An Introduction to Exponent

Hendrik Swanepoel

@hendrikswan www.tagtree.io

What You'll Learn in This Module

- Install
- Debugging
- Data Flow
- Generate a Project
- React Best Standards
- React Native Components

Why Use Exponent?

Less Scary

Environment setups are scary and prone to errors

Evaluation

Quickly evaluate React Native for your project's needs

One Environment

Avoid setting up an environment for each target platform

Summary

Installed the Exponent XDE app

Generated a new Exponent project

Installed the Exponent app on the simulator

Got our first app up and running on the simulator!

Summary

Installed the Exponent app on iPhone

Used LAN for hosting

Used the exp protocol to intercept links with exponent app

Sent link to phone to launch our app

Demo

Remove sample files

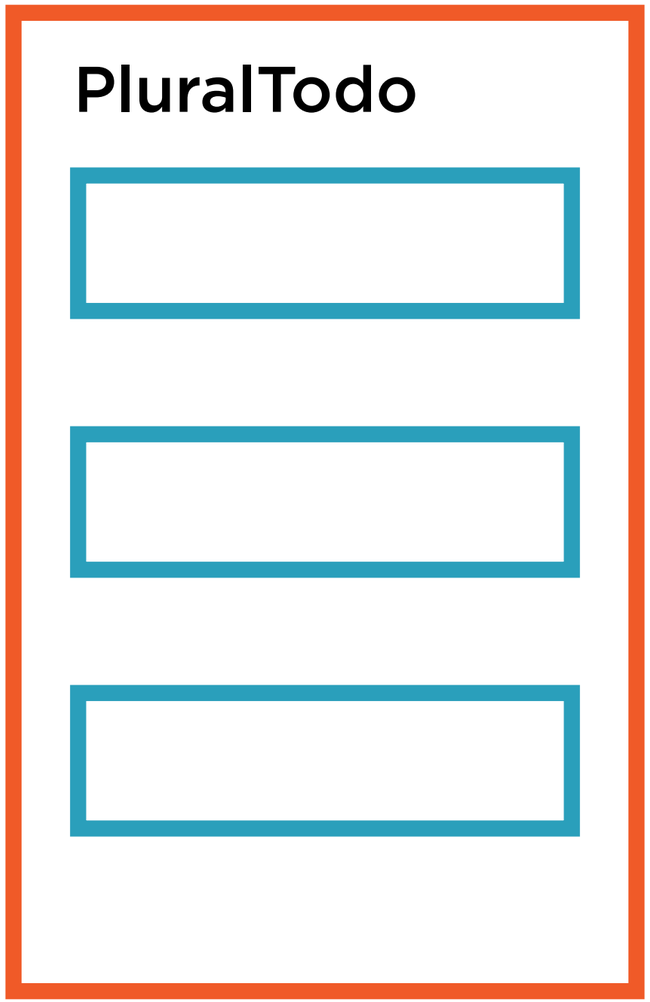
Render a Text element instead

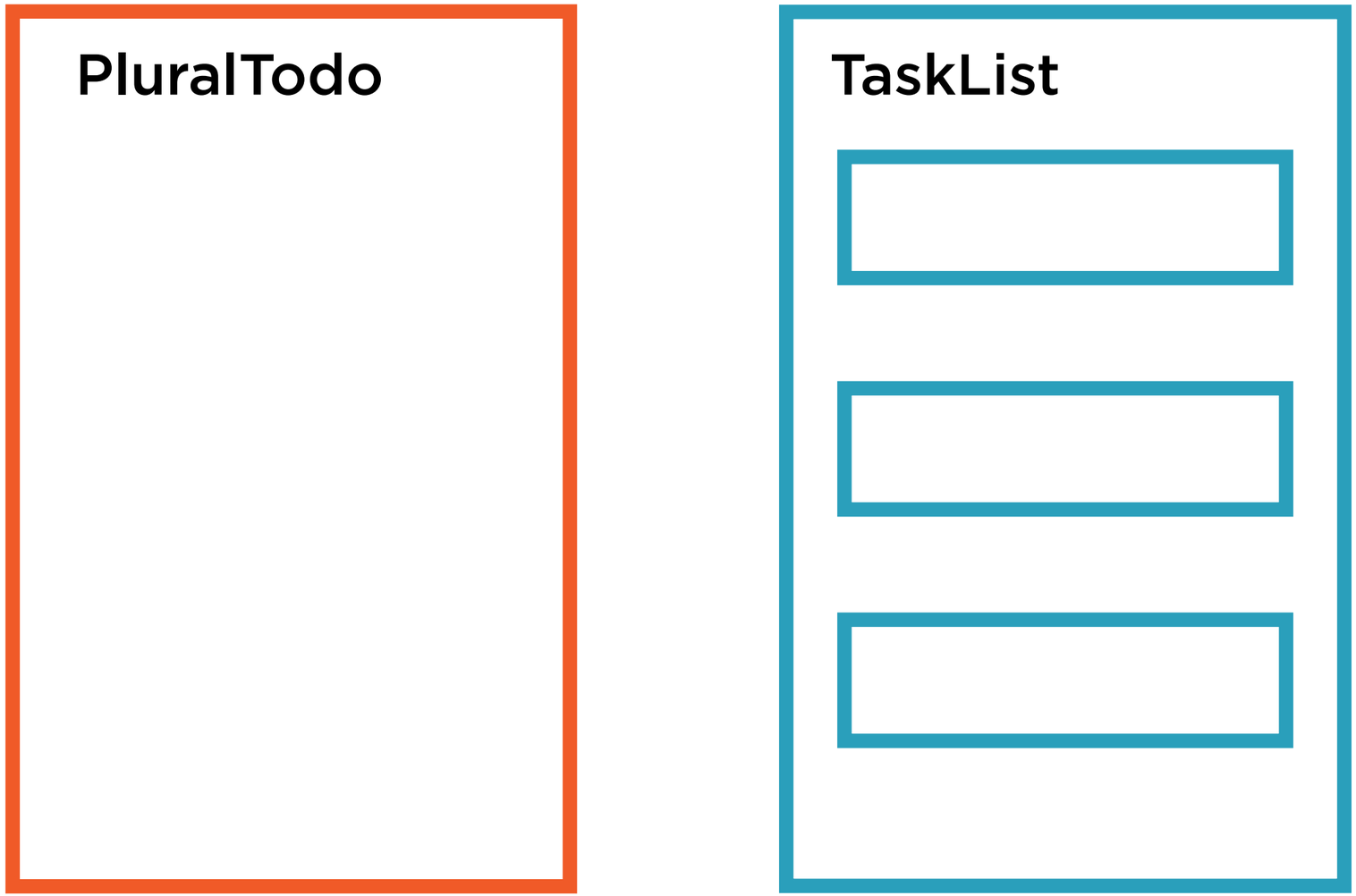
Summary

Removed generated files

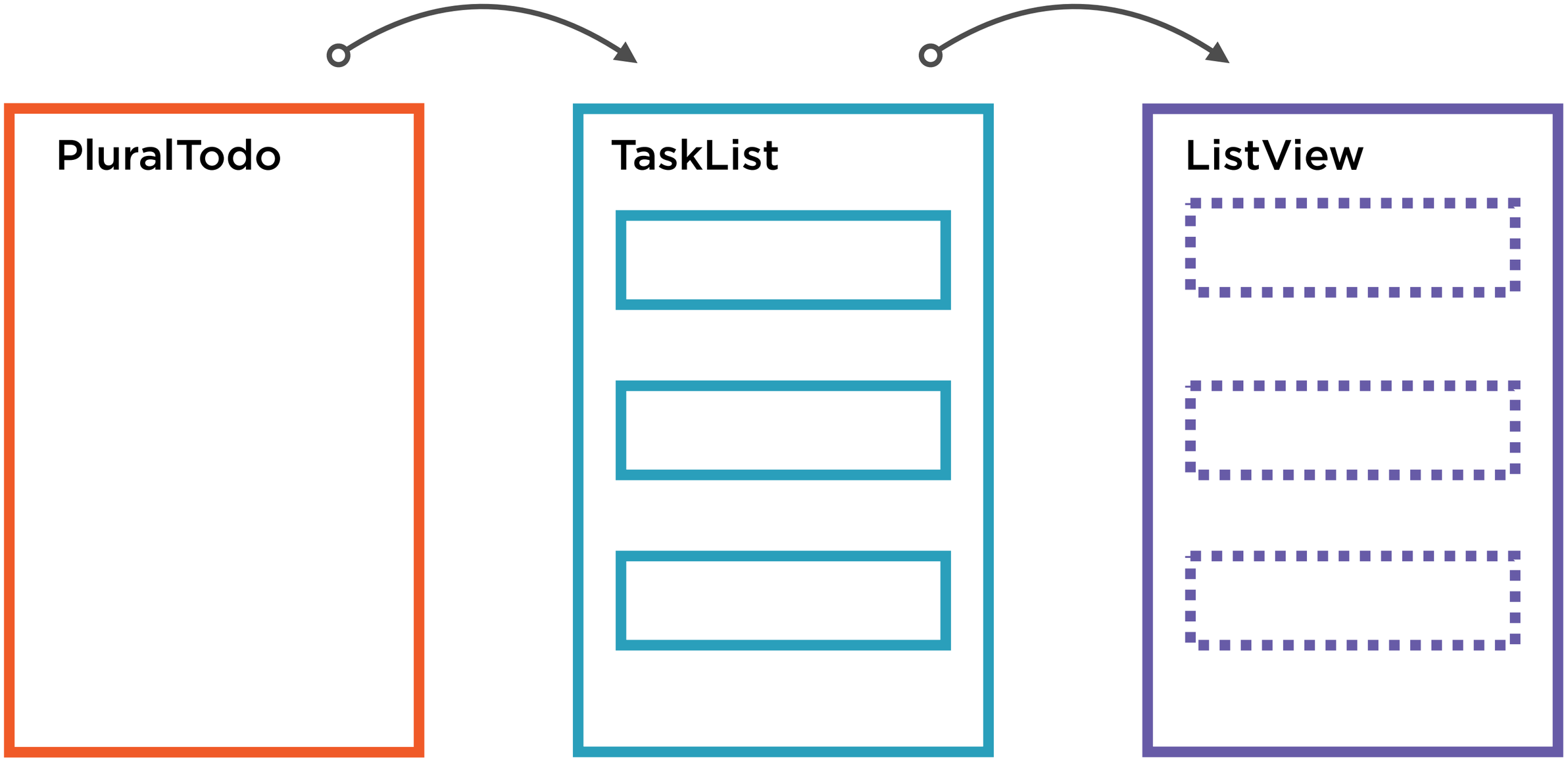
Kept main.js, but cleaned it out

Renamed main component to PluralTodo


PluralTodo


PluralTodo
TaskList


PluralTodo
TaskList
ListView


PluralTodo
TaskList
ListView

Container vs Presentation Components

Container Components

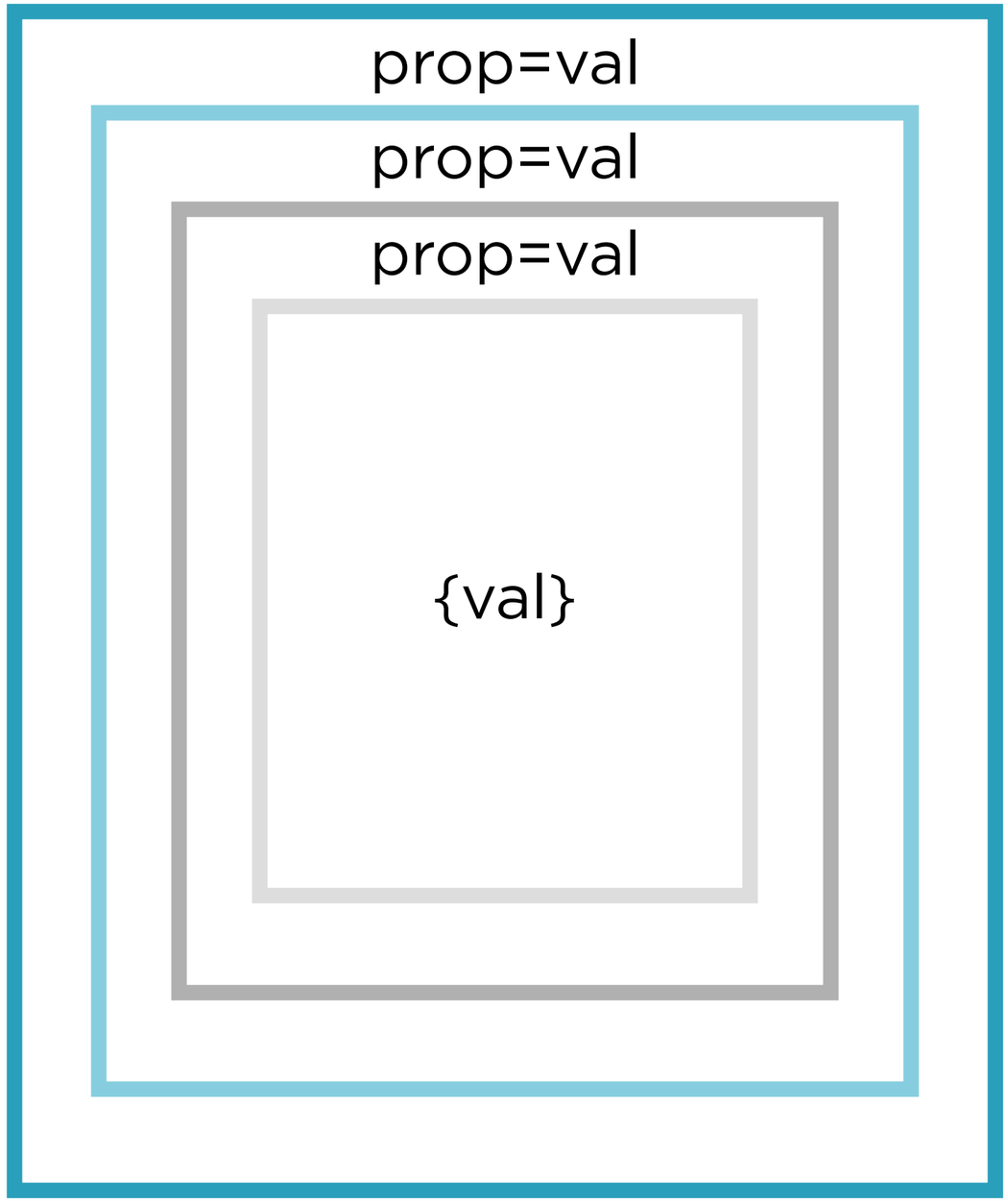
Can make changes to state

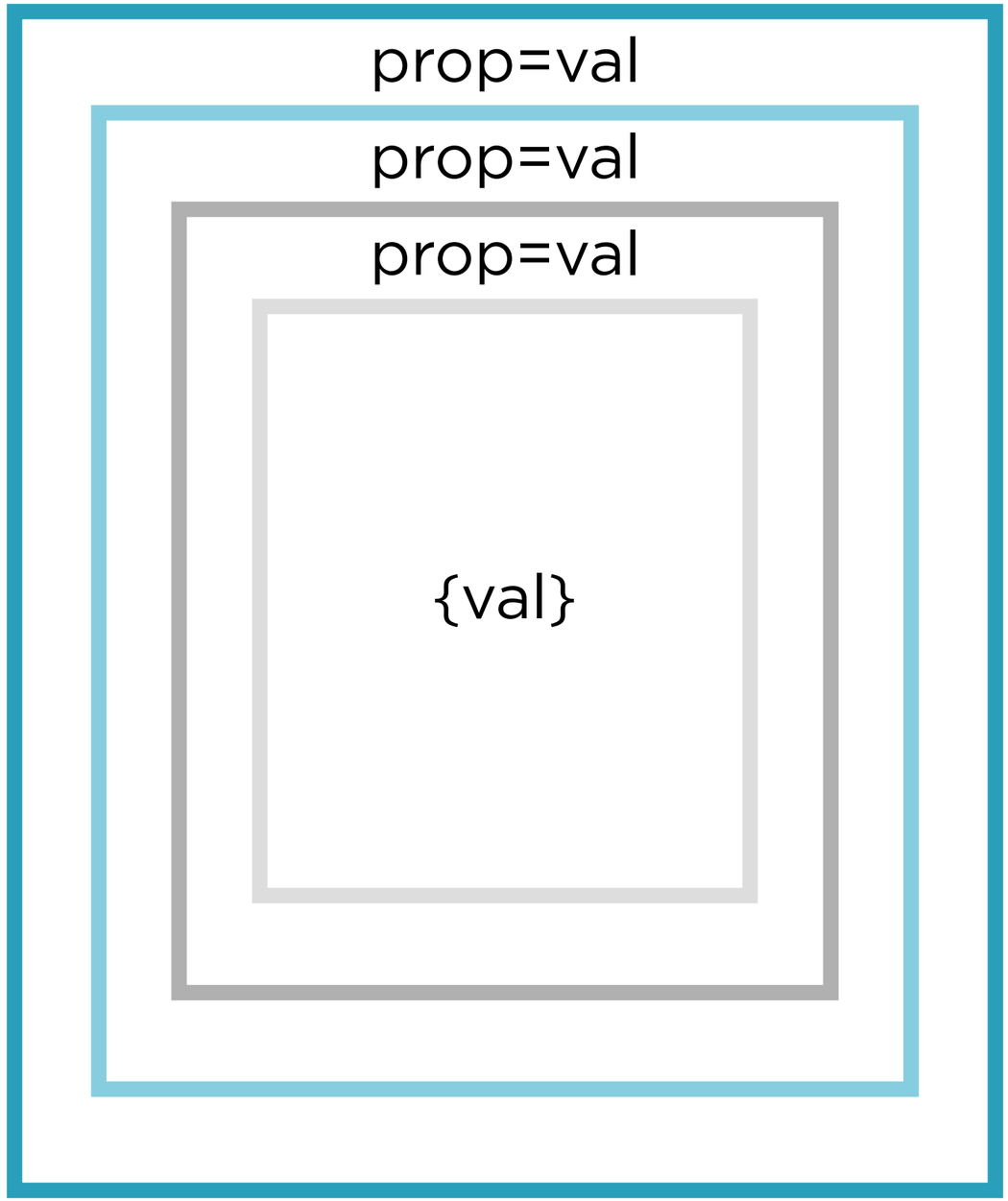
Pass functions and data to Presentation components through props

Presentation Components

Receive data only through props

Calls back on functions provided through props


prop=val
prop=val
prop=val
{val}




```
{
  prop=val
}
```

Demo

Add a list of todos to the PluralTodo component

The PluralTodo component will be our only container component

Create the TaskList component

Render TaskList in PluralTodo

Style the TaskList component

Summary

Add a list of todos to the PluralTodo component

Create the TaskList component

Render TaskList in PluralTodo

Style the TaskList component

Demo

Use todos array in TaskList component

Add validation for todos prop on TaskList

Use ListView to show items in array

Populate a datasource

Summary

Add prop validation for todos prop in TaskList

Create and populate datasource

Used cloneWithRows to populate datasource

Provided rowHasChanged function to datasource

Provided a renderRow prop on ListView

Demo

Extract a new component (TaskRow)

Add prop validation to expect a todo

Use the shape prop type to validate attributes on todo object

Add styling to TaskRow component

Summary

Extract granular components to have simple components

Use shape prop type to validate for attributes on an object prop

Pass state to new component (TaskRow) through parent component (TaskList)

Use Flexbox for styling with flow layout

Demo

Create a button and wire it up

Communicate via events with the Container component

Use the TouchableHighlight component to build a button

Demo

Create a button and wire it up

Communicate via events with the Container component

Use the TouchableHighlight component to build a button

Attach the Chrome debug tools

Summary

Use Text and TouchableHighlight to build buttons

Use onPress on TouchableHighlight

Added prop validation for onAddStarted

Use CTRL+CMD+Z to launch the developer tools

Demo

Use the Navigator to switch between views

Configure possible views with routes

Show a different view by calling the push function on the Navigator

Configure how the views transition

Summary

Use the cross platform Navigator component to switch between views

Use the renderScene prop to configure your routes

Use refs to have a programmatic reference to a JSX created element

Use .push when you want to transition to a different route

Use the configureScene prop to configure how the navigator transitions to a different route

Demo

Creating the TaskForm component

An approach to rapid layout and styling

Using TextInput, TouchableHighlight, and Text components

Summary

If possible, make your new view the default route of the navigator

Define the basic layout with associated style rules, then flesh out style with auto reload for super quick feedback

Demo

Wire up the TaskForm component with behavior

Use .pop() to close the TaskForm and go back to the TaskList

Pass the todo to the container component

Add the todo to state and rerender

Summary

Try specifying prop validation before using props

Debug tools show warnings for failed prop validation

Use .pop() to go back to the previous route in the Navigator

Use onChange of TextInput to track changes to it's value

Use prop functions to communicate up to the container component

Use componentWillReceiveProps when you want a ListView to rebind in a presentation component

Demo

Add a button to each todo record in list

Through props bubble up the done todo

Remove the affected todo from state in PluralTodo (container component)

Summary

Created and styled button in TaskRow component

Used prop validation to validated onAdd prop in TaskList and TaskRow

Use onPress event in TaskRow and bubbled up to PluralTodo (container)

Filter out the todo from the todos in container before calling setState

Summary

- Use Exponent to check out React Native with minimal environment setup
- Use prop validation to enforce that the correct props are set on your components
- Use the Navigator component for cross platform navigation
- Try to contain state permutation to container components
- Presentation components get all dependencies through props

Full Environment Setup

Xcode

Android SDK

Redux