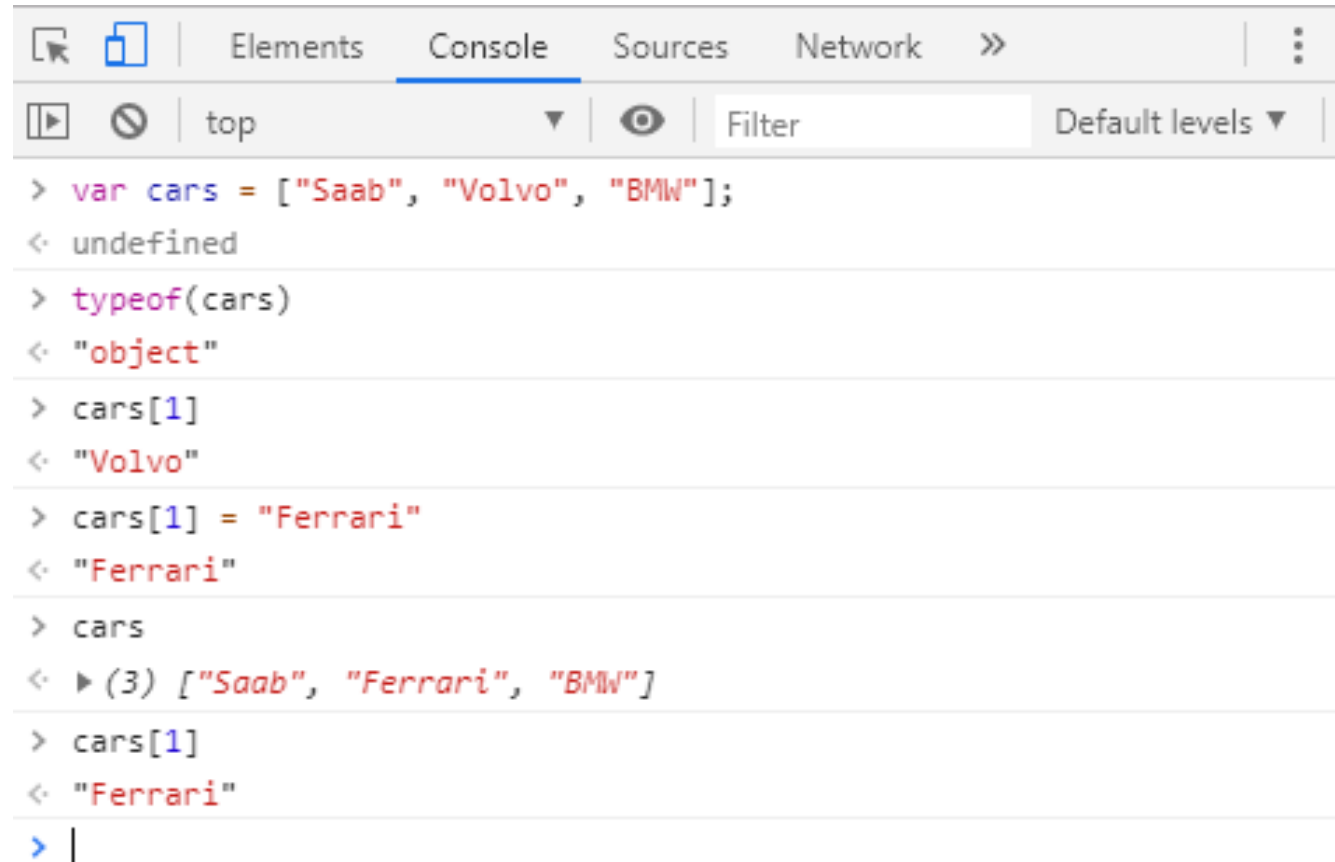
Operadores de Comparação

Operador de comparação	Operador	Descrição
Menor que	<	Usado para determinar se o operando da esquerda é menor que o operando da direita.
Maior que	>	Usado para determinar se o operando da esquerda é maior que o operando da direita.
Menor ou igual a	<=	Usado para determinar se o operando da esquerda é menor ou igual ao operando da direita.
Maior ou igual a	>=	Usado para determinar se o operando da esquerda é maior ou igual ao operando da direita.
Igual	==	Usado para determinar se o operando da esquerda é igual ao operando da direita.
Diferente de	!=	Usado para determinar se o operando da esquerda é diferente do operando da direita.
Estritamente igual	===	Verifica se os dois operandos são iguais sem permitir que haja qualquer coerção de tipo.
Estritamente diferente	!==	Verifica se os dois operandos são diferentes sem permitir que haja qualquer coerção de tipo.

Array

- O objeto Array do JavaScript é um objeto global usado na construção de 'arrays': objetos de alto nível semelhantes a listas.

Atenção a função
typeof



```
> var cars = ["Saab", "Volvo", "BMW"];
< undefined

> typeof(cars)
< "object"

> cars[1]
< "Volvo"

> cars[1] = "Ferrari"
< "Ferrari"

> cars
< ▶ (3) ["Saab", "Ferrari", "BMW"]

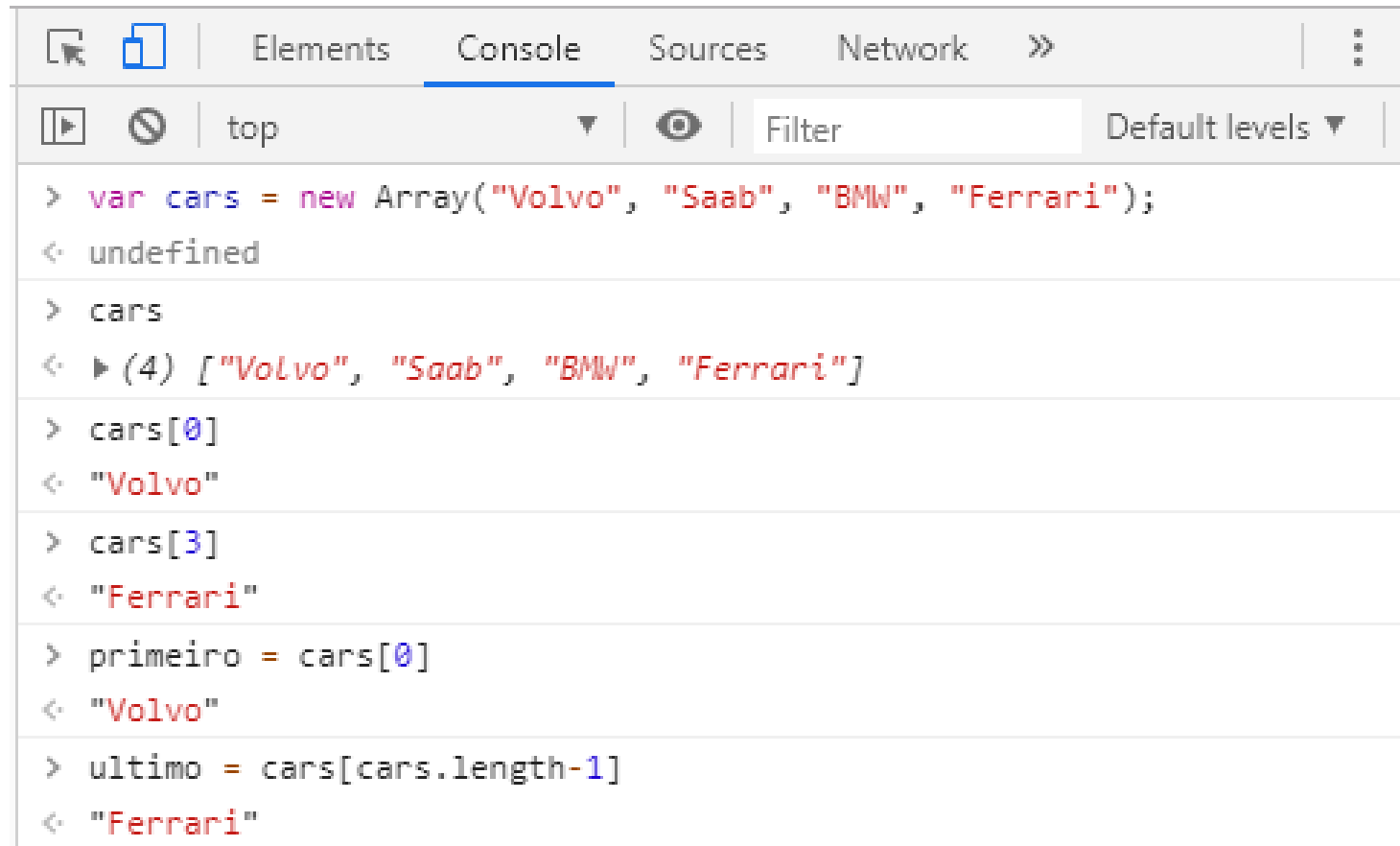
> cars[1]
< "Ferrari"

> |
```

Array

- O objeto Array do JavaScript é um objeto global usado na construção de 'arrays': objetos de alto nível semelhantes a listas.

Atenção a função
length



```
> var cars = new Array("Volvo", "Saab", "BMW", "Ferrari");
< undefined

> cars
< ▶ (4) ["Volvo", "Saab", "BMW", "Ferrari"]

> cars[0]
< "Volvo"

> cars[3]
< "Ferrari"

> primeiro = cars[0]
< "Volvo"

> ultimo = cars[cars.length-1]
< "Ferrari"
```

Array

Atenção a função
push, pop, shift e
unshift().

```
top Filter Default levels ▼  
> cars  
< ▶ (4) ["Volvo", "Saab", "BMW", "Ferrari"]  
> cars.push("McLaren")  
< 5  
> cars  
< ▶ (5) ["Volvo", "Saab", "BMW", "Ferrari", "McLaren"]  
> cars.pop()  
< "McLaren"  
> cars  
< ▶ (4) ["Volvo", "Saab", "BMW", "Ferrari"]  
> cars.unshift("McLaren")  
< 5  
> cars  
< ▶ (5) ["McLaren", "Volvo", "Saab", "BMW", "Ferrari"]  
> cars.shift()  
< "McLaren"  
> cars  
< ▶ (4) ["Volvo", "Saab", "BMW", "Ferrari"]
```

Objetos

- Um inicializador de objetos é uma lista delimitada por vírgula de zero ou mais pares de nomes de **propriedades** e **valores** associados a um objeto, entre chaves ({}).

O acesso a um elemento, pode ser feito por meio de sua propriedade (ou chave, também chamada).



```
> var object1 = {a: 'foo', b: 42, c: {}};
< undefined

> var object1 = {
  a: 'foo',
  b: 42,
  c: {}
};
< undefined

> object1;
< ▶ {a: "foo", b: 42, c: {}}

> object1.b
< 42

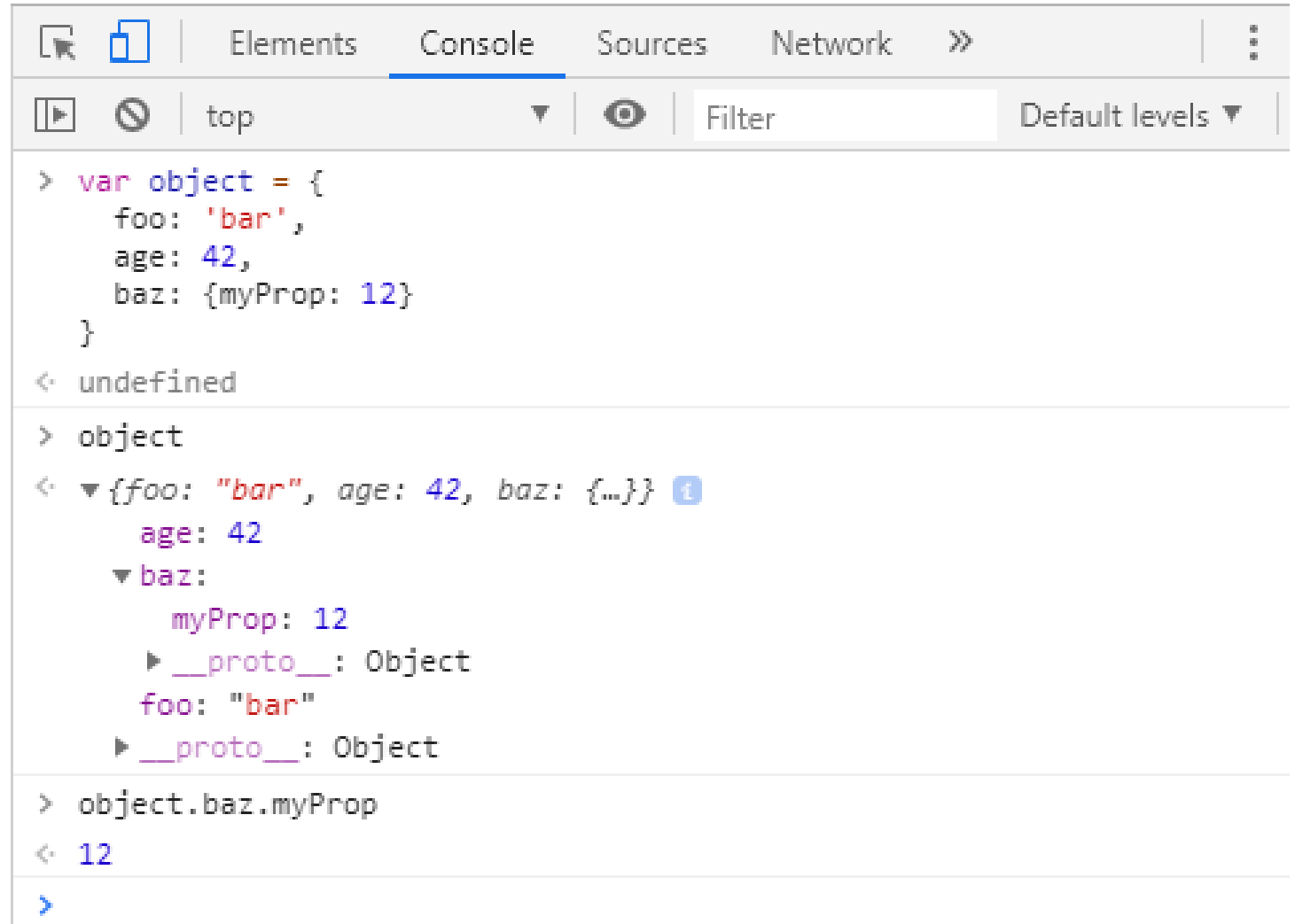
> object1.a
< "foo"

> object1.c
< ▶ {}
```

Objetos

- Um inicializador de objetos é uma lista delimitada por vírgula de zero ou mais pares de nomes de **propriedades** e **valores** associados a um objeto, entre chaves ({}).

Podemos ter também, propriedades, dentro de outras propriedades.



```
> var object = {
  foo: 'bar',
  age: 42,
  baz: {myProp: 12}
}
< undefined

> object
< {foo: "bar", age: 42, baz: {myProp: 12}}

> object.baz.myProp
< 12
```

The screenshot shows a browser's developer console with the 'Console' tab selected. It displays the execution of JavaScript code to create an object and access its nested properties. The first command creates an object with properties 'foo' (value 'bar'), 'age' (value 42), and 'baz' (an object with 'myProp' value 12). The second command logs the object, showing its structure. The third command accesses the nested property 'object.baz.myProp', which returns the value 12.

Array X Objetos

```
> var jogadores = [  
    {nome: "Igor", numero: 8, posicao: "meio"},  
    {nome: "Joao", numero: 4, posicao: "zagueiro"},  
    {nome: "Fabio", numero: 7, posicao: "atacante"},  
];  
< undefined  
  
> jogadores  
< ▼ (3) [{...}, {...}, {...}] ⓘ  
  ▶ 0: {nome: "Igor", numero: 8, posicao: "meio"}  
  ▶ 1: {nome: "Joao", numero: 4, posicao: "zagueiro"}  
  ▶ 2: {nome: "Fabio", numero: 7, posicao: "atacante"}  
    length: 3  
  ▶ __proto__: Array(0)  
  
> jogadores[0]  
< ▶ {nome: "Igor", numero: 8, posicao: "meio"}  
  
> |
```

Podemos ter também, objetos dentro de array, bem como o inverso, também é possível.

Array X Objetos



```
> jogadores
```

```
< ▼ (3) [{...}, {...}, {...}] ⓘ
```

- ▶ 0: {nome: "Igor", numero: 8, posicao: "meio"}
- ▶ 1: {nome: "Joao", numero: 4, posicao: "zagueiro"}
- ▶ 2: {nome: "Fabio", numero: 7, posicao: "atacante"}
- length: 3
- ▶ __proto__: Array(0)

```
> jogadores.unshift({nome: 'Rodrigo', numero: 10, posicao: 'atacante'});
```

```
< 4
```

```
> jogadores
```

```
< ▼ (4) [{...}, {...}, {...}, {...}] ⓘ
```

- ▶ 0: {nome: "Rodrigo", numero: 10, posicao: "atacante"}
- ▶ 1: {nome: "Igor", numero: 8, posicao: "meio"}
- ▶ 2: {nome: "Joao", numero: 4, posicao: "zagueiro"}
- ▶ 3: {nome: "Fabio", numero: 7, posicao: "atacante"}
- length: 4
- ▶ __proto__: Array(0)

```
> jogadores.pop()
```

Podemos ter também, objetos dentro de array, bem como o inverso, também é possível.

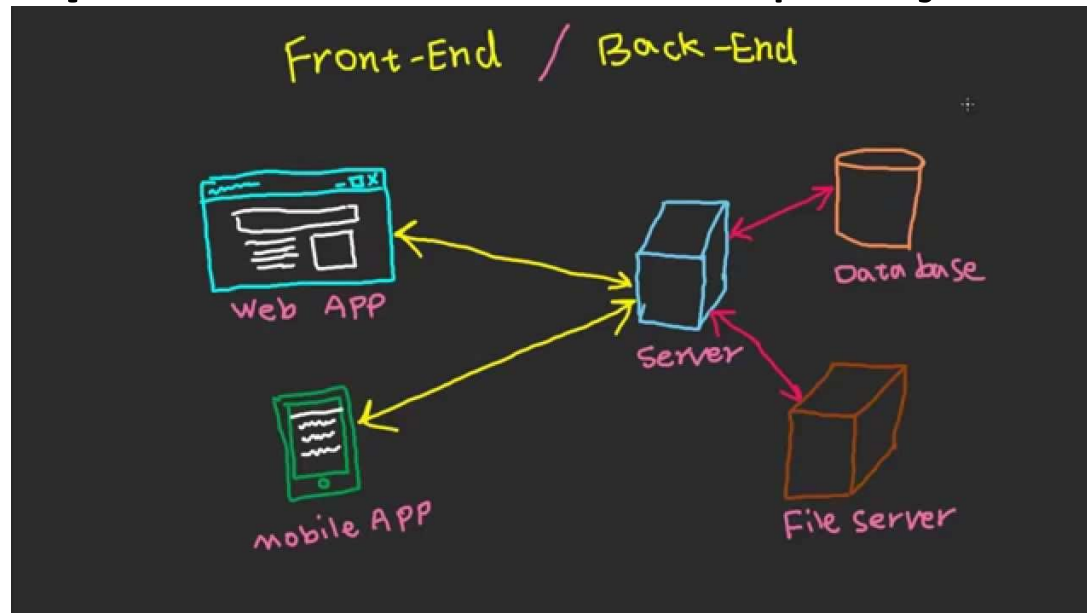
Welcome to Json

- O formato JSON (Javascript Object Notation) é derivado dos objetos literais da linguagem Javascript.
- Considerado subconjunto da linguagem Javascript, o JSON não tem nenhum recurso adicional que a linguagem Javascript já não tenha.
- Não é uma linguagem de programação, mas um **formato estruturado para troca de dados**.
- Devido a sua simplicidade atinge um alto nível de interoperabilidade entre aplicações.

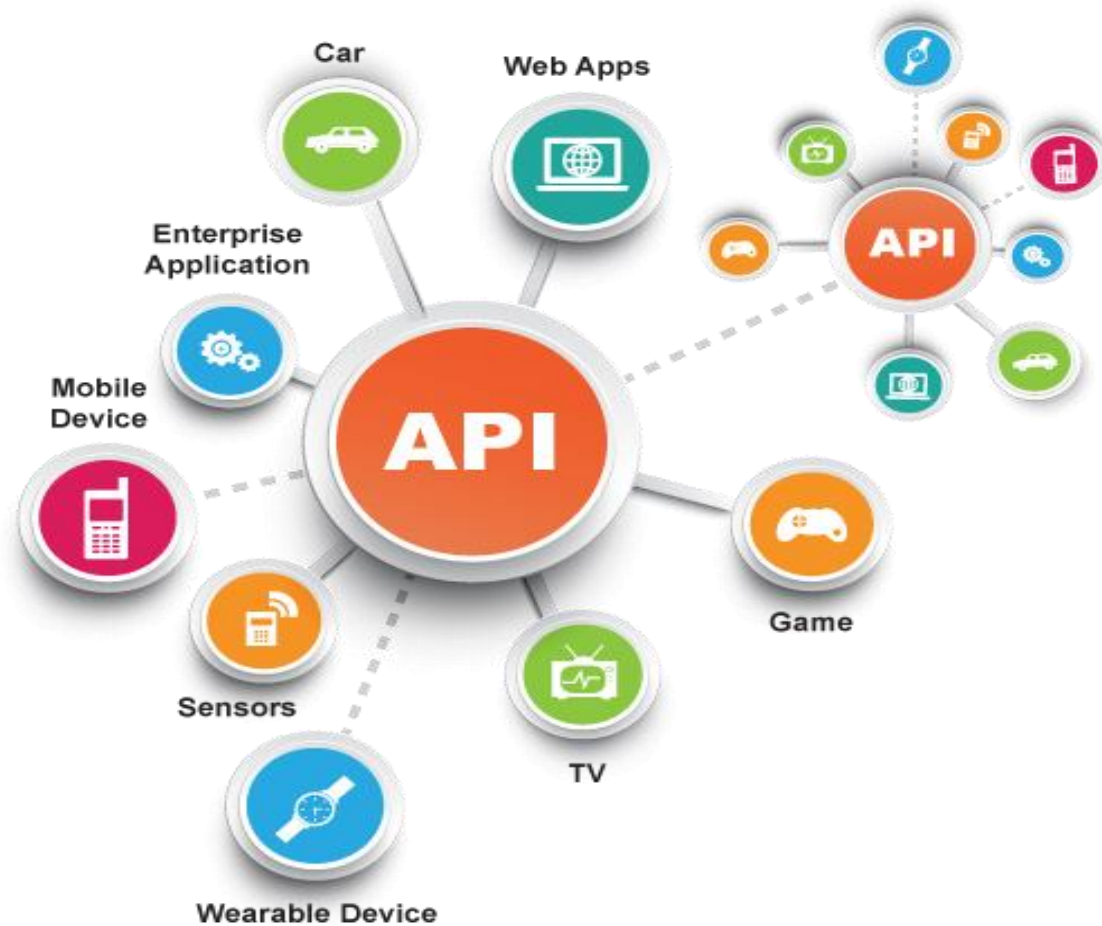


Interoperabilidade

- Conjunto de rotinas e padrões de programação para acesso a um aplicativo.
- Com o advento dos client-side frameworks, mais e mais desenvolvedores estão se preocupando em desenvolver o **back-end** que é **compartilhado** entre suas aplicações Web e outras aplicações Nativas



Interoperabilidade - Exemplos



Tipos de dados - JSON

- Strings.
- Number
- Boolean
- Array
- Object
- Null

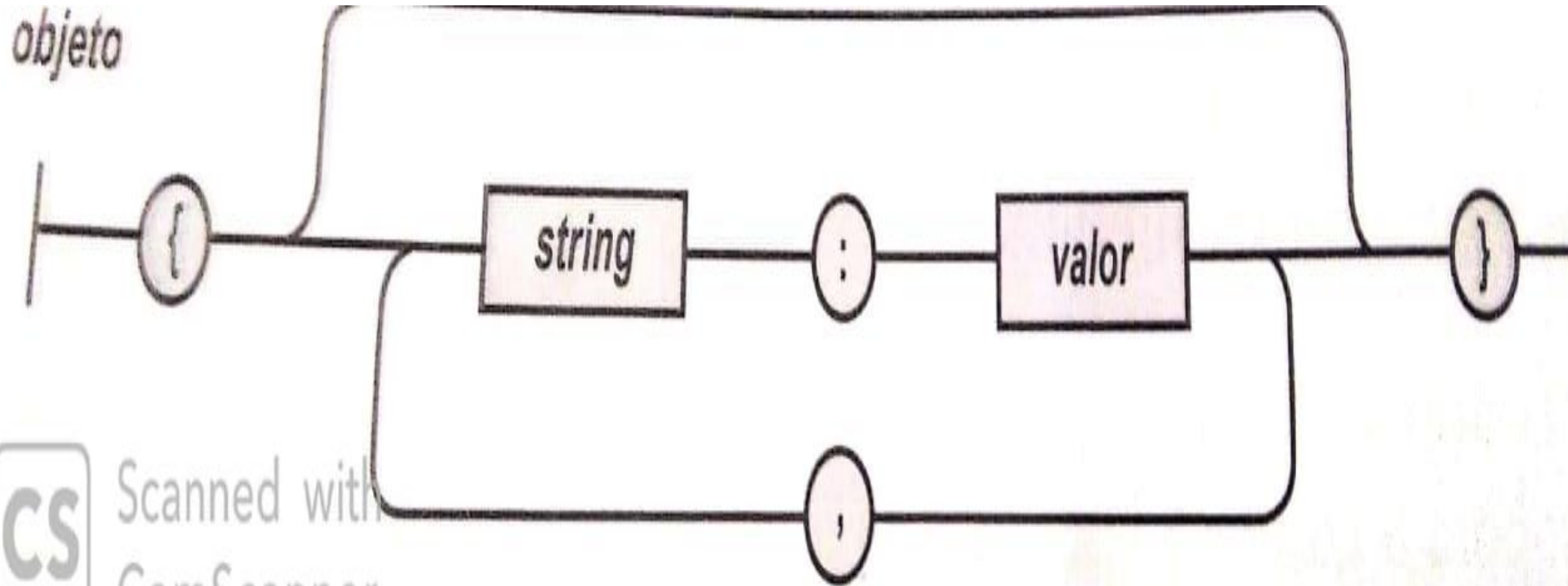


Gramática do JSON

- Em poucas palavras, o JSON é uma representação textual definida por um pequeno conjunto de regras segundo o qual os dados são estruturados. A especificação do JSON define que os dados podem ser estruturados de duas maneiras:
 1. Uma coleção de pares nome / valor (Objetos).
 2. Uma lista ordenada de valores (Array).

Estrutura JSON

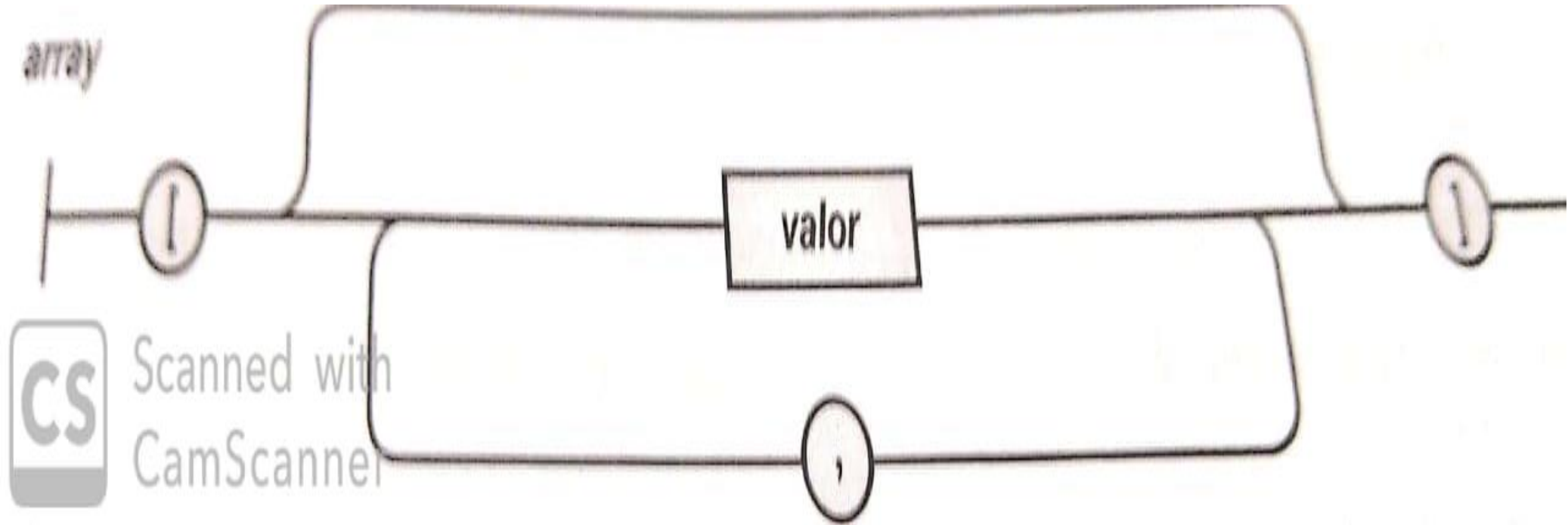
Coleção de
pares
nome /
valor.



Scanned with
CamScanner

Estrutura JSON

Coleção de
pares
nome /
valor.



Componentes estruturais JSON

Token	Literal	Nome
Início de array	[Colchete esquerdo
Fim de array]	Colchete direito
Início de objeto	{	Chave esquerda
Fim de objeto	}	Chave direita
Separador de nome/valor	:	Dois pontos
Separador de valor	,	Vírgula

Tecnologias e Linguagens

8th	ColdFusion	Nim
json	SerializeJSON	Module json
ActionScript	toJSON	Objective C
ActionScript3	D:	NSJSONSerialization
Ada	Libdjson	json-framework
GNATCOLL.JSON	Dart	JSONKit
AdvPL	json library	yajl-objc
JSON-ADVPL	Delphi	TouchJSON
ASP	Delphi Web Utils	OCaml
JSON for ASP	JSON Delphi Library	jsonm
JSON ASP utility class	E	PascalScript
AWK	JSON in TermL	JsonParser
JSON.awk	Fantom	Perl
rhawk	Json	CPAN
BlitzMax	FileMaker	Photoshop
bmx-rjson	JSON	JSON Photoshop Scripting
C	Fortran	PHP
JSON_checker	json-fortran	PHP 5.2
YAJL	YAJL-Fort	PicoLisp
LibU	Go	picolisp-json
json-c	package json	Pike
json-parser	Groovy	Public.Parser.JSON
jsonsl	groovy-io	Public.Parser.JSON2
WJElement	Haskell	PL/SQL
M's JSON parser	RJson package	pljson
cJSON	json package	Prolog
Jansson	Java	Jekejeke
jsmn	JSON-java	PureBasic
parson	JSONUtil	JSON
ujson4c	jsonp	Puredata
nxjson	Json-lib	PuRestJson
frozen	Stringtree	Python
microjson	SOJO	The Python Standard Library
mjson	json-taglib	simplejson
progbase	Flexjson	pyson
C++	Argo	Yajl-Py
JSONKit	jsonij	ultrajson
jsonme--	fastjson	metamagic.json
ThorsSerializer	mjson	progbase
JsonBox	jjson	R
jvar	json-simple	rjson

Case – Classificação raças de cachorro

IBM Watson Studio

Upgrade

Rodrigo Macedo's Account

RM

Projects / Dog Breed Classifier / Dog Breed / Edit and retrain

Drag and drop files from your project.

Search classes

5 classes | 0 incomplete classes | 0 unclassified images

New training data size: 0.0/250 MB

Create a class

GoldenRetriever (...)

Husky (50)

Labrador (171)

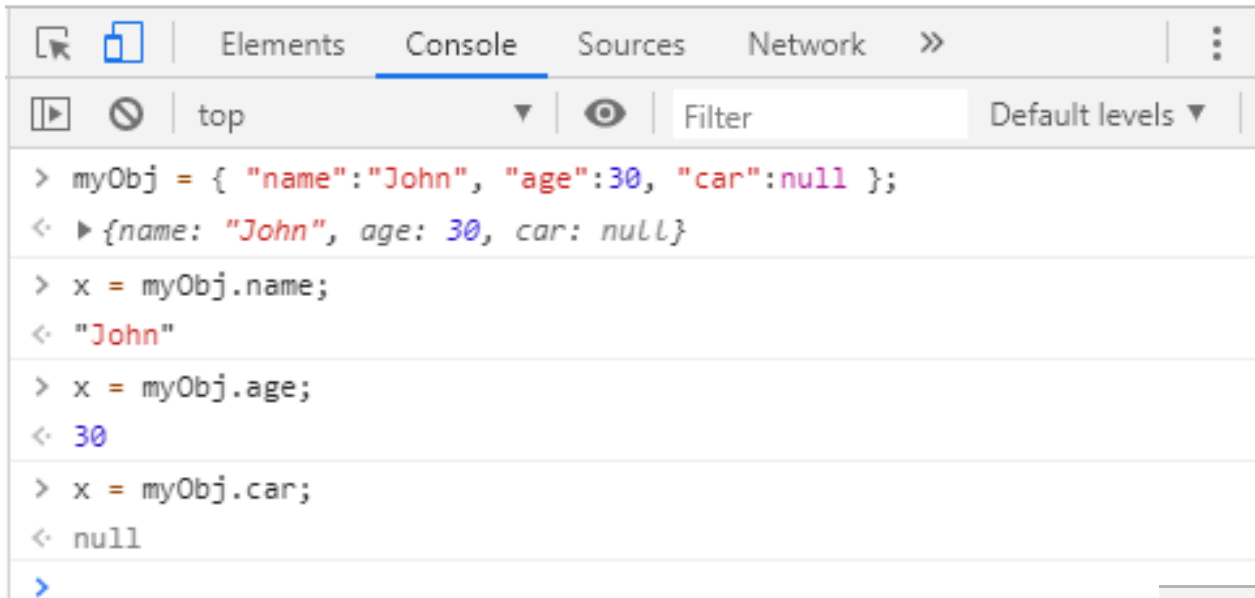
ShihTzu (214)

Negative (0)

Use the negative class to train the model on images that do not depict the visual subject of any of the positive classes.

This class is recommended but not

Exemplos JSON



```
> myObj = { "name": "John", "age": 30, "car": null };
< ▶ {name: "John", age: 30, car: null}

> x = myObj.name;
< "John"

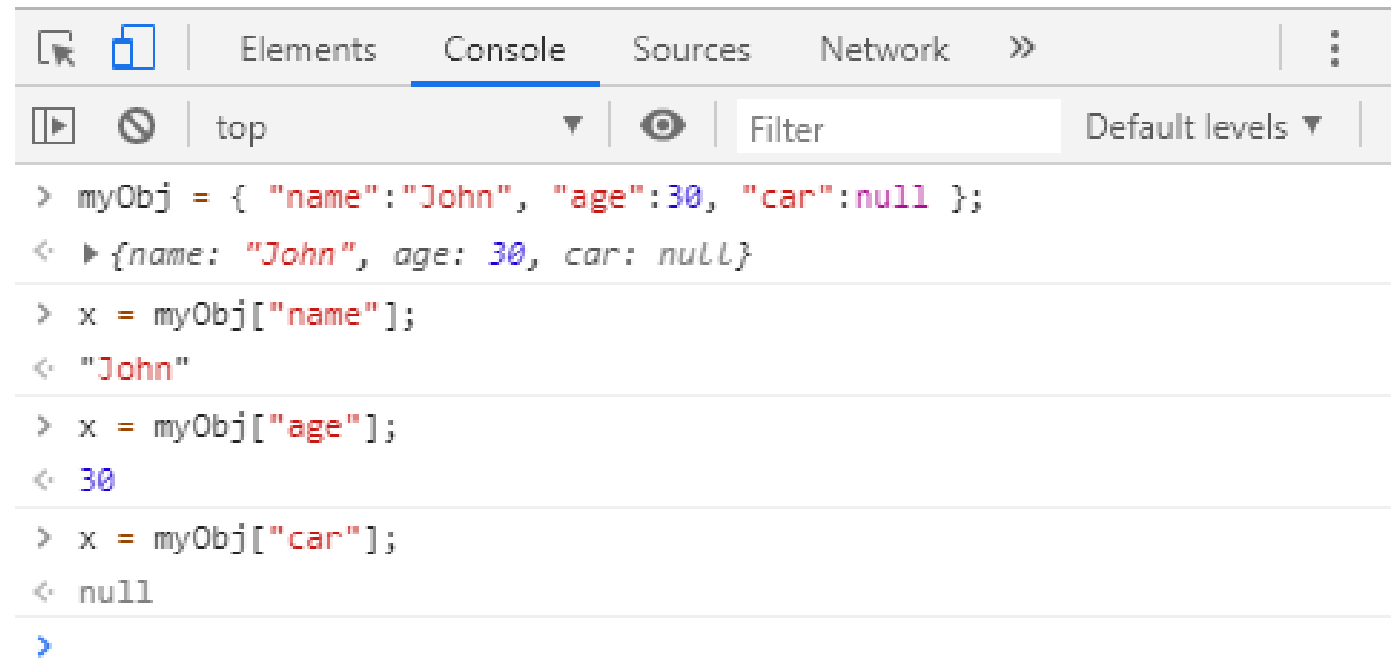
> x = myObj.age;
< 30

> x = myObj.car;
< null

>
```

Consulta objetos JSON pelo colchete ([]) + o nome da propriedade.

Consulta objetos JSON pelo ponto (.) + o nome da propriedade.



```
> myObj = { "name": "John", "age": 30, "car": null };
< ▶ {name: "John", age: 30, car: null}

> x = myObj["name"];
< "John"

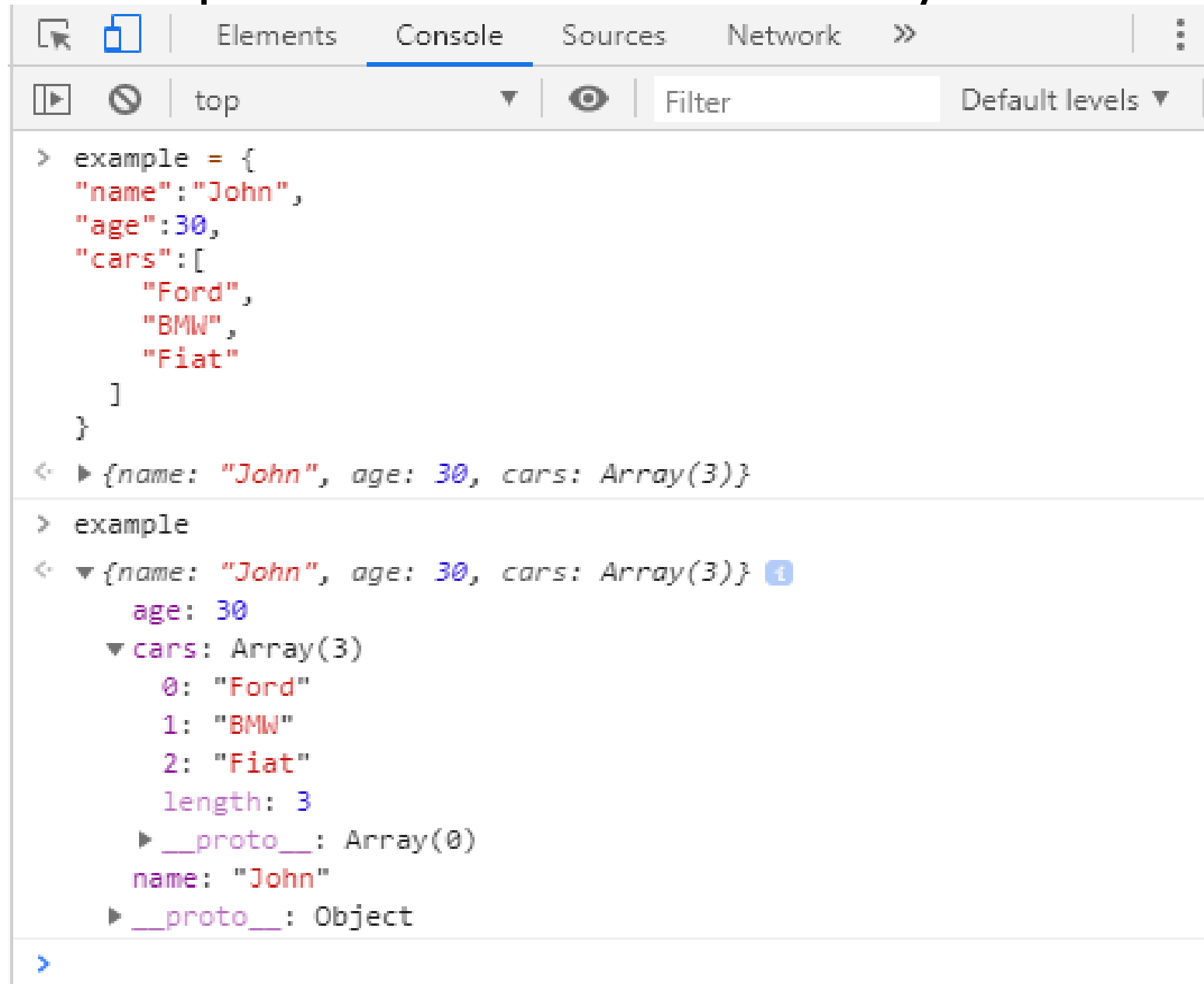
> x = myObj["age"];
< 30

> x = myObj["car"];
< null

>
```

Exemplos JSON com Arrays

A
propriedade
cars
representa
um array de
valores

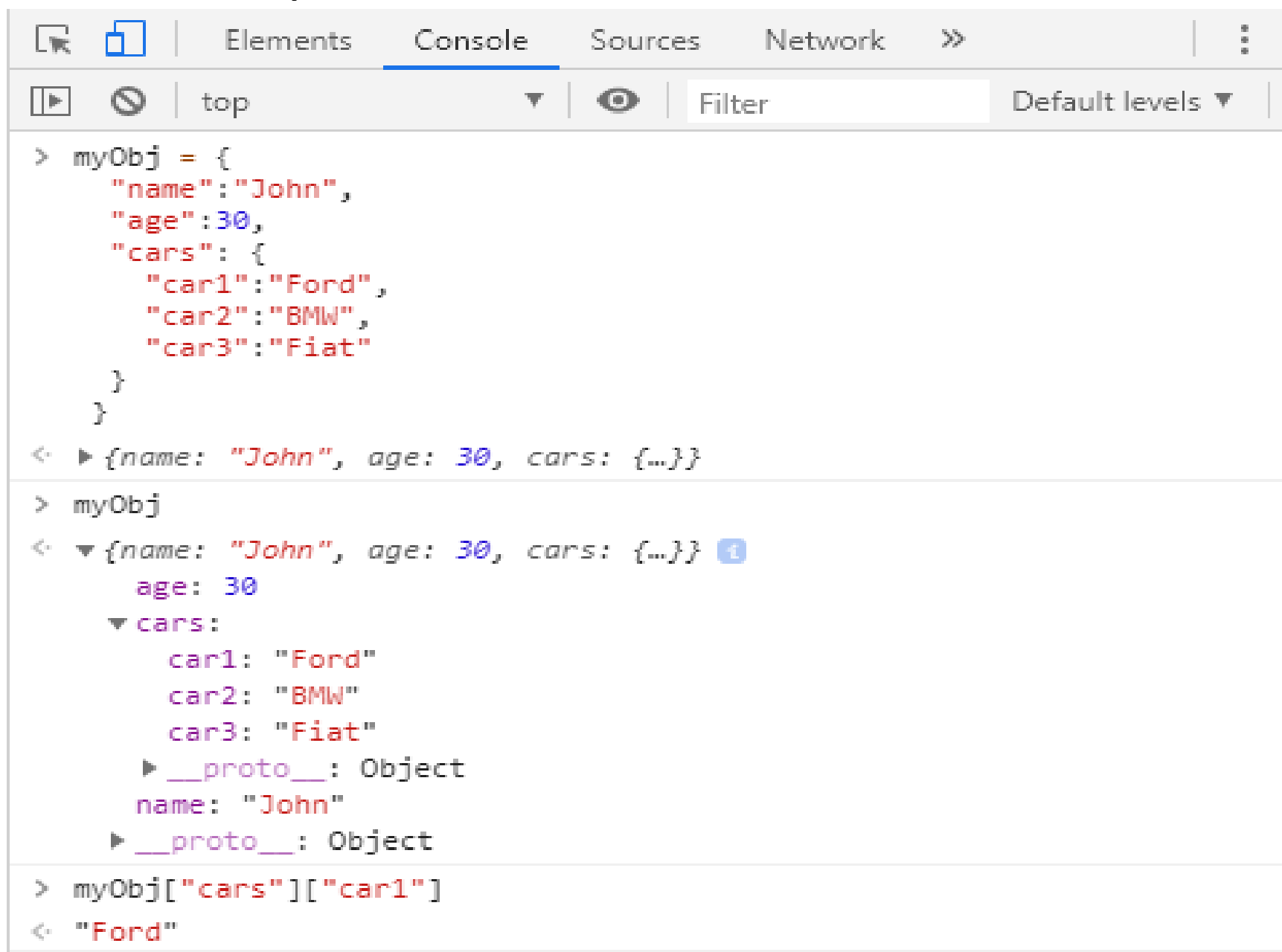


```
> example = {  
  "name": "John",  
  "age": 30,  
  "cars": [  
    "Ford",  
    "BMW",  
    "Fiat"  
  ]  
}  
< ▶ {name: "John", age: 30, cars: Array(3)}
```

```
> example  
< ▼ {name: "John", age: 30, cars: Array(3)} ⓘ  
  age: 30  
  ▼ cars: Array(3)  
    0: "Ford"  
    1: "BMW"  
    2: "Fiat"  
    length: 3  
    ▶ __proto__: Array(0)  
  name: "John"  
  ▶ __proto__: Object  
>
```

Exemplos JSON aninhados

Nesse caso, temos um json dentro de outro. O objeto cars, está dentro do objeto superior, myObj.



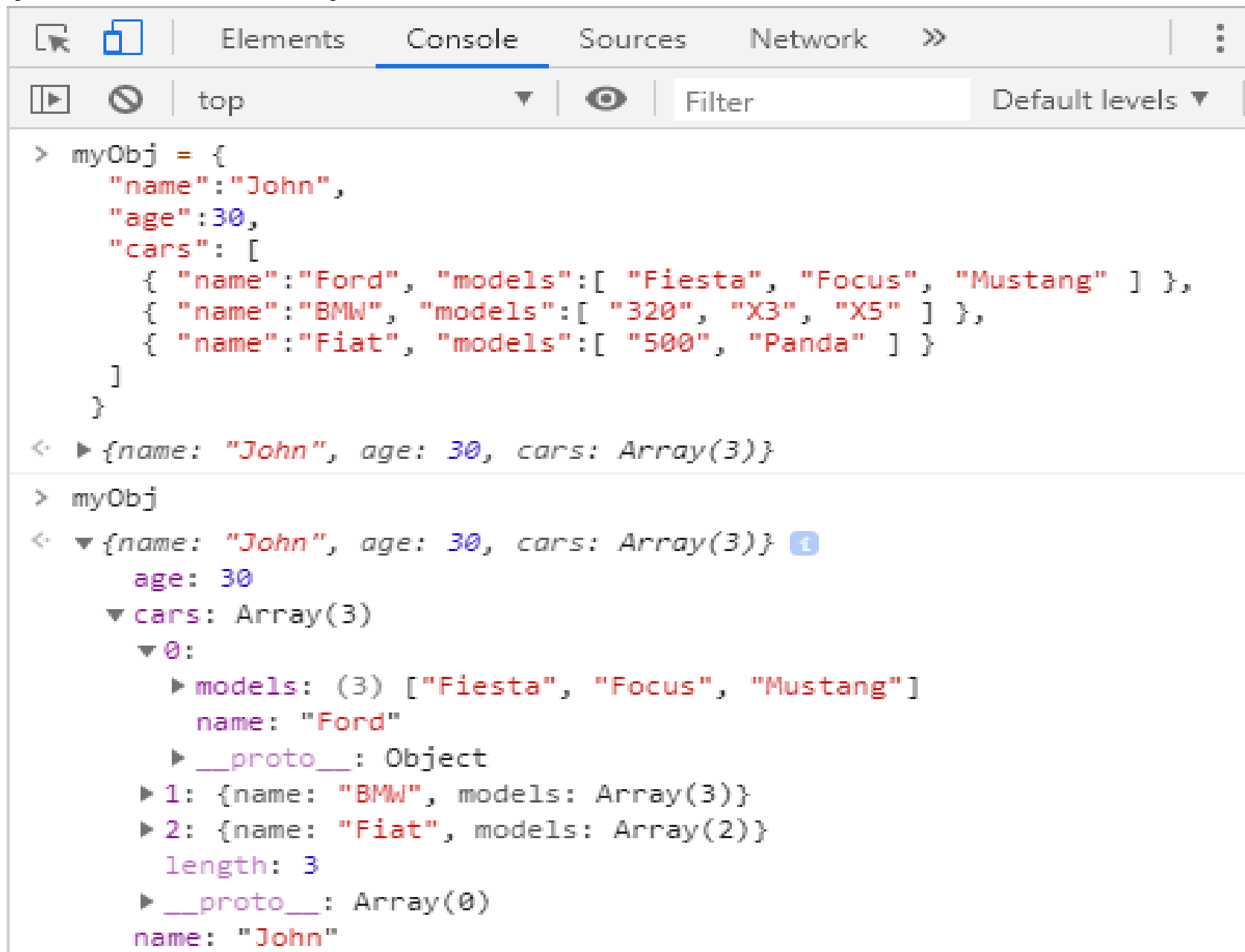
```
> myObj = {
  "name": "John",
  "age": 30,
  "cars": {
    "car1": "Ford",
    "car2": "BMW",
    "car3": "Fiat"
  }
}
< ▶ {name: "John", age: 30, cars: {...}}
```

```
> myObj
< ▼ {name: "John", age: 30, cars: {...}} ⓘ
  age: 30
  ▼ cars:
    car1: "Ford"
    car2: "BMW"
    car3: "Fiat"
    ▶ __proto__: Object
  name: "John"
  ▶ __proto__: Object
```

```
> myObj["cars"]["car1"]
< "Ford"
```

Exemplos Arrays aninhados em JSON

Nesse caso, temos um array dentro de outro array que está por sua vez dentro de um objeto JSON.



```
> myObj = {  
  "name": "John",  
  "age": 30,  
  "cars": [  
    { "name": "Ford", "models": [ "Fiesta", "Focus", "Mustang" ] },  
    { "name": "BMW", "models": [ "320", "X3", "X5" ] },  
    { "name": "Fiat", "models": [ "500", "Panda" ] }  
  ]  
}  
  
< ▶ {name: "John", age: 30, cars: Array(3)}  
  
> myObj  
< ▼ {name: "John", age: 30, cars: Array(3)} ⓘ  
  age: 30  
  ▼ cars: Array(3)  
    ▼ 0:  
      ▶ models: (3) ["Fiesta", "Focus", "Mustang"]  
        name: "Ford"  
      ▶ __proto__: Object  
    ▶ 1: {name: "BMW", models: Array(3)}  
    ▶ 2: {name: "Fiat", models: Array(2)}  
      length: 3  
    ▶ __proto__: Array(0)  
  name: "John"
```


JSON x XML

```
{
  "menu": {
    "id": "file",
    "value": "File",
    "popup": {
      "menuitem": [
        {
          "value": "New",
          "onclick": "CreateNewDoc()"
        },
        {
          "value": "Open",
          "onclick": "OpenDoc()"
        },
        {
          "value": "Close",
          "onclick": "CloseDoc()"
        }
      ]
    }
  }
}
```

```
<menu id="file" value="File">
  <popup>
    <menuitem value="New" onclick="CreateNewDoc()" />
    <menuitem value="Open" onclick="OpenDoc()" />
    <menuitem value="Close" onclick="CloseDoc()" />
  </popup>
</menu>
```

JSON x XML

```
{"employees":  
  { "firstName":"John", "lastName":"Doe" },  
  { "firstName":"Anna", "lastName":"Smith" },  
  { "firstName":"Peter", "lastName":"Jones" }  
}]}
```

```
<employees>  
  <employee>  
    <firstName>John</firstName> <lastName>Doe</lastName>  
  </employee>  
  <employee>  
    <firstName>Anna</firstName> <lastName>Smith</lastName>  
  </employee>  
  <employee>  
    <firstName>Peter</firstName> <lastName>Jones</lastName>  
  </employee>  
</employees>
```

JSON x XML

-> Semelhanças:

- JSON e XML são "autoexplicativos" (legíveis por humanos)
- JSON e XML são hierárquicos (valores dentro de valores)
- JSON e XML podem ser analisados e usados por muitas linguagens de programação
- JSON e XML podem ser buscados com um XMLHttpRequest

-> Diferenças:

- JSON não usa tag final.
- JSON é mais curto.
- JSON é mais rápido para ler e escrever.
- JSON pode usar arrays.

Stringify

- Converte valores em javascript para uma String JSON.



```
> var obj = { name: "John", age: 30, city: "New York" };
< undefined

> obj
< {name: "John", age: 30, city: "New York"}

> var myJSON = JSON.stringify(obj);
< undefined

> myJSON
< '{"name":"John","age":30,"city":"New York"}'

>
```

Parse / Eval

- Converte um texto JSON em um objeto JavaScript.



```
> var text = '{ "employees" : [' +
  '{ "firstName":"John" , "lastName":"Doe" },' +
  '{ "firstName":"Anna" , "lastName":"Smith" },' +
  '{ "firstName":"Peter" , "lastName":"Jones" } ]}';
< undefined

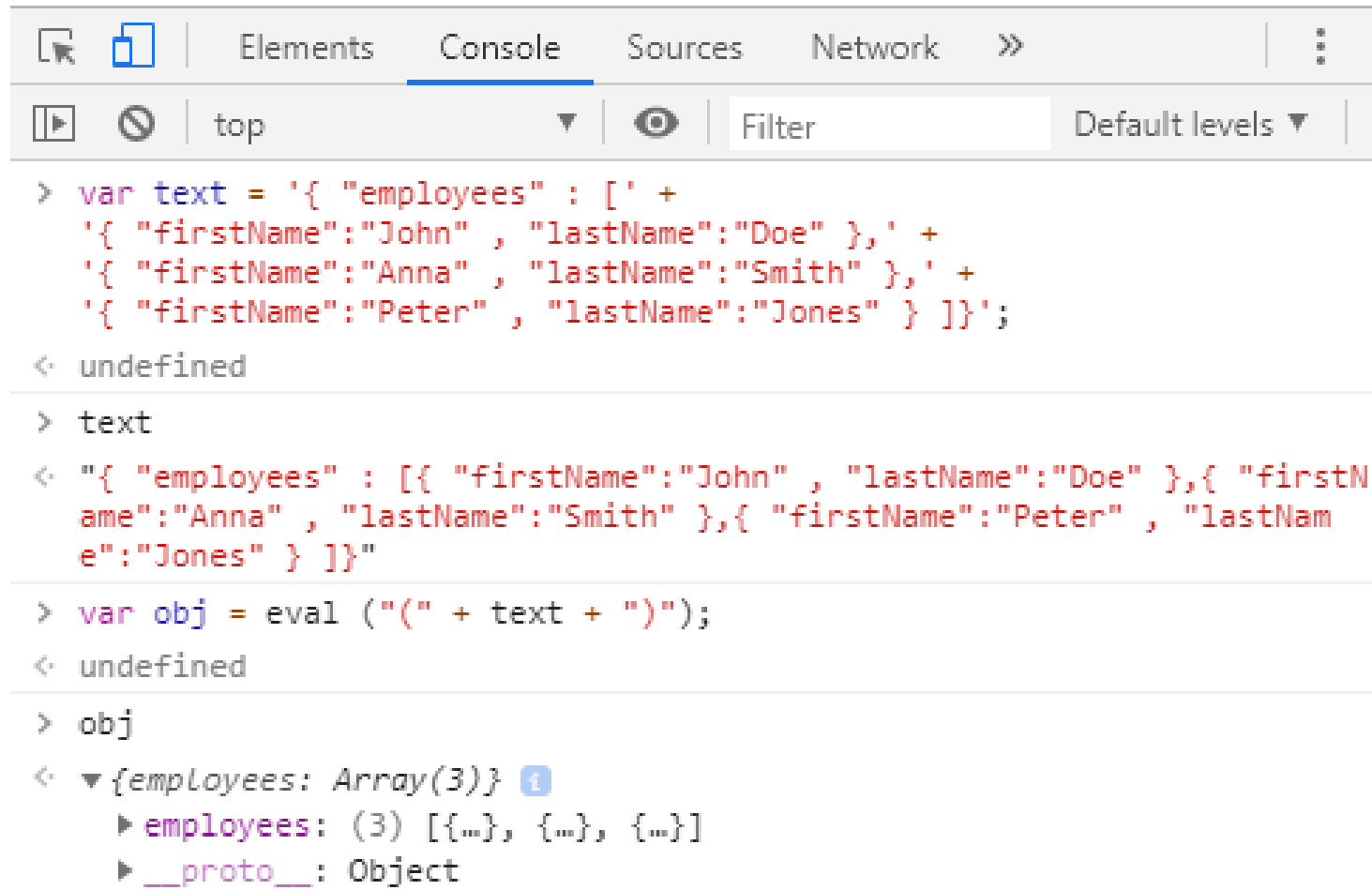
> text
< "{ \"employees\" : [{ \"firstName\":\"John\" , \"lastName\":\"Doe\" }, { \"firstN
ame\":\"Anna\" , \"lastName\":\"Smith\" }, { \"firstName\":\"Peter\" , \"lastNam
e\":\"Jones\" } ]}"

> var obj = JSON.parse(text);
< undefined

> obj
< {employees: Array(3)}
  ▶ employees: (3) [{...}, {...}, {...}]
  ▶ __proto__: Object
```

Parse / Eval

- Converte um texto JSON em um objeto JavaScript.



```
> var text = '{ "employees" : [' +
  '{ "firstName":"John" , "lastName":"Doe" },' +
  '{ "firstName":"Anna" , "lastName":"Smith" },' +
  '{ "firstName":"Peter" , "lastName":"Jones" } ]}';
< undefined

> text
< "{ \"employees\" : [{ \"firstName\":\"John\" , \"lastName\":\"Doe\" }, { \"firstN
ame\":\"Anna\" , \"lastName\":\"Smith\" }, { \"firstName\":\"Peter\" , \"lastNam
e\":\"Jones\" } ]}"

> var obj = eval ("(" + text + ")");
< undefined

> obj
< {employees: Array(3)}
  ▶ employees: (3) [{...}, {...}, {...}]
  ▶ __proto__: Object
```

Curiosidade JSON

- A API JSONP foi introduzida na linguagem JAVA, apenas em sua versão 7.
- O banco de dados PostgreSQL fornece suportes a consultas em JSON e JSONB (uma otimização do JSON).
- Em uma requisição AJAX, o objeto XMLHttpRequest, em Javascript pode retornar no campo **responseText** um resultado em JSON.





Q1) [CESPE TCE PA 2016] Web service é um software que, armazenado em um computador, pode ser acessado por outro software em outro computador por meio de uma rede. A partir dessa afirmação, julgue o item subsequente.

O formato JSON (JavaScript Object Notation) é uma especialização do XML e pode ser utilizado para representar dados.

Q2) [IFRS 2018] JSON (JavaScript Object Notation) é um formato para intercâmbio de dados baseado em texto e independente de linguagem. A respeito das características que apresenta este formato de acordo com o padrão ECMA-404, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F), e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo: JSON (JavaScript Object Notation) é um formato para intercâmbio de dados baseado em texto e independente de linguagem. A respeito das características que apresenta este formato de acordo com o padrão ECMA-404, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F), e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo:

() Suporta os seguintes tipos de dados: object, array, number, string, true, false e null.

() É indicado para aplicações que requerem transferência de dados binários.

() Não pode ser considerado uma especificação de intercâmbio completo de dados.

() Sua sintaxe é formada por chaves, colchetes, dois pontos, vírgulas e cifrões.

a) V - F - F - V

b) F - V - F - F

c) V - F - V - V

d) V - V - V - V

e) V - F - V - F

Q1) [CESPE TCE PA 2016] Web service é um software que, armazenado em um computador, pode ser acessado por outro software em outro computador por meio de uma rede. A partir dessa afirmação, julgue o item subsequente.

O formato JSON (JavaScript Object Notation) é uma especialização do XML e pode ser utilizado para representar dados. ERRADO.

Q2) [IFRS 2018] JSON (JavaScript Object Notation) é um formato para intercâmbio de dados baseado em texto e independente de linguagem. A respeito das características que apresenta este formato de acordo com o padrão ECMA-404, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F), e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo: JSON (JavaScript Object Notation) é um formato para intercâmbio de dados baseado em texto e independente de linguagem. A respeito das características que apresenta este formato de acordo com o padrão ECMA-404, classifique cada uma das afirmativas abaixo como verdadeira (V) ou falsa (F), e assinale a alternativa que apresenta a sequência CORRETA, de cima para baixo:

() Suporta os seguintes tipos de dados: object, array, number, string, true, false e null.

() É indicado para aplicações que requerem transferência de dados binários.

() Não pode ser considerado uma especificação de intercâmbio completo de dados.

() Sua sintaxe é formada por chaves, colchetes, dois pontos, vírgulas e cifrões.

a) V - F - F - V

b) F - V - F - F

c) V - F - V - V

d) V - V - V - V

e) V - F - V - F

Q3) [CESPE SLU DF 2019] Com relação a desenvolvimento de software, julgue o item a seguir.

A função `JSON.toObject()` permite converter um texto no formato JSON em um objeto JavaScript nativo.

Q4) [CESPE FUB 2018] Julgue o próximo item, a respeito de questões relacionadas ao desenvolvimento e testes de segurança de aplicações web.

Havendo necessidade de exportar dados do browser para o servidor, objetos JavaScript poderão ser transformados em JSON; nesse caso, JSON sempre vai excluir as funções dos objetos.

Q5) [CESPE STJ 2015] Julgue o item que se segue acerca de HTML5, CSS3 e JSon.

JSON (JavaScript Object Notation) é um formato de arquivo de texto para troca de dados em que um objeto é um conjunto de pares nome/valor.

Q3) [CESPE SLU DF 2019] Com relação a desenvolvimento de software, julgue o item a seguir.

A função `JSON.toObject()` permite converter um texto no formato JSON em um objeto JavaScript nativo. ERRADO.

Q4) [CESPE FUB 2018] Julgue o próximo item, a respeito de questões relacionadas ao desenvolvimento e testes de segurança de aplicações web.

Havendo necessidade de exportar dados do browser para o servidor, objetos JavaScript poderão ser transformados em JSON; nesse caso, JSON sempre vai excluir as funções dos objetos. ERRADO.

Q5) [CESPE STJ 2015] Julgue o item que se segue acerca de HTML5, CSS3 e JSon.

JSON (JavaScript Object Notation) é um formato de arquivo de texto para troca de dados em que um objeto é um conjunto de pares nome/valor. CERTO.

Q6) [CESPE TRE MT 2015] Assinale a opção que apresenta corretamente um objetivo seguido de uma representação em JSON (JavaScript Object Notation).

a) Representar o ano de 2015:

ano:= [2015]

b) Representar a cotação do dólar:

"dolar": 3.87

c) Representar a projeção do PIB brasileiro negativo:

{PIB-BR}= -3[%]

d) Representar valor booleano:

{recessão}:= [true]

e) Representar array de strings:

{ "[DF],[MT],[MS],[AM]" }

Q6) [CESPE TRE MT 2015] Assinale a opção que apresenta corretamente um objetivo seguido de uma representação em JSON (JavaScript Object Notation).

a) Representar o ano de 2015:

ano:= [2015]

b) Representar a cotação do dólar:

"dolar": 3.87

c) Representar a projeção do PIB brasileiro negativo:

{PIB-BR}= -3[%]

d) Representar valor booleano:

{recessão}:= [true]

e) Representar array de strings:

{"[DF],[MT],[MS],[AM]"}

Q7) [CCV UFC 2019] Utilizando JSON, na notação abaixo, Equipe é de qual tipo JSON?

```
{ "Equipe": [ "Carlos", "Pedro", "Francisco" ] }
```

- a) Um array.
- b) Um object.
- c) Um struct.
- d) Uma classe.
- e) Uma string JSON inválida.

Q8) [CESPE FUNPRES-PJUD 2016] A respeito das tecnologias relacionadas ao desenvolvimento web em Java, julgue o item a seguir.

Para lidar com um conjunto de objetos em JSon, é necessário utilizar um array que permita realizar, em uma única operação, a carga de todos os objetos.

Q7) [CCV UFC 2019] Utilizando JSON, na notação abaixo, Equipe é de qual tipo JSON?

```
{ "Equipe": [ "Carlos", "Pedro", "Francisco" ] }
```

a)Um array.

b)Um object.

c)Um struct.

d)Uma classe.

e)Uma string JSON inválida.

Q8) [CESPE FUNPRES-PJ 2016] A respeito das tecnologias relacionadas ao desenvolvimento web em Java, julgue o item a seguir.

Para lidar com um conjunto de objetos em JSon, é necessário utilizar um array que permita realizar, em uma única operação, a carga de todos os objetos.

ERRADO.

Q9) [UECE-CEV FUNCEME 2018] Atente ao que se diz a seguir sobre o JSON (JavaScript Object Notation) e assinale a afirmação verdadeira.

- a) É uma notação em formato texto criada para programas web desenvolvidos em JavaScript.
- b) Apesar de ser fácil de ler e escrever para humanos, é difícil de gerar e interpretar para máquinas.
- c) Diferentemente de XML, não pode ser utilizado como formato de representação de dados em web services.
- d) Sua estrutura é constituída por um conjunto de pares chave/valor, o que o torna um formato propício para troca de dados.

Q10)[CESPE TCE SC 2016] Julgue o item que se segue a respeito dos padrões XSLT e JSON.

Em navegadores que não possuem apoio para a função JavaScript `JSON.parse`, pode-se utilizar a função `eval` para converter um texto JSON em um objeto JavaScript, por meio da sintaxe apresentada a seguir.

```
var obj = eval ("(" + text + ")");
```

Q9) [UECE-CEV FUNCEME 2018] Atente ao que se diz a seguir sobre o JSON (JavaScript Object Notation) e assinale a afirmação verdadeira.

- a) É uma notação em formato texto criada para programas web desenvolvidos em JavaScript.
- b) Apesar de ser fácil de ler e escrever para humanos, é difícil de gerar e interpretar para máquinas.
- c) Diferentemente de XML, não pode ser utilizado como formato de representação de dados em web services.
- d) Sua estrutura é constituída por um conjunto de pares chave/valor, o que o torna um formato propício para troca de dados.

Q10)[CESPE TCE SC 2016] Julgue o item que se segue a respeito dos padrões XSLT e JSON.

Em navegadores que não possuem apoio para a função JavaScript `JSON.parse`, pode-se utilizar a função `eval` para converter um texto JSON em um objeto JavaScript, por meio da sintaxe apresentada a seguir.

`var obj = eval("(" + text + ")");` CERTO.

Q11) [CESPE TRE PE 2017] Assinale a opção que indica a descrição correta de um array denominado empregados que contenha três objetos compostos pelo registro do primeiro e do último nome de um empregado em uma matriz JSON.

a)

```
'empregados': (  
    [PrimeiroNome:'Marina',  
    UltimoNome:'Barbosa '],  
    [PrimeiroNome:'Paloma',  
    UltimoNome:'Duarte'],  
    [PrimeiroNome:'Emanuelle',  
    UltimoNome:'Araújo']  
)
```

b)

```
empregados:{  
    ("PrimeiroNome":"Marina",  
    "UltimoNome":"Barbosa "},  
    ("PrimeiroNome":"Paloma",  
    "UltimoNome":"Duarte"),  
    ("PrimeiroNome":"Emanuelle",  
    "UltimoNome":"Araújo")  
}
```

c)

```
empregados:{  
    ['PrimeiroNome':"Marina",  
    'UltimoNome':"Barbosa "],  
    ['PrimeiroNome':"Paloma",  
    'UltimoNome':"Duarte"]},  
    ['PrimeiroNome':"Emanuelle",  
    'UltimoNome':"Araújo"]  
}
```

d)

```
"empregados":[  
    {"PrimeiroNome":"Marina",  
    "UltimoNome":"Barbosa "},  
    {"PrimeiroNome":"Paloma",  
    "UltimoNome":"Duarte"},  
    {"PrimeiroNome":"Emanuelle",  
    "UltimoNome":"Araújo"}  
]
```

e)

```
empregados:[  
    {PrimeiroNome:'Marina',  
    UltimoNome:'Barbosa '},  
    {PrimeiroNome:'Paloma',  
    UltimoNome:'Duarte'},  
    {PrimeiroNome:'Emanuelle',  
    UltimoNome:'Araújo'}  
]
```

Q11) [CESPE TRE PE 2017] Assinale a opção que indica a descrição correta de um array denominado empregados que contenha três objetos compostos pelo registro do primeiro e do último nome de um empregado em uma matriz J S O N.

a)

```
'empregados': (  
    [PrimeiroNome:'Marina',  
    UltimoNome:'Barbosa '],  
    [PrimeiroNome:'Paloma',  
    UltimoNome:'Duarte'],  
    [PrimeiroNome:'Emanuelle',  
    UltimoNome:'Araújo']  
)
```

b)

```
empregados:{  
    ("PrimeiroNome":"Marina",  
    "UltimoNome":"Barbosa "},  
    ("PrimeiroNome":"Paloma",  
    "UltimoNome":"Duarte"),  
    ("PrimeiroNome":"Emanuelle",  
    "UltimoNome":"Araújo")  
}
```

c)

```
empregados:{  
    ['PrimeiroNome':"Marina",  
    'UltimoNome':"Barbosa "],  
    ['PrimeiroNome':"Paloma",  
    'UltimoNome':"Duarte"]},  
    ['PrimeiroNome':"Emanuelle",  
    'UltimoNome':"Araújo"]  
}
```

d)

```
"empregados":[  
    {"PrimeiroNome":"Marina",  
    "UltimoNome":"Barbosa "},  
    {"PrimeiroNome":"Paloma",  
    "UltimoNome":"Duarte"},  
    {"PrimeiroNome":"Emanuelle",  
    "UltimoNome":"Araújo"}  
]
```

e)

```
empregados:[  
    {PrimeiroNome:'Marina',  
    UltimoNome:'Barbosa '},  
    {PrimeiroNome:'Paloma',  
    UltimoNome:'Duarte'},  
    {PrimeiroNome:'Emanuelle',  
    UltimoNome:'Araújo'}  
]
```

Q12) [CESGRANRIO BB 2018] O seguinte item apresenta uma estrutura em formato JSON válido:

a) {
 nomeCompleto: {
 nome: Fulano,
 sobrenome: de Tal},
 idade: 26,
 email: {
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
 },
 endereco: {},
 preferencias: null
}

b) {
 nomeCompleto: {
 nome: Fulano,
 sobrenome: de Tal},
 idade: 26,
 email: [
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
],
 endereco: {},
 preferencias: null
}

c) {
 "nomeCompleto": {
 "nome": "Fulano",
 "sobrenome": "de Tal"},
 "idade": 26,
 "email": [
 "fulano@aqui.com.br",
 "fulano@ali.com.br",
 "fulano@la.com.br"
],
 "endereco": {},
 "preferencias": null
}

d) {
 "nomeCompleto": {
 "nome": "Fulano",
 "sobrenome": "de Tal"},
 "idade": 26,
 "email": {
 "fulano@aqui.com.br",
 "fulano@ali.com.br",
 "fulano@la.com.br"
 },
 "endereco": {},
 "preferencias": null
}

e) {
 "nomeCompleto": {
 "nome": Fulano,
 "sobrenome": de Tal},
 "idade": 26,
 "email": {
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
 },
 "endereco": {},
 "preferencias": null
}

Q12) [CESGRANRIO BB 2018] O seguinte item apresenta uma estrutura em formato JSON válido:

a) {
 nomeCompleto: {
 nome: Fulano,
 sobrenome: de Tal},
 idade: 26,
 email: {
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
 },
 endereco: {},
 preferencias: null
}

b) {
 nomeCompleto: {
 nome: Fulano,
 sobrenome: de Tal},
 idade: 26,
 email: [
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
],
 endereco: {},
 preferencias: null
}

c) {
 "nomeCompleto": {
 "nome": "Fulano",
 "sobrenome": "de Tal"},
 "idade": 26,
 "email": [
 "fulano@aqui.com.br",
 "fulano@ali.com.br",
 "fulano@la.com.br"
],
 "endereco": {},
 "preferencias": null
}

d) {
 "nomeCompleto": {
 "nome": "Fulano",
 "sobrenome": "de Tal"},
 "idade": 26,
 "email": {
 "fulano@aqui.com.br",
 "fulano@ali.com.br",
 "fulano@la.com.br"
 },
 "endereco": {},
 "preferencias": null
}

e) {
 "nomeCompleto": {
 "nome": Fulano,
 "sobrenome": de Tal},
 "idade": 26,
 "email": {
 fulano@aqui.com.br,
 fulano@ali.com.br,
 fulano@la.com.br
 },
 "endereco": {},
 "preferencias": null
}

Q13) - [CESGRANRIO BNDES 2011] Um programador, ao analisar determinado sistema WB, observa o arquivo a seguir.

```
{
  "menu": "m1",
  "acoes": [
    {
      "titulo": "X",
      "desc": "A"
    },
    {
      "titulo": "Y",
      "desc": "B"
    },
    {
      "titulo": "Z",
      "desc": "C"
    }
  ]
}
```

a) JSON

b) REST

c) SOAP

d) XLST

e) XML

Q13) - [CESGRANRIO BNDES 2011] Um programador, ao analisar determinado sistema WB, observa o arquivo a seguir.

```
{
  "menu": "m1",
  "acoes": [
    {
      "titulo": "X",
      "desc": "A"
    },
    {
      "titulo": "Y",
      "desc": "B"
    },
    {
      "titulo": "Z",
      "desc": "C"
    }
  ]
}
```

a) JSON

b) REST

c) SOAP

d) XLST

e) XML

Q14) [CESPE TRE MT 2015] Assinale a opção que apresenta corretamente um conteúdo com a notação JSON (JavaScript Object Notation).

a) Bob Moore USA Columbia 1985
Jimmy Greaves UK CBS 1988
John Barnes USA Rich CA 1982

b) nome,pais,origem, ano
Bob Moore,USA,Columbia,1985
Jimmy Greaves,UK,CBS,1988
John Barnes,USA,Rich CA,1982

c) <jogador>

<nome>Bob Moore </nome>

<pais>USA </pais>

<origem>Columbia </origem>

<ano>1985</ano>

</jogador>

d) {"jogador":
{"nome":"Bob Moore","pais":"USA","origem":"Columbia","ano":"1985"},
{"nome":"Jimmy Greaves","pais":"UK","origem":"CBS","ano":"1988"},
{"nome":"John Barnes","pais":"USA","origem":"Rich CA","ano":"1982"}
]}

e) [nome] => Jimmy Greaves
[pais] => UK
[origem] => CBS
[ano] => 1988
)
[2] => (
[nome] => John Barnes
[pais] => USA
[origem] => Rich CA
[ano] => 1982
)
)

Q14) [CESPE TRE MT 2015] Assinale a opção que apresenta corretamente um conteúdo com a notação JSON (JavaScript Object Notation).

a) Bob Moore USA Columbia 1985
Jimmy Greaves UK CBS 1988
John Barnes USA Rich CA 1982

b) nome,pais,origem, ano
Bob Moore,USA,Columbia,1985
Jimmy Greaves,UK,CBS,1988
John Barnes,USA,Rich CA,1982

c) <jogador>

<nome>Bob Moore </nome>

<pais>USA </pais>

<origem>Columbia </origem>

<ano>1985</ano>

</jogador>

d) {"jogador":
{"nome":"Bob Moore","pais":"USA","origem":"Columbia","ano":"1985"},
{"nome":"Jimmy Greaves","pais":"UK","origem":"CBS","ano":"1988"},
{"nome":"John Barnes","pais":"USA","origem":"Rich CA","ano":"1982"}
]}

e) [nome] => Jimmy Greaves
[pais] => UK
[origem] => CBS
[ano] => 1988
)
[2] => (
[nome] => John Barnes
[pais] => USA
[origem] => Rich CA
[ano] => 1982
)
)

Q15) [CCV UFC 2019] [FGV TCM SP 2015] Como resposta a uma requisição AJAX, o script executando no servidor envia para o cliente a cadeia de caracteres seguinte, no formato JSON:

```
{ "nome":"Martin", "id":1299, "v":[3,2] }
```

que é recebida no seguinte campo do objeto XMLHttpRequest, em Javascript:

- a) responseString;
- b) serverResponse;
- c) responseText;
- d) JSONResponse;
- e) JSONResponseString.

Q15) [CCV UFC 2019] [FGV TCM SP 2015] Como resposta a uma requisição AJAX, o script executando no servidor envia para o cliente a cadeia de caracteres seguinte, no formato JSON:

```
{ "nome":"Martin", "id":1299, "v":[3,2] }
```

que é recebida no seguinte campo do objeto XMLHttpRequest, em Javascript:

a) responseString;

b) serverResponse;

c) responseText;

d) JSONResponse;

e) JSONResponseString.

Q16) [CCV UFC 2016] Considerando a estrutura JSON (JavaScript Object Notation), marque o item correto.

- a) Segundo a especificação do JSON, não é possível a utilização de caracteres de escape, sendo uma limitação na troca de dados.
- b) Por sua notação simplista, somente é possível utilizar tipos de dados primitivos: caractere, números e valores lógicos (Verdadeiro ou Falso).
- c) Na especificação dos dados no formato JSON, os valores são especificados em pares (chave:valor), onde a chave é identificada por números sequenciais.
- d) A partir da troca de arquivos com dados no formato JSON, aplicações escritas em linguagens diferentes podem trocar informações de forma estruturada e simples
- e) Como o JSON é uma notação recente, ela foi especificada baseada nas linguagens mais novas, sendo compatíveis somente com as linguagens originadas a partir da linguagem C.

Q16) [CCV UFC 2016] Considerando a estrutura JSON (JavaScript Object Notation), marque o item correto.

- a) Segundo a especificação do JSON, não é possível a utilização de caracteres de escape, sendo uma limitação na troca de dados.
- b) Por sua notação simplista, somente é possível utilizar tipos de dados primitivos: caractere, números e valores lógicos (Verdadeiro ou Falso).
- c) Na especificação dos dados no formato JSON, os valores são especificados em pares (chave:valor), onde a chave é identificada por números sequenciais.
- d) A partir da troca de arquivos com dados no formato JSON, aplicações escritas em linguagens diferentes podem trocar informações de forma estruturada e simples
- e) Como o JSON é uma notação recente, ela foi especificada baseada nas linguagens mais novas, sendo compatíveis somente com as linguagens originadas a partir da linguagem C.

Q17) [CCV UFC 2013] Um Analista de TI, ao analisar um determinado sistema WEB, observa o arquivo a seguir.

```
{  
  "id":1,  
  "nome":"José Patriolino Silveira",  
  "endereco":"Campus do Pici, Bloco 02 A"  
}
```

Podemos afirmar que este arquivo está no formato:

- a) XML
- b) JSON
- c) DOM
- d) XSLT
- e) SOAP

Q17) [CCV UFC 2013] Um Analista de TI, ao analisar um determinado sistema WEB, observa o arquivo a seguir.

```
{  
  "id":1,  
  "nome":"José Patriolino Silveira",  
  "endereco":"Campus do Pici, Bloco 02 A"  
}
```

Podemos afirmar que este arquivo está no formato:

a) XML

b) JSON

c) DOM

d) XSLT

e) SOAP

Q18) [SUGEP-UFRPE 2018] Sobre JSON, analise as afirmativas abaixo.

- 1) É um formato de arquivo de texto para troca de dados em que um objeto é um conjunto de pares nome/valor.
- 2) A tecnologia JSON é uma especialização do formato XML para representação de dados.
- 3) Para lidar com um conjunto de objetos em JSON, é necessário utilizar um array que permita realizar, em uma única operação, a carga de todos os objetos.

Está(ão) correta(s), apenas:

- a)1
- b)2
- c)1 e 2
- d)1 e 3
- e)3

Q18) [SUGEP-UFRPE 2018] Sobre JSON, analise as afirmativas abaixo.

- 1) É um formato de arquivo de texto para troca de dados em que um objeto é um conjunto de pares nome/valor.
- 2) A tecnologia JSON é uma especialização do formato XML para representação de dados.
- 3) Para lidar com um conjunto de objetos em JSON, é necessário utilizar um array que permita realizar, em uma única operação, a carga de todos os objetos.

Está(ão) correta(s), apenas:

a)1

b)2

c)1 e 2

d)1 e 3

e)3

Q19) [CESGRANRIO LIQUIGAS 2018] Interoperabilidade refere-se ao processo de comunicação entre sistemas sem que seja criada uma dependência tecnológica entre eles. Uma maneira de implementar a interoperabilidade é pela transferência de dados pelo uso de uma linguagem de intercâmbio de dados como

a)JSON

b)GTFS

c)SPARQL

d)XML Schema

e)XSLT

Q20) [FCC SEMEF 2019] Na sintaxe do padrão de troca de dados JSON, o tipo de dados que um campo pode conter é, dentre outros

a)float.

b)vector.

c)date.

d)boolean.

e)function.

Q19) [CESGRANRIO LIQUIGAS 2018] Interoperabilidade refere-se ao processo de comunicação entre sistemas sem que seja criada uma dependência tecnológica entre eles. Uma maneira de implementar a interoperabilidade é pela transferência de dados pelo uso de uma linguagem de intercâmbio de dados como

a)JSON

b)GTFS

c)SPARQL

d)XML Schema

e)XSLT

Q20) [FCC SEMEF 2019] Na sintaxe do padrão de troca de dados JSON, o tipo de dados que um campo pode conter é, dentre outros

a)float.

b)vector.

c)date.

d)boolean.

e)function.

Q21) [CESPE CGE-CE 2019] As estruturas básicas que constituem um JSON são

- a) uma lista de valores e uma coleção de strings/inteiros.
- b) uma lista de inteiros e uma coleção de pares de strings.
- c) uma lista de valores ordenados e uma coleção de pares nome/valor.
- d) uma lista de strings e uma coleção de pares de booleanos.
- e) uma lista de booleanos e uma coleção de listas de strings.

Q22) [FUNRIO MPOG 2013] A função que invoca o compilador Java Script, que pode ser usada para converter um texto JSON em um objeto, denomina-se

- a) decode.
- b) encode.
- c) process.
- d) eval.
- e) map.

Q21) [CESPE CGE-CE 2019] As estruturas básicas que constituem um JSON são

- a) uma lista de valores e uma coleção de strings/inteiros.
- b) uma lista de inteiros e uma coleção de pares de strings.
- c) uma lista de valores ordenados e uma coleção de pares nome/valor.
- d) uma lista de strings e uma coleção de pares de booleanos.
- e) uma lista de booleanos e uma coleção de listas de strings.

Q22) [FUNRIO MPOG 2013] A função que invoca o compilador Java Script, que pode ser usada para converter um texto JSON em um objeto, denomina-se

- a) decode.
- b) encode.
- c) process.
- d) eval.
- e) map.

Q23) [CCV UFC 2019] Uma das formas de realizar a troca de informações entre dispositivos é através de mensagens com conteúdo JSON (Javascript Object Notation). Sobre o JSON, é correto afirmar:

- a) Uma semelhança entre o XML e o JSON é a categorização das duas como sendo linguagens de marcação.
- b) É uma notação textual baseada na especificação de um conjunto de elementos organizados na forma chave-valor.
- c) Assim como o XML, o JSON não permite que seja especificado o tipo dos elementos que estão sendo especificados devido o seu formato textual.
- d) É uma especificação onde os dados do arquivo são compilados e processados gerando o arquivo binário que será utilizado na troca de informações.
- e) Uma das desvantagens no uso do JSON é a sua dependência da plataforma sobre a qual está executando, devido à necessidade do arquivo ser compilado para cada plataforma diferente.

Q23) [CCV UFC 2019] Uma das formas de realizar a troca de informações entre dispositivos é através de mensagens com conteúdo JSON (Javascript Object Notation). Sobre o JSON, é correto afirmar:

a) Uma semelhança entre o XML e o JSON é a categorização das duas como sendo linguagens de marcação.

b) É uma notação textual baseada na especificação de um conjunto de elementos organizados na forma chave-valor.

c) Assim como o XML, o JSON não permite que seja especificado o tipo dos elementos que estão sendo especificados devido o seu formato textual.

d) É uma especificação onde os dados do arquivo são compilados e processados gerando o arquivo binário que será utilizado na troca de informações.

e) Uma das desvantagens no uso do JSON é a sua dependência da plataforma sobre a qual está executando, devido à necessidade do arquivo ser compilado para cada plataforma diferente.

Q24) [FGV TJ-GO 2014] Como resposta a uma requisição AJAX, o *script* executando no servidor envia para o cliente a cadeia de caracteres seguinte, no formato JSON:

```
{"nome": "Ypojuca", "id": 1503, "xy": [1,4]}
```

Em Javascript, o campo do objeto **XMLHttpRequest** onde essa cadeia é recebida é:

- a) `responseText`;
- b) `serverResponse`;
- c) `responseString`;
- d) `JSONResponse`;
- e) `JSONResponseString`.

Q25) [CESPE SUFRAMA 2014] Julgue os itens que se seguem, relativos a JSE, JME e JEE.

O Java EE 7.0 possui API específica para a leitura e a validação de formato JSON (JavaScript Object Notation).

Q24) [FGV TJ-GO 2014] Como resposta a uma requisição AJAX, o *script* executando no servidor envia para o cliente a cadeia de caracteres seguinte, no formato JSON:

```
{"nome": "Ypojuca", "id": 1503, "xy": [1,4]}
```

Em Javascript, o campo do objeto **XMLHttpRequest** onde essa cadeia é recebida é:

- a) `responseText`;
- b) `serverResponse`;
- c) `responseString`;
- d) `JSONResponse`;
- e) `JSONResponseString`.

Q25) [CESPE SUFRAMA 2014] Julgue os itens que se seguem, relativos a JSE, JME e JEE.

O Java EE 7.0 possui API específica para a leitura e a validação de formato JSON (JavaScript Object Notation). CERTO.

Q26) [CESGRANRIO BB 2018] Um desenvolvedor Web precisa transformar em JSON a seguinte variável, em um programa JavaScript:

```
var d1 = { "nome":"Ana Anacleto", "idade":31 };
```

Qual fragmento de código faz isso corretamente?

a)var d2 = d1.stringify(d1)

b)var d2 = d1.parse()

c)var d2 = JSON.stringify(d1);

d)var d2 = JSON.parse(dados);

e)var d2 = d1.toJSON();

Q27) [CESPE STM 2018] A respeito de construção de sistemas, julgue o item subsequente.

O protocolo JSON é derivado da linguagem de programação Java e sua utilização é restrita a sistemas desenvolvidos em Java ou JavaScript.

Q26) [CESGRANRIO BB 2018] Um desenvolvedor Web precisa transformar em JSON a seguinte variável, em um programa JavaScript:

```
var d1 = { "nome":"Ana Anacleto", "idade":31 };
```

Qual fragmento de código faz isso corretamente?

a)var d2 = d1.stringify(d1)

b)var d2 = d1.parse()

c)var d2 = JSON.stringify(d1);

d)var d2 = JSON.parse(dados);

e)var d2 = d1.toJSON();

Q27) [CESPE STM 2018] A respeito de construção de sistemas, julgue o item subsequente.

O protocolo JSON é derivado da linguagem de programação Java e sua utilização é restrita a sistemas desenvolvidos em Java ou JavaScript. **ERRADO.**

Q28) [CESPE ANATEL 2014] No que se refere à arquitetura e à programação Java, julgue o item que se segue.

Na plataforma JEE (Java Enterprise Edition) versão 6, não é possível encontrar bibliotecas da própria plataforma para o consumo dos serviços REST no formato JSON.

Q29) [CESPE MEC 2015] No que se refere à arquitetura Java EE 7, julgue o próximo item.

O JSON (JavaScript object notation) é um formato leve para troca de dados que implementa o modelo MVC (model-view-control) em substituição ao framework JSF.

Q30) [CESPE ANCINE 2013] A respeito de arquitetura e engenharia de software, julgue o item seguinte.

O JSON-P, que é uma API Java EE 7, permite que aplicações Java EE analisem, transformem e consultem dados no formato JSON.

Q28) [CESPE ANATEL 2014] No que se refere à arquitetura e à programação Java, julgue o item que se segue.

Na plataforma JEE (Java Enterprise Edition) versão 6, não é possível encontrar bibliotecas da própria plataforma para o consumo dos serviços REST no formato JSON. CERTO.

Q29) [CESPE MEC 2015] No que se refere à arquitetura Java EE 7, julgue o próximo item.

O JSON (JavaScript object notation) é um formato leve para troca de dados que implementa o modelo MVC (model-view-control) em substituição ao framework JSF. ERRADO.

Q30) [CESPE ANCINE 2013] A respeito de arquitetura e engenharia de software, julgue o item seguinte.

O JSON-P, que é uma API Java EE 7, permite que aplicações Java EE analisem, transformem e consultem dados no formato JSON. CERTO.

Q31) [CCV UFC 2018] O que acontece quando executamos o código abaixo em um navegador com suporte à ECMAScript versão ES6 ou superior?

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var obj = { name: "Joao", age: 20, city: "Fortaleza" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

- a) Será exibido na janela do navegador o seguinte texto: [object Object].
- b) Será exibido no console do navegador o seguinte texto: [object Object].
- c) Será exibido na janela do navegador o seguinte texto: Joao, 20, Fortaleza.
- d) Será exibido na janela do navegador o seguinte texto:
{"name":"Joao","age":20,"city":"Fortaleza"}.
- e) Será exibido no console do navegador o seguinte texto:
{"name":"Joao","age":20,"city":"Fortaleza"}.

Q31) [CCV UFC 2018] O que acontece quando executamos o código abaixo em um navegador com suporte à ECMAScript versão ES6 ou superior?

```
<!DOCTYPE html>
<html>
<body>

<p id="demo"></p>

<script>
var obj = { name: "Joao", age: 20, city: "Fortaleza" };
var myJSON = JSON.stringify(obj);
document.getElementById("demo").innerHTML = myJSON;
</script>

</body>
</html>
```

- a) Será exibido na janela do navegador o seguinte texto: [object Object].
- b) Será exibido no console do navegador o seguinte texto: [object Object].
- c) Será exibido na janela do navegador o seguinte texto: Joao, 20, Fortaleza.
- d) Será exibido na janela do navegador o seguinte texto:
{"name":"Joao","age":20,"city":"Fortaleza"}.
- e) Será exibido no console do navegador o seguinte texto:
{"name":"Joao","age":20,"city":"Fortaleza"}.

Q32) [FGV IBGE 2016] Considere o objeto JSON:

```
{ "lista": { "feira": "domingo", "produtos": { "fruta": [
  { "id": "1", "nome": "Uva", "origem": "SC", "preço": "2.5" },
  { "id": "2", "nome": "Abacate", "preço": "2.3", "produtor":
    "Acme" }, ] } } }
```

a)

```
<?xml version="1.0" encoding="UTF-8" ?>
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
    </fruta>
  </produtos>
</lista> ;
```

b)

```
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
    </fruta>
  </produtos>
</lista> ;
```

c)

```
<?xml version="1.0" encoding="UTF-8" ?>
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <origem>SC</origem>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
      <produtor>Acme</produtor>
    </fruta>
  </produtos>
</lista> ;
```

Q32) [FGV IBGE 2016] Considere o objeto JSON:

```
{ "lista": { "feira": "domingo", "produtos": { "fruta": [
  { "id": "1", "nome": "Uva", "origem": "SC", "preço": "2.5" },
  { "id": "2", "nome": "Abacate", "preço": "2.3", "produtor":
    "Acme" }, ] } } }
```

d)

```
<?xml version="1.0" encoding="UTF-8" ?>
<lista>
  <produtos>
    <feira>domingo
      <fruta>
        <id>1</id>
        <nome>Uva</nome>
      </fruta>
      <fruta>
        <id>2</id>
        <nome>Abacate</nome>
        <preço>2.3</preço>
        <produtor>Acme</produtor>
      </fruta>
    </feira>
  </produtos>
</lista>;
```

e) Não é possível converter para o formato XML, pois o objeto JSON não está bem formado.

Q32) [FGV IBGE 2016] Considere o objeto JSON:

```
{ "lista": { "feira": "domingo", "produtos": { "fruta": [
  { "id": "1", "nome": "Uva", "origem": "SC", "preço": "2.5" },
  { "id": "2", "nome": "Abacate", "preço": "2.3", "produtor":
    "Acme" }, ] } } }
```

a)

```
<?xml version="1.0" encoding="UTF-8" ?>
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
    </fruta>
  </produtos>
</lista> ;
```

b)

```
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
    </fruta>
  </produtos>
</lista> ;
```

c)

```
<?xml version="1.0" encoding="UTF-8" ?>
<lista>
  <feira>domingo</feira>
  <produtos>
    <fruta>
      <id>1</id>
      <nome>Uva</nome>
      <origem>SC</origem>
      <preço>2.5</preço>
    </fruta>
    <fruta>
      <id>2</id>
      <nome>Abacate</nome>
      <preço>2.3</preço>
      <produtor>Acme</produtor>
    </fruta>
  </produtos>
</lista> ;
```

Q33) [CESPE MPE PI 2018] O que acontece quando executamos o código abaixo em um navegador com suporte à ECMAScript versão ES6 ou superior?

```
<html>
<body>
<p id="prova"></p>

<script>
var myObj, x;
myObj = { nome: "MP", uf: "PI", codigo: 10 };
x = myObj.uf;
document.getElementById("prova").innerHTML = x;
</script>
</body>
</html>
```

O resultado da execução, com JSON, do código JavaScript antecedente será PI

Q33) [CESPE MPE PI 2018] O que acontece quando executamos o código abaixo em um navegador com suporte à ECMAScript versão ES6 ou superior?

```
<html>
<body>
<p id="prova"></p>

<script>
var myObj, x;
myObj = { nome: "MP", uf: "PI", codigo: 10 };
x = myObj.uf;
document.getElementById("prova").innerHTML = x;
</script>
</body>
</html>
```

O resultado da execução, com JSON, do código JavaScript antecedente será PI.
CERTO.

Q34) [CESPE SLU DF 2019] No que diz respeito a ferramentas de desenvolvimento, julgue o item a seguir.

O PostgreSQL é um sistema gerenciador de banco de dados objeto-relacional (ORDBMS) que oferece suporte a tipos de dados especializados como o JSON e o JSONB.

Q35) [CESPE SLU DF 2019] No que diz respeito a ferramentas de desenvolvimento, julgue o item a seguir.

O Laravel é um framework Java, livre e open source, que utiliza a arquitetura MVC (model, view, controller) e trabalha com serviços RESTful e dados no formato JSON.

Q36) [CONSULPLAN Câmara MG 2018] “É uma alternativa à XML para representar dados; comumente utilizado em aplicativos Ajax. Seus objetos são representados como uma lista de nomes e valores de propriedade entre colchetes. Trata-se de um formato simples que facilita leitura, criação e análise de objetos e, ainda, permite que programas transmitam dados eficientemente pela internet.” Trata-se de:

- a) POJO – Plain Old Java Object.
- b) JSON – JavaScript Object Notation.
- c) SOAP – Simple Object Access Protocol.
- d) REST – Representational State Transfer.

Q34) [CESPE SLU DF 2019] No que diz respeito a ferramentas de desenvolvimento, julgue o item a seguir.

O PostgreSQL é um sistema gerenciador de banco de dados objeto-relacional (ORDBMS) que oferece suporte a tipos de dados especializados como o JSon e o JSonb. CERTO.

Q35) [CESPE SLU DF 2019] No que diz respeito a ferramentas de desenvolvimento, julgue o item a seguir.

O Lavravel é um framework Java, livre e open source, que utiliza a arquitetura MVC (model, view, controller) e trabalha com serviços RESTful e dados no formato JSon. ERRADO.

Q36) [CONSULPLAN Câmara MG 2018] “É uma alternativa à XML para representar dados; comumente utilizado em aplicativos Ajax. Seus objetos são representados como uma lista de nomes e valores de propriedade entre colchetes. Trata-se de um formato simples que facilita leitura, criação e análise de objetos e, ainda, permite que programas transmitam dados eficientemente pela internet.” Trata-se de:

- a) POJO – Plain Old Java Object.
- b) JSON – JavaScript Object Notation.
- c) SOAP – Simple Object Access Protocol.
- d) REST – Representational State Transfer.

Q37) [CESPE TRT8 2016] <time> <jogador> <nome>Bob Moore</nome> <pais>USA</pais>
<origem>Columbia</origem> <ano>1985</ano> </jogador> <jogador> <nome>Jimmy
Greaves</nome> <pais>UK</pais> <origem>CBS</origem> <ano>1988</ano> </jogador>
<jogador> <nome>John Barnes</nome> <pais>USA</pais> <origem>Rich CA</origem>
<ano>1982</ano> </jogador> </time>

Considerando o arquivo XML, assinale a opção que apresenta o conteúdo JSON associado, com os mesmos nomes de atributos e seus conteúdos e com a sintaxe correta.

- a) {jogador:[
 {nome:Bob Moore,pais:USA,origem:Columbia, ano:1985},
 {nome:Jimmy Greaves,pais:UK,origem:CBS , ano:1988},
 {nome:John Barnes,pais:USA,origem:Rich CA, ano:1982}
]}
- b) {"jogador":[
 {"nome":"BobMoore":"/nome","pais":"USA":"/pais","origem":"Columbia":"/origem","
 ano":"1985":"/ano"},
 {"nome":"Jimmy Greaves":"/nome","pais":"UK":"/pais","origem":"CBS ":"/origem","
 ano":"1988":"/ano"},
 {"nome":"John Barnes":"/nome","pais":"USA":"/pais","origem":"Rich CA":"/origem","
 ano":"1982":"/ano"}
]"/jogador"}
- c) {"jogador":[
 {"nome":"Bob Moore","pais":"USA","origem":"Columbia"," ano":"1985"},
 {"nome":"Jimmy Greaves","pais":"UK","origem":"CBS "," ano":"1988"},
 {"nome":"John Barnes","pais":"USA","origem":"Rich CA"," ano":"1982"}
]}

Q37) [CESPE TRT8 2016] <time> <jogador> <nome>Bob Moore</nome>
<pais>USA</pais> <origem>Columbia</origem> <ano>1985</ano> </jogador>
<jogador> <nome>Jimmy Greaves</nome> <pais>UK</pais>
<origem>CBS</origem> <ano>1988</ano> </jogador> <jogador> <nome>John
Barnes</nome> <pais>USA</pais> <origem>Rich CA</origem> <ano>1982</ano>
</jogador> </time>

Considerando o arquivo XML, assinale a opção que apresenta o conteúdo JSON associado, com os mesmos nomes de atributos e seus conteúdos e com a sintaxe correta.

d) {"jogador"=[
{"nome"="Bob Moore","pais"="USA","origem"="Columbia"," ano"="1985"},
{"nome"="Jimmy Greaves","pais"="UK","origem"="CBS "," ano"="1988"},
{"nome"="John Barnes","pais"="USA","origem"="Rich CA"," ano"="1982"}
] }

e) {"jogador"=[
{"nome"="Bob Moore";"pais"="USA";"origem"="Columbia";" ano"="1985"};
{"nome"="Jimmy Greaves";"pais"="UK";"origem"="CBS ";" ano"="1988"};
{"nome"="John Barnes";"pais"="USA";"origem"="Rich CA";" ano"="1982"}
] }

Q37) [CESPE TRT8 2016] <time> <jogador> <nome>Bob Moore</nome> <pais>USA</pais>
<origem>Columbia</origem> <ano>1985</ano> </jogador> <jogador> <nome>Jimmy
Greaves</nome> <pais>UK</pais> <origem>CBS</origem> <ano>1988</ano> </jogador>
<jogador> <nome>John Barnes</nome> <pais>USA</pais> <origem>Rich CA</origem>
<ano>1982</ano> </jogador> </time>

Considerando o arquivo XML, assinale a opção que apresenta o conteúdo JSON associado, com os mesmos nomes de atributos e seus conteúdos e com a sintaxe correta.

- a) {jogador:[
 {nome:Bob Moore,pais:USA,origem:Columbia, ano:1985},
 {nome:Jimmy Greaves,pais:UK,origem:CBS , ano:1988},
 {nome:John Barnes,pais:USA,origem:Rich CA, ano:1982}
]}
- b) {"jogador":[
 {"nome":"BobMoore":"/nome","pais":"USA":"/pais","origem":"Columbia":"/origem","
 ano":"1985":"/ano"},
 {"nome":"Jimmy Greaves":"/nome","pais":"UK":"/pais","origem":"CBS ":"/origem","
 ano":"1988":"/ano"},
 {"nome":"John Barnes":"/nome","pais":"USA":"/pais","origem":"Rich CA":"/origem","
 ano":"1982":"/ano"}
]"/jogador"}
- c) {"jogador":[
 {"nome":"Bob Moore","pais":"USA","origem":"Columbia"," ano":"1985"},
 {"nome":"Jimmy Greaves","pais":"UK","origem":"CBS "," ano":"1988"},
 {"nome":"John Barnes","pais":"USA","origem":"Rich CA"," ano":"1982"}
]}

GABARITO

Q1 – ERRADO.	Q14 – LETRA D.
Q2 - LETRA E.	Q15 – LETRA C.
Q3 – ERRADO.	Q16 - LETRA D.
Q4 - ERRADO.	Q17 – LETRA B.
Q5 - CERTO.	Q18 – LETRA A.
Q6 - LETRA B.	Q19 - LETRA A.
Q7 – LETRA A.	Q20 - LETRA D.
Q8 – ERRADO.	Q21 - LETRA C.
Q9 - LETRA D.	Q22 - LETRA D.
Q10 - CERTO.	Q23 - LETRA B.
Q11 – LETRA D.	Q24 - LETRA A.
Q12 - LETRA C.	Q25 - CERTO.
Q13 - LETRA A.	Q26 - LETRA C.
Q27 - ERRADO.	Q28 - CERTO
Q29 - ERRADO	Q30 - CERTO
Q31 - LETRA D	Q32 - LETRA C.
Q33 – CERTO	Q34 - CERTO
Q35 - ERRADO	Q36 - LETRA B
Q37 - LETRA C.	