We use components to tell React what we want to see on the screen

In particular, render returns a **React element**, which is a lightweight description of what to render.

JSX comes with the full power of JavaScript. You can put any JavaScript expressions within braces inside JSX.

To “remember” things, components use **state**.

All React component classes that have a constructor should start with a super(props) call.

---

From the documentation:

API : Apllication Programming Interface. It helps the request and services are in order and correctly delivers the jobs.

While working with react, we can use both function components and class components.

Tags in HTML called “Core Components” in react native.

<div> (unscrollable) = <Viev>

<div> (scrollable) = <ScrollView)>

<p> = <Text>

<img> = <Image>

<input type= “text> = <TextInput>

Whatever a function returns is rendered as “react element”.

Javascript expressions will work between curly braces{};

Creating functions: const(optional) funcname = (parameters) => {

Return ..

}

Assume we have created a function called cat, later in the code we can easily call its return value by <cat/>

State: usage: const [getter,setter] = useState(true or 0 or a string whatever);

Here an example: const[isHungry, setIsHungry] = useState(true);

Use state creates a isHungry state variable and sets its value true. Also it creates a state function to change its value which is setIsHungry;

Adjacent jsx elements always must be wrapped in an enclosing tag. So sometimes we do it like this:

const Cafe = () => {

return (

<> (JSX encloser tags)

<Cat name="Munkustrap" />

<Cat name="Spot" />

</>

);

};

You can create your own styles in this form:

const styles = StyleSheet.create({

  container: {

   flex: 1,

   paddingTop: 22

  },

  item: {

    padding: 10,

    fontSize: 18,

    height: 44,

  },

});

In react, css codes are usually written in camel casing. E.g: backgroundColor instead of background-color.

In react flex-direction is by default column. In css it was row.