# TypeScript switch case

**Summary**: in this tutorial, you will about the TypeScript `switch...case` statement.

## Introduction to TypeScript switch case statement

The following shows the syntax of the `switch...case` statement:

```
switch ( expression ) {
   case value1:
       // statement 1
       break;
   case value2:
       // statement 2
       break;
   case valueN:
       // statement N
       break;
   default:
       //
       break;
}
```

How it works:

First, the `switch...case` statement evaluates the `expression`.

Then, it searches for the first `case` clause whose expression evaluates to the same value as the value ( `value1`, `value2`, … `valueN` ).

The `switch...case` statement will execute the statement in the first `case` clause whose value matches.

If no matching `case` clause is found, the `switch...case` statement looks for the optional `default` clause. If the `default` clause is available, it executes the statement in the `default` clause.

The `break` statement that associates with each `case` clause ensures that the control breaks out of the `switch...case` statement once the statements in the `case` clause complete.

If the matching case clause doesn't have the `break` statement, the program execution continues at the next statement in the `switch...case` statement.

By convention, the `default` clause is the last clause in the `switch...case` statement. However, it doesn't need to be so.

## TypeScript switch case statement examples

Let's take some examples of using the `switch...case` statement.

### 1) A simple TypeScript switch case example

The following example shows a simple `switch...case` example that shows a message based on the target id:

```
let targetId = 'btnDelete';

switch (targetId) {
    case 'btnUpdate':
        console.log('Update');
        break;
    case 'btnDelete':
        console.log('Delete');
        break;
    case 'btnNew':
        console.log('New');
        break;
}
```

Output:

```
Delete
```

In this example, the `targetId` is set to `btnDelete`.

The `switch...case` statement compares the `targetId` with a list of values. Because the `targetId` matches the `'btnDelete'` the statement in the corresponding `case` clause executes.

## 2) Grouping case example

If you have a code that is shared by multiple cases, you can group them. For example:

```
// change the month and year
let month = 2,
    year = 2020;

let day = 0;
switch (month) {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
        day = 31;
        break;
    case 4:
    case 6:
    case 9:
    case 11:
        day = 30;
        break;
    case 2:
        // leap year
        if (((year % 4 == 0) &&
            !(year % 100 == 0))
            || (year % 400 == 0))
            day = 29;
        else
```

```
            day = 28;
        break;
    default:
        throw Error('Invalid month');
}

console.log(`The month ${month} in ${year} has ${day} days`);
```

Output:

```
The month 2 in 2020 has 29 days
```

This example returns the days of a specific month and year.

If the month is 1,3, 5, 7, 8, or 12, the number of days is 31. If the month is 4, 6, 9, or 11, the number of days is 30.

If the month is 2 and the year is a leap year, it returns 29 days, otherwise, it returns 28 days.