

---

## Console object

---

شیء کنسول ، امکان دسترسی به کنسول اشکال زدایی مرورگر را فراهم میکند.

شیء کنسول . یکی از ویژگی های شیء ویندو میباشد.


شیء کنسول با دستورات زیر قابل دسترسی هستند:

`window.console` or `console`

---

### Console object methods

---

<b><code>assert()</code></b>	تنها یک مقدار بولین را دریافت می کند و میتواند یک شرط را نیز مستقیما دریافت کند و تنها در صورتی کار میکند که شرط نادرست باشد
<b><code>clear()</code></b>	کنسول را پاک میکند
<b><code>count()</code></b>	برای شمردن تعداد لاگ ها در کنسول مرورگر مورد استفاده قرار میگیرد
<b><code>error()</code></b>	نمایش پیام ارور در کنسول
<b><code>group()</code></b>	برای رفتن خروجی در کنسول به سطح بعدی استفاده میشود 
<b><code>groupCollapsed()</code></b>	مانند <code>group()</code> عمل میکند
<b><code>groupEnd()</code></b>	پس از استفاده از <code>group()</code> برای اتمام سطح بندی و بازگشت به سطح اولیه از <code>groupEnd()</code> استفاده میشود
<b><code>info()</code></b>	پیام های که جنبه اطلاع رسانی دارند را در کنسول به نمایش میگذارد
<b><code>log()</code></b>	انواع مختلف داده را در کنسول و خطاهایی که برنامه نویس مستقیما مرتکب نشده به نمایش می گذارد
<b><code>table()</code></b>	داده هایی را که قابل منظم شدن یا قرار گرفتن در جدول هستند (مثل آرایه ها و آبجکت ها) را در به صورت جدول بندی شده در کنسول به نمایش میگذارد
<b><code>time()</code></b>	تایمر را استارت میکند(میتواند مدت زمان صرف شده برای عملیات را پیگیری کند).
<b><code>timeEnd()</code></b>	تایمری را که با <code>console.time()</code> استارت شده را متوقف میکند
<b><code>trace()</code></b>	جز اینکه یک پیام در کنسول به نمایش می گذارد مسیر طی شده برای رسیدن متد <code>trace()</code> را در کنسول نشان میدهد
<b><code>warn()</code></b>	پیام هشدار دهنده را در کنسول به نمایش میگذارد

---

## Console assert()

---

این متد در خروجی تنها یک نوع داده دریافت می کند و آن داده از نوع Boolean است البته شما میتوانید به صورت مستقیم یک شرط را نیز در آن قرار دهید متد `assert()` تنها یک مقدار **False** را نشان میدهد اگر شرط شما نادرست باشد این متد در کنسول نمایش داده میشود .

`console.assert(expression , message)`

```
<script>
  var x = 2,
  y = 3;
  console.assert(x + y == 4, 'expression is false');
```

✖ ▶ Assertion failed: expression is false

index2.html:12

---

## Console clear()

---

تمام پیام هایی که در کنسول وجود دارند را پاک میکند

`Console.clear()`

✖ ▶ Assertion failed: expression is false

index2.html:12

✖ ▶ GET <https://www.facebook.com/tr?id=1731524830476313&ev=PageView>  
net::ERR\_TIMED\_OUT

[www.facebook.com/tr?id=1731524830476313&ev=PageView:1](https://www.facebook.com/tr?id=1731524830476313&ev=PageView)

✖ ▶ WebSocket connection to 'ws://192.168.92.203:9222/ws' failed:

reload.js:22

> console.clear()

Console was cleared

VM131:1

< undefined

> |

---

## Console count()

---

برای شمردن لاگ ها در کنسول مرورگر مورد استفاده قرار میگیرد این متد تعداد بارهایی که خودش فراخوانی میشود را می‌شمرد مثلا اگر در یک حلقه استفاده شود هر بار که حلقه اجرا میشود تعداد دفعات اجرا شده حلقه را در کنسول نشان میدهد

### Console.count()

```
9      <script>
10      |   for(i = 1 ; i <= 5; i++){
11      |       console.count();
12      |   }
13      </script>
```

default: 1	<a href="#">index2.html:11</a>
default: 2	<a href="#">index2.html:11</a>
default: 3	<a href="#">index2.html:11</a>
default: 4	<a href="#">index2.html:11</a>
default: 5	<a href="#">index2.html:11</a>

---

## Console error()

---

این متد برای نمایش ارور در کنسول مرورگر مورد استفاده قرار میگیرد.

```
9      <script>
10      |   console.error("process undefined");
11      </script>
```

✖ process undefined index2.html:10

---

## Console group()

---

با استفاده از این متد میتوان تعدادی از پیام هایی را که در کنسول نمایش داده میشود را در گروه جداگانه قرار داد به صورتی که نسبت به بقیه پیام هایی که در کنسول نمایش داده میشود یک مرحله

بیرون

```
<script>
  console.group();
  console.error("process undefined");
  console.log('apple');
  console.log('banana');
  console.groupEnd();
  console.log('cucumber');
</script>
```

console.group

✖ process undefined

apple

banana

cucumber

---

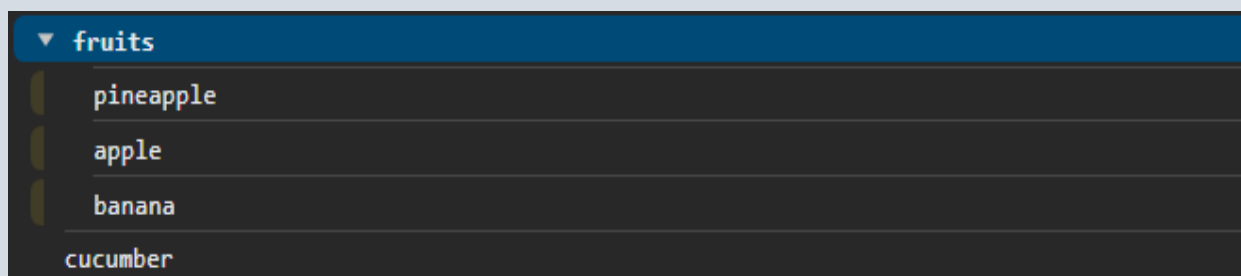
## Console groupCollapsed()

---

با استفاده از این متد میتوان یک گروه به اسم Collapse group در کنسول ایجاد کرد و تعدادی از پیام هایی را که در کنسول نمایش داده میشود در این گروه قرار داد که همانند گروه یک سطح از بقیه پیام های کنسول جلوتر هستند.

Console.groupCollapsed(label)

```
<script>
  console.groupCollapsed('fruits');
  console.log('pineapple');
  console.log('apple');
  console.log('banana');
  console.groupEnd();
  console.log('cucumber');
</script>
```



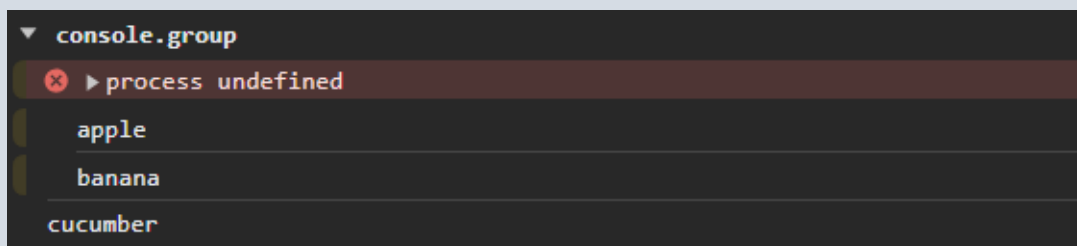
---

## Console groupEnd()

---

با استفاده از این متد میتوان محدوده یک گروه از پیام ها را که در کنسول قرار دارد پایان

```
<script>
  console.group();
  console.log('pineapple');
  console.log('apple');
  console.log('banana');
  console.groupEnd();
  console.log('cucumber');
</script>
```



---

## Console info()

---

یک پیام اطلاعاتی را در خروجی کنسول مرورگر نمایش میدهد

Console.info()

```
<script>
  console.info('some information !!!');
</script>
```

some information !!!

index2.html:10

---

## Console log()

---

این متد پر استفاده ترین متد کنسولی جاوا اسکریپت است که توسط توسعه دهندگان بسیار زیادی استفاده می‌شود. با استفاده از این متد شما می‌توانید انواع مختلف داده‌ای جاوا اسکریپت را در قسمت کنسول به نمایش بگذارید. جدای از آن این متد وظیفه نشان داده خطاهایی را دارد که خود برنامه نویس ممکن است به صورت مستقیم مرتکب آن‌ها نشده باشد. استفاده از این روش برای خطایابی بسیار کم دردرس است

```
<script>
  var testVariable = 404;
  console.log("This is triggered by console.log");
  console.log("checking the variable value",testVariable);
</script>
```

This is triggered by console.log

index2.html:11

checking the variable value 404

index2.html:12

---

## Console table()

---

داده‌هایی را که قابلیت نمایش داده شدن به صورت جدول را دارند (آبجکت‌ها و آرایه‌ها) در کنسول مرورگر به صورت جدول منظم نمایش میدهد.

```
<script>
  var fruits = ['apple' , 'orange' , 'pineapple' , 'banana']
  console.table(fruits);
</script>
```

(index)	Value
0	'apple'
1	'orange'
2	'pineapple'
3	'banana'

► Array(4)

---

## Console time()

---

متد تایم یک تایمر را در کنسول مرورگر استارت میکند مثلا میتوان تایمر را برای راه اندازی یک حلقه for برای ۱۰۰ بار استارت کرد و تا پایان حلقه ادامه داد و در انتهای حلقه از متد Timeend() برای پایان حلقه استفاده کرد.

```
<script>
  console.time();
  for(var i=0; i<100; i++) {
    console.log('salam');
  }
  console.timeEnd();
</script>
```

100 salam

default: 12.659912109375 ms

---

## Console trace()

---

کاری که این متد انجام میدهد جدای اینکه یک پیام را چاپ میکند این است که یک Stack trace را ایجاد میکند در این حالت ، متد trace مسیر طی شده برای رسیدن به متد Trace را نشان خواهد داد

```
<script>
  console.log('salam');
  console.log(2+2);
  console.log(2**3);
  console.trace();
</script>
```

```
salam                                                                    index2.html:10
4                                                                        index2.html:11
8                                                                        index2.html:12
▼ console.trace                                                         index2.html:13
  (anonymous) @ index2.html:13
```

---

## Console warn()

---

این متد به ما امکان میدهد تا یک پیام به سبک وارنینگ را نشان دهیم برای این کار در داخل متد هرنوع داده ی را وارد کرده و بعد در قسمت کنسول مرورگر پیام خروجی را به صورت وارنینگ نمایش میدهد.

```
<script>
  console.warn('important!!!')
</script>
```

▶ important!!!

index2.html:10