Contents

[**React introduction** 2](#_Toc101237532)

[**1.What is React** 2](#_Toc101237533)

[Basics needed to learn React 2](#_Toc101237534)

[2.Data Types 2](#_Toc101237535)

[3.ES5 version 4](#_Toc101237536)

[4.ES6 5](#_Toc101237537)

# **React introduction**

# **1.What is React**

**React** is a JavaScript library for building user interfaces.

### Basics needed to learn React

* HTML
* CSS
* Javascript(ES6)

**JavaScript fundamental concepts**

* Variable Declaration
* Data Types
* Arrays and Objects
* Functions
* Conditional statements
* Looping concept

**Variable Declaration**

var username="Mahendra";

console.log(username);

# 2.Data Types

4 different types of datatypes

* number
* string
* boolean
* undefined

var mynum=1;

console.log(typeof mynum);

var myStr='1';*//single code or double code*

console.log(typeof myStr);

var myBool=true; //never use uppercase like TRUE, always use lower true or false

console.log(typeof myBool);

var abc=undefined;

console.log(typeof abc);

Arrays

var myarr=[1,2,3,4,5];

console.log(myarr);

console.log(myarr[0]);

myarr.forEach((*e*)=>{

    console.log(*e*);

});

Object (JSON Object)

var myObject = [

    { color: "red", value: "#f00" },

    { color: "green", value: "#0f0" },

    { color: "blue", value: "#00f" },

    { color: "cyan", value: "#0ff" },

    { color: "magenta", value: "#f0f" },

    { color: "yellow", value: "#ff0" },

    { color: "black", value: "#000" }

];

console.log(myObject);

console.log(myObject[0]);

myObject.forEach((*e*) => {

    console.log(*e*.color + ':' + *e*.value);

});

Function (executes specific task)

function test(){

    console.log('Hello');

}

test();

function add(*num1*,*num2*){

    console.log(*num1*+*num2*);

}

add(1,2);

Conditional Statements

*// if if-else if-else if-else*

Loops

*//for loop*

*//while loop*

# 3.ES5 version

|  |  |  |
| --- | --- | --- |
| **Array Features** | **JSON Features** | **Date features** |
| isArray() | stringify() | valueOf() |
| map() |  |  |
| filter() |  |  |
| forEach() |  |  |
| every() |  |  |
| some() |  |  |
| indexOf() |  |  |
| lastIndexOf() |  |  |

**Array.isArray**

var myname='Mahendra';

var colors=['RED','GREEN'];

console.log(Array.isArray(myname));

console.log(Array.isArray(colors));

map()

we can loop and array using map at the same time we can even return the modified array.

With forEach we can simple loop, but can’t modify the array

var out=mynumarr.map((*e*)=>{ return *e*\*2});*//looping and modifying*

console.log(out);

mynumarr.map((*e*)=>console.log(*e*));*//simply looping without modifying*

filter()

var mynumarr1=[1,2,3,4];

var out2=mynumarr1.filter((*e*)=>{ return *e*%2===0});

console.log(out2);

forEach()

var mynumarr2=[1,2,3,4];

mynumarr2.forEach((*e*)=>console.log(*e*));

every() [if condition is satisfied for all the elements then it returns true else false]

var mynumarr3=[2,4,6,9];

var out3=mynumarr3.every((*e*)=>{return *e*%2===0});

console.log(out3);

some()

If at least one condition satisfies on all elements, then some will return true

var mynumarr4=[2,4,6,9];

var out4=mynumarr4.some((*e*)=>{return *e*%2===0});

console.log(out4);

indexOf and lastindexOf

var mynumarr5=[2,4,6,9,4];

console.log(mynumarr5.indexOf(4));*//gives first index*

console.log(mynumarr5.lastIndexOf(4));*//gives last index position of element*

stringify() : converts object to string

var myObject={name:"mahendra",age:28};

console.log(JSON.stringify(myObject));

var today= new Date();

console.log(today); *//2022-04-19T09:08:38.827Z*

console.log(today.valueOf());*//1650359341418 provides date in millis*

# 4.ES6/ECMAScript 2015

Let &const

Arrow function

Classes

For..of

Default Parameter

Rest Operator

Spread Operator

[Destructuring](https://www.google.com/search?rlz=1C1ONGR_enUS996US996&sxsrf=APq-WBtU2zSO5JR6vN1vEbGFfXieBK3gbQ:1650359675991&q=Destructuring&spell=1&sa=X&ved=2ahUKEwiXv_Tx5J_3AhWzHTQIHesNB18QkeECKAB6BAgBEDI)

**Let & const**

Var variable can be used outside block (like a global level declaration), but let and const can’t be.

for (var v = 1; v < 3; v++) {

    console.log('inside var:'+v);

}

console.log('outside var:'+v);

console.log('------------------------------');

for (let l = 1; l < 3; l++) {

    console.log(l);

}

console.log(l);

inside var:1

inside var:2

outside var:3

------------------------------

1

2

console.log(l);

            ^

ReferenceError: l is not defined

    at Object.<anonymous> (*C*:\learning\react\code\001.js basics\example003\_es6.js:11:13)

    at Module.\_compile (*internal*/*modules*/*cjs*/loader.js:1085:14)

    at Object.Module.\_extensions..js (*internal*/*modules*/*cjs*/loader.js:1114:10)

    at Module.load (*internal*/*modules*/*cjs*/loader.js:950:32)

    at Function.Module.\_load (*internal*/*modules*/*cjs*/loader.js:790:12)

    at Function.executeUserEntryPoint [as runMain] (*internal*/*modules*/run\_main.js:75:12)

    at internal/main/run\_main\_module.js:17:47

Const can’t be reassinable

const c=10;

c=c+1;

console.log(c);

c=c+1;

 ^

TypeError: Assignment to constant variable.

    at Object.<anonymous> (*C*:\learning\react\code\001.js basics\example003\_es6.js:14:2)

    at Module.\_compile (*internal*/*modules*/*cjs*/loader.js:1085:14)

    at Object.Module.\_extensions..js (*internal*/*modules*/*cjs*/loader.js:1114:10)

    at Module.load (*internal*/*modules*/*cjs*/loader.js:950:32)

    at Function.Module.\_load (*internal*/*modules*/*cjs*/loader.js:790:12)

    at Function.executeUserEntryPoint [as runMain] (*internal*/*modules*/run\_main.js:75:12)

    at internal/main/run\_main\_module.js:17:47

**Arrow function**

const user=()=>console.log('hello');

user();

const add=(*i*,*j*)=>{return *i*+*j*;};

console.log(add(7,6));

For single parameter, we nod need to use brackets (like i=> below)

const square=*i*=>{return *i*\**i*;};

console.log(square(7));

**Classes**

You have 2 use constructor to declare class variables.

class User {

    constructor(*firstName*,*lastName*,*dob*){

**this**.firstName=*firstName*;

**this**.lastName=*lastName*;

**this**.dob=*dob*;

    }

    printHello(){

        console.log('Hello');

    }

}

let u= new User();

u.firstName='John';

u.lastName='Manda';

console.log(u);

console.log(JSON.stringify(u));

u.printHello();

Below can be only used in typescript.

class User {

    firstName: string,

    lastName: string,

    dob?: string;

}

For of and for in

For of gives value of array, but for in gives index of element

const myarray =[10,11,12,13,14,15];

for(const a of myarray){

    console.log(a);

}

for(const a in myarray){

    console.log(a);

}

Default parameter: See below the second parameter is set with default value true.If user wont pass any value then the default value is used

function printMe(*name*,*aggreeedTerms*=true){

    console.log(*name*+':'+*aggreeedTerms*);

}

printMe('John');

printMe('John',false);

Rest Operator

Rest Operator and Spread operator both have 3 dots, but little differences

REST Operator (rest =rest of values or remaining operators)

function printMe(*param1*,*param2*,...*remaining*){

    console.log(*param1*);

    console.log(*param2*);

    console.log(*remaining*);

}

console.log(printMe(1,2,3,4,5));

output

1

2

[3,4,5]

function addAll(...*i*){

    return *i*.reduce((*a*,*b*)=>*a*+*b*,0);

}

console.log(addAll(1,2,3,4,5));

Spread Operator : In below we copied iArr and jArr into kArr

let iArr=[1,2,3,4];

let jArr=[5,6,7,8];

let kArr=[...iArr,...jArr];

kArr.forEach((*e*)=>console.log(*e*));

Destructuring (storing the values of an object into separate properties)

let person={

    firstName:"Mahendra",

    lastName:"Pappu"

}

const {firstName,lastName} =person;

console.log(firstName+':'+lastName);

*//array Destructuring*

let numbers=[1,2,3,4]

const [i,j,k] =numbers;

console.log(i+':'+j+':'+k);