# ReactJS Learning -

# **Components, JSX, props**

Self notes, Key concepts to remember

1. index.js renders root component App.js on the root html body element.

2. App.js imports all components and combines them.

3. components are simply javascript functions returning HTML element.

4. components return only one HTML element.

5. components need to be exported and imported to be used.

6. DOM manipulation by javascript is imperative approach, component system in react is declarative.

7. JSX (javascript XML) is special javascript syntax to insert HTML in javascript.

8. we use ( ) parenthesis and wrapping  to insert multiple lines of HTML in component.

9. we create css file next to Component.js and import to style HTML.

10. css file doesn't need export default at the end, but needs to be imported.

11. component names start with uppercase letter.

12. custom HTML element name inside JSX starts with uppercase letter.

13. components are inserted in JSX as custom HTML elements, like- <Component></Component>.

14. they can also be written as self closing- <Component/>.

15. for css styling, 'className' is used instead of 'class' attribute, as JSX is simply javascript and class in javascript is a reserved word.

16. we can write any javascript (functions, variable decl.) inside Component.js.
16. but we must use { } curly braces to write javascript inside JSX.

17. props is used to make components dynamic.

18. props is needed to use one component's data in other components, to input data from outside.

19. props is inserted as attribute to custom HTML components.

20. props is simply an object structure of data.

21. we usually use { } curly braces to insert dynamic data into props.

22. we set props data to child components in parent component, and get access to the data in child Component.js files.

23. props object is passed as parameter to child component functions to get access to props data. Example:

in App.js (setting props):

```
<Expense
    title = {expenses[0].title}
    name = {expenses[0].name}>
    </Expense>
```

in Expense.js (getting data):

```
function Expense(props) {
  return <h1>props.name</h1>
}
```

here props object:

```
props = {
  title: expenses[0].title,
  name: expenses[0].name
}
```

24. building components from smaller components and passing data through props is called ‘composition’.

25. composition concept has an interesting feature- shell/wrapper component.

26. wrapper component wraps other components in place of  wrapper to reduce code duplication.

27. usually Card.js with Card.css is the wrapper component.

28. Card.js receives its className ‘card’ containing repeated styles.

29. to make the wrapper work, {props.children} data is passed between s of card component.

30. children is a common props all components of react get by default.

31. {props.children} = wrapped components.

32. <Card ...> wrapper gets another className from replaced <div ...> wrapper along with its own ‘card’ className.

33. ‘className’ attribute is a css class to normal HTML elements, but to custom HTML component it becomes a props.

34. to make it all work, we add props.className to ‘card’ class.

example:

```
const classes = 'card' + props.className;
function Card(props) {
  return <div className={classes}>
    {props.children}
  </div>;
}
```

35. we write import React from ‘react’; at the start of every Component.js file containing some JSX codes, because JSX has return React.createElement( ) statement inside it.