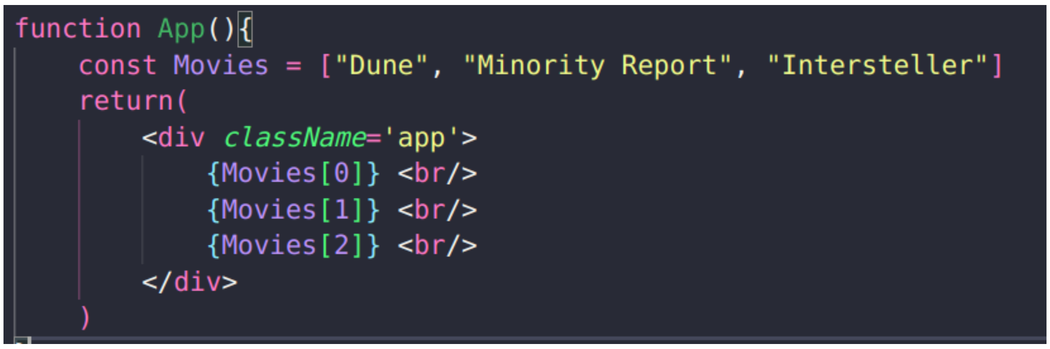
List in React and map function

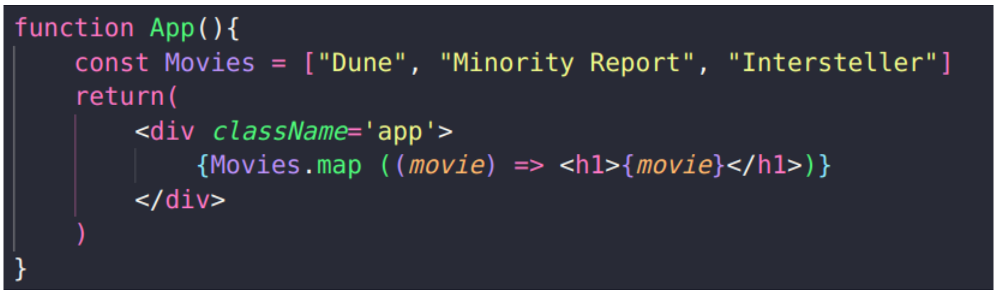
=> you can render the elements of the list one by one



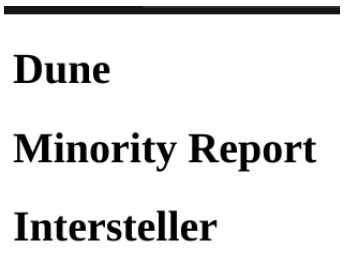
```
function App(){
    const Movies = ["Dune", "Minority Report", "Intersteller"]
    return(
        <div className='app'>
            {Movies[0]} <br/>
            {Movies[1]} <br/>
            {Movies[2]} <br/>
        </div>
    )
```

-- but the problem is that what if numbers are in 100's or ...
-- so we need function :-- map

-- map function returns the list/ array



```
function App(){
    const Movies = ["Dune", "Minority Report", "Intersteller"]
    return(
        <div className='app'>
            {Movies.map ((movie) => <h1>{movie}</h1>)}
        </div>
    )
}
```

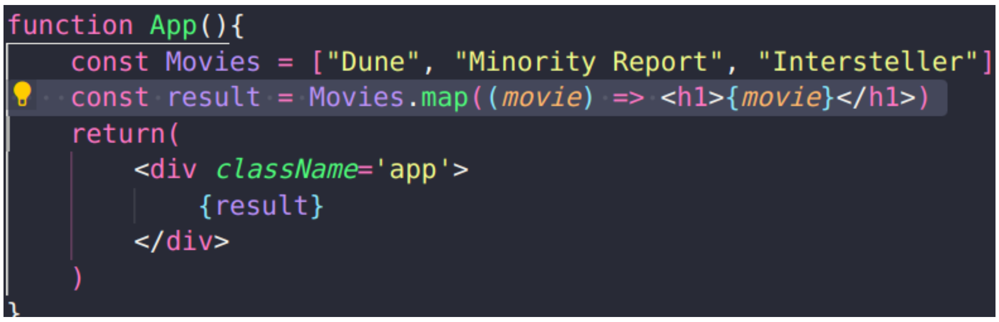


Dune

Minority Report

Intersteller

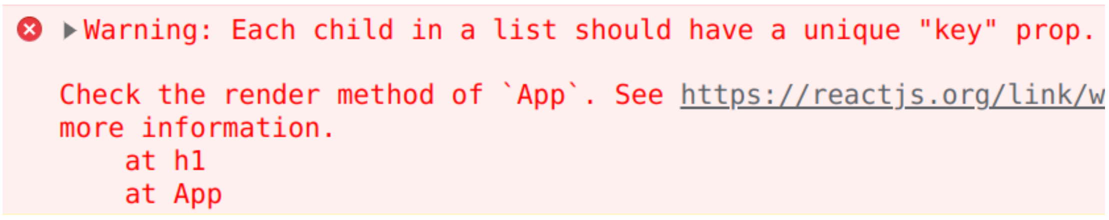
-- or you can put outside of the jsx as



```
function App(){
    const Movies = ["Dune", "Minority Report", "Intersteller"]
    const result = Movies.map((movie) => <h1>{movie}</h1>)
    return(
        <div className='app'>
            {result}
        </div>
    )
}
```

-- and here it doesn't need to be wrapped inside the curley braces because it is outside of the JSX .
-- and it works in the same way so you can do in any way

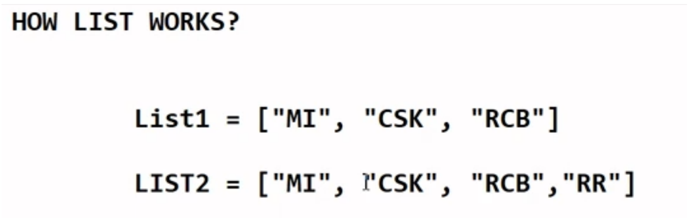
=> and it was how you will render the list element but there is a problem of



```
▶Warning: Each child in a list should have a unique "key" prop.

 Check the render method of `App`. See https://reactjs.org/link/w
 more information.
     at h1
     at App
```

-- to identify each child uniquely it must be associated with some key.

=> Keys :
“”” in JS we know that when some change happens then entire DOM tree re-renders / crested again. but in react because of the virtual DOM concept it renders only the modified/manipulated part. “””

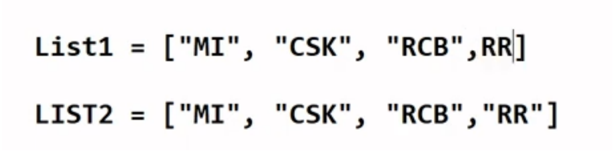


```
HOW LIST WORKS?

        List1 = ["MI", "CSK", "RCB"]

        LIST2 = ["MI", "CSK", "RCB","RR"]
```

-- suppose we have two lists and
we want to update our list1 with list2.

-- so you need to add RR

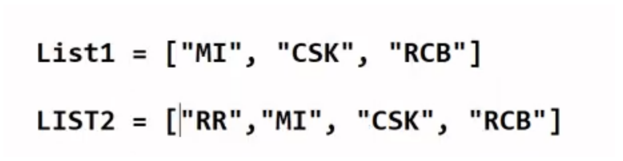
=> how it will work ??
MI -matched with ----> MI okay
CSK -matched with ----> CSK
similarly RCB with RCB



```
List1 = ["MI", "CSK", "RCB",RR]

LIST2 = ["MI", "CSK", "RCB","RR"]
```

-- but RR is not in lsit1 but in list2 so we will update it with this value

=> but assume if list is like this in the order :--

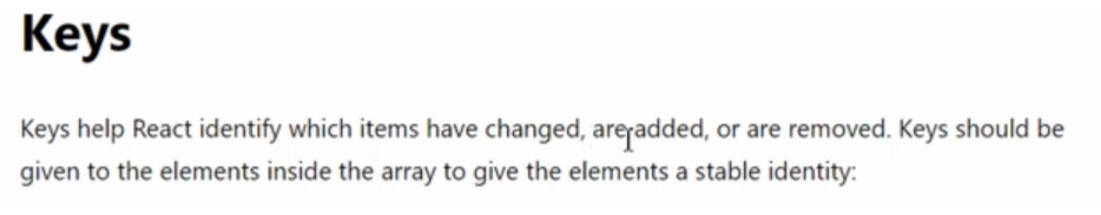


```
List1 = ["MI", "CSK", "RCB"]

LIST2 = ["RR","MI", "CSK", "RCB"]
```

then MI != RR
CSK != MI
RCB != CSK

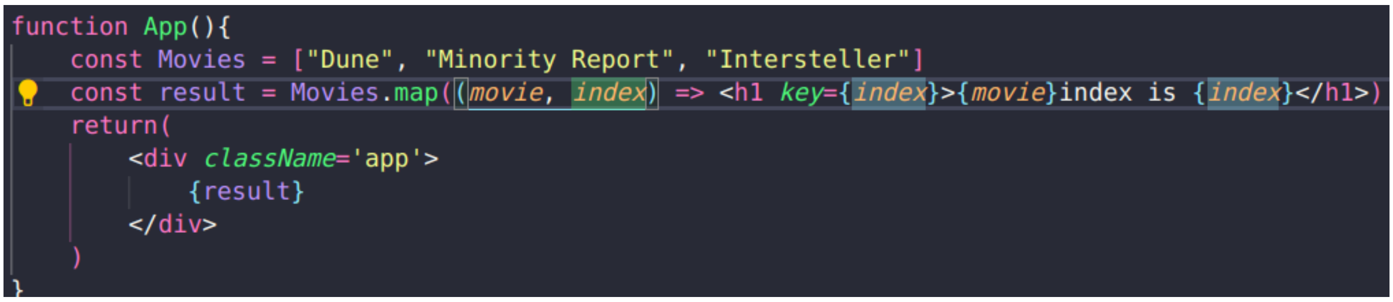
-- and it will think nothing is matching so it will update the whole list1 by creating the all the values and will delete existing values
---> list1 = [“RR”, “MI”, “CSK”, “RCB]

-- while we just needed to add the RR only but due to ordering whole list is recreated which is violating the virtual dom and react concept .



Keys

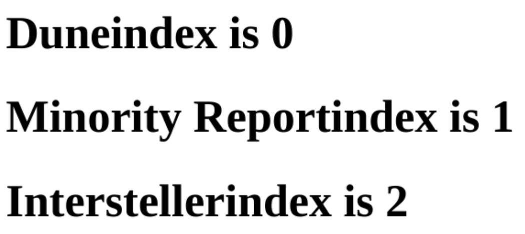
Keys help React identify which items have changed, are added, or are removed. Keys should be given to the elements inside the array to give the elements a stable identity:



```
function App(){
    const Movies = ["Dune", "Minority Report", "Intersteller"]
    const result = Movies.map((movie, index) => <h1 key={index}>{movie}index is {index}</h1>)
    return(
        <div className='app'>
            {result}
        </div>
    )
}
```

-- since we know that index is unique so we can use it as a key.
=> html(h1) have property key .

-- and warning is removed.



Duneindex is 0

Minority Reportindex is 1

Interstellerindex is 2

- **React Virtual DOM**: It's a lightweight representation of actual DOM, stored in memory and is never rendered.
- **Reconciliation in React**: The process of syncing Virtual DOM with the real DOM.
- **Diffing Algorithm**: The algorithm to find the minimum number of steps needed to update the real DOM.
- **Assumptions for using the Diffing Algorithm**:
  - Two elements of different types will produce different trees.
  - The developer can hint at which child elements may be stable across different renders with a `key` attribute.

-- why we should not use index as keys :-

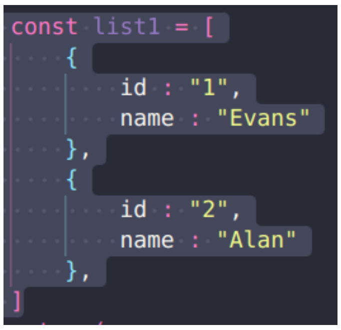
- Performance Issues due to unnecessary re-renders.- Issues in data mapping in case list items are sorted, filtered, or deleted.

=> so when we shuuld/can use index as key : -

-- if some unique elements are not present in the list / array .

=> here already we have the id and which can be used as

the key



```
const list1 = [
    {
        id : "1",
        name : "Evans"
    },
    {
        id : "2",
        name : "Alan"
    },
]
```