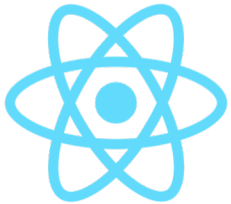


React & React Native

Fundamentals

avanade

©2019 Avanade Inc. All Rights Reserved.

Overview

- No experience with react is assumed

- My goal is for you to understand how everything works,
  so please ask questions anytime

- Bootcamp divided into 2 parts:
  - Part 1: React Fundamentals (can follow along in either web or native)
  - Part 2: React Native APIs

avanade

©2019 Avanade Inc. All Rights Reserved.

Part 1: Topics

- What is React?
  - What is it for?
  - How does it work?
  - React vs React Native
- Syntax and core constructs
  - Rendering and updating (no compilers)
  - Components & JSX
- General design principles
  - State management and updating
  - The Redux philosophy
- Development tools (native and/or web)
- Working with remote data

avanade

©2019 Avanade Inc. All Rights Reserved.

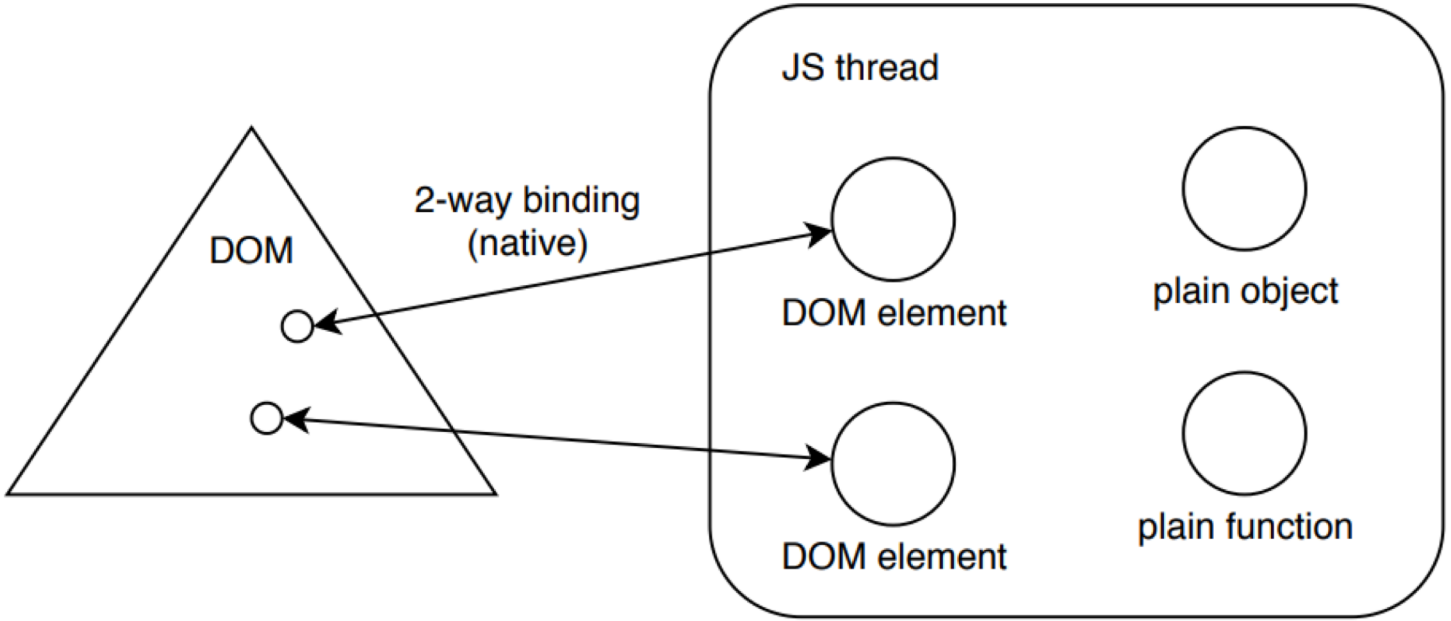
What is React?

- **Abstract Declarative UI Library**
  - Benefits
    - Portability
    - Efficient updating
      - Accessing native resources is expensive, use in-memory representation and take deltas
- Enforces a state management pattern that is simple and scalable
  - Benefits
    - Efficient updating
      - Only run reconciliation algorithm when relevant state has changed
    - Scalable pattern

avanade

©2019 Avanade Inc. All Rights Reserved.

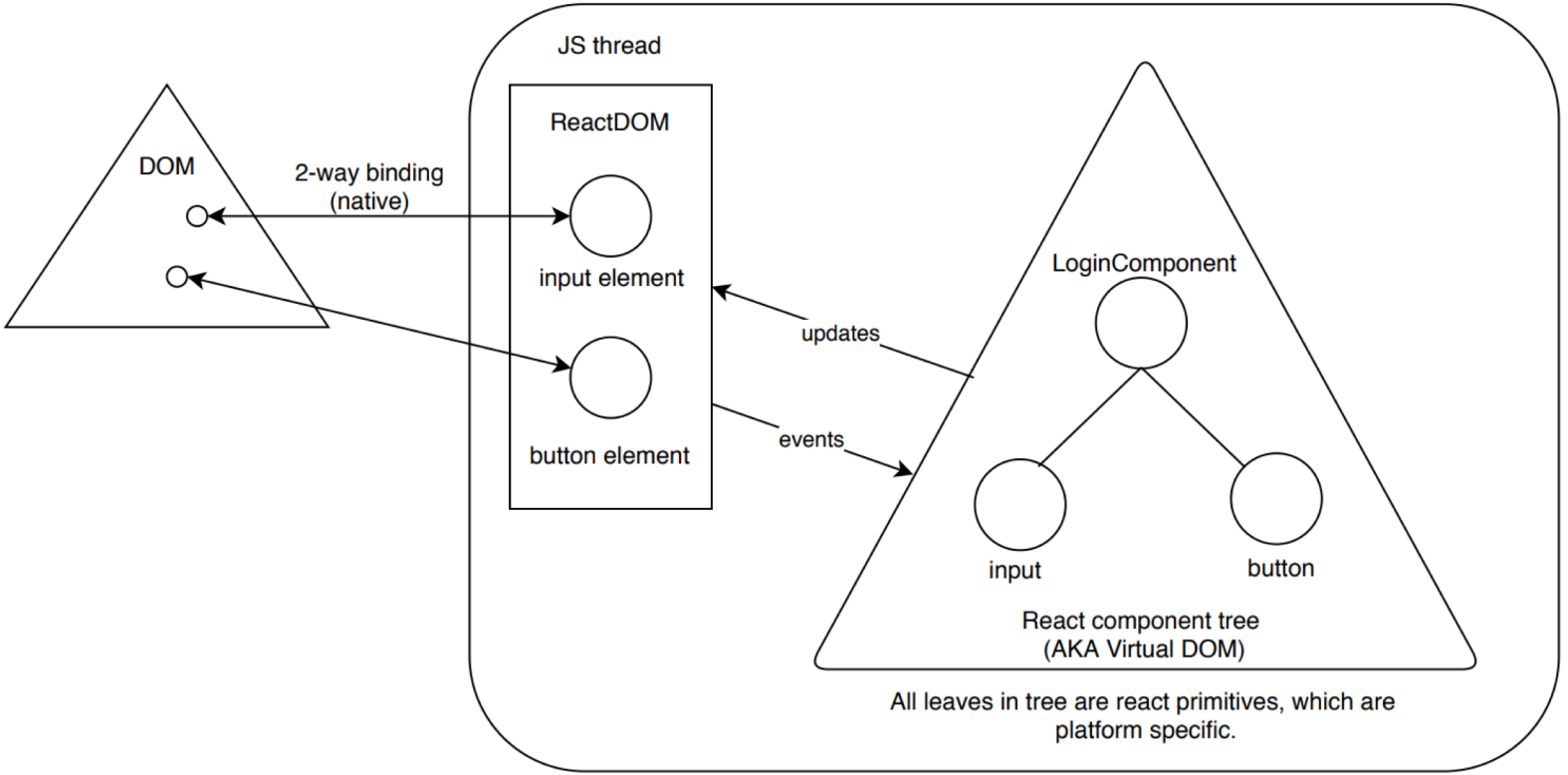
Freeform DOM manipulation



avanade

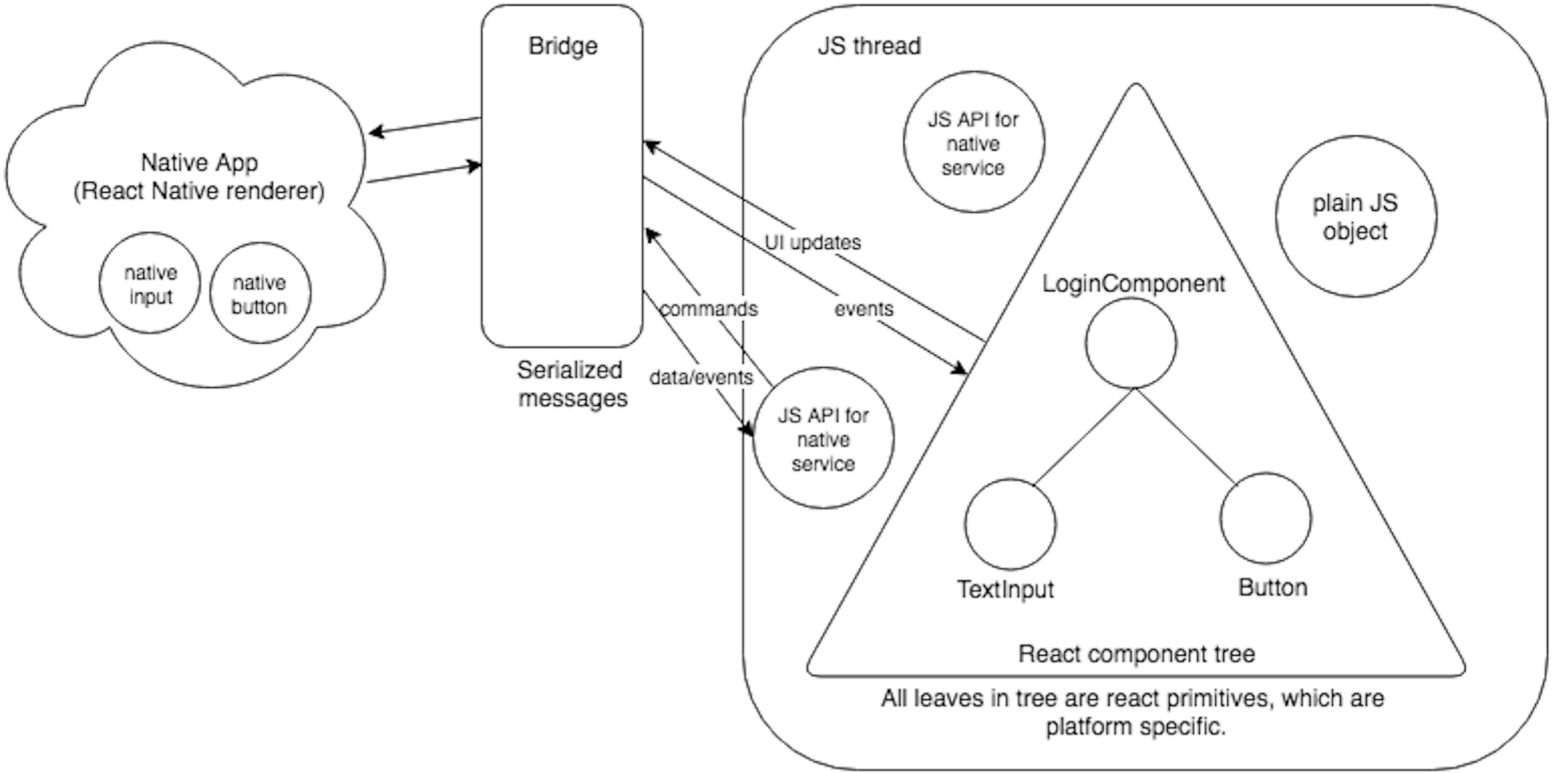
©2019 Avanade Inc. All Rights Reserved.

React web app



avanade
©2019 Avanade Inc. All Rights Reserved.

React native app



avanade

©2019 Avanade Inc. All Rights Reserved.

State management non-strategies

- Keep all state in native primitives
  - Slow updates
  - Updating more than necessary
  - Have to interact with native APIs to perform business logic

- Maintain state in memory freeform, update native primitives directly when necessary
  - getElementById() / jQuery / getViewById()
  - Too much code
  - Complicated, not scalable

avanade

©2019 Avanade Inc. All Rights Reserved.

Component pattern

- Each component only knows about its internal state and its direct children
- Components output abstract view representation of its current state upon request
  - You can remember the last abstract state of a component and compute the delta
- Update native primitives by providing this abstract representation to a renderer

avanade

©2019 Avanade Inc. All Rights Reserved.

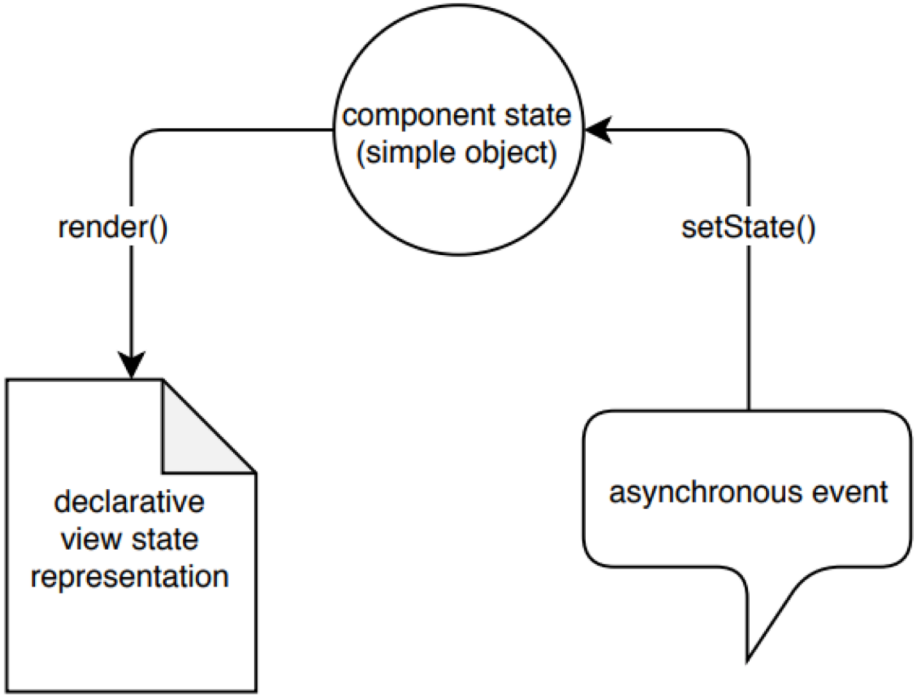
Updating strategies

- Manual flush
  - Responsibility on dev
    - Examples: Templating libraries, D3.js
- Angular
  - Every time anything happens, check everything for deltas
  - Allows for freeform state representation within component
- React
  - Keep all state in single plain object
  - Only update that object by calling a function provided by React which updates the object as requested, and then rerenders that component and its children

avanade

©2019 Avanade Inc. All Rights Reserved.

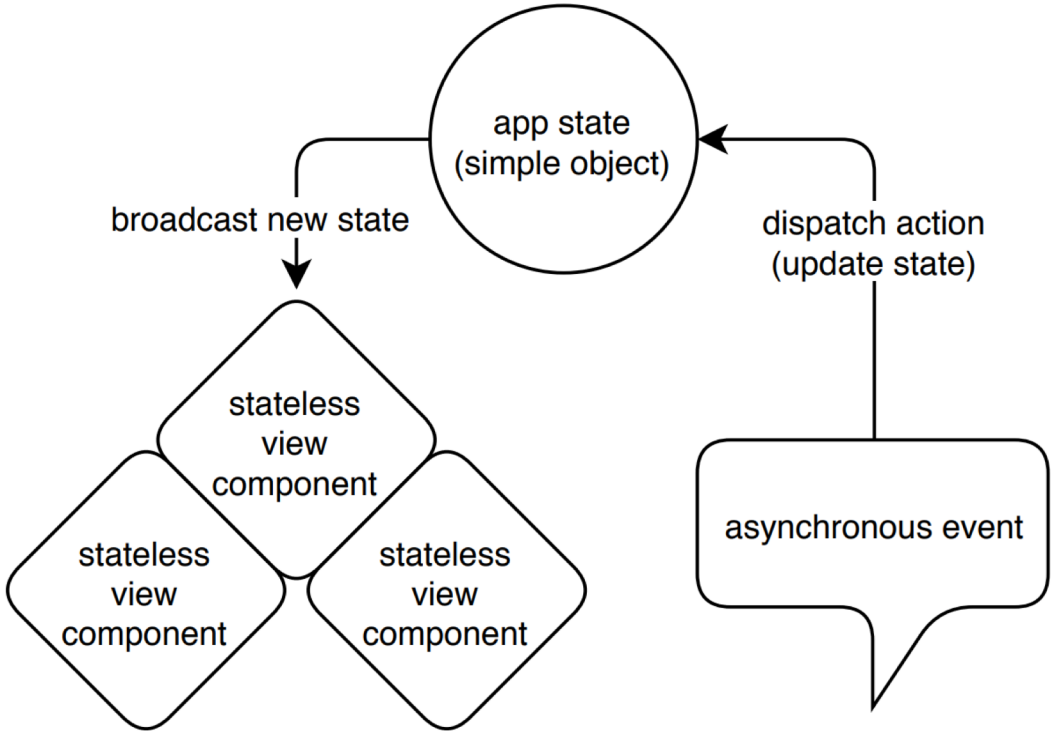
React.Component internal state management



avanade

©2019 Avanade Inc. All Rights Reserved.

Redux application state management



avanade

©2019 Avanade Inc. All Rights Reserved.

coding session #1

react-native-fun/1_bare-bones

avanade

©2019 Avanade Inc. All Rights Reserved.

coding session #2

react-native-fun/2_simple-app

avanade

©2019 Avanade Inc. All Rights Reserved.