EN
English

Universidad de Oviedo

School of Computer Science

SOFTWARE ARCHITECTURE

# Software Architecture

Lab. 03

React

Solid

2021-22

Jose Emilio Labra Gayo
Pablo González
Irene Cid
Hugo Lebredo

## What is React.js?

React is a JavaScript library for building user interfaces for the web as well as mobile applications

- Open source
- Created by Facebook
- Based on components

## Why React?

Some reasons to use React:

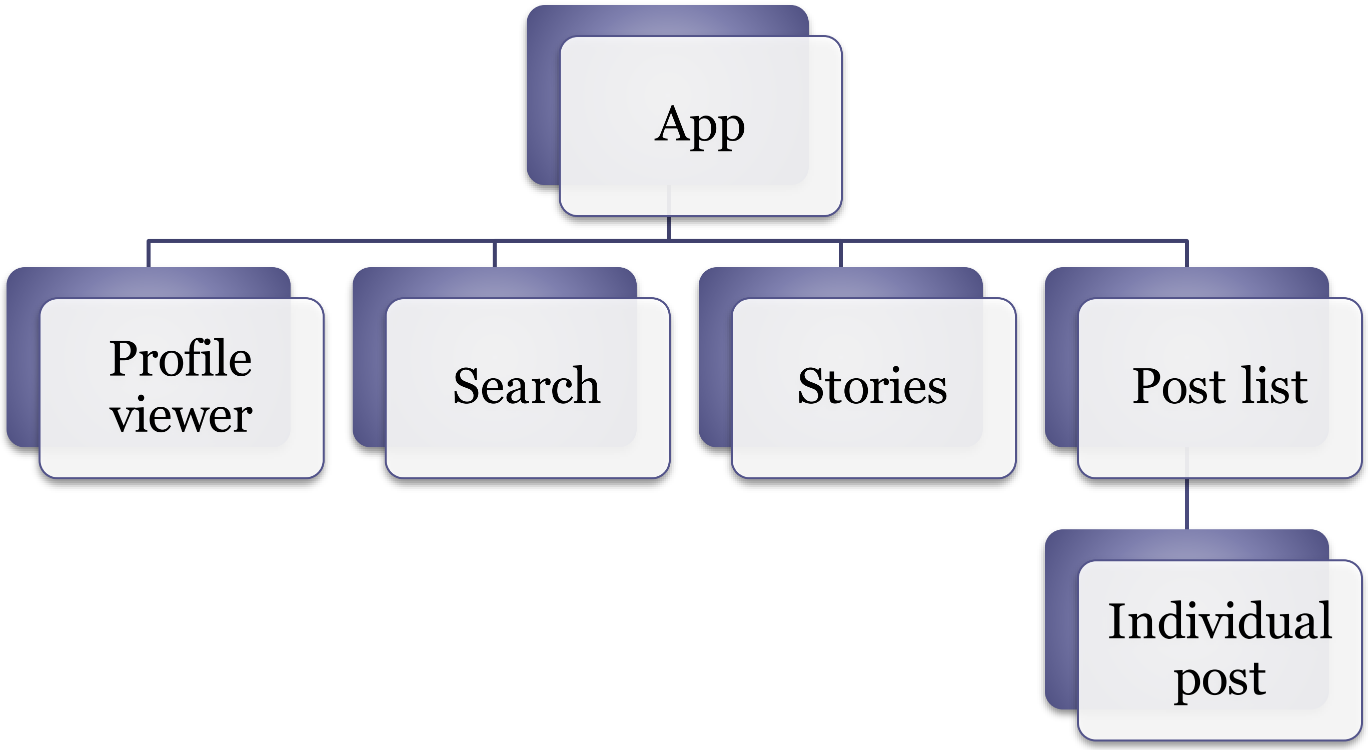
- Simplicity and easy to learn
- Reusable components
- Native approach (React Native)
- Lots of resources and tools for development
- Testability

## Components

Pages are modelled using components

A component is a part of the user interface

- Example: Instagram



## Components

A component can be implemented as a JavaScript class

- It has a state
- And a `render` method that controls what is displayed in UI
- When the state changes, react updates the element and its child's in memory
- This element representation in memory is called Virtual DOM

```
class ProfileViewer{
  state = {}
  render(){
  }
}
```

React **reacts** to changes

# JSX

JSX is a syntax extension to Javascript

- Looks like HTML
- JSX files are compiled to Javascript

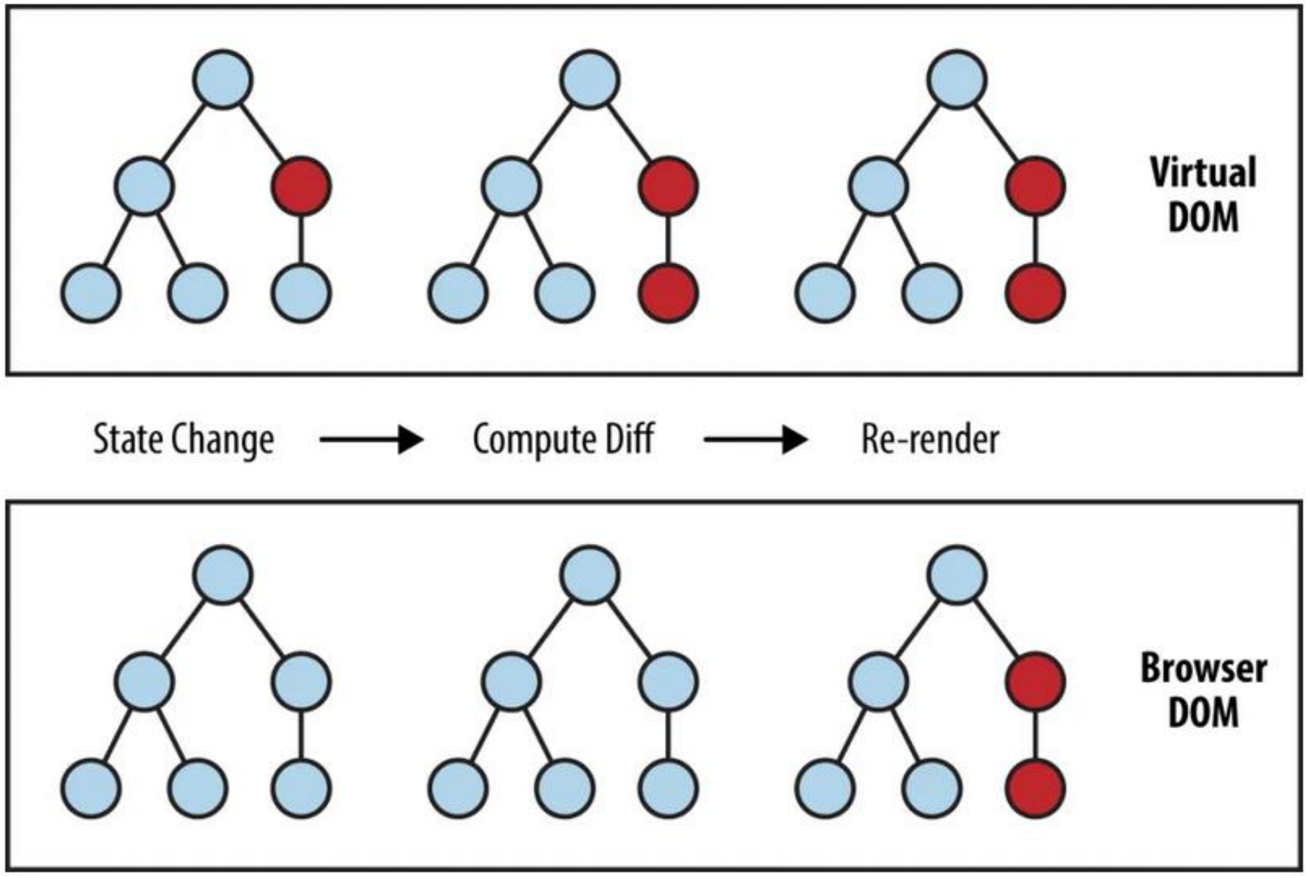
JSX

```
const element = (
  <h1 className="greeting">
    Hello, world!
  </h1>
);
```

Javascript

```
const element = React.createElement(
  'h1',
  {className: 'greeting'},
  'Hello, world!'
);
```

# Virtual DOM



source: https://www.oreilly.com/library/view/learning-react-native/9781491929049/ch02.html

## We also have Hooks

They replace classes by functions

- In the following example, we use the `useState` hook to handle the name changes in the app
- Once the button is clicked, the state is changed, the virtual DOM updated, and the page is automatically refreshed

```
const App = () => {
    const [name, setName] = useState('World');
    return (
        <div className="App">
            <h1>Hello, {name}!</h1>
            <button onClick={() => setName('James')}>
                Click me to change the name
            </button>
        </div>
);}
```

## Typescript with React

Adds optional **static typing** to Javascript

```
type UserListProps = {
  users: User[];
}
```

```
export type User = {
    name:string;
    email:string;
  }
```

```
function UserList(props: UserListProps): JSX.Element {
  return (
    <>
      <List>
          {props.users.map((user,i)=>{
             return (
                   <ListItem key={user.email}>
                       <ListItemIcon><ContactPageIcon/></ListItemIcon>
                       <ListItemText primary={user.name} secondary={user.email}/>
                   </ListItem>
             )}))}
      </List>
    </>
  );
}
```

This component will render an array of Users that will receive as a component property. Check the full example in **your repository**!

## SOLID

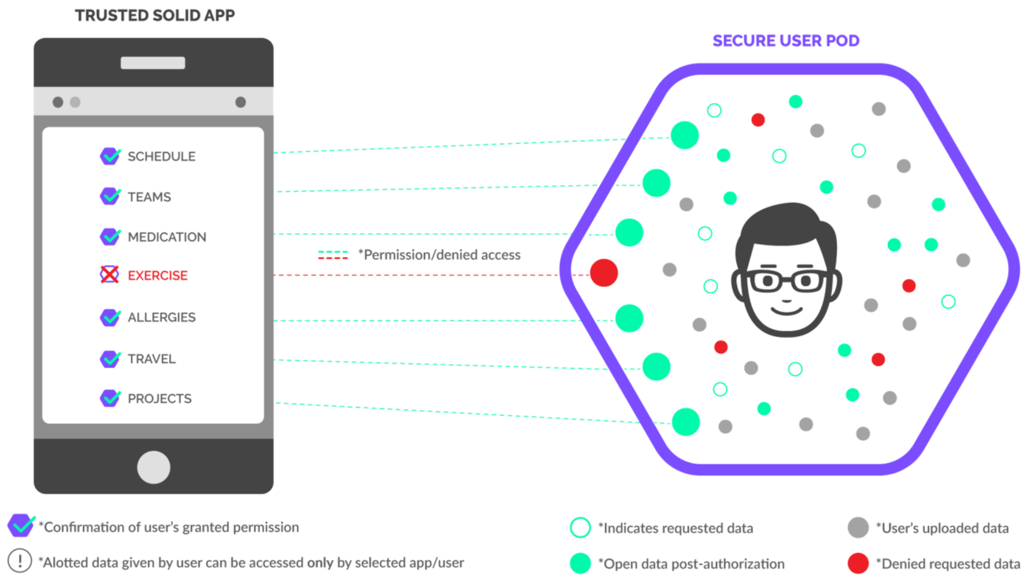
Is a technology for organizing apps, information and identities in a decentralized way

- Initially started at MIT
- Directed by Tim Berners-Lee

# Let's create a POD

For creating a POD, we need a POD provider. We can use an external one or host our own POD server

- https://inrupt.net/


TRUSTED SOLID APP
SCHEDULE
TEAMS
MEDICATION
EXERCISE
ALLERGIES
TRAVEL
PROJECTS
*Permission/denied access
SECURE USER POD
*Confirmation of user's granted permission
*Alotted data given by user can be accessed only by selected app/user
*Indicates requested data
*Open data post-authorization
*User's uploaded data
*Denied requested data

## How to combine Solid and React?

- React components
  https://github.com/inrupt/solid-ui-react
- Example project (javascript)
  https://github.com/pglez82/solid-react-example
- Example project (with typescript)
  https://github.com/Arquisoft/solid-react-example/
- Solid Documentation from Inrupt
  https://docs.inrupt.com/
- Awesome Solid
  https://github.com/pdsinterop/awesome-solid
- Videos introducing Solid by Jackson Morgan
  https://www.youtube.com/playlist?list=PLtNrk03_ElXBGf5fmrqqkYew9Z3L0YifN

## Additional documentation

Exercises about React state (in Spanish)

I. Ej1 Create a counter
II. Ej2 Complex states(objects)
III. Ej3 Different handlers()
IV. Ej4 Adding elements to an array
V. Ej5 Change a component's behaviour (background color)

## Additional documentation

Exercises rendering in React

I. Ej1 Array rendering
II. Ej2 Refactoring
III. Ej3 Adding elements to the array
IV. Ej4 Adding elements from a form

## Additional documentation

Asynchronous programming

I. Ej1 Fetch() -> Do an API request
II. Ej2 useEffect()
III. Ej3 Conditional rendering
IV. Ej4 One Refactoring
V. Ej5 Requests using axios library

## Additional documentation

Exercises using Typescript + React

I. Ej1 Counter with typescript
II. Ej2 2nd exercise
III. Ej3 Example of an interface

## Additional documentation

More links

I. Course Bootcamp Fullstack

II. First Node.js conference by Ryan Dahl

End of presentation