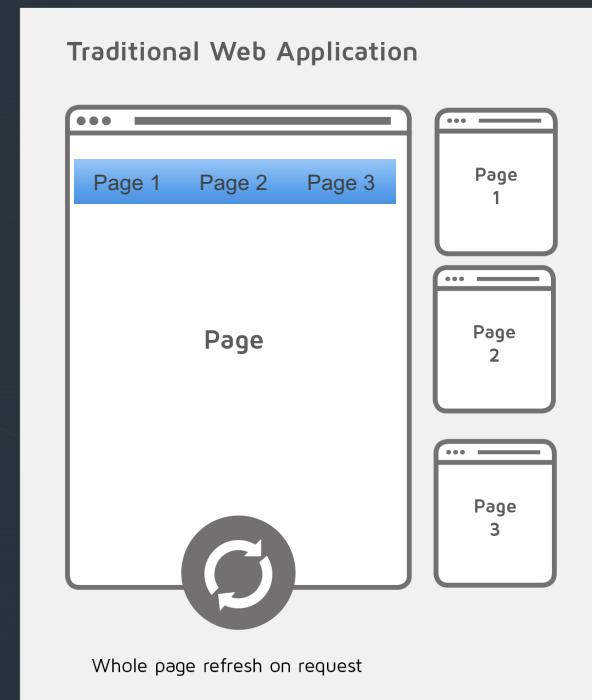
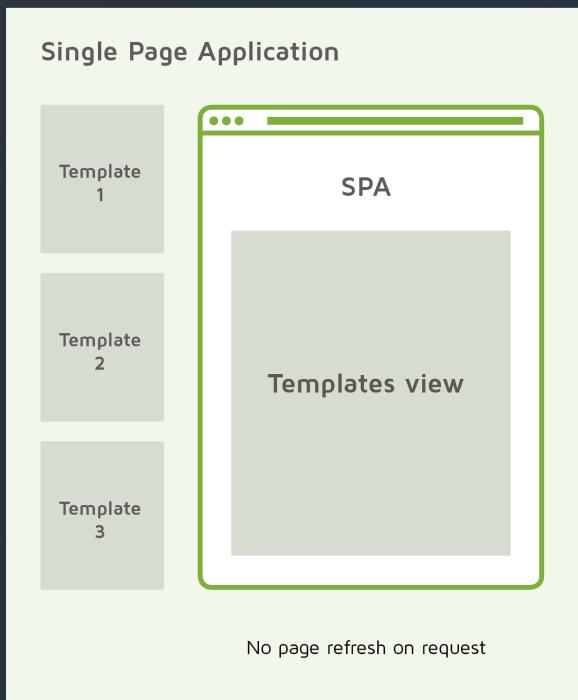


# Single Page Application

# What is a single page application?

Single page application is a kind of web apps, that dynamically generate the content of the page instead of loading different web pages. Therefore we don't need any page reload in this type of applications. (e.g Gmail, Facebook, ...)



# Pros of Single-Page Application

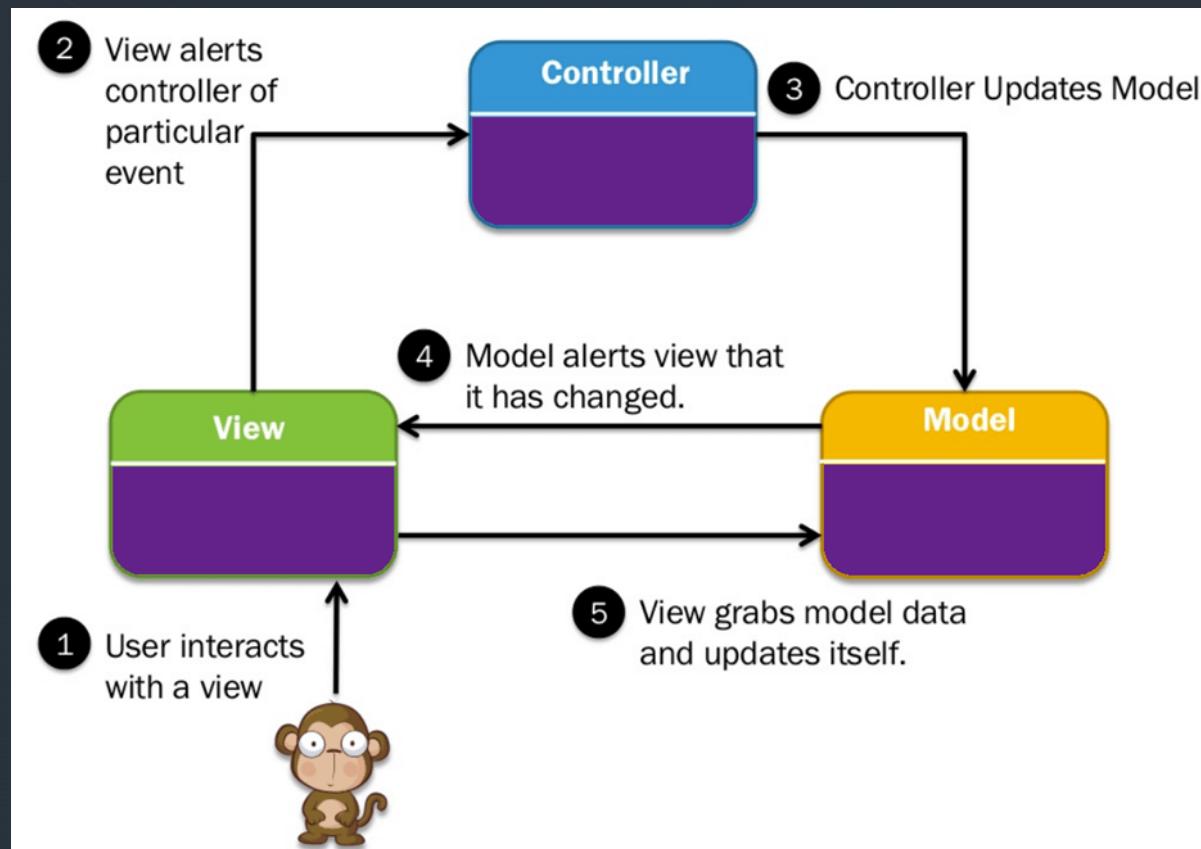
- SPA is fast because most of page content will be loaded once and only data will be transmitted back and forth.
- The development is simplified, because no need to write code to render page on the server.
- Easy to debug, because chrome web development tools let you monitor the network operations.
- Easier to develop mobile applications, because same backend can be used for web and mobile app.
- SPA can cache any local storage effectively, therefore can be used even offline.

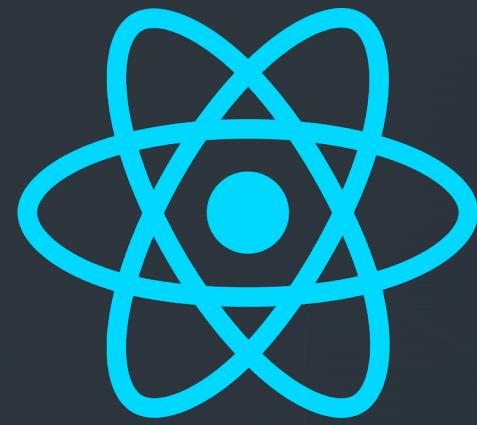
# Cons of Single-Page Application

- It is very tricky and not easy to implement
- It is slow to download because of heavy client framework need to be loaded to the client.
- It requires Javascript to be present and enabled.

# Model-View-Controller (MVC)

- It's a software architecture pattern for implementing user interfaces and split the application into three interconnected parts.

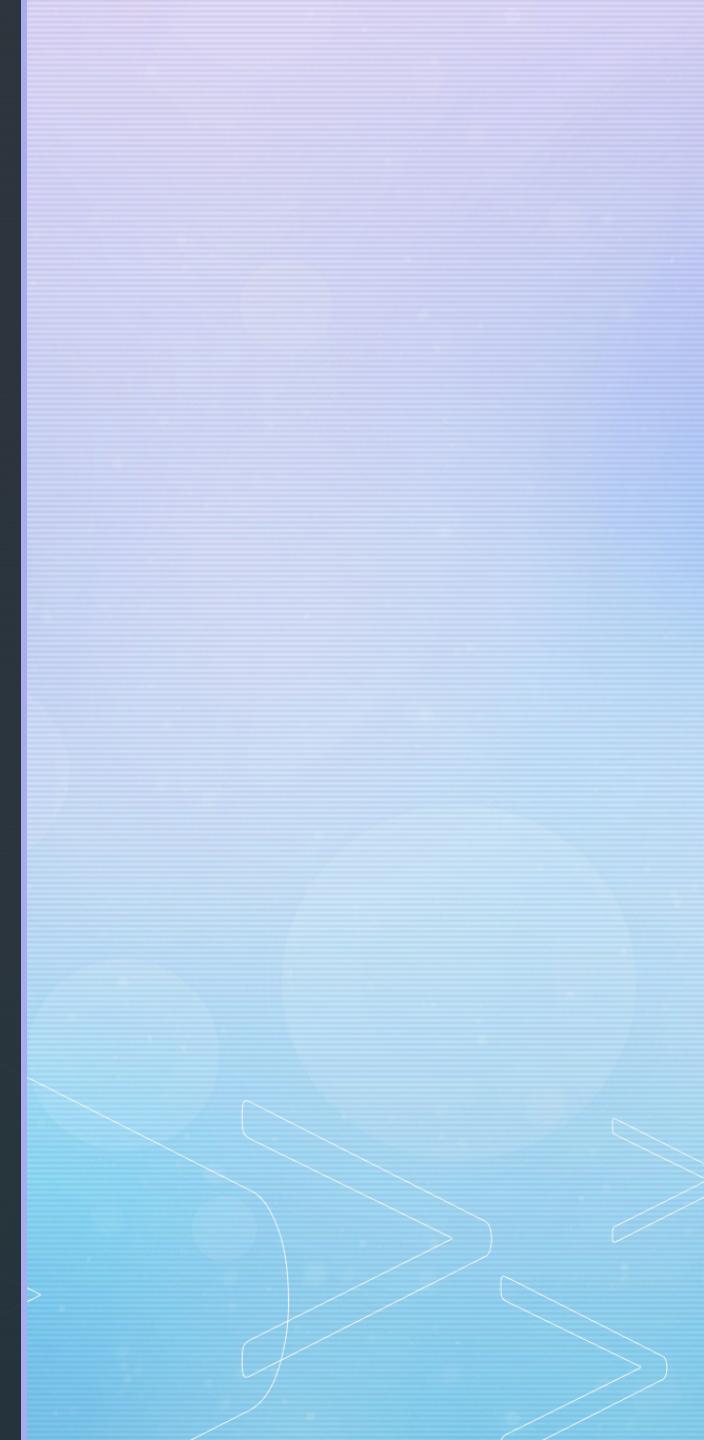




# React

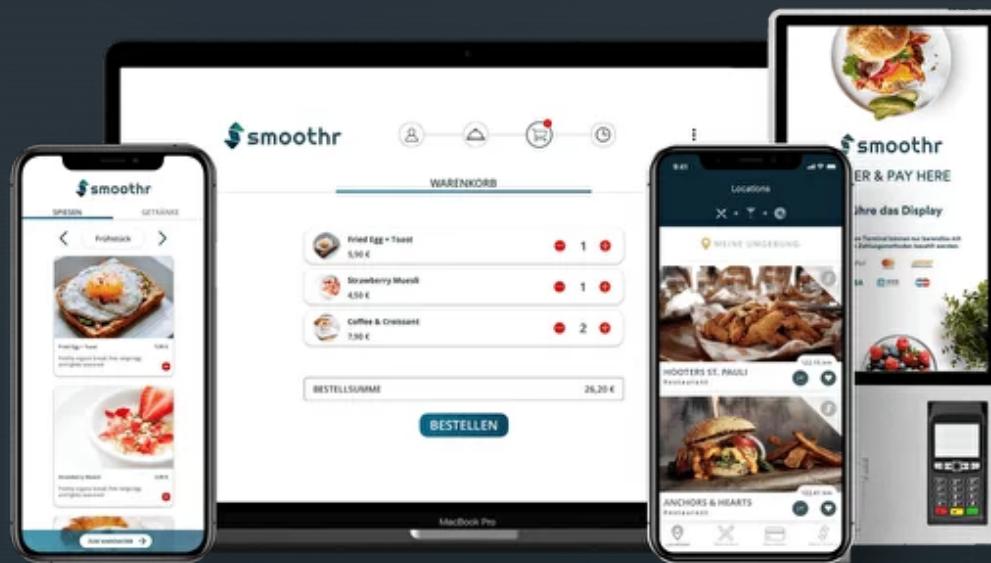


## ReactJS Introduction



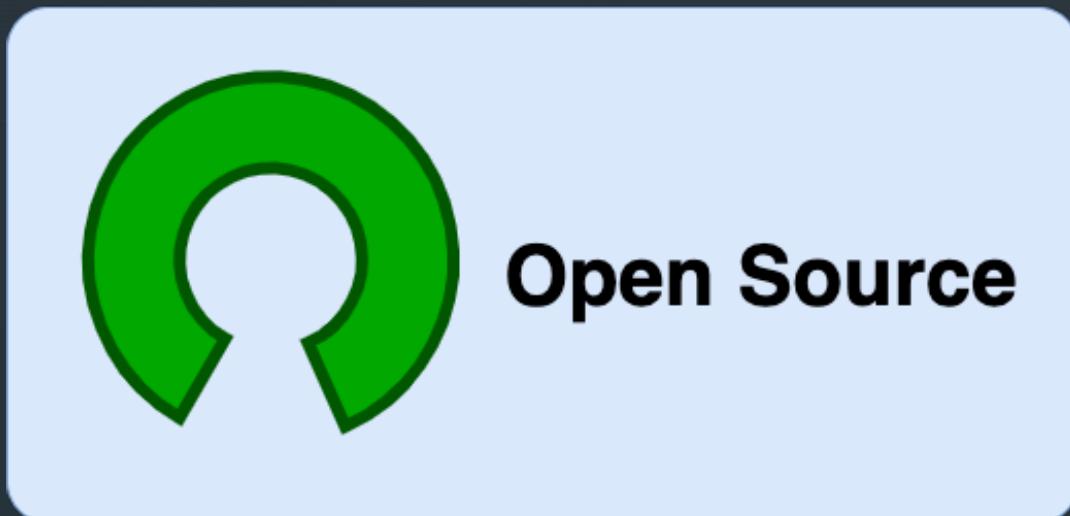
# What is React?

- React is a Javascript library for building fast and interactive user interfaces for the web as well as mobile applications



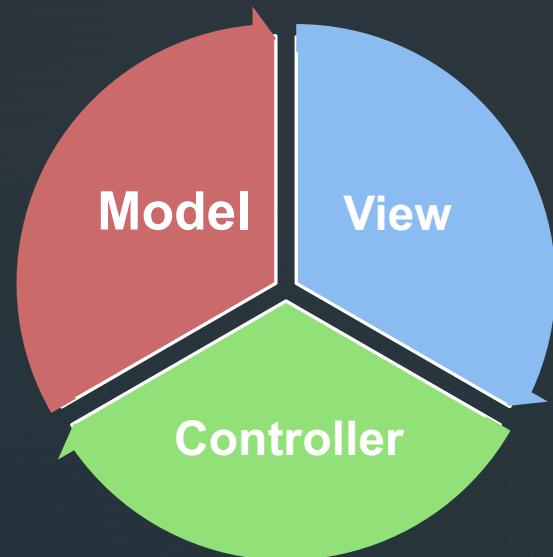
# What is React?

- React is a Javascript library for building fast and interactive user interfaces for the web as well as mobile applications
- It is an open-source reusable component-based front-end library



# What is React?

- React is a Javascript library for building fast and interactive user interfaces for the web as well as mobile applications
- It is an open-source reusable component-based front-end library
- In a Model View Controller architecture, React is the View which is responsible for how the app looks and feels.



# What is React?



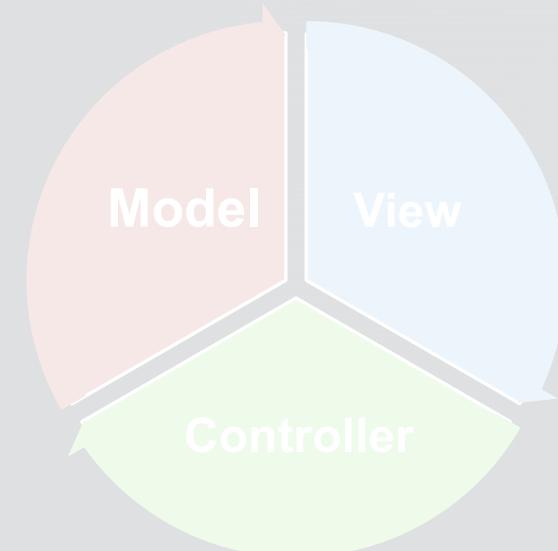
## Did You Know?

React was created by **Jordan Walker**, who was a software engineer at Facebook

- React is used for building fast and interactive user interfaces for web applications

It is an open-source reusable component-based front-end library

In a Model View Controller architecture, React is the View which is responsible for how the app looks and feels.



# What is React?

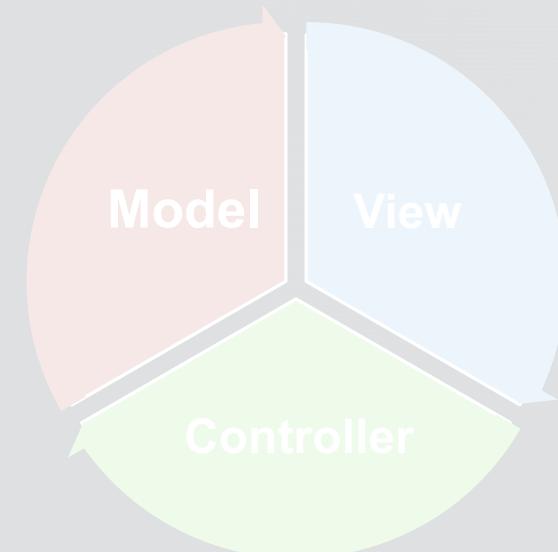
Let's see how React works  
in real-time

- React is a library for building fast and interactive user interfaces for the web.



React is an open-source reusable component-based front-end library.

In a Model View Controller architecture, React is the View which is responsible for how the app looks and feels.





Take an **example** of an **Instagram** webpage which is entirely built using React

A screenshot of an Instagram profile for a user named "jordan". The profile picture shows a close-up of a camera lens. The bio reads: "Jordan A. Graphic designer and photographer www.defectsdesigns.com". Below the bio are several circular category icons: "Landscape", "Cali", "City", "Plants", "Portraits", "NYC", and "Chicago". A large arrow points from the character's speech bubble towards the Instagram profile.

Instagram profile for **jordan**

Follow ...

1,048 posts 13.5k followers 22 following

Jordan A. Graphic designer and photographer  
www.defectsdesigns.com

Landscape Cali City Plants Portraits NYC Chicago >

4,938 192



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

Instagram

Search

jordan

Follow

1,048 posts    13.5k followers    22 following

Jordan A. Graphic designer and photographer  
[www.defectsdesigns.com](http://www.defectsdesigns.com)

Landscape    Cali    City    Plants    Portraits    NYC    Chicago

4,938 likes    192 comments



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

Search Component

The image shows a mockup of an Instagram profile page for a user named "jordan". The profile picture is a circular image of a person holding a Nikon camera. The bio reads: "Jordan A. Graphic designer and photographer www.defectsdesigns.com". Below the bio are several circular thumbnail images representing different photo categories: Landscape, Cali, City, Plants, Portraits, NYC, and Chicago. A right-pointing arrow indicates more categories are available. At the bottom, three full-size photos are displayed: a woman in a hat, a woman in a garden, and a woman sitting on a bench. Each photo has a heart icon and a number indicating likes (4,938, 192). The top navigation bar includes the Instagram logo, a search bar with the placeholder "Search", and icons for notifications, likes, and profile.

Instagram

jordan

Follow

1,048 posts   13.5k followers   22 following

Jordan A. Graphic designer and photographer  
www.defectsdesigns.com

Landscape   Cali   City   Plants   Portraits   NYC   Chicago

4,938   192



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

Search Component

The image shows a mockup of the Instagram 2018 web profile. At the top left is the Instagram logo. Below it is a search bar with the placeholder "Search". To the right of the search bar are three icons: a person, a heart, and a camera. A red box highlights the search bar, and a blue arrow points from the text "Search Component" to it. In the center, there's a profile card for a user named "jordan". The card includes a profile picture of a camera, the name "jordan", a "Follow" button, a dropdown menu, and three dots. It also displays statistics: "1,048 posts", "13.5k followers", and "22 following". Below the stats is a bio: "Jordan A. Graphic designer and photographer" followed by a website link "www.defectsdesigns.com". A red box highlights the entire profile card area, and a blue arrow points from the text "Profile Description Component" to it. Below the profile card is a grid of seven circular thumbnails with labels: "Landscape", "Cali", "City", "Plants", "Portraits", "NYC", and "Chicago". At the bottom, there are three large image cards: a woman in a hat, a woman in a field, and a woman sitting on the ground. The image card for the woman sitting on the ground has a heart icon with the number "4,938" and a circle with the number "192".

Profile Description Component



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

The image shows a mockup of the Instagram 2018 web profile. At the top left is the Instagram logo. To its right is a search bar with the placeholder "Search". Above the search bar is a blue callout labeled "Search Component" with an arrow pointing to it. Below the search bar is a user profile for "jordan". The profile includes a circular profile picture of a camera, the username "jordan", a "Follow" button, a dropdown menu, and three dots. It also shows statistics: "1,048 posts", "13.5k followers", and "22 following". A bio reads "Jordan A. Graphic designer and photographer" followed by a website link "www.defectsdesigns.com". This entire section is highlighted with a red rounded rectangle and has a blue callout labeled "Profile Description Component" with an arrow pointing to it. Below the profile is a horizontal row of seven circular thumbnails with labels: "Landscape", "Cali", "City", "Plants", "Portraits", "NYC", and "Chicago". A right-pointing arrow is positioned next to the "Chicago" thumbnail. This row is highlighted with a red rounded rectangle and has a blue callout labeled "Stories Component" with an arrow pointing to it. At the bottom of the page are three large, vertically stacked image cards. The first card shows a woman in a hat. The second card shows a woman in a field. The third card shows a woman sitting on the ground with a heart icon and the number "4,938" and a circle icon with the number "192" overlaid.

Search Component

Profile Description Component

Stories Component



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

The image shows a mockup of the Instagram 2018 web profile interface. Several components are highlighted with red boxes and labeled with blue text:

- Search Component**: Points to the search bar at the top center.
- Profile Description Component**: Points to the user profile card for "jordan".

jordan Follow ...

1,048 posts 13.5k followers 22 following

Jordan A. Graphic designer and photographer  
www.defectsdesigns.com
- Stories Component**: Points to the grid of thumbnail images representing different story categories.
- Single Post Component**: Points to a single post featuring a woman wearing a hat.



# Instagram 2018 Mockup

FREE WEB PROFILE PSD

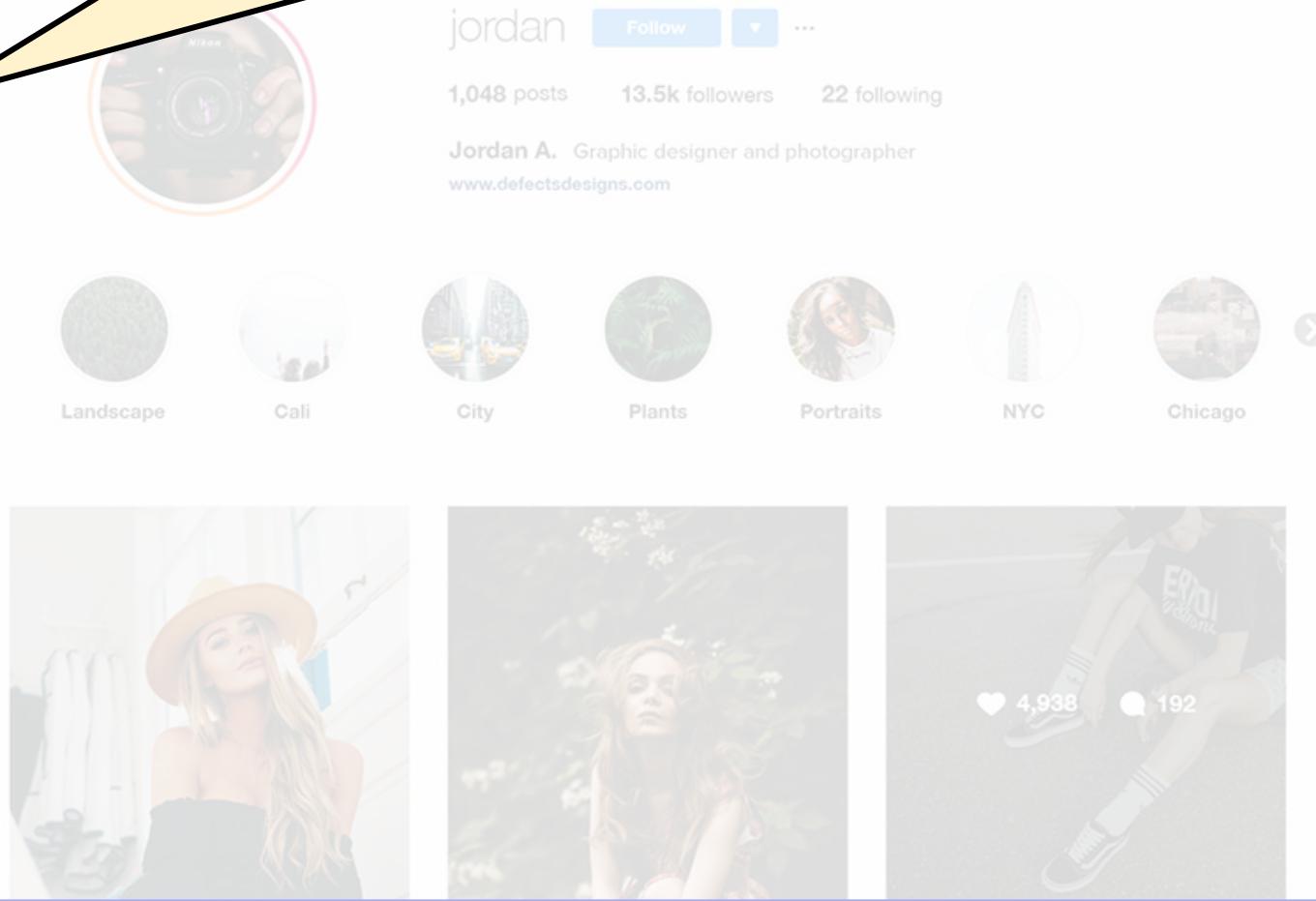
The image shows a mockup of the Instagram 2018 web profile interface. Several components are highlighted with red boxes and labeled with blue text:

- Search Component**: Points to the search bar at the top center.
- Profile Description Component**: Points to the user profile section for "jordan".

jordan [Follow](#) [...](#)  
1,048 posts 13.5k followers 22 following  
Jordan A. Graphic designer and photographer  
[www.defectsdesigns.com](http://www.defectsdesigns.com)
- Stories Component**: Points to the grid of thumbnail images representing different story categories.
- Single Post Component**: Points to a detailed view of a single post featuring a woman wearing a hat.
- Post List Component**: Points to a horizontal list of three posts.



React divides the UI into multiple components, which makes the code easier to debug, and each component has its own property and function



# Why React?



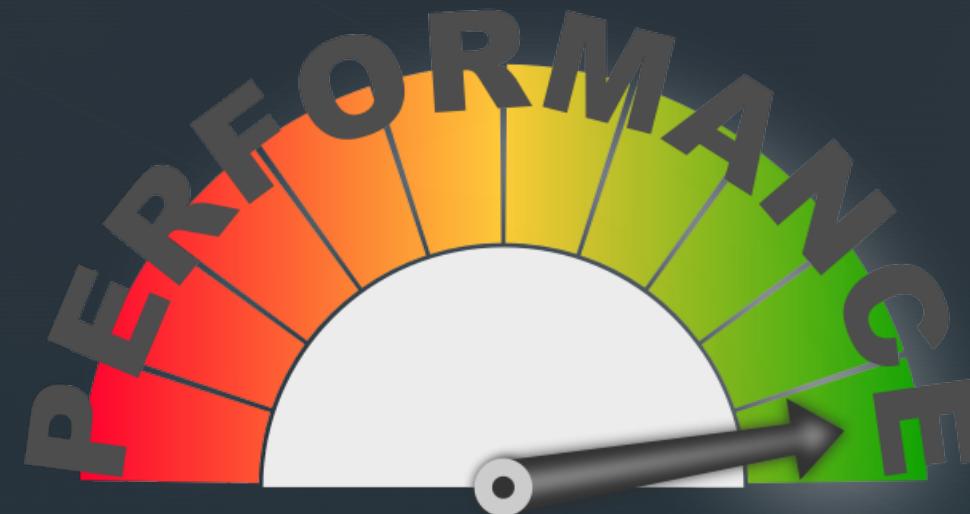
# Why React?

- Easy creation of dynamic web applications



# Why React?

- Easy creation of dynamic web applications
- Performance enhancement



# Why React?

- Easy creation of dynamic web applications
- Performance enhancement
- Reusable components



# Why React?

- Easy creation of dynamic web applications
- Performance enhancement
- Reusable components
- Can be used for mobile apps

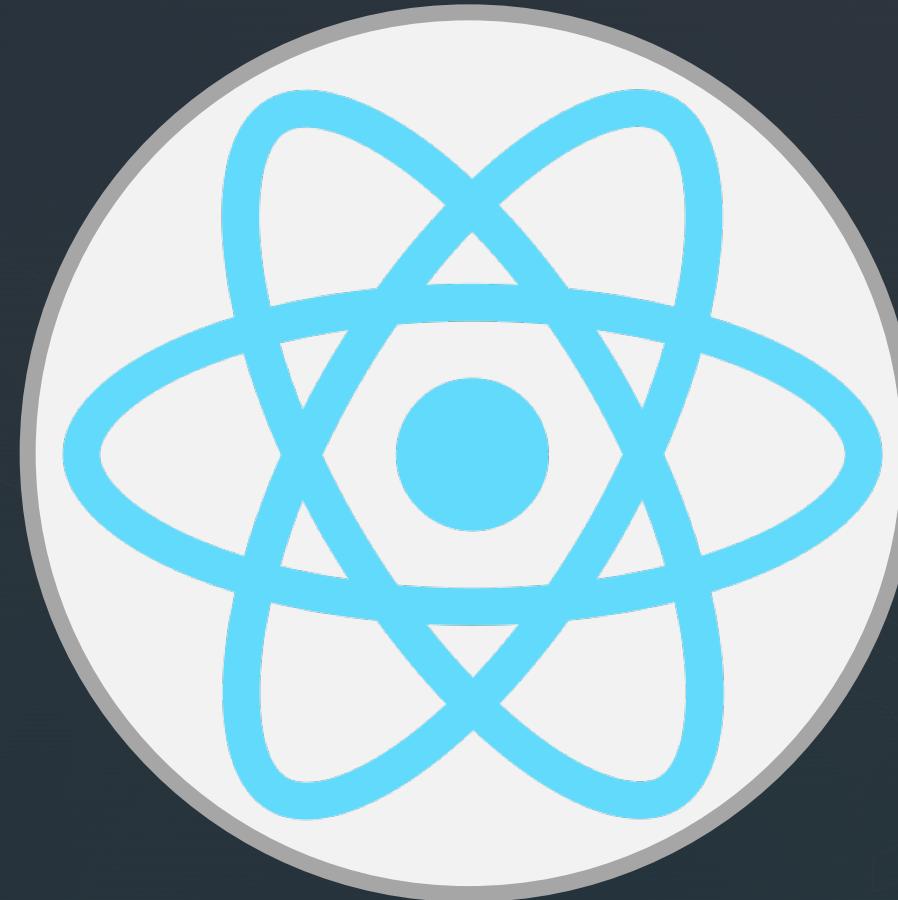


# Why React?

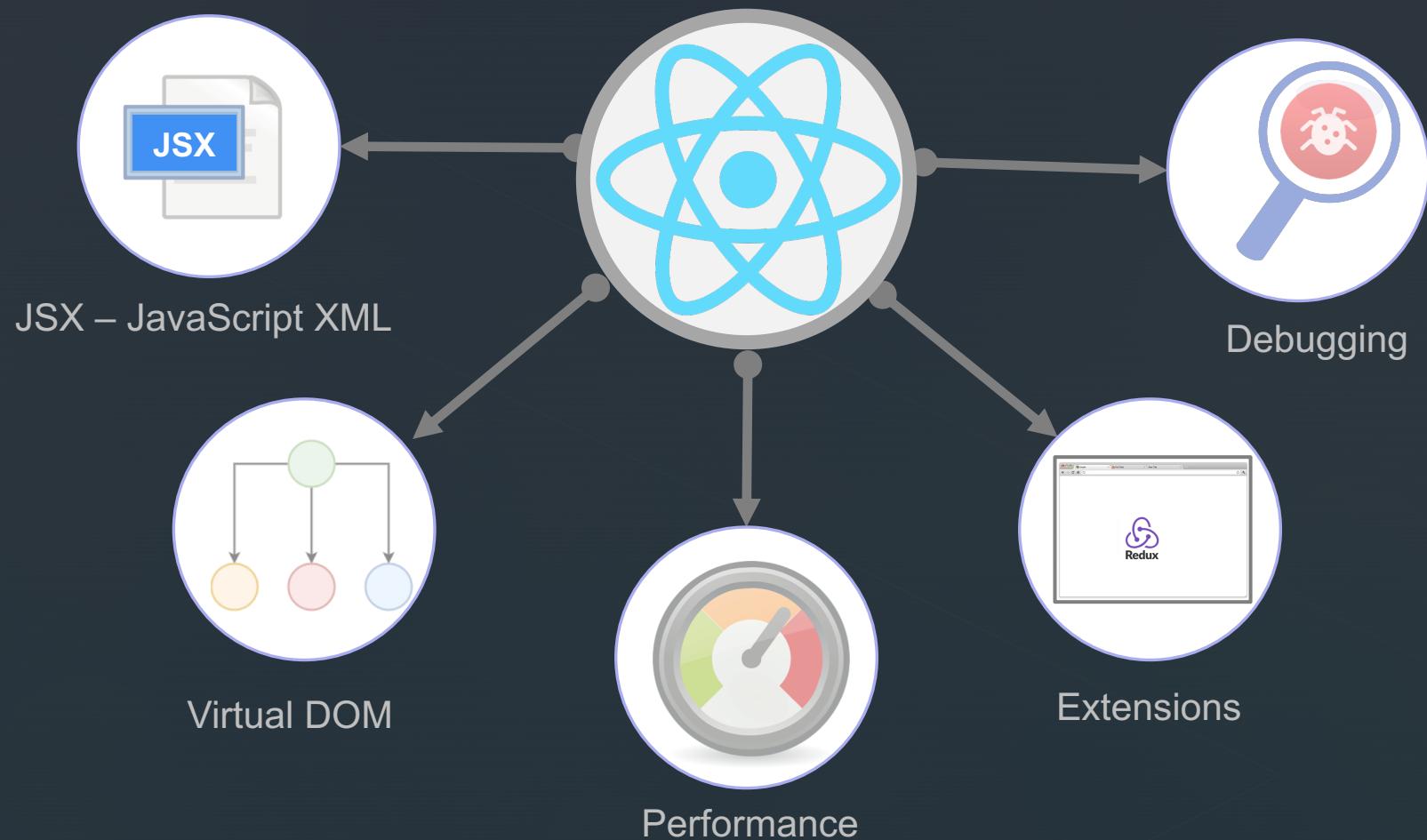
- Easy creation of dynamic web applications
- Performance enhancement
- Reusable components
- Can be used for mobile apps
- Dedicated tools for debugging



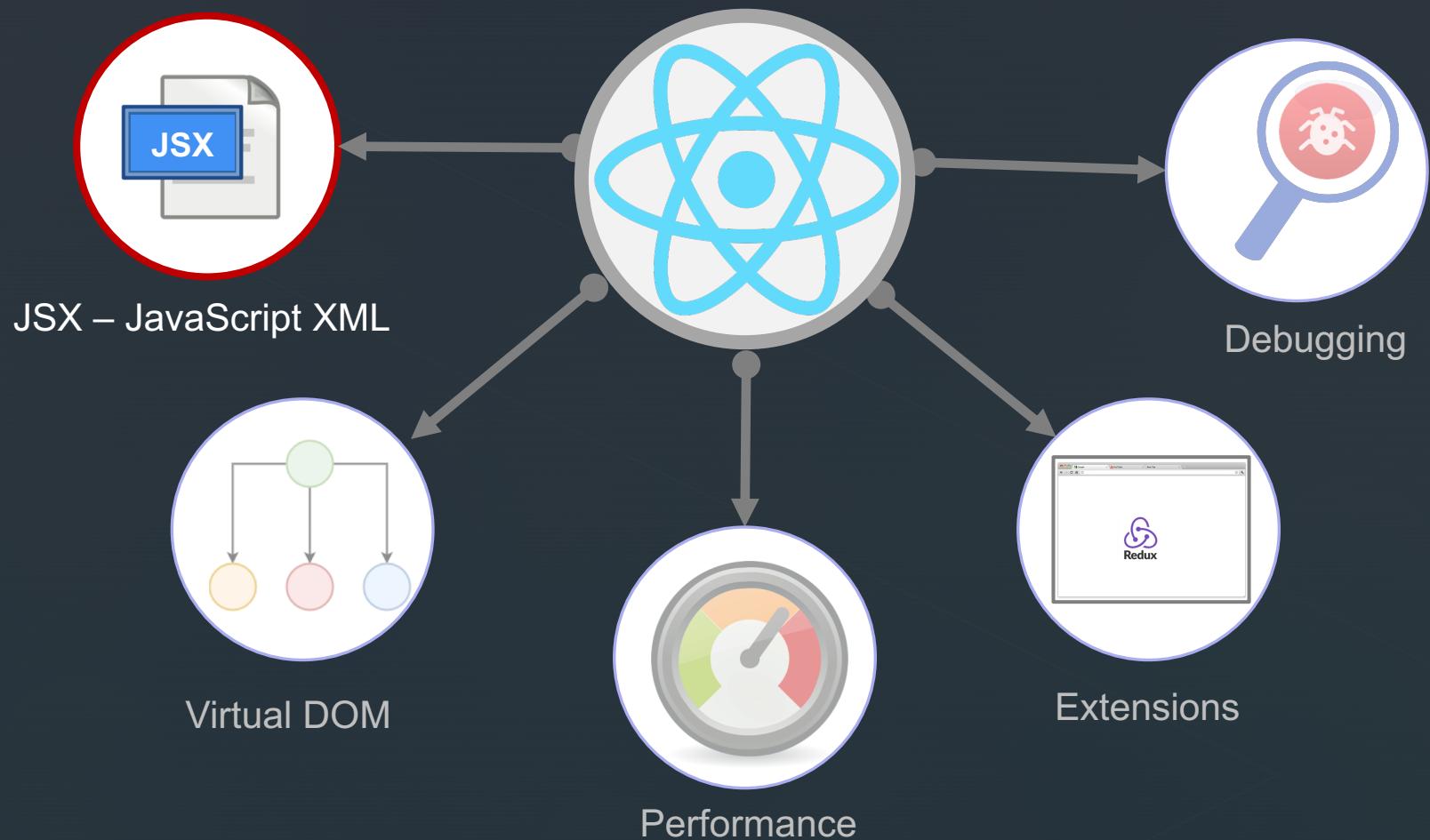
# Features of React



# Features of React



# Features of React



# What is JSX?

- JSX is a syntax extension to JavaScript. It is used with React to describe what the User Interface should look like.
- By Using JSX, you can write HTML structure in the same file that contains JavaScript code.
- JSX helps in making the code easier to understand and debug as it avoids usage of JS DOM structures which are rather complex.

**JS + HTML = JSX**



# What is JSX?



You got it! JSX is basically a combination of Javascript and HTML

- JSX is used in the User Interface and look like.

By Using JSX, you can write HTML structure in the same file that contains JavaScript code.

JSX helps in making the code easier to understand and debug as it avoids usage of JS DOM structures which are rather complex.

**JS + HTML = JSX**



Let's look at this basic  
JSX syntax and  
understand how it works

```
const simple = <h1>Hello World!</h1>;
```



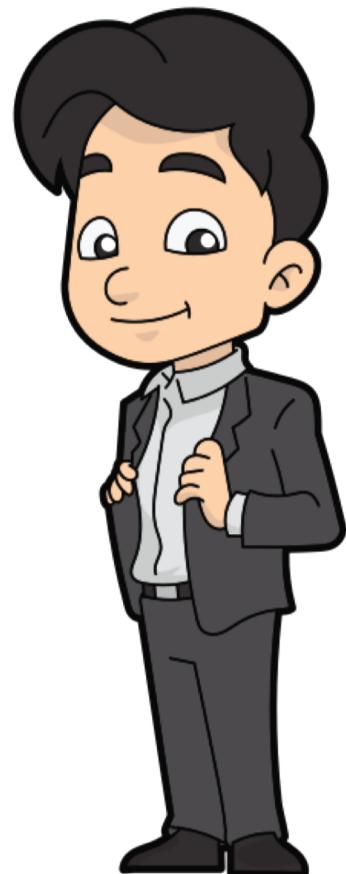
“Const simple =“ is a  
JavaScript code

```
const simple = <h1>Hello World!</h1>;
```

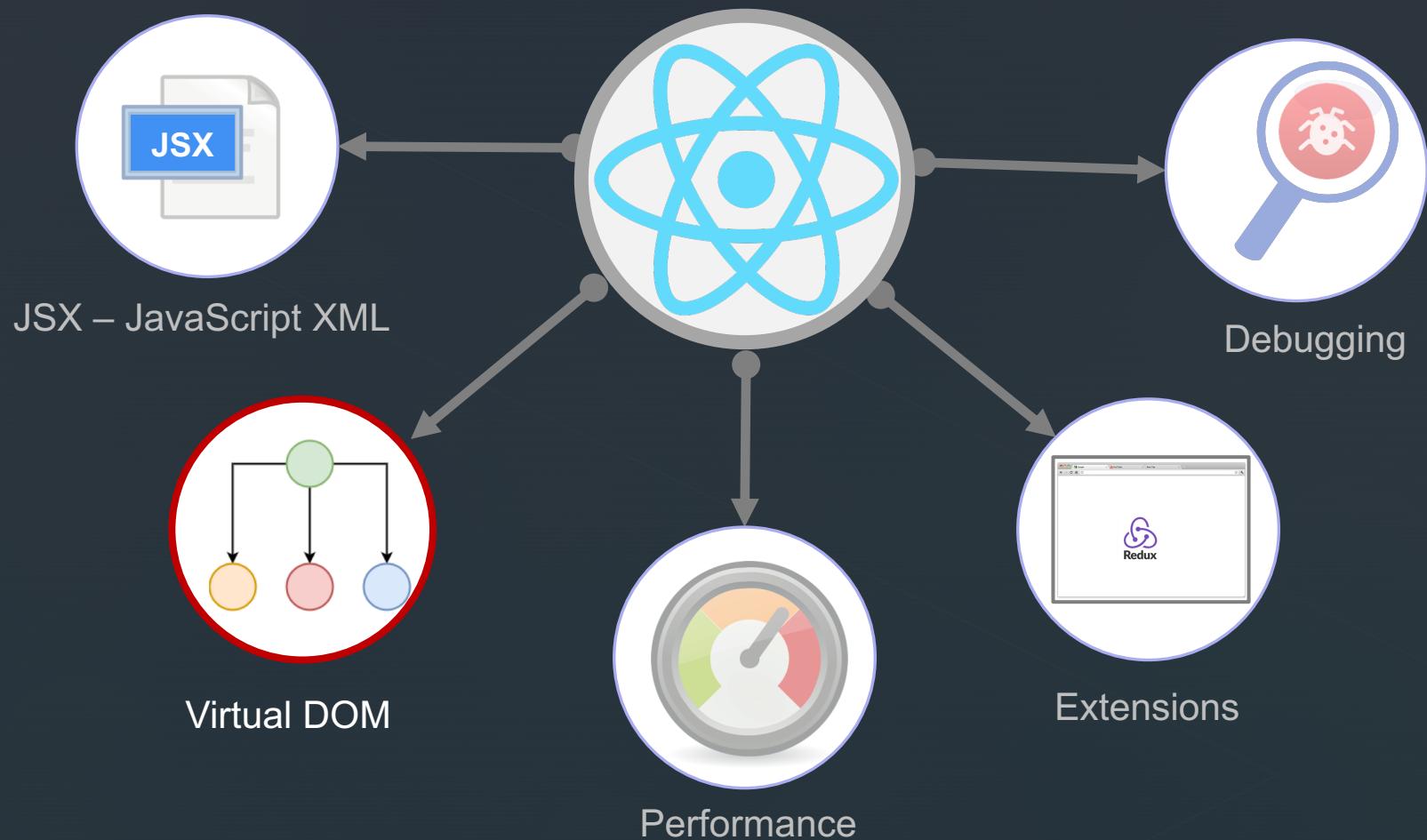


“<h1>Hello World</h1>”  
is an **HTML** code

```
const simple = <h1>Hello World!</h1>;
```

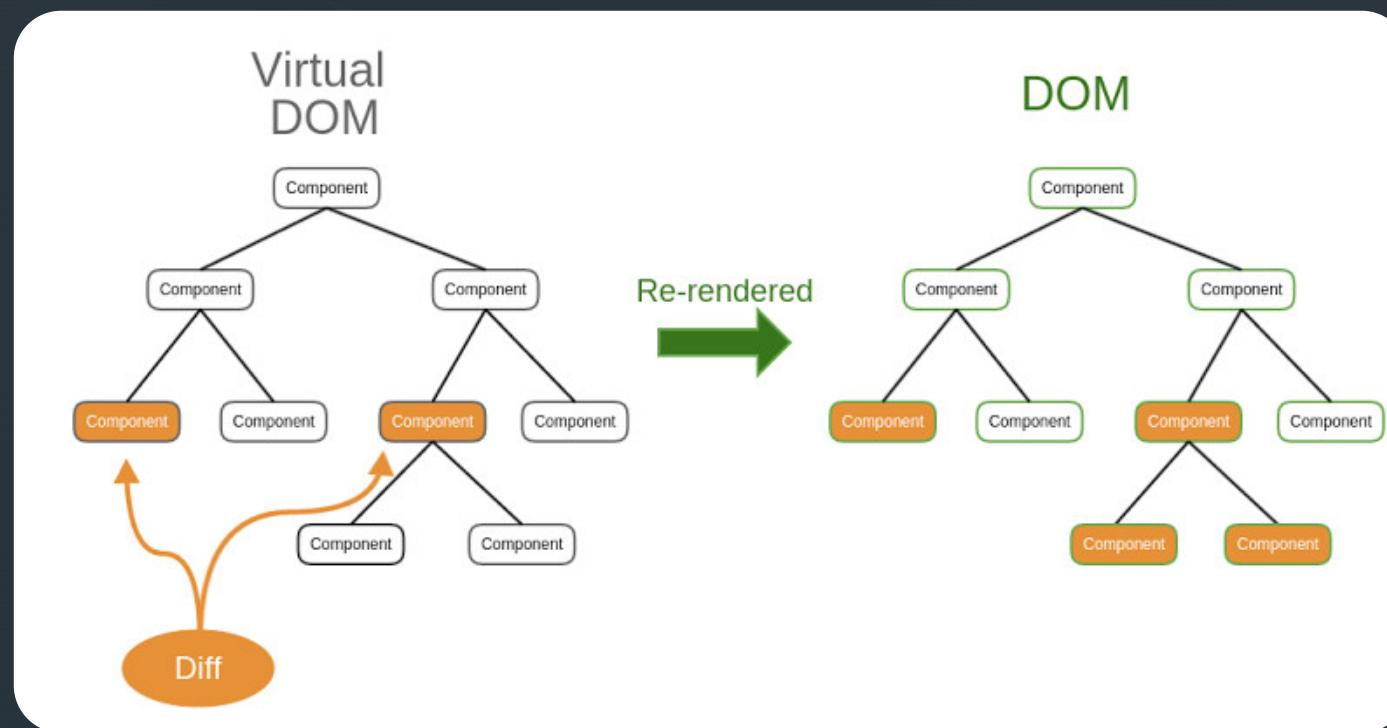


# Features of React



# Virtual DOM (React DOM)

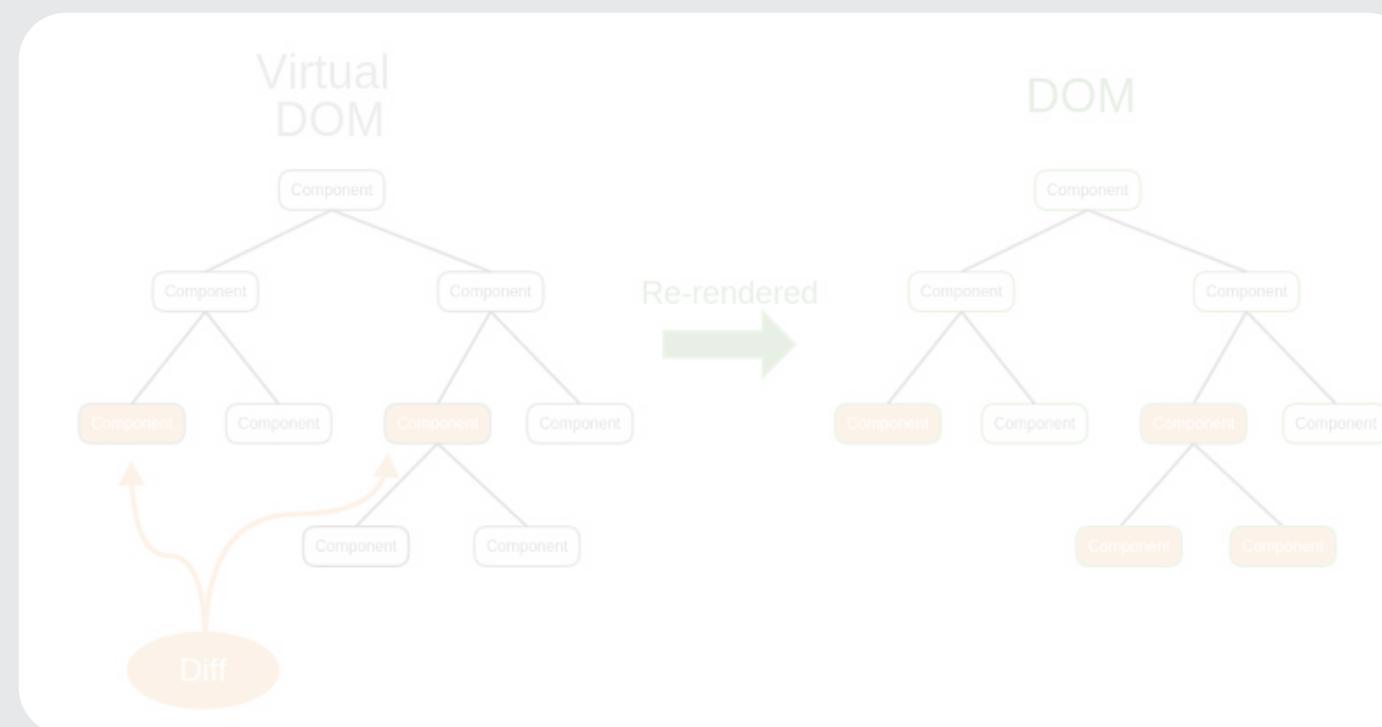
- React Keeps a lightweight representation of the Real DOM in the memory, and that is known as the Virtual DOM



# Virtual DOM

- Read the Real DOM from the Virtual DOM

DOM treats an XML or HTML document as a tree structure in which each node is an object representing a part of the document

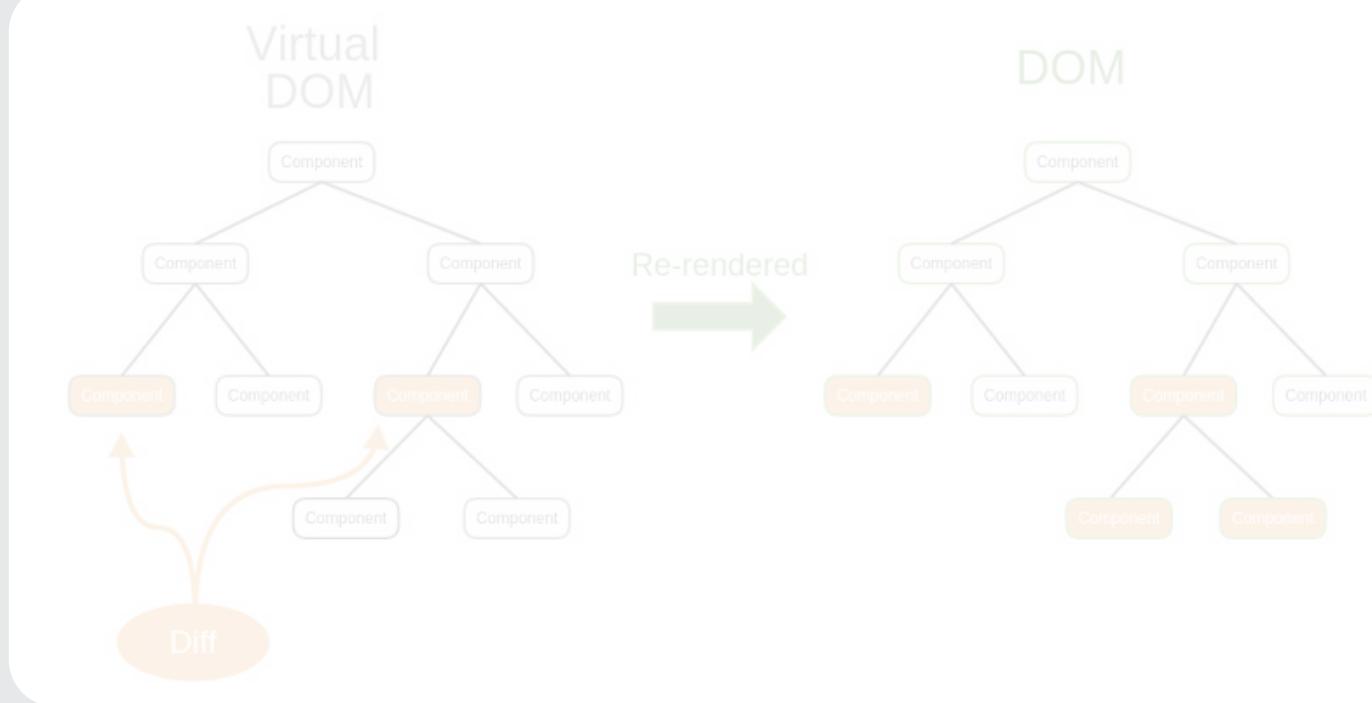


# Virtual DOM



Notice that Virtual DOM is the exact copy of the real DOM

- Read the Real DOM and the Virtual DOM



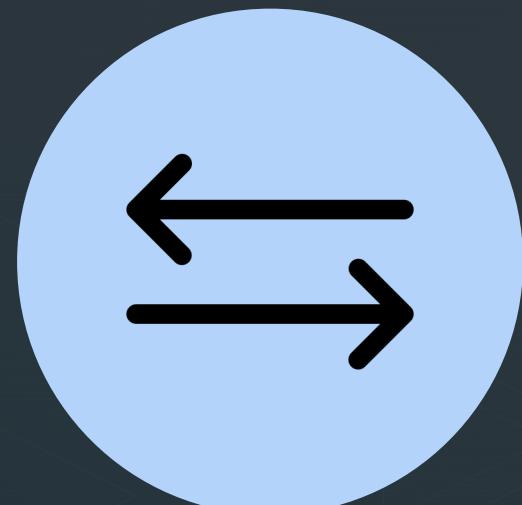
# Virtual DOM (React DOM)

- React Keeps a lightweight representation of the Real DOM in the memory, and that is known as the Virtual DOM
- Manipulating Real DOM is much slower than manipulating virtual DOM, because nothing gets drawn onscreen.

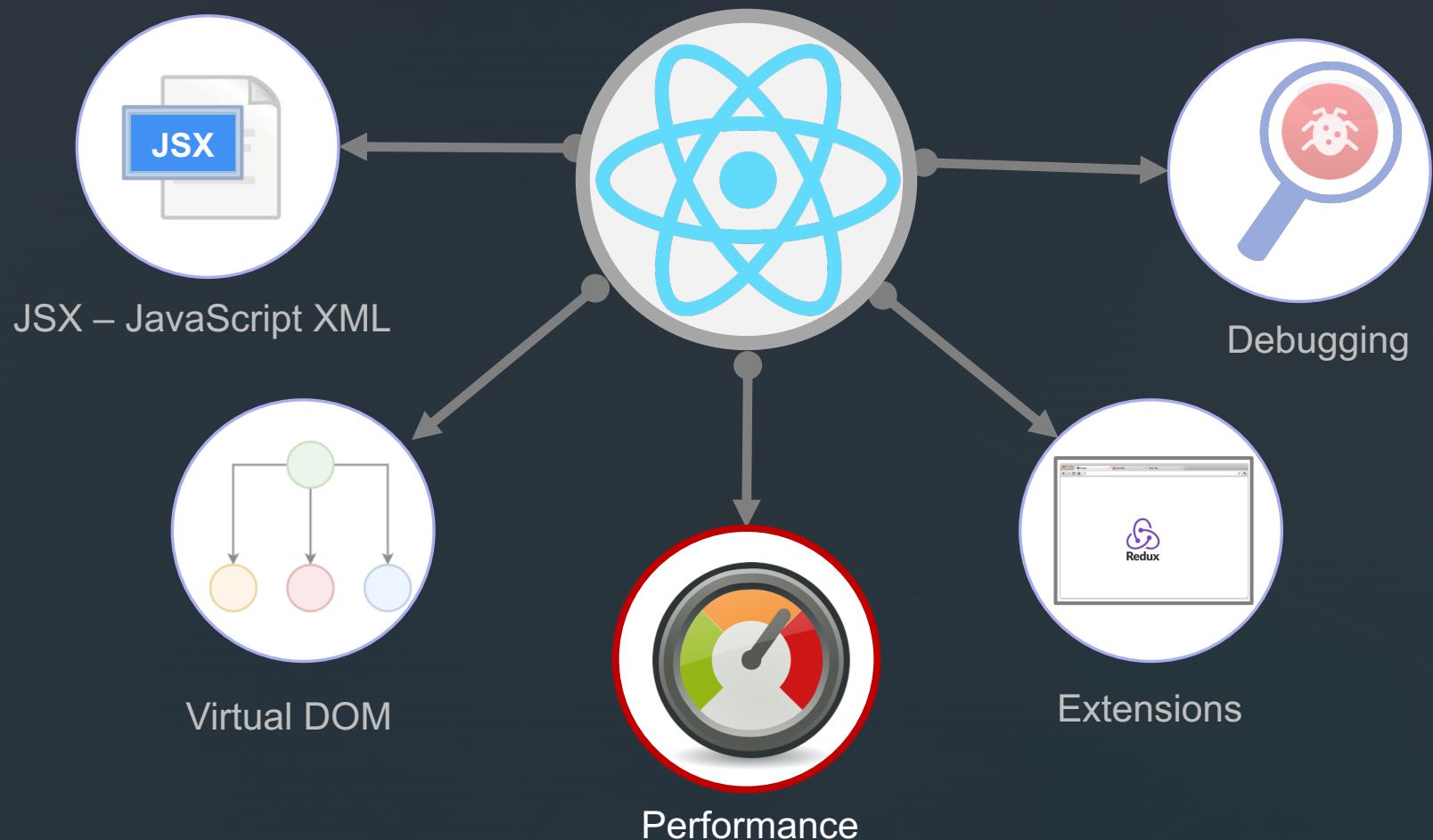


# Virtual DOM (React DOM)

- React Keeps a lightweight representation of the Real DOM in the memory, and that is known as the Virtual DOM
- Manipulating Real DOM is much slower than manipulating virtual DOM, because nothing gets drawn onscreen.
- When the state of an object changes, Virtual DOM changes only that object in the real DOM instead of updating all the objects

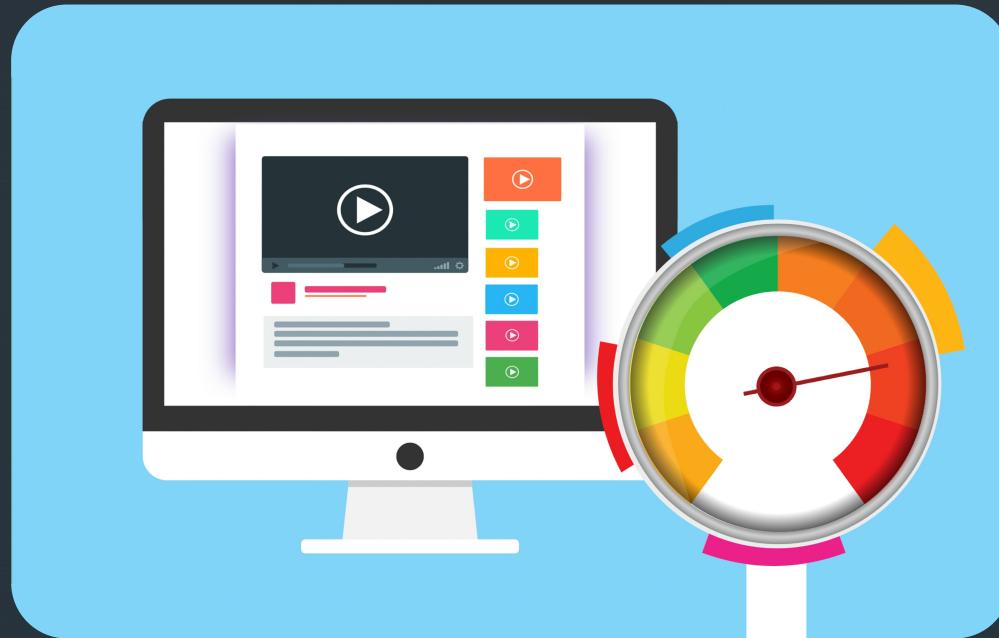


# Features of React

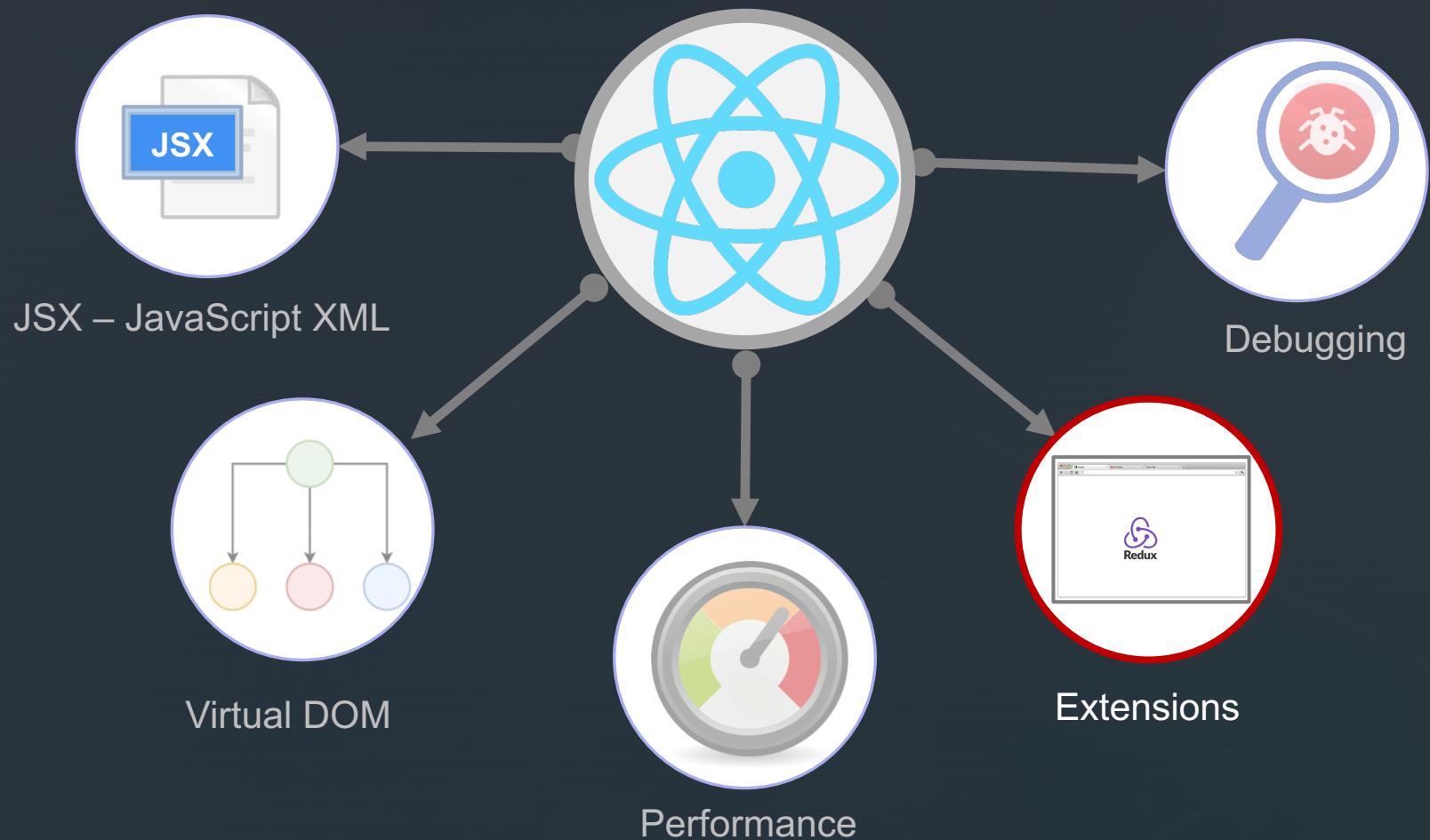


# Performance

- React uses Virtual DOM that makes the web apps fast.
- Complex User Interface is broken down into individual components allowing multiple users to work on each component simultaneously.

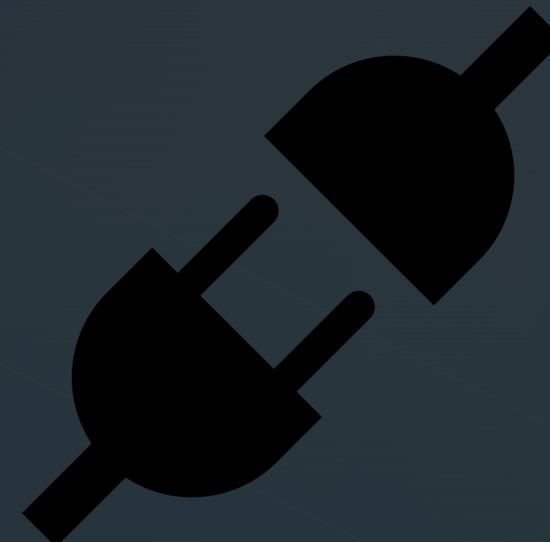


# Features of React

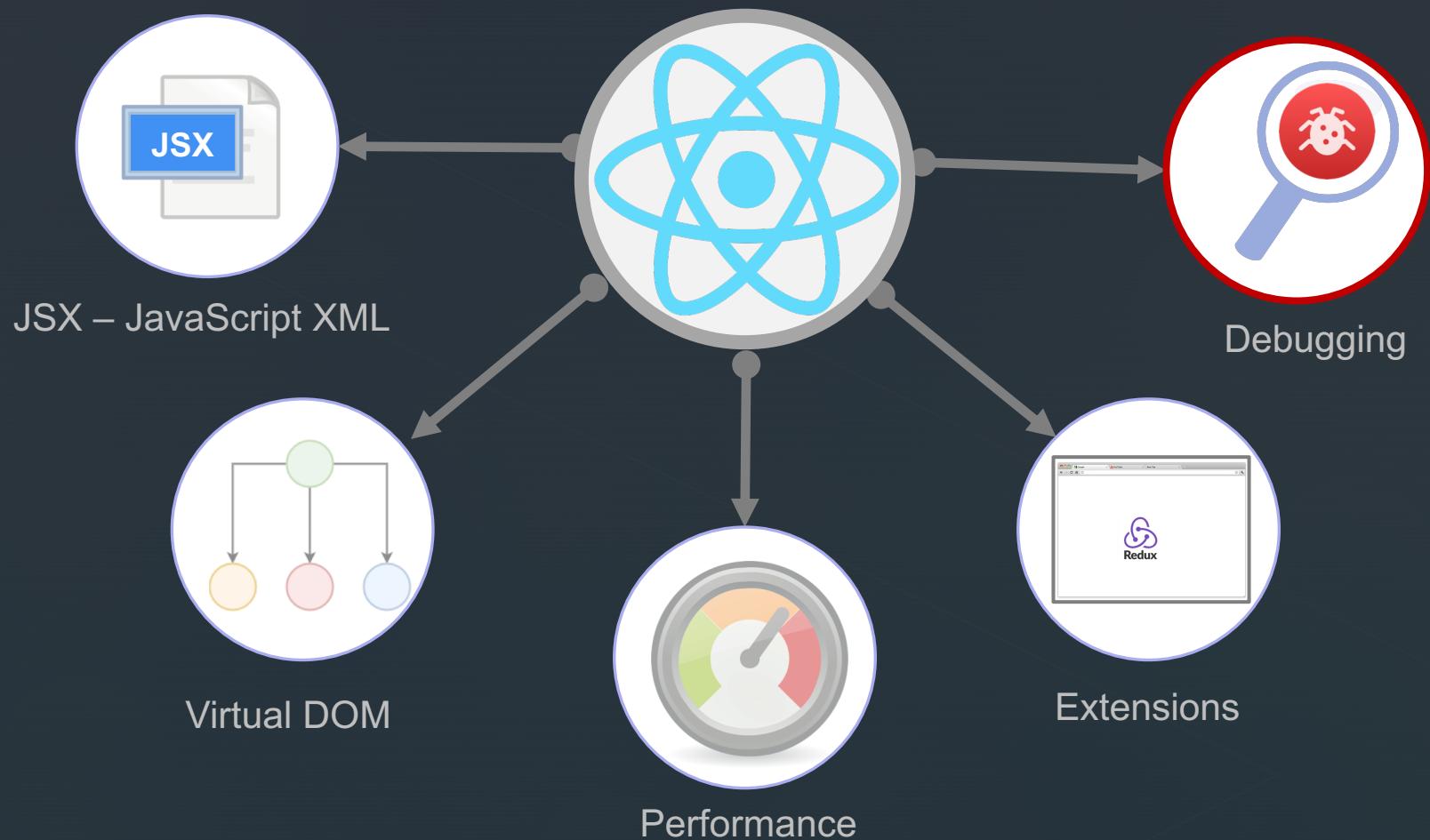


# Extensions

- React goes beyond simple UI and has many extensions for complete application architecture support.
- It provides server-side rendering
- Supports mobile app development
- Extended with Redux, among others

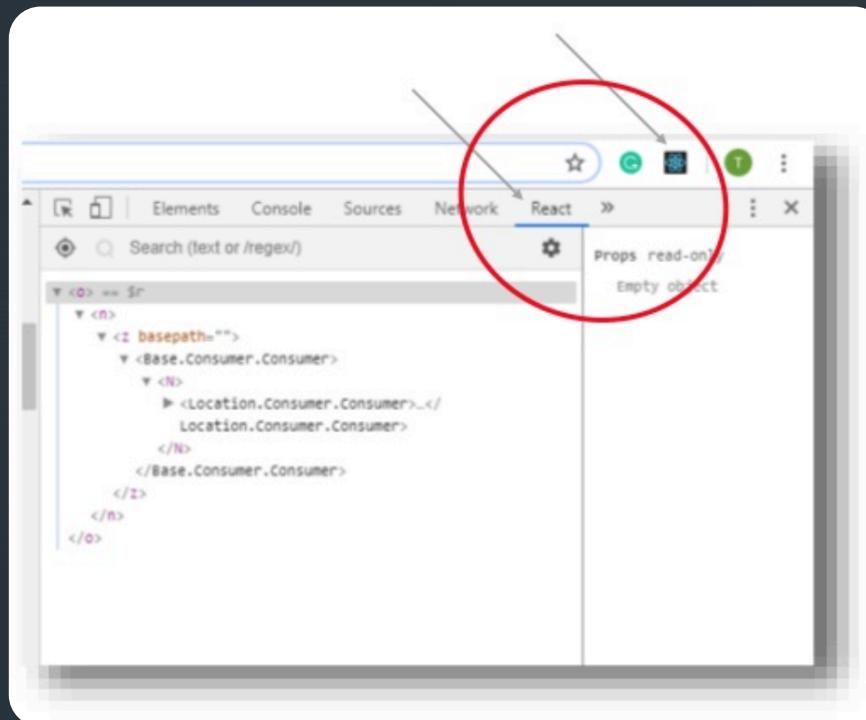


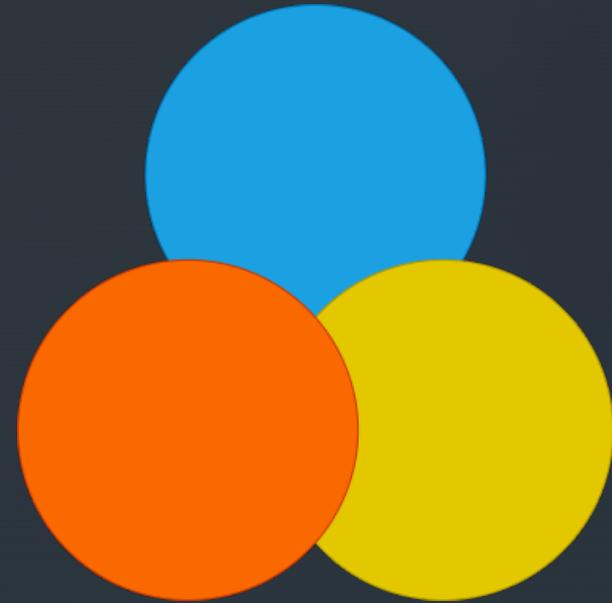
# Features of React



# Debugging

- React applications are extremely easy to test due to a large developer community
- Facebook even provides a small browser extension that makes React debugging faster and easier (React Developer Extension)

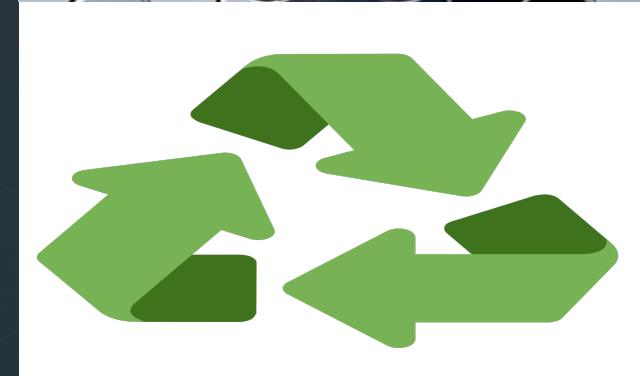




# Components, State & Props

# Reac Components

- Components are the building blocks of any React application, and a single app usually consist of multiple components
- A component is sessentially a piece of the User Interface. It is like a function/class that return HTML elements.
- It splits the User Interface into independent reusable pieces that can be processed seperately



# Reac Components cont...

A component is implemented as a Javascript class having some state and a render method

```
import React, { Component } from 'react';

export default class Simple extends Component {
  render() {
    return (
      <div>
        <h1>This is a Component</h1>
      </div>
    );
  }
}
```



# Reac Components cont...

```
import React, { Component } from 'react';

export default class Simple extends Component {
  constructor(){
    this.state={ }
  }
  render() {
    return (
      <div>
        <h1>This is a Component</h1>
      </div>
    );
  }
}
```

State Object of a component is mutable. It means only methods of the same component can modify it's content.



```
import React, { Component } from 'react';

export default class Simple extends Component {
  state = { }
  render() {
    return (
      <div>
        <h1>This is a Component</h1>
      </div>
    );
  }
}
```

# Reac Components cont...

Render method is responsible for how the User Interface looks and feels to the user.

```
import React, { Component } from 'react';

export default class Simple extends Component {
  state = { }
  render() {
    return (
      <div>
        <h1>This is a Component</h1>
      </div>
    );
  }
}
```



# State of a Component

- State of a component is an object that holds some data
- This data influence the output of a component
- To modify state properties:
  1. Use `this.setState({propName:value})`
  2. Use spread operator if property assigned to array or object.  
Example:  
`this.setState({propName: [...array, newElm]})`
  3. You can use push, pop, shift, unshift, splice to manipulate the state object  
Example:  
`this.state.users.push(newUserObject);`



# State of a Component

- This is how we can access State's properties:

```
import React, { Component } from 'react'

export default class App extends Component {
  state = {
    name: 'fahim'
  }

  render() {
    return (
      <div>
        |  Hi, {this.state.name}. Welcome here!
      </div>
    )
  }
}
```

# Props object of a component

- Props is short for properties, that allows us to pass arguments or data to components
- Props are passed to components in the way similar to that HTML tag attributes

## Parent Comp

```
import React, { Component } from 'react'
import Child from 'Child.js'

export default class App extends Component {
  state = {
    name: 'fahim'
  }

  render() {
    return (
      <div>
        <h3>Hi, {this.state.name}. Welcome here!</h3>
        1 <Child name={this.state.name} />
      </div>
    )
  }
}
```

## Child Comp

```
import React, { Component } from 'react'

export default class Child extends Component {
  render() {
    return (
      <div>
        2 <h5>Hey {this.props.name}.
          This is child-component!</h5>
      </div>
    )
  }
}
```

# Bracket Notation

- Since React let us to write HTML and JavaScript in same file so we are using JSX.
- JSX code needed to be written inside ( )
- If you want to write JavaScript between JSX use following syntax

```
( <JSX> { JavaScript } </JSX> )
```

Note: Sometime we need to use array.map(()=>{})

- In such cases if you need to write JSX in the callback, do it such as:

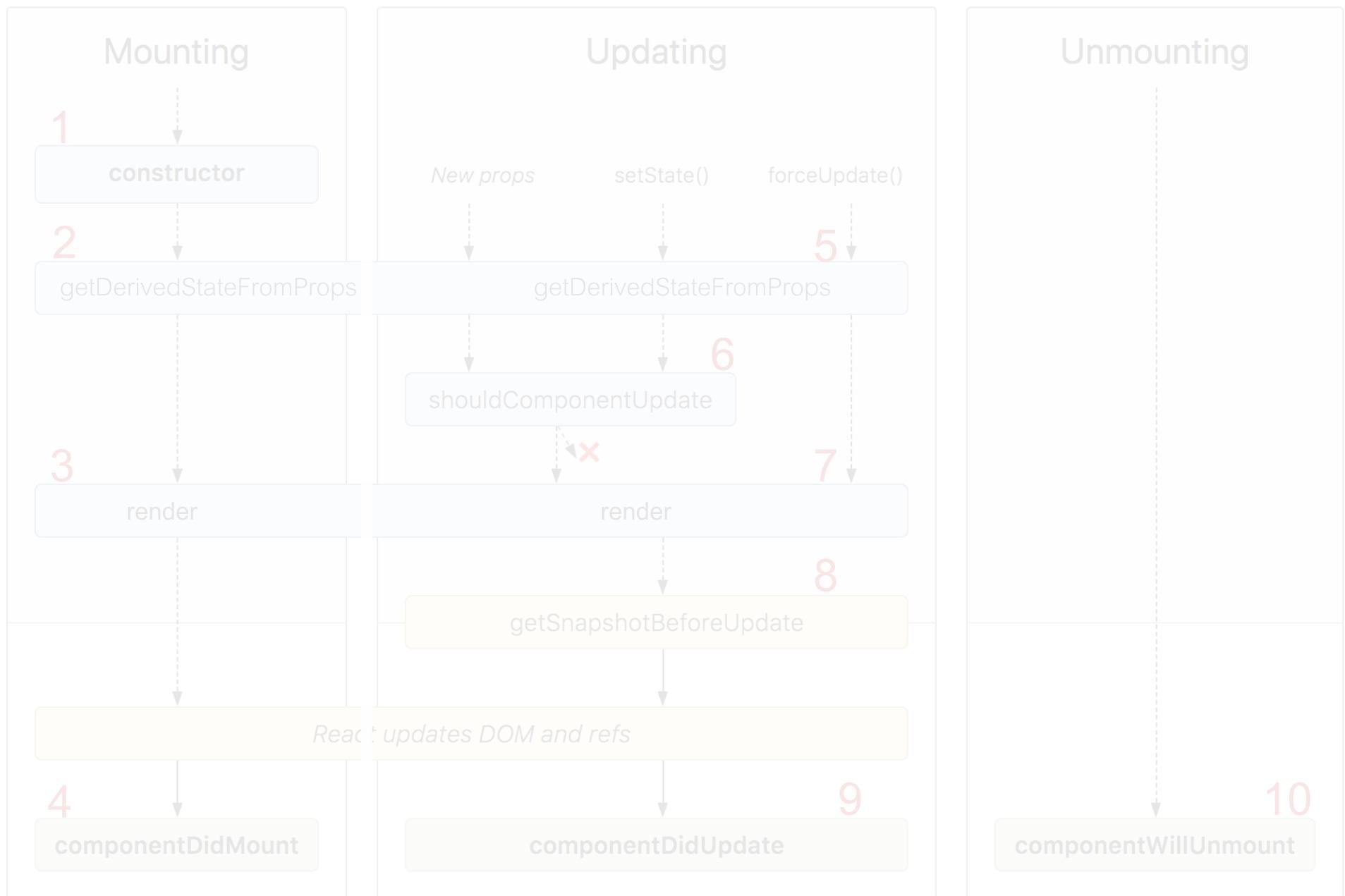
```
{array.map(()=>{ return (<JSX> </JSX>) }) }  
{array.map(()=>(<JSX></JSX>))}
```

# React Life Cycle



# React Life Cycle

- Series of pre-defined events that triggers from initiating to death of a react component.
- React life cycle consist of three phases:
  - **Mounting:** Component will render for the first time (Birth phase).
  - **Updating:** By state change or forceUpdate needs to render the components again and again. (Growth phase)
  - **Unmounting:** When no need to display the component anymore and remove it from user interface. (Death phase)



# Mounting Phase – constructor()

- Constructor is the first method of a component that will execute in mounting phase.
- This method is a good place to initialize properties and methods of the class component.
- The method is not mandatory for component's definition.

```
constructor() {  
  super();  
  this.state = {  
    name: "fahim",  
    email: "fahim@gmail.com",  
    users: [],  
  };  
  console.log("Constructor called");  
}
```

# Mounting Phase – getDerivedStateFromProps()

- This method helps the programmer to set properties of state object from props object or statically.
- This method calls immediately before render() method.
- This method is **static** method. Means no need for instance of the class to invoke it.
- It returns an object to update the state, but returns null when there is no changes.

```
static getDerivedStateFromProps(props, state) {  
  return {  
    //name is a property of state object  
    name: props.name  
  }  
}
```

# Mounting Phase – render()

- This method is the only required method in class components and it is responsible to handle rendering of component to UI.
- Return value of this method is a single JSX element that can contains child.
- React requires that your render() be pure.
  - **Pure functions** are those that do not have any side-effects and will always return the same output when the same inputs are passed
  - This means setState() can not be called within render().

# Mounting Phase – componentDidMount()

- This method calls when the component mounted successfully and being visible for the end-user.
- This method is a good place to sends requests to APIs and update the state object.

```
componentDidMount() {  
  console.log("ComponentDidMount");  
  axios({  
    method: "GET",  
    url: "https://randomuser.me/api/?results=5",  
    headers: { "Content-type": "application/json" },  
  }).then((res) => {  
    this.setState({ users: [...res.data.results] });  
    this.displayUsers(this.state.users);  
  });  
}
```

# Updating Phase – shouldComponentUpdate()

- This method calls before render method and let the programmer to cancel rendering on state/props changes manually.
- This method helps to increase performance by preventing from un-necessary rendering.
- The method returns a boolean value that specify needs to render or not.

```
shouldComponentUpdate(nextProps, nextState) {  
  return this.props.title !== nextProps.title ||  
    this.state.input !== nextState.input }
```

# Updating Phase – componentDidUpdate()

- This method calls in the update phase.
- The common use-case for it is updating DOM in response to state/prop changes.
- setState() is allowed here but needs to be wrapped into an if condition.

```
componentDidUpdate(prevProps) {  
  //Typical usage, don't forget to compare the props  
  if (this.props.userName !== prevProps.userName) {  
    this.fetchData(this.props.userName);  
  }  
}
```

# Unmounting Phase – componentWillUnmount()

- This method calls immediately before unmounting the component from user interface.
- It's a good place to do the cleanup actions such as clearing timers, cancelling api calls or clearing caches in storage.
- setState() is not allowed here. Because there is no more render for the component, when this method called.

```
componentWillUnmount() {  
  window.removeEventListener('resize', this.resizeListener)  
}
```