Video 3.1

React.js: Introduction

Chris Murphy

Review

- **JavaScript:** a general-purpose, easy-to-use programming language

- **DOM:** representation of structure of HTML page, which can be manipulated using JavaScript

- **jQuery:** library that simplifies accessing/using the DOM

What is React?

- JavaScript library for building user interfaces
- HTML page is composed of recyclable, interactive **'components'** that have a lifecycle during which the state of the component changes
- Highly efficient because of notion of **VirtualDOM**
- Created and maintained by Facebook
- Used in production by many well known companies

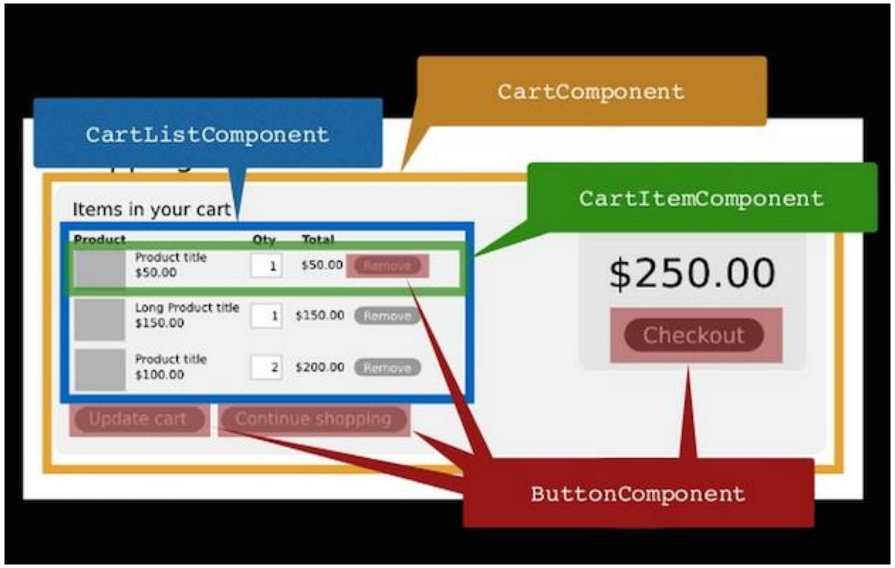
  - Netflix
  - WhatsApp, Instagram
  - Atlassian (BitBucket, HipChat, Jira)
  - Codecademy
  - Airbnb
  - Pinterest
  - Dropbox
  - PayPal
  - Reddit
  - Salesforce
  - Squarespace
  - New York Times
  - Treehouse
  - eBay
  - Trulia
  - Expedia
  - Visa
  - Wolfram Alpha

Why React?

- **Modularity:** organize code into reusable components that can work together
- **Lifecycle maintenance:** modifying component based on state; event listeners; simplified conditional rendering
- **JSX:** write HTML within JavaScript

Components

- Building blocks of React
- Make up the nodes included in the VirtualDOM
- Include and maintain a **state** that changes with events
- Each component maintains state independently
- Applications can be configured to respond to component level events



VirtualDOM

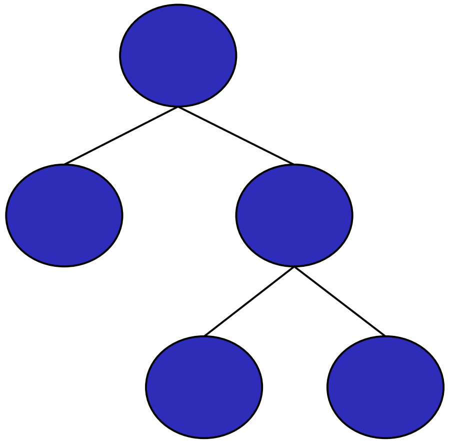
- Node tree that represents HTML elements, their attributes, and content as objects and properties
- **Selectively** renders and re-renders **subtrees** of nodes based on state changes
- Efficient because it does the least amount of DOM manipulation to update components
- Provides a layer of abstraction to the developer, providing simpler programming model and high performance

Normal DOM – How it Works

- When a node is updated, the browser updates (re-renders) **all** nodes

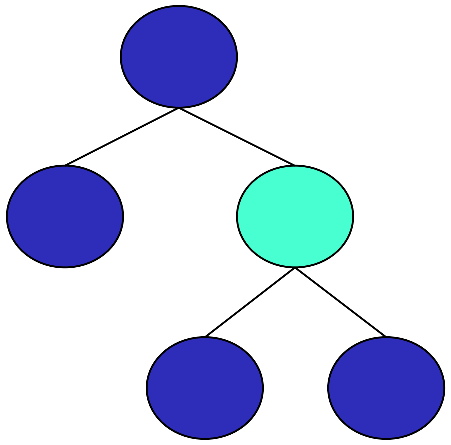
Normal DOM – How it Works

- When a node is updated, the browser updates (re-renders) **all** nodes



Normal DOM – How it Works

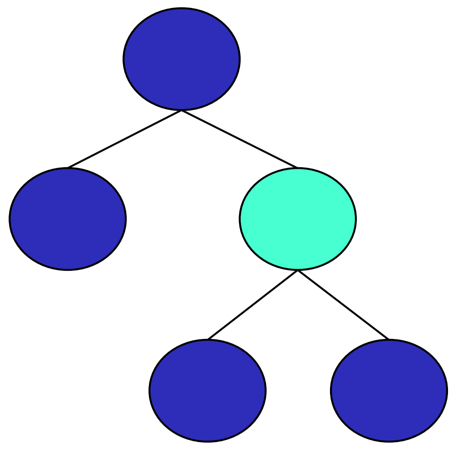
- When a node is updated, the browser updates (re-renders) **all** nodes

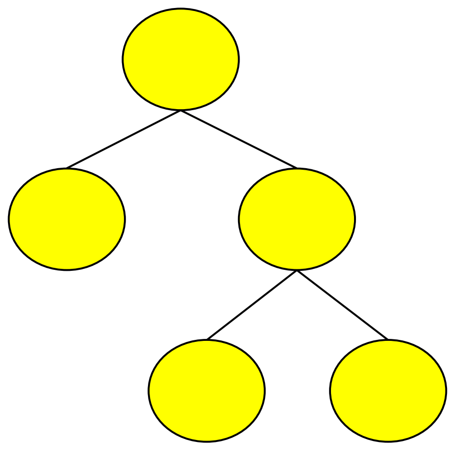


Change has been made to any given node

Normal DOM – How it Works

- When a node is updated, the browser updates (re-renders) **all** nodes


Change has been made to any given node

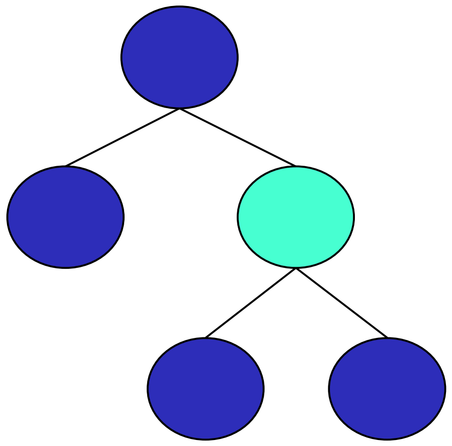

*Re-render **all** nodes to reflect the change*

VirtualDOM – How it Works

- When a node is updated, two things occur:
  - **‘diff’** to determine which nodes within DOM have changed
  - **‘reconciliation’** to update the nodes that are affected

VirtualDOM – How it Works

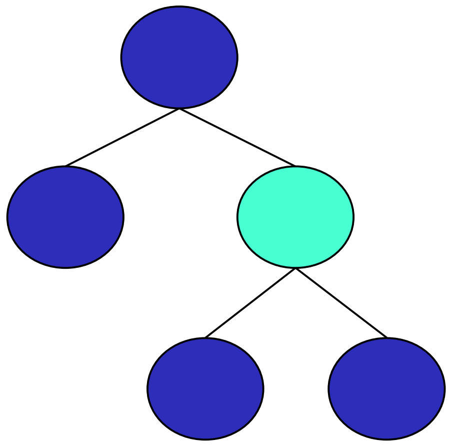
- When a node is updated, two things occur:
  - **‘diff’** to determine which nodes within DOM have changed
  - **‘reconciliation’** to update the nodes that are affected



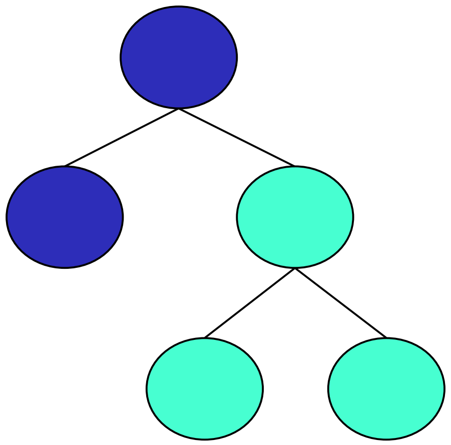
*Identify nodes that have changed (**‘diff’**)*

VirtualDOM – How it Works

- When a node is updated, two things occur:
  - **‘diff’** to determine which nodes within DOM have changed
  - **‘reconciliation’** to update the nodes that are affected



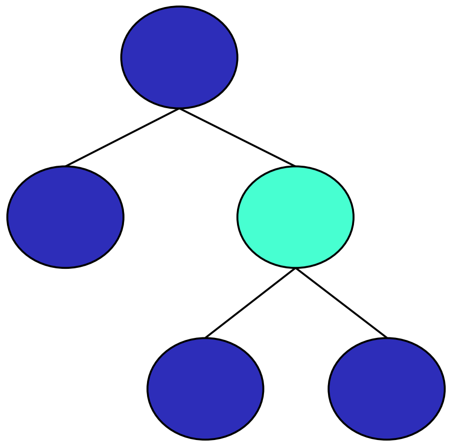
*Identify nodes that have changed (**‘diff’**)*



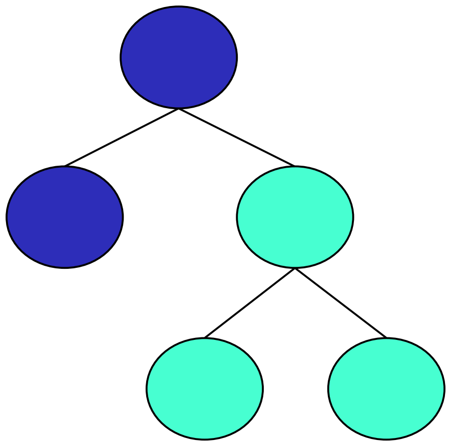
*Identify nodes that are affected by the change (**reconciliation**)*

VirtualDOM – How it Works

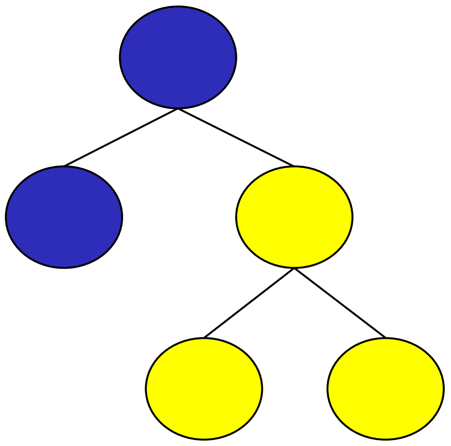
- When a node is updated, two things occur:
  - **'diff'** to determine which nodes within DOM have changed
  - **'reconciliation'** to update the nodes that are affected



*Identify nodes that have changed ('**diff**')*



*Identify nodes that are affected by the change (**reconciliation**)*



*Re-render **ONLY** the nodes that were affected by change*

Developing with React

1. Within the page's HTML, allocate a position on the page in which the desired React component will be rendered, e.g. a `div`

2. Create a React component in JavaScript
   - Establish an initial state
   - Define any events that could change the component's state over its lifecycle
   - Define the function to render the HTML

3. Drop the component into position allocated in Step 1

Getting Started

- Create a **div** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **div**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **div** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **div**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **div** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **div**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **div** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **div**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **div** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **div**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **`div`** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **`div`**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a `div` in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in `div`

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

Getting Started

- Create a **`div`** in the HTML to represent the location where the React component will be placed
- Write JavaScript code to create and display component in **`div`**

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id="container"></div>
      <script type="text/jsx">
         <!-- Insert React code here -->
      </script>
  </body>
</html>
```

JSX

- JSX – JavaScript XML Syntax Transform
- Allows user to write HTML-like tags within JavaScript
- Converts text (HTML) to React code

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Rendering Elements using JSX

```
<div id="container"></div>
<script type='text/jsx'>

  ReactDOM.render(
      <h1>Hello, World!</h1>,
      document.getElementById('container')
  );

</script>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
         ReactDOM.render(
            <h1> Hello, React! </h1>,
            document.getElementById('container')
         );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
         ReactDOM.render(
            <h1> Hello, React! </h1>,
            document.getElementById('container')
         );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
         ReactDOM.render(
            <h1> Hello, React! </h1>,
            document.getElementById('container')
         );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
         ReactDOM.render(
            <h1> Hello, React! </h1>,
            document.getElementById('container')
         );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
        ReactDOM.render(
          <h1> Hello, React! </h1>,
          document.getElementById('container')
        );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
        ReactDOM.render(
          <h1> Hello, React! </h1>,
          document.getElementById('container')
        );
      </script>
  </body>
</html>
```

Hello, React!

```
<!DOCTYPE html>
<html>
  <head>
      <title>ReactJS Example</title>
      <script src="react.js"></script>
      <script src="react-dom.js"></script>
  </head>
  <body>
      <div id='container'></div>
      <script type='text/jsx'>
         ReactDOM.render(
            <h1> Hello, React! </h1>,
            document.getElementById('container')
         );
      </script>
  </body>
</html>
```

Looking Ahead

- Defining React components
- Reacting to user events
- Interaction between React components
- Developing large applications with React

Looking Ahead

- Defining React components
- Reacting to user events
- Interaction between React components
- Developing large applications with React