**React has three principles.**

-Don’t touch the DOM

-Your first duty as a react programmer is to decide what is going be a component.

-Your second duty as a react programmer is to decide where the state is going to live.

-Your third duty is to determine what is going to change when the state change.

**React is** a library that behaves as a framework that let us build spa, managing the part of the V in MVC. With react we can create webpages that has various components that we can build, as we deem necessary, to create a full webpage or app.

In order to use react we have **to import react and component from the react dependency**, we can use pure component too, by doing so we can have a component that handles by himself the “should component update”.

**React components have two parts** regarding data management: **state and properties (props)**. **State is** the internal data of the component and it is local to it, meaning that sibling components do not know what the state of the others siblings is. **The properties of the component** are the data that is passed to them when they are called, like the parameters of a function.

**React has a dataflow like a cascade**; it can just go down, passed from parents to children in a lineal manner. Props is the data that is pass down from a parent to a child.

**React components can be a class or a functional component**, meaning that they can be declare as a class or as a function (even an arrow function).

A functional component has props, but does not have lifecycle methods or state like a class component.

**A lifecycle method** are methods that react has inside its components that can manage the build of the component along its use, from render to unmount.

**State should not be changed directly**, but if we need to change the state, we can use setState. setState is a function that provides a way to change state without touching it directly.

**React has controlled components**; a **controlled component is** a component that has its value set by using the state of react, **an uncontrolled** one has his value handled by itself.

**React use what is called jsx**, this means javascript extended, it is a superset of javascript that has some built in functionalities and sintaxis that help us use html inside javascript.

**Jsx use camel case** to name its properties, some words like **class change to className** because it is a reserved word in js. Once we are inside a render method, the place where we write our html, we have to use brackets in order to use javascript.

**React needs to return an element** inside de render method in order to work.

We can change dynamically the classes and properties of the elements, we can iterate over them and do many things that only the power of js can offer and can’t be done with html.

**We can just return one element in react,** if we want to render multiple elements we need **to wrap it inside a container** or use the **fragment** from the react library.

**React use what is call keys**, which are ids that identify elements inside a list and are necessary so react can differentiate between them.

An key has only to be called directly on the children of the father that will contain the list.

**For routing react uses react-router-dom**, that has a bunch of tools for routing.

**React-router-dom has access to the history api**, that the browsers provide and has data concerning to the location of the page and what has happened in the pages.

**For define a navigation block we need to use Router**, we can use browserlist that don’t need further configuration before calling it from the react-router-dom

**Inside this router we can define a switch that is a list of routes**, **and inside it we can call our actual routes or links**. The routes take a exact property that only renders the element it its path property match exacly that of the url and the component property that define which component is going to be renderer once this url gets accessed

**If we want to make a element be always present** we can just put it outside the switch and inside the router and we can then freely use link and etc inside this elements.