Reflection

introduction to reflection and using the method `Reflect.apply`

Reflection in a programming language is the act of inspecting, dynamically calling, and modifying classes, objects, properties, and methods. In other words, reflection is the ability of the programming language to *reflect* on the structure of the code.

The ES6 Reflect API gives you a Reflect object that lets you call methods, construct objects, getting and setting prototypes, manipulating and extending properties.

Reflection is important, because it lets you write programs and frameworks that are able to handle a non-static code structure.

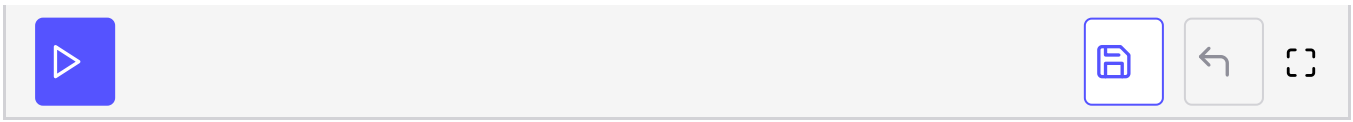
One use case is **automated testing**. You don't have to do expensive and complicated setup operations in order to test some methods of a class in isolation.

We can also use the Reflect API in proxies. We will learn about proxies in the next chapter.

`Reflect.apply` calls functions or methods.

```
let target = function getArea( width, height ) {
    return `${ width * height }${this.units}`;
}
let thisValue = { units: 'cm' };
let args = [ 5, 3 ];


console.log('Area: '+Reflect.apply( target, thisValue, args ));
```



Now let's learn how to create objects in reflection.