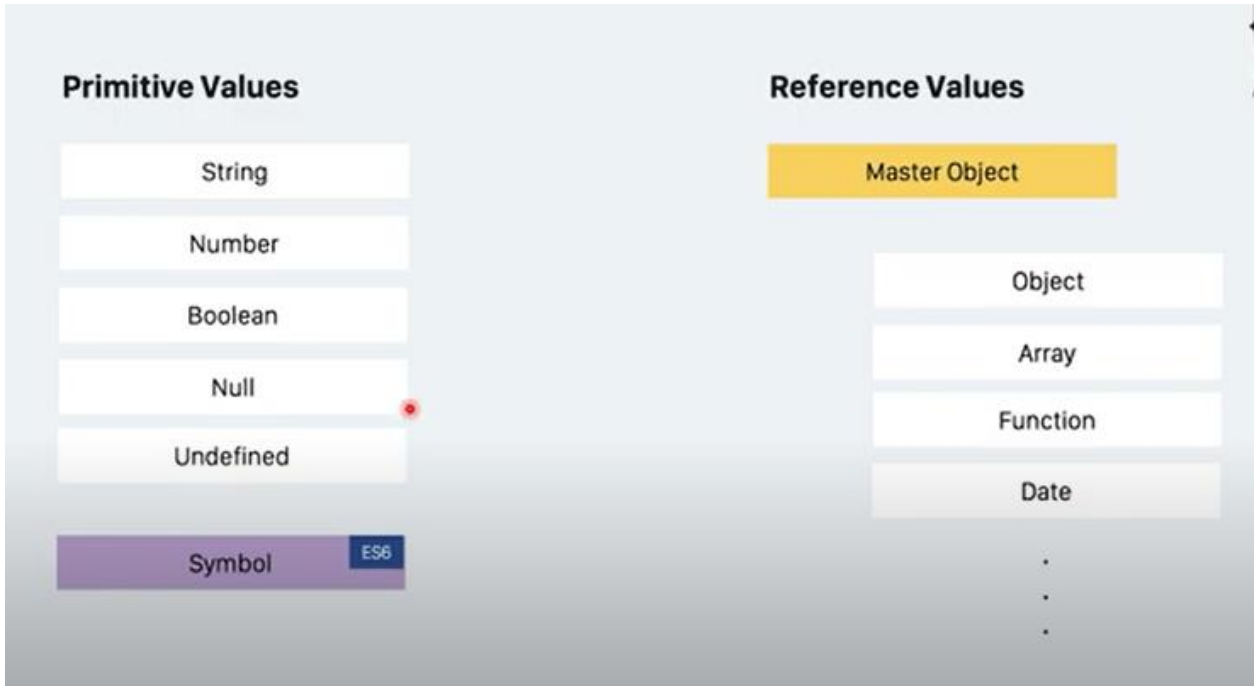
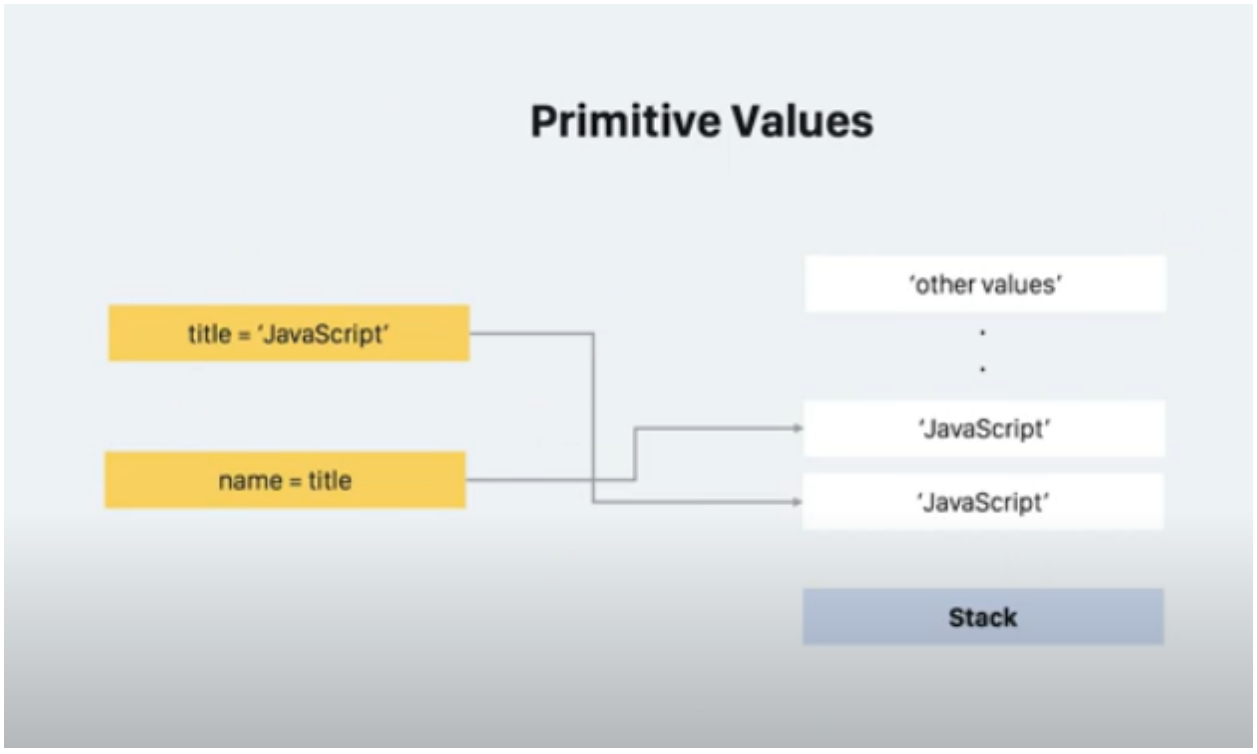
Primitive VS Reference




```
//Reference value ekta mashter object theke ashe
//primitive value gula stack er moto structure a rakhe & Reference value gula Heap er moto structure a rakhe
```

```
var a = 5;
var b = a;
console.log(a);//5
console.log(b);//5



//a k change korbo
var a = 5;
var b = a;
a = 6;
console.log(a);//6
console.log(b);//5


//a er modda ekta object nilam
var a = {
    name: 'sakib',
}
var b = a;

a.name = "Tamim";

console.log(a);
console.log(b);
//output: {name: "Tamim"} {name: "Tamim"}
//b o tamim hoiya gese. ai outpit unexpectrd



//a er modda array nibo
var a = ['sakib', "tamim"];
var b = a;

a.push('Mashrafi');
console.log(a);//["sakib", "tamim", "Mashrafi"]
console.log(b);//["sakib", "tamim", "Mashrafi"]
//Object & array er khatre ek rokom bihave reference type
//Number, string er khatre er ek rokom behave karon ai 2 ta 2 dohoren er data type er modda pore



//Reference value ekta mashter object theke ashe
//primitive value gula stack er moto structure a rakhe & Reference value gula Heap er moto structure a rakhe

//stack limited ekta jaiga niya kaj kore
```

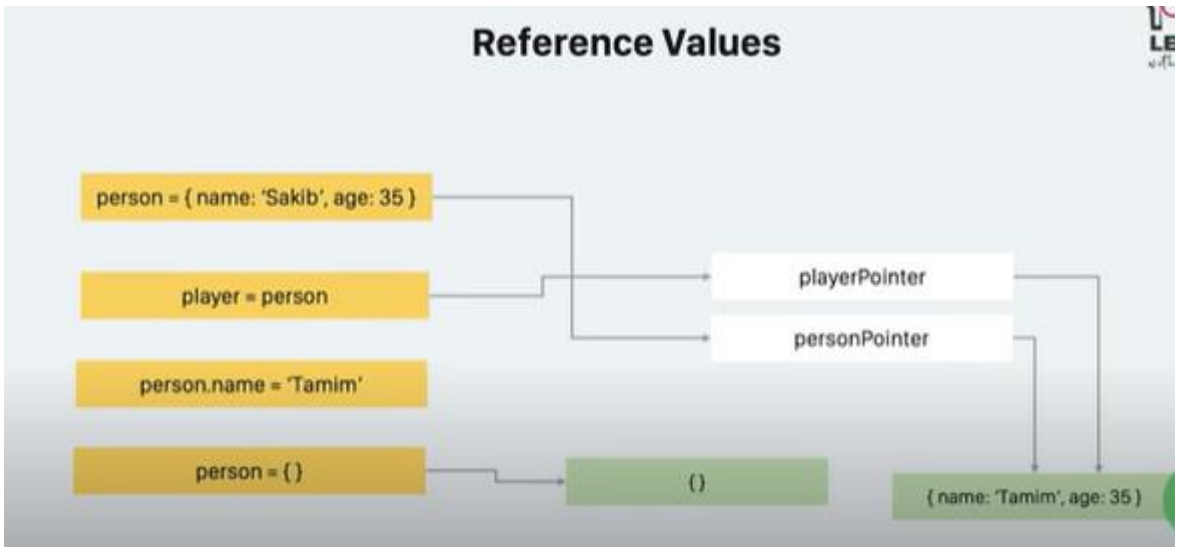


Primitive stack data structure a memory te data rakhe. Memory te ekta jaigai javascript value ta rakhese.

Title ai javascript value take point kore ase.

Name = title korle ja hoi….she title er javasript value take copy kore memory er stack er moto er ekta slot

a rakhe . So 2 ta value same but 2 tar jaiga difference.



Reference er khetre amra ekta object nilam person. So ai object take she heap er modda rakbe. Heap memory te ekta randon jaiga.

Person directly point korbe na name:sakib' age:24 she pointer k point kore rakhe. Player notun kore er ekta pointer toiri kore

```
person.name = 'Tamim'//mutation/update
Person = {}//Assignment....tai notun object create holo
Player = person; player Tamim e pass korse karon player pointer take e point kore chilo.
```

```
//mutation vs assignment
var a = ['sakib', 'Tamim'];
var b = a;//ai khane a pointer variable(aita value na) b pointer a chole jabe
a.push('Mushfique'); //change/mutation
a = []; //Assignment
console.log(a);//[]
console.log(b);//(3) ["sakib", "Tamim", "Mushfique"]
//Reference a assignment & mutation 2 ta difference



//copy reference immutably
var language = {
    name: "javascript",
    estd: "1995"
};

//immutably mane same jinish alada alada ekjon change hole o er ekjon change hobe na(like spread operator)
var language2 = {...language};//immutably
language.funder = "Brendan Iceh";//language.push korte parbo na karon aita to array na
console.log(language);//{name: "javascript", estd: "1995", funder: "Brendan Iceh"}
```

```
console.log(language2);//{name: "javascript", estd: "1995"}



var language = {
    name: "javascript",
    estd: "1995",
    libraries: ["react", "Vue"],//aita er ekta reference type aita k she diply copy korbe na
};


var language2 = {...language};//immutably
language.libraries.push = "JQuery";

console.log(language);//{name: "javascript", estd: "1995", libraries: Array(2)}

console.log(language2);//{name: "javascript", estd: "1995", libraries: Array(2)}
// reference type copy korer shomoi shotorko thakte hobe karon jokhon e copy hoi pointer ta copy hoi tai mutated korle
ektar shate er ekta o change hoiya jai

// //onek boro nested object hole lodash use korbo immutabely korer jonno
var language2 = _.cloneDeep(language);//cloneDeep lodash er ekta function // lodash librey html file link html a rakte
hobe then use krte hobe // tahole ami ekta poribotton korle o er ekta poribotton hobe na



//primitive wraper type
// String, number, Boolean ader object representation ase, there own representation
//primitive type er to prototype nai she rakbe kothai? So wraper theke niya ase wraper er kase excessable
var a = "sakib";//string er prototype nai
console.dir(a);
console.log(a.charAt(2); //wraper er jonno korte parsi
var b = new String("Sakib");//er prototype ase so b te onek buildin function user korte parbo ja a  er jonno parbo na
//wrapper type aita ignore kora e valo //aita o string aita ekta object o.
console.dir(b);
console.log(b.charAt(2));




//Primitives are not mutable
var a = "sakib";
a.test = "Tamim";
console.log(a);//output sakib e error ashbe na karon ignore kore chole gese



//pass by reference/value
//reference type copy korer shomoi shotorko thakte hobe karon jokhon e copy hoi pointer ta copy hoi tai mutated korle ek
tar shate er ekta o change hoiya jai so aita function er parameter er khatre ki hoi dakhi....

let a = 1;//a primitive type

//js a funcion er modda jokhon amra parameter pass korbo sheta premitive type hok er jatai hok always pass by reference
let change = (val) => {
    val = 2;
}

change(a);//output 1// dakhe mone hoi value pass kora hoiyase but passed by reference

console.log(a);
```

```
//pass by reference er prove
let a = {//ekta object nilam
    num: 1,
};


let change = (val) => {//object pathisi
    //aita ekta local variable create holo, ashse kintu reference hishabe e
    val ={};//output: 1 // val assignment hoiyase jar karone aita ekta notun object //aita ekta separeate scope
   //ait obj object
   //so reference hishebe gele change hower kotha chilo

    val.num = 2; //output: 2 karon mutated korlam //so pass by reference hoiyase

    }

    change(a);

    console.log(a);
```

Conculation:

1. Function er khatre value always pass by reference hobe
2. Primitive value k kokhono muted kora jai na, muted korle o ignore kore
3. Reference type gula always muted korle ek rokom, assignment korle er ek rokom kaj korbe
4. Perfectly/ Deply value copy korte chile lodash er library
5. Ek lever er array/obj k amra spread operator maddome copy korte pari