**# Group Activity**

> Work in group to solve these tasks.

1. What is the difference between the following statements:

```js

import {getImageUrl} from './utils.js';

import getImageUrl from './utils.js';

```

2. Is there anything wrong with the following component? Explain.

```js

const App = () => {

    return

    (

        <ul>

            <li>Katherine Johnson: mathematician</li>

            <li>Mario Pasquel: chemist</li>

            <li>Lavon Julian: chemist</li>

        </ul>

    )

  };

```

3.

- Give an example how [reduce()]((https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Array/Reduce)) works.

- calculate the sum of exercises in the following array using the reduce method.

```js

const parts = [

      {

        name: "Fundamentals of React",

        exercises: 10,

      },

      {

        name: "Using props to pass data",

        exercises: 7,

      },

      {

        name: "State of a component",

        exercises: 14,

      },

    ]

```

4.

- Give an example how [map()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Array/map) works.

- Difference between map() and [foreach()](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global\_Objects/Array/forEach)

- Why do we need key attribute when mapping? Refer to the reading material below:

---

One of the more tricky things with React is the requirement of a `key` prop when you attempt to render a list of elements.

If we want to render a list like this, then there's no problem:

```jsx

const ui = (

  <ul>

    <li>One</li>

    <li>Two</li>

    <li>Three</li>

  </ul>

)

```

But rendering an array of elements is very common:

```jsx

const list = ['One', 'Two', 'Three']

const ui = (

  <ul>

    {list.map(listItem => (

      <li>{listItem}</li>

    ))}

  </ul>

)

```

Those will generate the same HTML, but what it actually does is slightly different. Let's re-write it to see that difference:

```jsx

const list = ['One', 'Two', 'Three']

const listUI = list.map(listItem => <li>{listItem}</li>)

// notice that listUI is an array

const ui = <ul>{listUI}</ul>

```

So we're interpolating an array of renderable elements. This is totally acceptable, but it has interesting implications for when things change over time.

If you re-render that list with an added item, React doesn't really know whether you added an item in the middle, beginning, or end. And the same goes for when you remove an item (it doesn't know whether that happened in the middle, beginning, or end either).

**## ref**

- [Workshop](https://github.com/kentcdodds/react-fundamentals)

1. What is the difference between the following statements:

```js

import {getImageUrl} from './utils.js';

import getImageUrl from './utils.js';

So, if you wish to import the complete library (or its default export) you would use

import x from 'lib';

2. Is there anything wrong with the following component? Explain.

```js

const App = () => {

    return

    (

        <ul>

            <li>Katherine Johnson: mathematician</li>

            <li>Mario Pasquel: chemist</li>

            <li>Lavon Julian: chemist</li>

        </ul>

    )

  };

Attempted import error: './App' does not contain a default export (imported as 'App').

WARNING in [eslint]

src\App.js

Line 998:7: 'App' is assigned a value but never used no-unused-vars

Line 1000:3: Unreachable code no-unreachable

ERROR in ./src/index.js 71:82-85

export 'default' (imported as 'App') was not found in './App' (module has no exports)

webpack compiled with 1 error and 1 warning

ADDED  export default App

Also an error

src\App.js

Line 1005:5: Unreachable code no-unreachable

Search for the keywords to learn more about each warning.

To ignore, add // eslint-disable-next-line to the line before.

WARNING in [eslint]

src\App.js

Line 1005:5: Unreachable code no-unreachable

webpack compiled with 1 warning

ANSWER:

const App = () => {

    return

    (

should be

const App = () => {

    return (

this works

  const App = () => {

    return (

        <ul>

            <li>Katherine Johnson: mathematician</li>

            <li>Mario Pasquel: chemist</li>

            <li>Lavon Julian: chemist</li>

        </ul>

    )

  };

  export default App

3.

- Give an example how [reduce()]((<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/Reduce)>)

works.

- calculate the sum of exercises in the following array using the reduce method.

```js

const parts = [

      {

        name: "Fundamentals of React",

        exercises: 10,

      },

      {

        name: "Using props to pass data",

        exercises: 7,

      },

      {

        name: "State of a component",

        exercises: 14,

      },

    ]

ANSwer:

const array1 = [1, 2, 3, 4];

// 0 + 1 + 2 + 3 + 4

const initialValue = 0;

const sumWithInitial = array1.reduce(

(previousValue, currentValue) => previousValue + currentValue,

initialValue

);

console.log(sumWithInitial);

// expected output: 10

import React from 'react'

const Part = ({ part }) => {

  return (

    <p>{part.name} {part.exercises}</p>

  )

}

export default Part;

import React from 'react'

import Part from './Part'

const Content = ({ parts }) => {

  const rows = () => parts.map(part =>

    <Part

      key={part.id}

      part={part}

    />

  )

  const sum = () => parts.reduce((acc, part) =>

  acc + part.exercises, 0)

  // var sum = 0

  // for (var i = 0; i < parts.length; i++){

  //   sum = sum + parts[i].exercises

  // }

  // seuraavassa ota pois sulut sum()

  return (

    <>

      {rows()}

      <p><b>total of {sum()} exercises</b></p>

    </>

  )

}

export default Content;

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

4.

- Give an example how [map()](<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/map>)

works.

- Difference between map() and [foreach()](<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/forEach>)

- Why do we need key attribute when mapping? Refer to the reading material below:

---

One of the more tricky things with React is the requirement of a `key` prop when you attempt to render a list of elements.

If we want to render a list like this, then there's no problem:

```jsx

const ui = (

  <ul>

    <li>One</li>

    <li>Two</li>

    <li>Three</li>

  </ul>

)

```

But rendering an array of elements is very common:

```jsx

const list = ['One', 'Two', 'Three']

const ui = (

  <ul>

    {list.map(listItem => (

      <li>{listItem}</li>

    ))}

  </ul>

)

```

Those will generate the same HTML, but what it actually does is slightly different. Let's re-write it to see that difference:

```jsx

const list = ['One', 'Two', 'Three']

const listUI = list.map(listItem => <li>{listItem}</li>)

// notice that listUI is an array

const ui = <ul>{listUI}</ul>

```

So we're interpolating an array of renderable elements. This is totally acceptable, but it has interesting implications for when things change over time.

If you re-render that list with an added item, React doesn't really know whether you added an item in the middle, beginning, or end. And the same goes for when you remove an item (it doesn't know whether that happened in the middle, beginning, or end either).

**## ref**

- [Workshop](<https://github.com/kentcdodds/react-fundamentals>)

MAP

const array1 = [1, 4, 9, 16];

// pass a function to map

const map1 = array1.map(x => x \* 2);

console.log(map1);

// expected output: Array [2, 8, 18, 32]

FOREACH

const array1 = ['a', 'b', 'c'];

array1.forEach(element => console.log(element));

// expected output: "a"

// expected output: "b"

// expected output: "c"