

Programa o Estruturada

Aula 06 - Operadores JavaScript: Aritm tica

Videoaula 05 - Infinity e -Infinity



Videoaula 05 - Infinity e -Infinity

Outros dois valores numéricos especiais de JavaScript são o `Infinity` e o `-Infinity`. Ambos são do tipo `number` e são retornados caso algum número calculado esteja fora do intervalo permitido para números JavaScript. Vamos aos exemplos!

Vou iniciar o exemplo onde podemos atingir rapidamente o valor limite, ou seja, o maior número positivo de JavaScript, e veremos o que acontece quando escrevemos esse valor que já passou dos limites de JavaScript na tela.

Então, terei duas variáveis, a variável `texto` começa com a *string* vazia, e a variável `x` começa com o valor 10, e aí atribuímos a `texto` o valor de `x`, concatenado com a tag `
`, que é a quebra de linha de HTML. Na linha 19, vou elevar o valor de `x` a ele mesmo, 10 elevado a 10 nos dará um número enorme, e depois colocar esse valor no `texto` (linha 20).

Na linha 21, vou pegar aquele valor enorme e elevá-lo a ele mesmo e aí, com certeza, vamos extrapolar os limites, o maior número positivo de JavaScript, e eu vou imprimir na tela esse valor que já passou do maior valor que JavaScript suporta, e espero que essa escrita aconteça, que o valor `Infinity` seja escrito na tela.

Código 1 - 06_6 Number.html

```
1 <html>
2   <head>
3     <meta charset="UTF-8" />
4     <title>Programação Estruturada - Aula 06</title>
5   </head>
6   <body>
7     <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9     <h1>Number</h1>
10    <p id="texto"></p>
11
12    <script>
```

```

13 var texto = "";
14 var x = 10;
15 var texto = x + "<br>";
16
17 x = x ** x;
18 texto = texto + x + "<br>";
19 x = x ** x;
20 texto = texto + x + "<br><br>";
21
22 //var y = 2 / 0;
23 //texto = texto + "2 / 0 = " + y + "<br>";
24 //var z = -2 / 0;
25 //texto = texto + "-2 / 0 = " + z + "<br><br>";
26
27 texto = texto + "typeof Infinity = " + typeof Infinity;
28
29 document.getElementById("texto").innerHTML = texto;
30 </script>
31 </body>
32 </html>
33

```

E então podemos ver que, de fato, ao carregarmos essa página HTML, temos o primeiro valor que demos a x , que é 10, depois nós fizemos 10 elevado a 10, que deu esse número enorme (10 bilhões). Depois, nós elevamos 10 bilhões a 10 bilhões, resultando em um número enorme que ultrapassa o limite que JavaScript suporta, e aí o valor escrito na tela é, de fato, o `Infinity` positivo.

Figura 1 - Valor `Infinity`

Number

```

10
10000000000
Infinity

```

Se, por outro lado, optamos por definir dois valores, sendo a variável `y` o resultado de 2 dividido por 0, que valor seria esse, $2 / 0$? Ao carregar a página HTML, observe que esse valor também é o valor `Infinity`. E, se por acaso, eu fizer o valor `z`, que é -2 dividido por 0, o que teremos é o valor `-Infinity`, ok?

Figura 2 - Valor `Infinity`

Number

$2 / 0 = \text{Infinity}$

Figura 3 - Valor `-Infinity`

Number

$-2 / 0 = \text{-Infinity}$

Por fim, apenas para podermos conferir qual é o tipo desse valor `Infinity`, eu vou imprimir na tela o texto `"typeof Infinity ="` e vou acrescentar essa *string*, o resultado da aplicação da função `typeof` ao valor `Infinity`, e aí a gente vê que, de fato o tipo do valor `Infinity` é o tipo `number`.

Código 2 - 06_6 Number.html

```
1 <html>
2   <head>
3     <meta charset="UTF-8" />
4     <title>Programação Estruturada - Aula 06</title>
5   </head>
6   <body>
```

```
7 <noscript>Seu navegador não suporta JavaScript ou ele está desabilitado.</noscript>
8
9 <h1>Number</h1>
10 <p id="texto"></p>
11
12 <script>
13 var texto = "";
14 texto = texto + "typeof Infinity = " + typeof Infinity;
15
16 document.getElementById("texto").innerHTML = texto;
17 </script>
18 </body>
19 </html>
20
```

Figura 4 - Tipo Infinity

Number

```
typeof Infinity = number
```



Em JavaScript, os números são normalmente valores primitivos criados a partir de numerais, os quais não podem ter propriedades e métodos (porque não são objetos). Porém, com JavaScript, métodos e propriedades também estão disponíveis para valores primitivos, porque o JavaScript trata valores primitivos como objetos ao executar métodos e propriedades. Dessa forma, sempre que estiver trabalhando com um número, você poderá usar os métodos e propriedades do objeto `Number` nele. Na próxima videoaula, você aprenderá como fazer isso. Até lá!