

# React 4: Events & Conditional Rendering

IN608: Intermediate Application Development Concepts

Kaiako: Tom Clark & Grayson Orr

# Last Session's Content

- State
- Lifecycle methods
  - Mounting
  - Updating
  - Unmounting
- Data flow
- React hooks

# Today's Content

- File structure
- Events
  - Binding
- Conditional rendering

# Events

# Events

- Event handling with React elements is very similar to event handling on DOM elements
- There are two key syntax differences:
  - Camelcase is the naming convention for React events, not lowercase
  - With JSX, a function is passed as the event handler, not a string

```
// HTML  
<button onclick="someEvent()"></button>
```

```
// JSX  
<button onClick={someEvent}></button>
```

# Events

- Create a new component file called `Register.js`
- Renders a button which the user can toggle between a “Registered” & “Unregistered” state

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

const Register = () => {
  const [isRegistered, setIsRegistered] = useState(true)

  const handleRegisteredChange = () => setIsRegistered(!isRegistered)

  return (
    <React.Fragment>
      <button onClick={handleRegisteredChange}>
        {isRegistered ? 'Registered' : 'Unregistered'}
      </button>
    </React.Fragment>
  )
}

export default Register
```

# Events

- In App.js

```
import React from 'react'
import Clock from './Clock'
import Owner from './Owner'
import Register from './Register'
import afghanHoundImg from '../img/afghan-hound.jpg'

const dog = {
  name: 'Bingo',
  breed: 'Afghan Hound',
  img: afghanHoundImg
}

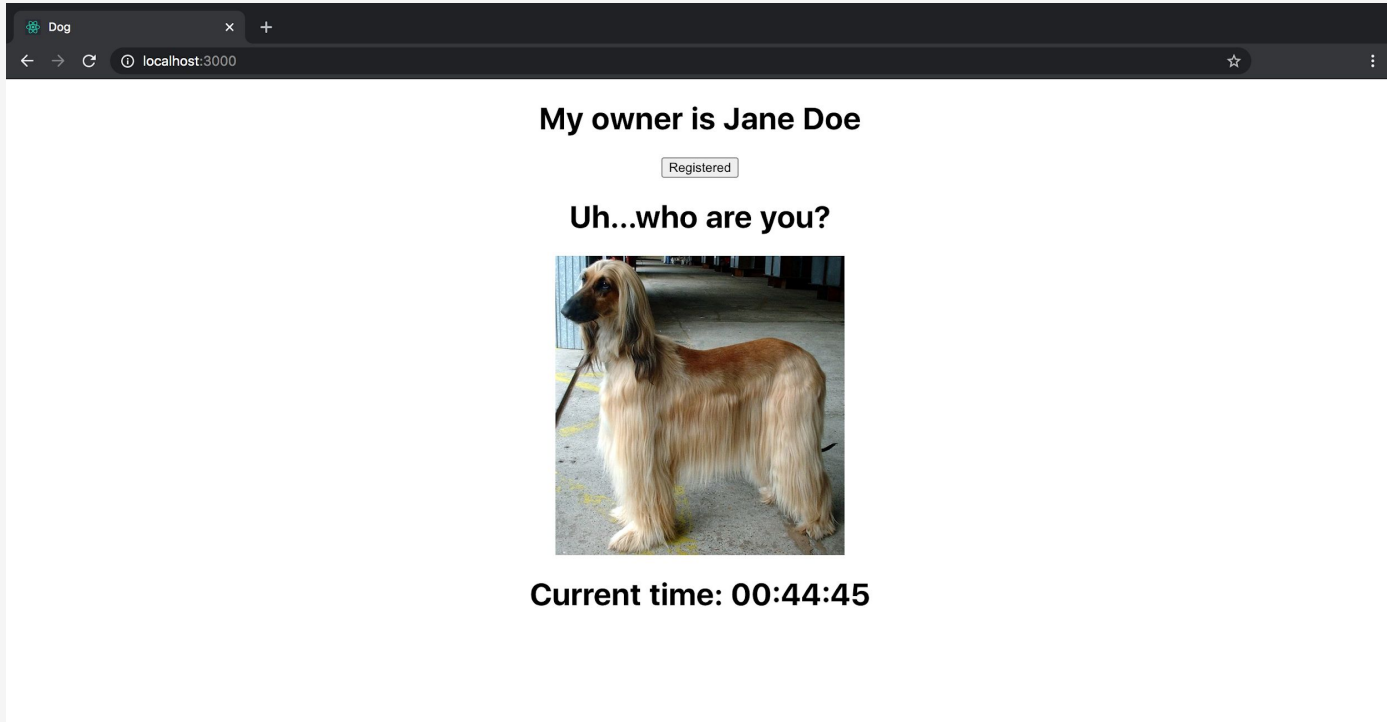
const formatDog = (dog) => `Woof woof, my name is ${dog.name} & my breed is an ${dog.breed}`

const getGreeting = (dog) => {
  if (dog) {
    return <h1>{formatDog(dog)}</h1>
  }
  return <h1>Uh...who are you?</h1>
}

const App = () => {
  return (
    <div className='container'>
      <Owner />
      <Register />
      {getGreeting()}
      <img src={dog.img} alt='afghan hound' width='300' />
      <Clock />
    </div>
  )
}

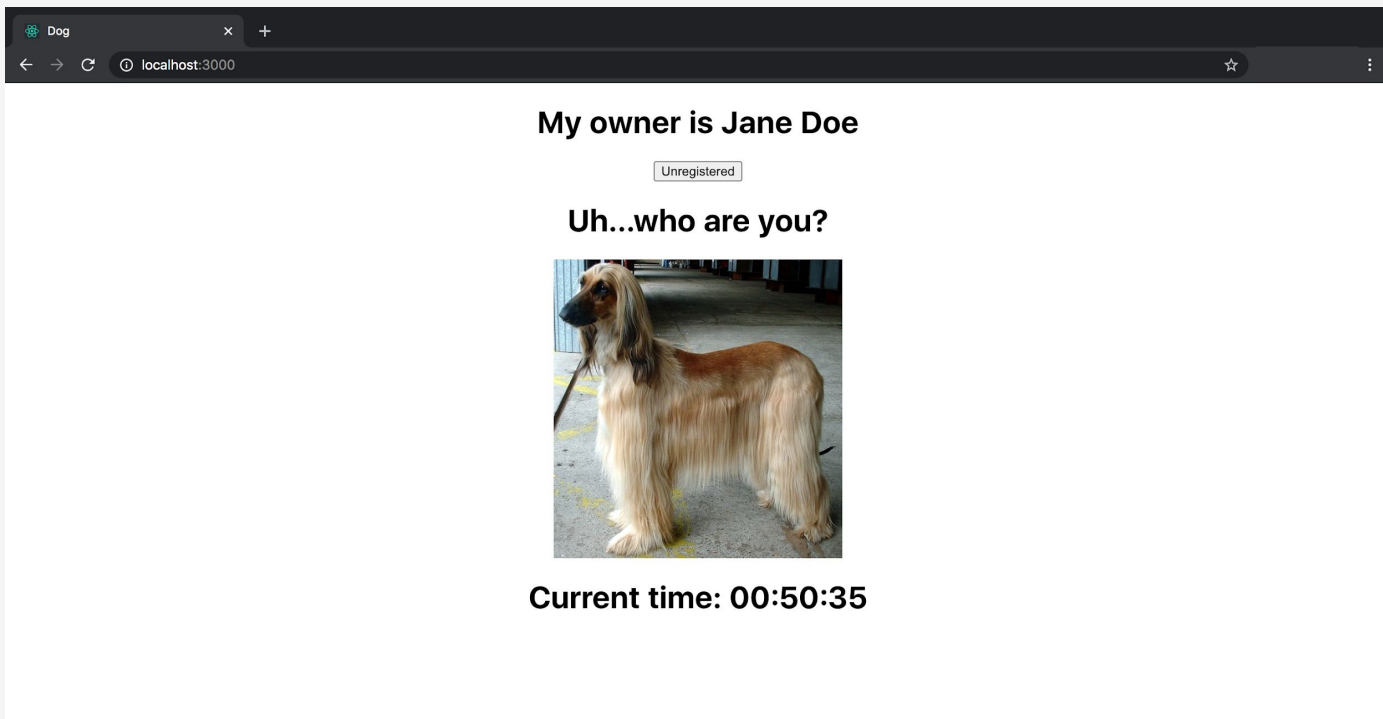
export default App
```

# Events





# Events



# Conditional Rendering

# Conditional Rendering

- Conditional rendering in React works the same as in JavaScript
  - You can use conditional statements, i.e., `if`, `else`, `else if`, `switch` or the ternary operator
- A component decides based on one or several conditions which element it will return, then later render
- Consider the following two components:

```
import React from 'react'

const GuestGreeting = () => <h1>Please sign up</h1>

const UserGreeting = () => <h1>Welcome back</h1>

export default GuestGreeting
export default UserGreeting
```

- **Note:** These two components are in separate files. `GuestGreeting.js` & `UserGreeting.js`
- We will create a component which renders either `GuestGreeting` or `UserGreeting` based on whether a user is logged in

# Conditional Rendering

- Create a new component file called `Greeting.js`
- Renders a different greeting depending on the value of `isLoggedIn` prop

```
import React from 'react'
import GuestGreeting from './GuestGreeting'
import UserGreeting from './UserGreeting'

const GuestGreeting = (props) => {
  const isLoggedIn = props.isLoggedIn
  return isLoggedIn ? <UserGreeting /> : <GuestGreeting />
}

export default Greeting
```

# Conditional Rendering

- Consider the following two components:

```
import React from 'react'  
  
const LoginButton= (props) => <button onClick={props.onClick}>Login</button>  
const LogoutButton= (props) => <button onClick={props.onClick}>Logout</button>  
  
export default LoginButton  
export default LogoutButton
```

- Note:** These two components are in separate files

# Conditional Rendering

- Create a new component file called `LoginControl.js`

```
import React, { useState } from 'react'
import Greeting from './Greeting'
import LoginButton from './LoginButton'
import LogoutButton from './LogoutButton'

const LoginControl = () => {
  const [isLoggedIn, setIsLoggedIn] = useState(true)

  const handleLoginClick = () => setIsLoggedIn(!isLoggedIn) // true
  const handleLogoutClick = () => setIsLoggedIn(!isLoggedIn) // false

  const button = isLoggedIn ? (
    <LogoutButton onClick={handleLogoutClick} />
  ) : (
    <LoginButton onClick={handleLoginClick} />
  )

  return (
    <React.Fragment>
      <Greeting isLoggedIn={isLoggedIn} />
      {button}
    </React.Fragment>
  )
}

export default LoginControl
```

# Conditional Rendering

- Renders `Greeting` & either `LoginButton` or `LogoutButton` depending on its current state, i.e., `isLoggedIn`
- `React.Fragment` - group a list of children without adding extra node to the DOM

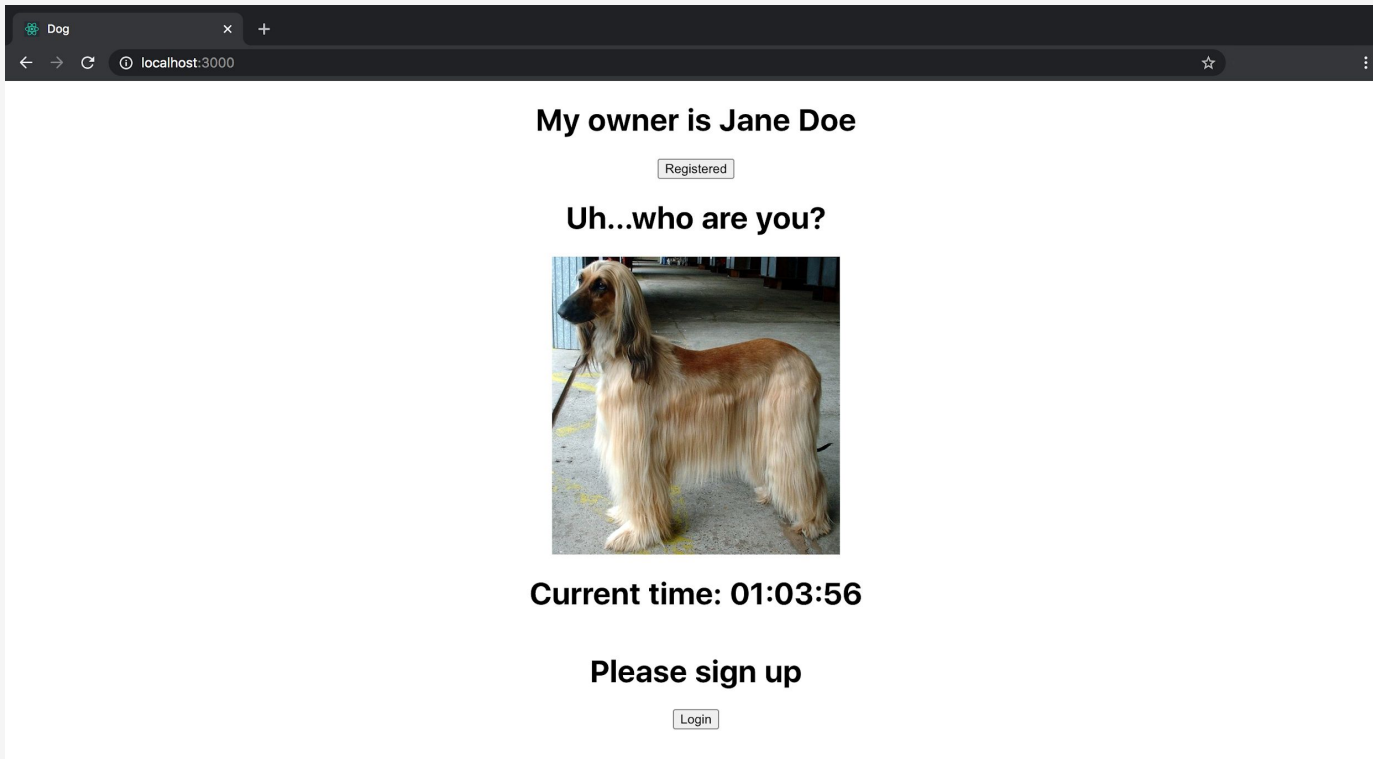
# Conditional Rendering

- In App.js
- `import LoginControl from './LoginControl'`
- Declare `<LoginControl />` in App()

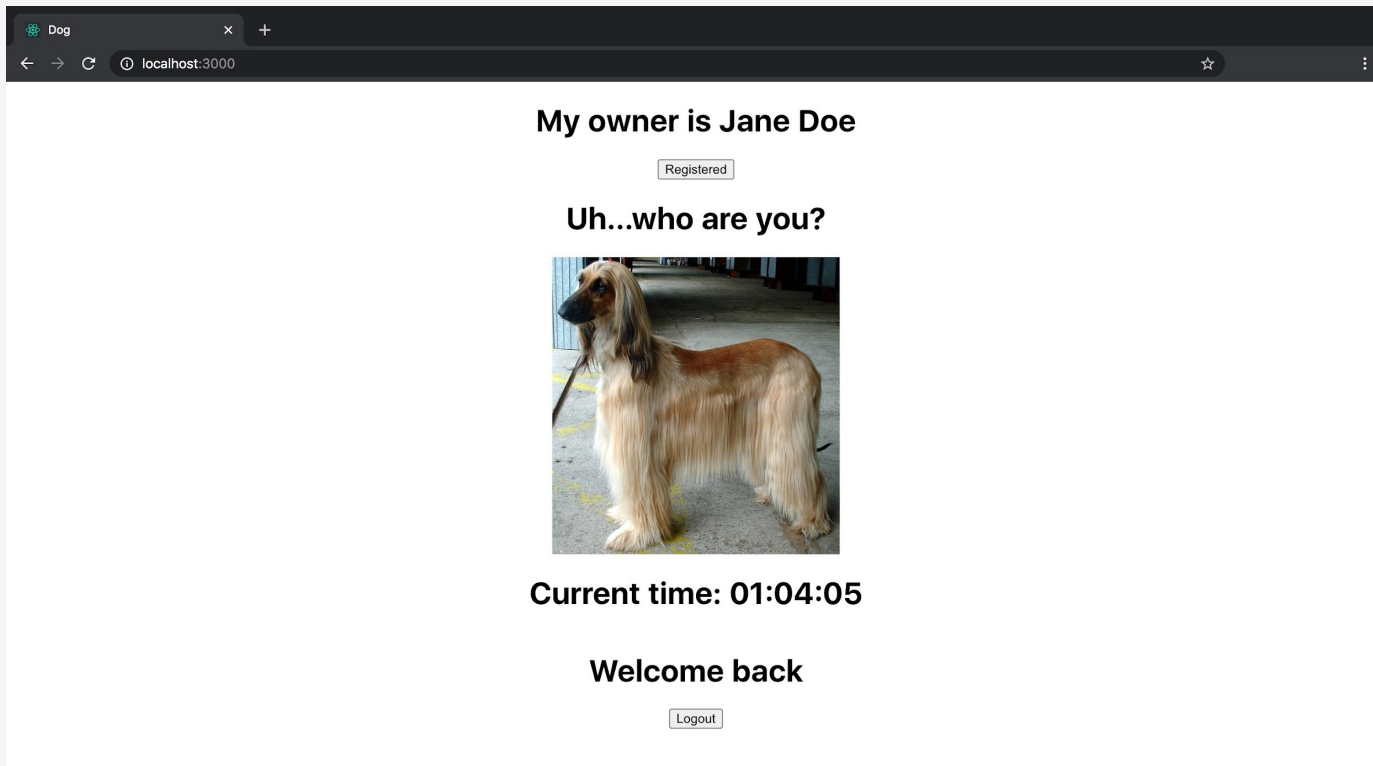
```
const App = () => {  
  return (  
    <div className='container'>  
      <Owner />  
      <Register />  
      {getGreeting()}  
      <img src={dog.img} alt='afghan hound' width='300' />  
      <Clock />  
      <LoginControl />  
    </div>  
  )  
}
```



# Conditional Rendering



# Conditional Rendering



# Programming Activity

- Checkout to master - `git checkout master`
- Create a new branch called 19-practical - `git checkout -b 19-practical`
- Open the file `19-practical.pdf` and work on the tasks described