

# TypeScript Types

If this site saves you hours of work, please  
**whitelist it in your ad blocker** 🙏 to  
**support us** ❤️  
in creating more helpful and free content  
in the future.

**Summary:** in this tutorial, you'll learn about the TypeScript types and their purposes.

## What is a type in TypeScript

In TypeScript, a type is a convenient way to refer to different **properties** and **functions** that a **value** has.

A value is anything you can assign to a variable e.g., a number, a string, an array, an object, and a function.

For example, see the following value:

```
'Hello'
```

When you look at this value, you can say it is a string. This value has properties and methods that a string has.

For example, the `'Hello'` value has a property called `length` that returns the number of characters:

```
console.log('Hello'.length); // 5
```

It also has many methods like `match()`, `indexOf()`, and `toLocaleUpperCase()`. For example:

```
console.log('Hello'.toLocaleUpperCase()); // HELLO
```

When you look at the value `'Hello'` and describe it by listing the properties and methods, it would be inconvenient.

A shorter way to refer to a value is to assign it a type. In this example, you say `'Hello'` is a string. Then, you know that you can use the properties and methods of a string for the value `'Hello'`.

In conclusion, in TypeScript:

- a type is a label that describes the different properties and methods that a value has
- every value has a type.

## Types in TypeScript

TypeScript inherits the built-in types from JavaScript. TypeScript types are categorized into:

- Primitive types
- Object types

### Primitive types

The following illustrates the primitive types in TypeScript:

Name	Description
<code>string</code>	Represent text data.
<code>number</code>	Represent numeric values.
<code>boolean</code>	Have true and false values.
<code>null</code>	Have one value: null.
<code>undefined</code>	Have one value: undefined. It is the default value of an uninitialized variable.
<code>symbol</code>	Represent a unique constant value.

### Object types

Object types are functions, arrays, classes, etc. Later, you'll learn how to create custom object types.

## Purposes of types in TypeScript

There are two main purposes of types in TypeScript:

- First, types are used by the TypeScript compiler to analyze your code for errors.
- Second, types allow you to understand what values are associated with variables.

## Examples of TypeScript types

The following example uses the `querySelector()` method to select the `<h1>` element:

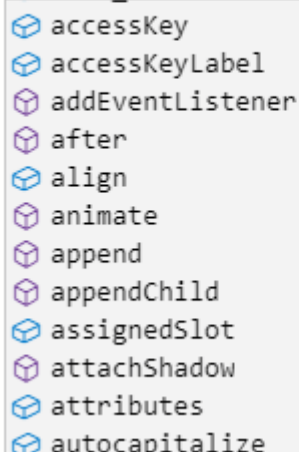
```
const heading = document.querySelector('h1');
```

The TypeScript compiler knows that the type of `heading` is `HTMLHeadingElement`:

```
const heading = document.querySelector('h1');  
const heading: HTMLHeadingElement | null
```

And it shows a list of methods for the `HTMLHeadingElement` type that `heading` can access:

```
const heading = document.querySelector('h1');  
heading?.
```



A screenshot of an IDE showing a list of methods for the `HTMLHeadingElement` type. The list includes: `accessKey`, `accessKeyLabel`, `addEventListener`, `after`, `align`, `animate`, `append`, `appendChild`, `assignedSlot`, `attachShadow`, `attributes`, and `autocapitalize`. Each method is preceded by a small blue icon.

If you try to access a property or method that doesn't exist, the TypeScript compiler will show an error. For example:

```
const heading = document.querySelector('h1');  
heading.rotate()
```

## Summary

- Every value in TypeScript has a type.
- A type is a label that describes the properties and methods that a value has.
- TypeScript compiler uses types to analyze your code for hunting bugs and errors.