

## Switch

@alithecodeguy

یک عبارت switch می‌تواند جایگزین چندین if شود و راه مشروح‌تری، برای مقایسه یک مقدار با چندین مقدار دیگر را در اختیار ما قرار می‌دهد.

### گرامر

عبارت switch می‌تواند یک یا چندین بلاک case و یک بلاک default دلبخواه داشته باشد و به شکل زیر نوشته می‌شود :

```
switch(x) {  
  case 'value1': // if (x === 'value1')  
    ...  
    [break]  
  case 'value2': // if (x === 'value2')  
    ...  
    [break]  
  default:  
    ...  
    [break]  
}
```

- مقدار **x** توسط **===** با مقدار اولین case مقایسه می‌شود یعنی مقدار value1. اگر برابر بودند، کد موجود در بدنه case تا نزدیکترین break انجام می‌شود سپس از بدنه switch خارج می‌شود، در غیر اینصورت سراغ case بعدی می‌رود.
  - این فرآیند را تا زمان پیدا کردن case مورد نظر ادامه می‌دهد.
  - در صورت عدم یافتن case برابر، بدنه موجود در default اجرا می‌شود.
- مثال :

```
let a = 2 + 2;  
switch (a) {  
  case 3:  
    alert('Too small');  
    break;  
  case 4:  
    alert('Exactly!');  
    break;  
}
```

@alithecodeguy

case 5:

```
alert( 'Too large' );
```

```
break;
```

default:

```
alert( "I don't know such values" );
```

```
}
```

در این مثال مقدار a ابتدا با case اول و مقدار 3 چک می شود و چون برابر نیستند به سراغ case دوم و مقدار 4 می رود و چون در این حالت برابر هستند پس بدنه case دوم تا نزدیکترین break اجرا می شود.

اگر break را ننویسیم ، اجرای کد تا رسیدن به break یا انتهای switch ادامه پیدا می کند. مثال :

```
let a = 2 + 2;
```

```
switch (a) {
```

case 3:

```
alert( 'Too small' );
```

case 4:

```
alert( 'Exactly!' );
```

case 5:

```
alert( 'Too big' );
```

default:

```
alert( "I don't know such values" );
```

```
}
```

اگر مثال فوق را اجرا کنید می بینید که دستورات به ترتیب زیر اجرا می شود :

```
alert( 'Exactly!' );
```

```
alert( 'Too big' );
```

```
alert( "I don't know such values" );
```

---

هم در switch و هم case های آن می توانید از عبارات مختلف استفاده کنید. برای مثال :

```
switch (+a) {
```

case b + 1:

```
alert("this runs, because +a is 1, exactly equals b+1");
```

```
break;
```

default:

```
alert("this doesn't run");
```

```
}
```

در این مثال +a مقدار 1 را بر می گرداند و این مقدار با مقدار b + 1 مقایسه می شود و در صورتی که برابر باشند بدنه case اجرا می شود.

## گروه بندی case ها

چندین case که کد مشترکی را اجرا کنند می توانند با یکدیگر ، یک گروه را تشکیل دهند . در مثال زیر اگر case 3 و case 5 کد مشترکی را اجرا کنند می تواند به شکل زیر نوشته شود :

```
let a = 3;
switch (a) {
  case 4:
    alert('Right!');
    break;
  case 3: // (*) grouped two cases
  case 5:
    alert('Wrong!');
    alert("Why don't you take a math class?");
    break;
  default:
    alert('The result is strange. Really.');
```

```
}
```

در این حالت ، case 3 و case 5 هر دو کد یکسانی را اجرا می کنند .

قابلیت گروه بندی یکی از ویژگی های جانبی استفاده نکردن از break در switch است . در مثال فوق اجرای case 3 از خط \* شروع شده و از خط case 5 عبور می کند چرا که به هیچ break نمی رسد .

## نوع داده اهمیت دارد

چک کردن برابری همیشه با === انجام می شود بنابراین برای اینکه مقادیر برابر باشند ، باید نوع یکسانی نیز داشته باشند . مثال :

```
let arg = prompt("Enter a value?");
switch (arg) {
  case '0':
  case '1':
    alert( 'One or zero' );
    break;
  case '2':
    alert( 'Two' );
    break;
  case 3:
    alert( 'Never executes!' );
    break;
```

default:

```
alert('An unknown value');  
}
```

- برای مقادیر رشته ای 0 و 1 دستور alert اول اجرا می شود.
- برای مقدار رشته ای 2 دستور alert دوم اجرا می شود.
- case 3 هیچ گاه اجرا نمی شود چرا که مقدار بازگشتی از prompt به صورت یک رشته است. پس اگر مقدار 3 را وارد کنیم بدنه قسمت default اجرا می شود.

@alithecodeguy