

1. ECMAScript 6 also known as ES6 is a significant update to the JavaScript language that introduced many new features aimed at simplifying coding practices, improving readability, and enhancing performance.

Key features include:

* **Block-scoped declarations**: Introduction of ‘let’ and ‘const’ for defining variables with block scope.
* **Arrow functions**: A shorter syntax for writing function expressions and resolving the ‘this’ context automatically.
* **Template literals**: Simplified string interpolation using backticks (`).
* **Default parameters**: Functions can now have default values for parameters.
* **Destructuring**: Allows unpacking values from arrays or properties from objects into distinct variables.

1. The let keyword is used to declare variables that are scoped to the block in which they are defined. Unlike var, which has a function-level scope, variables declared with let are accessible only within the block or statement where they are created. This helps avoid common issues related to variable hoisting and unintended global declarations. Additionally, let prevents re-declaration of the same variable within the same scope, improving code reliability and clarity.

|  |  |
| --- | --- |
| **var** | **let** |
| It is function-scoped. | It is block-scoped. |
| It is hoisted and initialized as undefined. | It is hoisted but not initialized. |
| Redeclaration is allowed within the same scope. | Redeclaration is not allowed within the same scope. |
| It creates a property on the global object. | It does not creates a property on the global object. |

1. The const keyword in JavaScript is used to declare variables whose identifiers are meant to be constant and immutable. Once a variable is declared using const, it must be assigned an initial value, and its reference cannot be changed throughout the program. However, it is important to note that const does not make the content immutable if the variable is an object or an array—it only prevents reassignment of the reference. Like let, const is block-scoped and does not support re-declaration in the same block.
2. ES6 introduced the class syntax to simplify the creation of objects and the implementation of inheritance in JavaScript. A class is essentially a blueprint for creating objects with shared properties and methods.

A class can include:

* A constructor method for initializing object state.
* Instance methods defined inside the class body.
* Static methods using the static keyword, accessible without creating an object.

Example:

class Player {

constructor(name) {

this.name = name;

}

display() {

console.log(`Player: ${this.name}`);

}

}

1. ES6 allows one class to inherit the features of another using the extends keyword. The derived class (also known as the child class) inherits the properties and methods of the base class (or parent class). Within the derived class, the super() function is used to call the constructor of the parent class.

Example:

class Player {

constructor(name) {

this.name = name;

}

}

class Batsman extends Player {

constructor(name, runs) {

super(name); // Call to parent constructor

this.runs = runs;

}

}

1. Arrow functions are a concise way to write function expressions in ES6. They use a new syntax (=>) and differ from traditional functions in several important ways. Most notably, arrow functions do not have their own this context and instead inherit it from the parent scope, making them particularly useful in callbacks and event handlers.

Example: const greet = name => `Hello, ${name}`;

1. Set: A Set is a built-in JavaScript object introduced in ES6 that allows storing unique values of any type. Duplicate values are automatically ignored.

Example: const numbers = new Set([1, 2, 3, 2]); // {1, 2, 3}

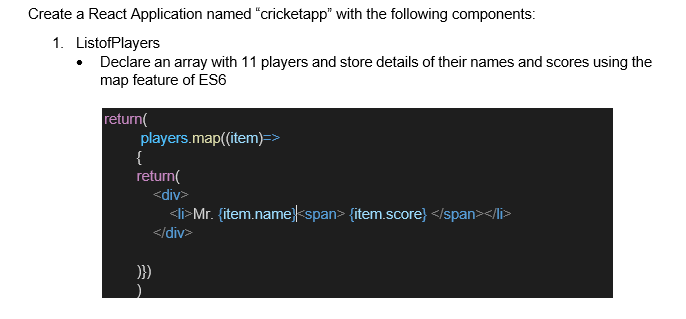
Map: A Map is a collection of key-value pairs where keys can be of any type. It maintains the insertion order of the keys and provides efficient lookup.

Example:

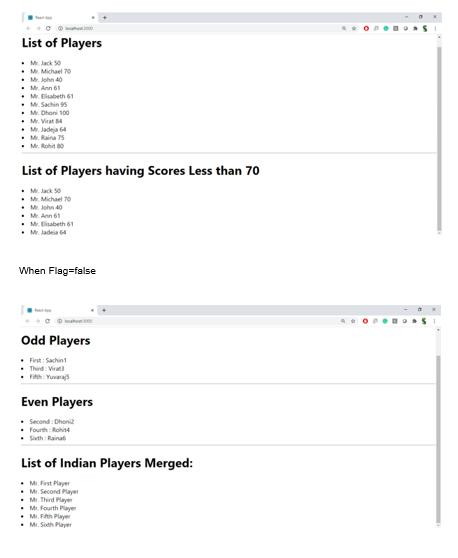
const details = new Map();

details.set('name', 'Rahul');

details.set('score', 95);









**Solution**

**ListofPlayers.js**

import React from 'react';

const ListofPlayers = ({ players }) => {

return (

<div>

<ul>

{players.map((item) => (

<li>Mr. {item.name}<span> {item.score} </span></li>

))}

</ul>

</div>

);

};

export default ListofPlayers;

**Scorebelow70.js**

import React from 'react';

const Scorebelow70 = ({ players }) => {

const players70 = players.filter(item => item.score < 70);

return (

<div>

<ul>

{players70.map((item) => (

<li>Mr. {item.name}<span> {item.score} </span></li>

))}

</ul>

</div>

);

};

export default Scorebelow70;

**OddPlayers.js**

import React from 'react';

export function OddPlayers({ team }) {

return(

<div>

<li> First : {team[0]} </li>

<li> Third : {team[2]}</li>

<li> Fifth : {team[4]}</li>

</div>

)

}

export default OddPlayers;

**EvenPlayers.js**

import React from 'react';

export function EvenPlayers({ team }) {

return(

<div>

<li> Second : {team[1]} </li>

<li> Fourth : {team[3]}</li>

<li> Sixth : {team[5]}</li>

</div>

)

}

export default EvenPlayers;

**ListofIndianPlayers.js**

import React from 'react';

const ListofIndianPlayers = ({ players }) => {

return (

<div>

<ul>

{players.map((item, index) => (

<li key={index}>Mr. {item}</li>

))}

</ul>

</div>

);

};

export default ListofIndianPlayers;

**App.js**

import React from 'react';

import './App.css';

import ListofPlayers from './ListofPlayers';

import Scorebelow70 from './Scorebelow70';

import { OddPlayers } from './OddPlayers';

import { EvenPlayers } from './EvenPlayers';

import ListofIndianPlayers from './ListofIndianPlayers';

const players = [

{ name: 'Yasashvi Jaiswal', score: 50 },

{ name: 'Abhishek Sharma', score: 70 },

{ name: 'Sanju Samson', score: 40 },

{ name: 'Hardik Pandya', score: 65 },

{ name: 'Surya Kumar Yadav', score: 83 },

{ name: 'Akshar Patel', score: 95 },

{ name: 'Kuldeep Yadav', score: 90 },

{ name: 'Jaspreet Bumrah', score: 84 },

{ name: 'Arsdhdeep Singh', score: 91 },

{ name: 'Ravi Bishnoi', score: 75 },

{ name: 'Shubhman Gill', score: 80 },

];

const T20players=['First Player','Second Player','Third Player'];

const RanjiTrophyPlayers=['Fourth Player','Fifth Player','Sixth Player'];

const IndianPlayers=[...T20players, ...RanjiTrophyPlayers];

const IndianTeam = ['Sachin Tandulkar', 'Virat Kohli', 'Mahendra Singh Dhoni', 'Shikhar Dhawan', 'Yuvraj Singh', 'Harbajan Singh'];

function App() {

var flag=true;

if(flag===true){

return(

<div>

<h1> List of Players: </h1>

<ListofPlayers players={players}/>

<h1> List of Players having Scores Less than 70 </h1>

<Scorebelow70 players={players}/>

</div>

)

}

else{

return(

<div>

<h1> Odd Players </h1>

<OddPlayers team={IndianTeam}/>

<hr/>

<h1> Even Players </h1>

<EvenPlayers team={IndianTeam}/>

<hr/>

<h1> List of Indian Players Merged: </h1>

<ListofIndianPlayers players={IndianPlayers}/>

</div>

)

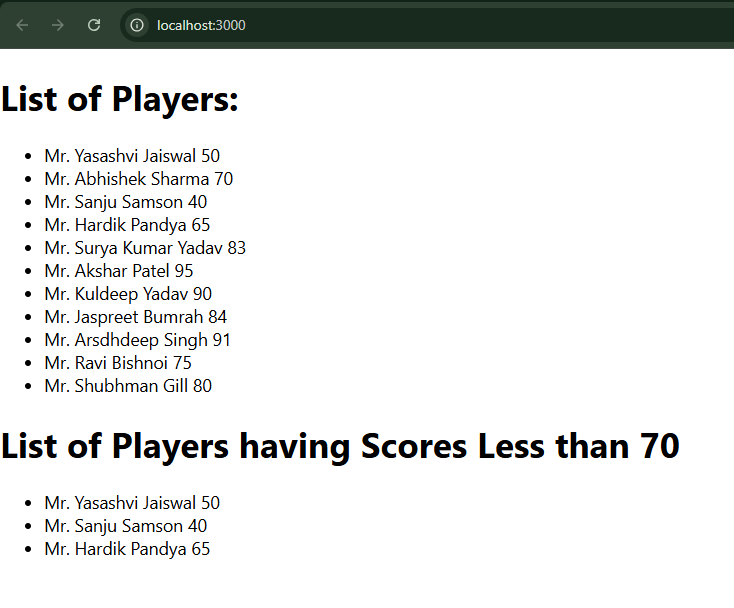
}

}

export default App;

**Output**

When flag=true



When flag=false

