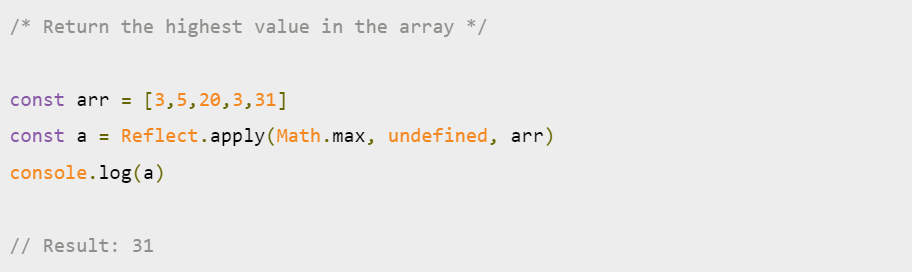
**What is JavaScript Reflect?**

JavaScript Reflect is an inbuilt ES6 global object that provides the ability to manipulate properties, variables, and object methods at runtime. It is not a constructor, therefore you cannot use the [new](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/new) operator with it.

**Methods:**

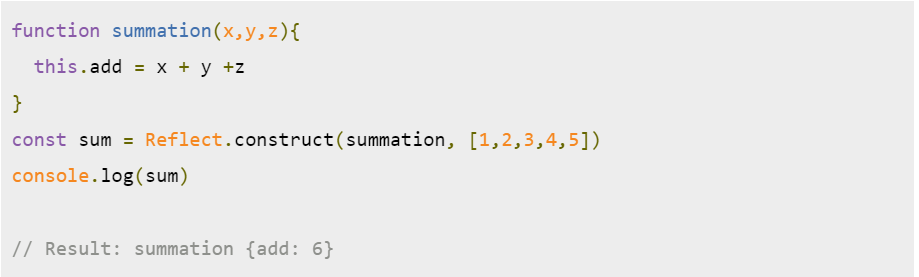
1. **Reflect.apply()**

* Syntax : Reflect.apply(target, thisArgument, argumentsList)
* Description
  + In ES5, you typically use the Function.prototype.apply() method to call a function with a given this value and arguments provided as an array (or an array-like object).
* Parameters :
* Target: The target function to call.
* thisArgument: The value of this provided for the call to target.
* argumentsList: An array-like object specifying the arguments with which target should be called.
* Return Value
  + The result of calling the given target function with the specified this value and arguments.
* Examples:



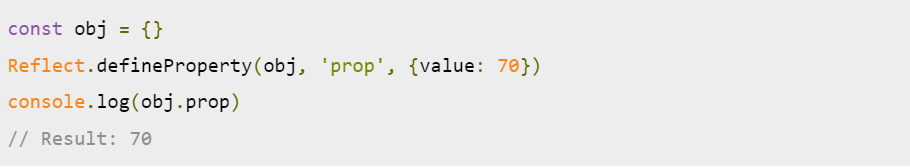
1. **Reflect.construct()**

* Syntax : Reflect.construct(target, argumentsList) || Reflect.construct(target, argumentsList)
* Description:
  + allows you to invoke a constructor with a variable number of arguments. (This would also be possible by using the spread syntax combined with the new operator.)
* Parameters :
* Target: The target function to call.
* argumentsList: An array-like object specifying the arguments with which target should be called.
* newTarget (optinal): The constructor whose prototype should be used. See also the new.target operator. If newTarget is not present, its value defaults to target.
* Return Value:
  + A new instance of target (or newTarget, if present), initialized by target as a constructor with the given argumentsList.
* Examples:



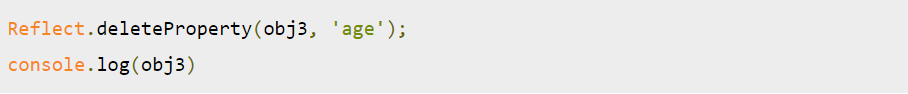
1. **Reflect.** **defineProperty ()**

* Syntax : Reflect.defineProperty(target, propertyKey, attributes)
* Description:
  + The Reflect.defineProperty method allows precise addition to or modification of a property on an object. For more details, see the Object.defineProperty which is similar.
* Parameters :
* Target: The target object on which to define the property.
* propertyKey: The name of the property to be defined or modified.
* attributes: The attributes for the property being defined or modified.
* Return Value:
  + A [Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean) indicating whether or not the property was successfully defined.
* Example:



1. **Reflect.deleteProperty()**

* Syntax : Reflect.deleteProperty(target, propertyKey)
* Description:
  + The Reflect.deleteProperty method allows you to delete a property on an object. It returns a [Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean) indicating whether or not the property was successfully deleted. It is almost identical to the non-strict [delete operator](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/delete).
* Parameters :
* Target: The target object on which to delete the property.
* propertyKey: The name of the property to be deleted.
* Return Value:
  + A [Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean) indicating whether or not the property was successfully deleted.
* Examples:



1. **Reflect.get()**

* Syntax : Reflect.get(target, propertyKey) || Reflect.get(target, propertyKey, receiver)
* Description: The Reflect.get method allows you to get a property on an object. It is like the [property accessor](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Property_Accessors) syntax as a function
* Parameters :
* Target: The target object on which to get the property.
* propertyKey: The name of the property to get.
* Reciever (optional) : The value of this provided for the call to target if a getter is encountered. When used with [Proxy](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Proxy), it can be an object that inherits from target.
* Return Value: The value of the property.
* Examples:



1. **Reflect. getOwnPropertyDescriptor**

* [Syntax](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#syntax): Reflect.getOwnPropertyDescriptor(target, propertyKey)
* [Parameters](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#parameters)
  + Target: The target object in which to look for the property.
  + propertyKey: The name of the property to get an own property descriptor for.
* [Return value](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#return_value)
  + A property descriptor object if the property exists in target object; otherwise, [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined).
* [Exceptions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#exceptions)
  + A [TypeError](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/TypeError), if target is not an [Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object).
* [Description](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#description)
  + The Reflect.getOwnPropertyDescriptor method returns a property descriptor of the given property if it exists in the target object, [undefined](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/undefined) otherwise. The only difference to [Object.getOwnPropertyDescriptor()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/getOwnPropertyDescriptor) is how non-object targets are handled.
* [Examples](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/getOwnPropertyDescriptor#examples)



1. **Reflect.set()**

## [Syntax](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#syntax):

* + Reflect.set(target, propertyKey, value)
  + Reflect.set(target, propertyKey, value, receiver)

### [Parameters](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#parameters)

* + Target: The target object on which to set the property.
  + propertyKey:The name of the property to set.
  + Value: The value to set.
  + receiver Optional: The value of this provided for the call to the setter for propertyKey on target. If provided and target does not have a setter for propertyKey, the property will be set on receiver instead.

### [Return value](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#return_value)

* + A [Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean) indicating whether or not setting the property was successful.

### [Exceptions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#exceptions)

* + A [TypeError](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/TypeError), if target is not an [Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object).

## [Description](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#description)

* + The Reflect.set method allows you to set a property on an object. It does property assignment and is like the [property accessor](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/Property_Accessors) syntax as a function.

## [Examples](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/set#examples)

## 

1. **Reflect.has()**

* [Syntax](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#syntax): Reflect.has(target, propertyKey)
* [Parameters](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#parameters)
  + Target: The target object in which to look for the property.
  + propertyKey : The name of the property to check.
* [Return value](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#return_value)
  + A [Boolean](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Boolean) indicating whether or not the target has the property.
* [Exceptions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#exceptions)
  + A [TypeError](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/TypeError), if target is not an [Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object).
* [Description](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#description)
  + The Reflect.has method allows you to check if a property is in an object. It works like the [in operator](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Operators/in) as a function.
* [Examples](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/has#examples):



1. **Reflect.ownKeys()**

* [Syntax](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#syntax): Reflect.ownKeys(target)
* [Parameters](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#parameters)
  + Target: The target object from which to get the own keys.
* [Return value](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#return_value)
  + An [Array](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array) of the target object's own property keys.
* [Exceptions](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#exceptions)
  + A [TypeError](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/TypeError), if target is not an [Object](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object).
* [Description](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#description)
  + The Reflect.ownKeys method returns an array of the target object's own property keys. Its return value is equivalent to [Object.getOwnPropertyNames(target)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/getOwnPropertyNames).concat([Object.getOwnPropertySymbols(target)](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Object/getOwnPropertySymbols)).
* [Examples](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Reflect/ownKeys#examples)

