TypeScript readonly

Summary: in this tutorial, you will learn how to use the TypeScript `readonly` access modifier to mark class properties as immutable properties.

TypeScript provides the `readonly` modifier that allows you to mark the properties of a class immutable. The assignment to a `readonly` property can only occur in one of two places:

- In the property declaration.
- In the constructor of the same class.

To mark a property as immutable, you use the `readonly` keyword. The following shows how to declare a `readonly` property in the `Person` class:

```
class Person {
    readonly birthDate: Date;

    constructor(birthDate: Date) {
        this.birthDate = birthDate;
    }
}
```

In this example, the `birthdate` property is a `readonly` property that is initialized in the constructor of the `Person` class.

The following attempts to reassign the `birthDate` property that results in an error:

```
let person = new Person(new Date(1990, 12, 25));
```

```
person.birthDate = new Date(1991, 12, 25); // Compile error
```

Error:

```
Cannot assign to 'birthDate' because it is a read-only property.
```

Like other access modifiers, you can consolidate the declaration and initialization of a `readonly` property in the constructor like this:

```
class Person {
    constructor(readonly birthDate: Date) {
    }
}
```

readonly vs. const

The following shows the differences between `readonly` and `const`:

	`readonly`	`const`
Use for	Class properties	Variables
Initialization	In the declaration or in the constructor of the same class	In the declaration

Summary

- Use the `readonly` access modifier to mark a class property as immutable.
- A `readonly` property must be initialized as a part of the declaration or in the constructor of the same class.