# Classes



Prior to ES6, we implemented Classes by creating a constructor function and adding properties by extending the prototype:

```
function Person(name, age, gender) {
    this.name   = name;
    this.age    = age;
    this.gender = gender;
}

Person.prototype.incrementAge = function () {
    return this.age += 1;
};
```

And created extended classes by the following:

```
function Personal(name, age, gender, occupation, hobby) {
    Person.call(this, name, age, gender);
    this.occupation = occupation;
    this.hobby = hobby;
}

Personal.prototype = Object.create(Person.prototype);
Personal.prototype.constructor = Personal;
Personal.prototype.incrementAge = function () {
    Person.prototype.incrementAge.call(this);
    this.age += 20;
    console.log(this.age);
};
```

ES6 provides much needed syntactic sugar for doing this under the hood. We can create Classes directly:

```
class Person {
    constructor(name, age, gender) {
        this.name   = name;
        this.age    = age;
        this.gender = gender;
    }

    incrementAge() {
      this.age += 1;
    }
}
```

And extend them using the `extends` keyword:

```
class Personal extends Person {
    constructor(name, age, gender, occupation, hobby) {
        super(name, age, gender);
        this.occupation = occupation;
        this.hobby = hobby;
    }

    incrementAge() {
        super.incrementAge();
        this.age += 20;
        console.log(this.age);
    }
}
```

> **Best Practice**: While the syntax for creating classes in ES6 obscures how implementation and prototypes work under the hood, it is a good feature for beginners and allows us to write cleaner code.