Undefined

As the name suggests, undefined means "not defined". So we declare a variable but do not assign a value to it, the variable becomes undefined.

Unlike null, the value of an undefined variable is set by JavaScript as undefined. The variable gets created at the run-time. When we do not pass an argument for a function parameter, the default value is taken as undefined. Besides, when a function doesn't return a value, it returns undefined.

Null

Null is basically an assignment value given to a variable. The variable which has been assigned as null contains no value.

When we assign null as a value to any variable, it means that it is empty or blank. It is to show that the variable has no value. Also, null is an object in JavaScript. When it gets assigned to a variable, it represents no value. The setting of the value must be done manually by the user as JavaScript never sets the value as null. An object can be emptied by setting it to null.

**0** is a number which is a falsy value in javascript where as **null**,**undefined** are also falsy values but represents a lack of value. **""** (empty string) is a string which is also a falsy value in javascript, it represents an empty string.

difference between var, let, and const :

The differences between **var**, **let**, and **const** variable declaration in JavaScript include:  
Variables declared with **var** and **const** are scoped to the immediate function body.  
Variables declared with the **var** keyword are *hoisted*. Hoisting means that the variable can be accessed in their enclosing scope even before they are declared.  
Variables declared with the **let** keyword are block-scoped, which means the variables will have scope to the immediate enclosing block.

Var variable :

The **var** keyword has traditional variable declaration syntax. It is optional to initialize a variable with a value if it is declared with the **var** keyword. If developers do not initialize it with a value, JavaScript will automatically assign **undefined** to it.

Hoisting of var in JavaScript

Variables declared with the **var** keyword are subject to hoisting. If we declare a variable (but do not initialize it) at the end of the function, the JavaScript runtime will move it to the top of its scope and, therefore, there will be no complaint by the runtime if we use that variable before being declared.

Let variable :

**let** is the improved version of **var**.

### Hoisting of let

Variables declared with the **let** keyword are not subject to hoisting. This means you cannot use a variable unless it is declared and initialized.

Const variable :

The **const** keyword follows the same rules as the **let** keyword. The only difference with **const** is that **const** is used to define *only* constant values in JavaScript programs.