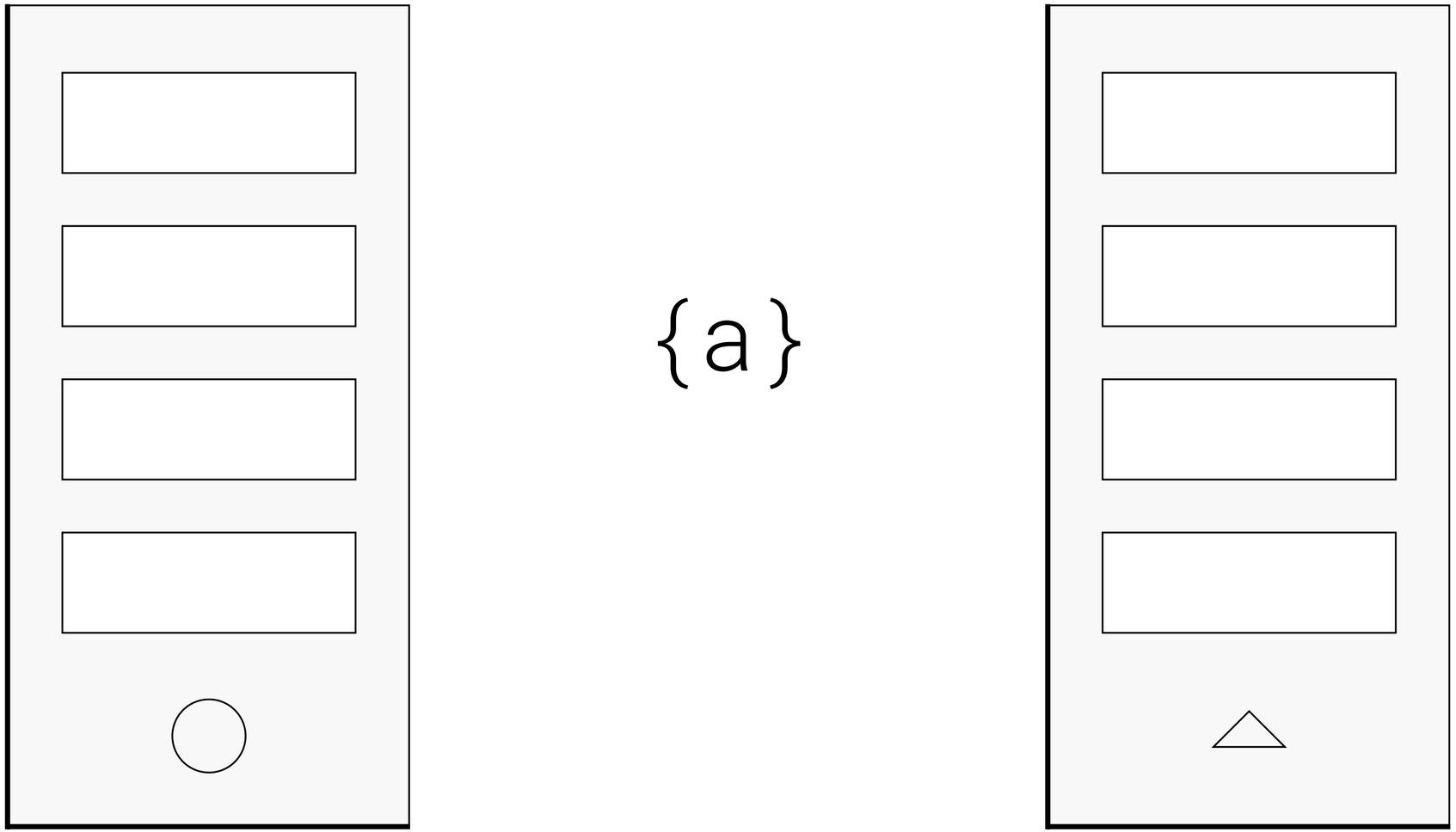
Platform Targeting

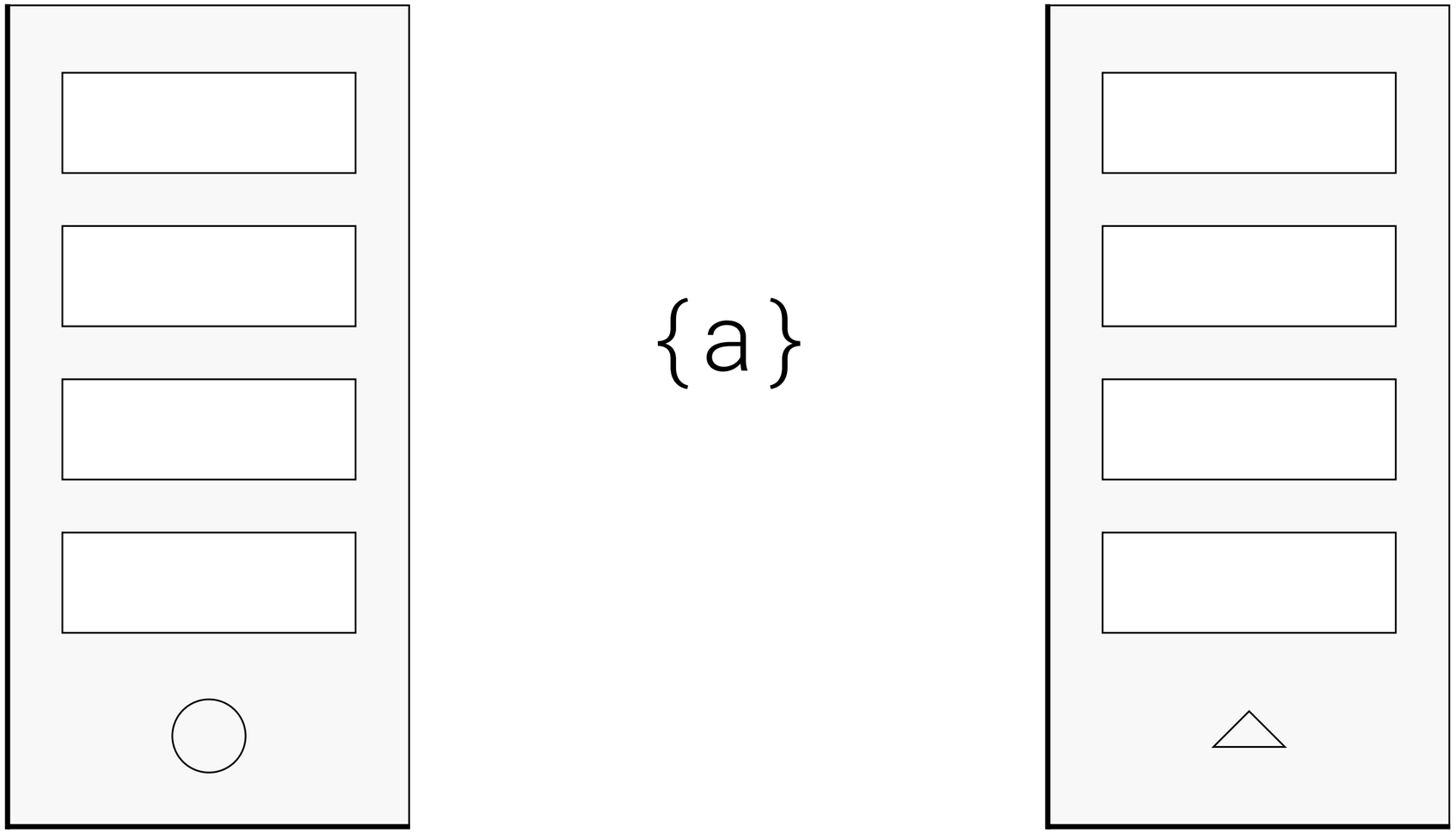
Hendrik Swanepoel

@hendrikswan www.tagtree.io

Write Once Run Anywhere

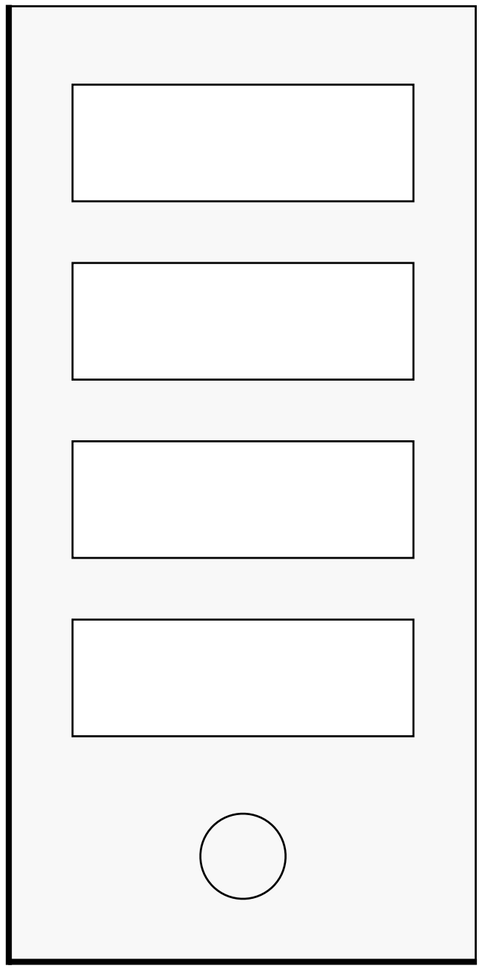


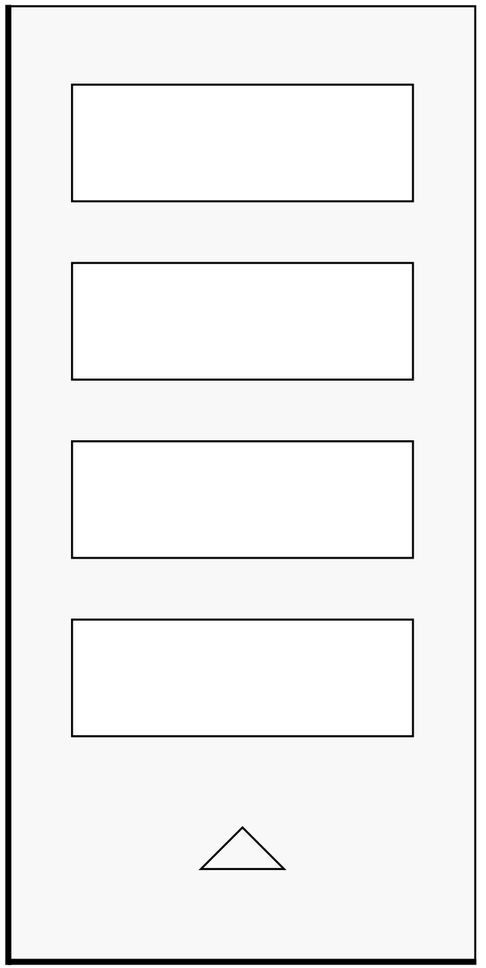
~~Write Once Run Anywhere~~



Learn Once Write Anywhere

{a}

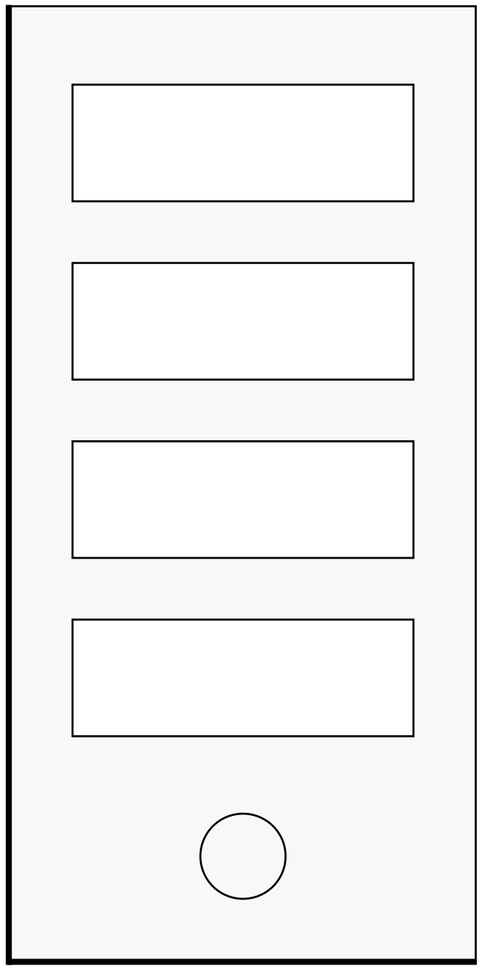


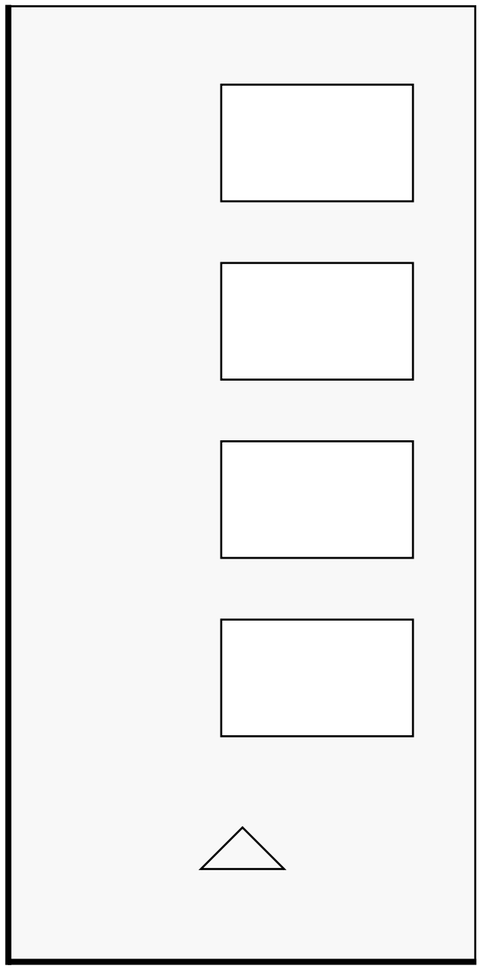


{b}

Learn Once Write Anywhere

{a}

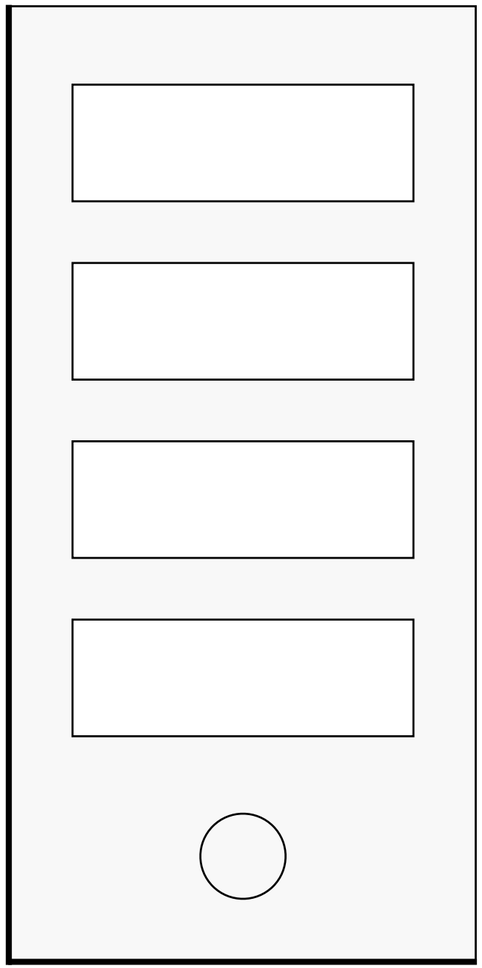


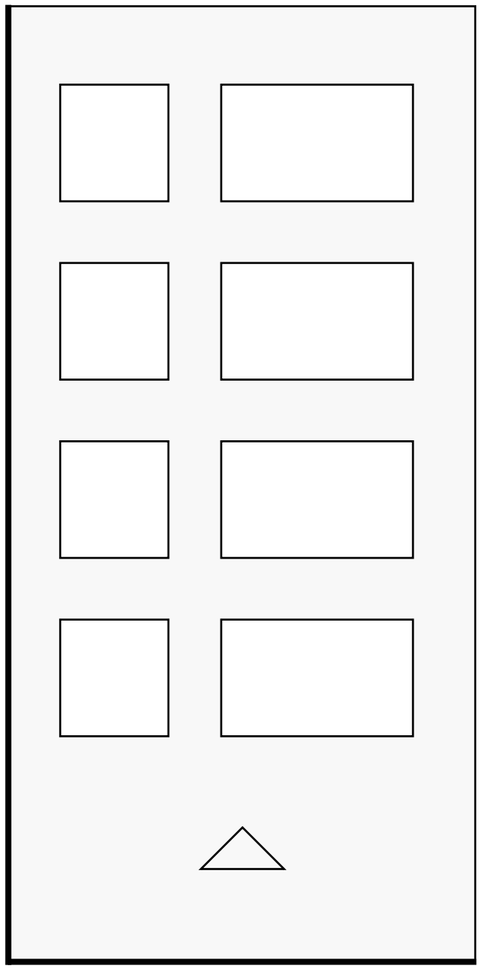


{b}

Learn Once Write Anywhere

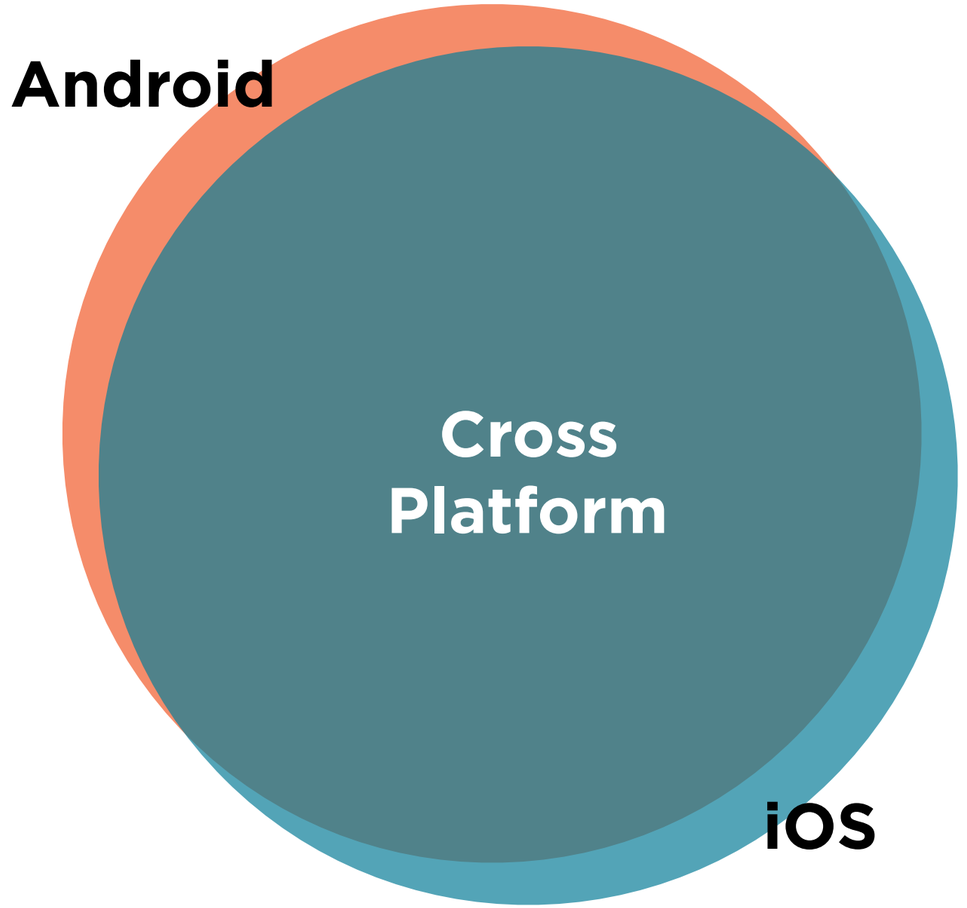
{a}





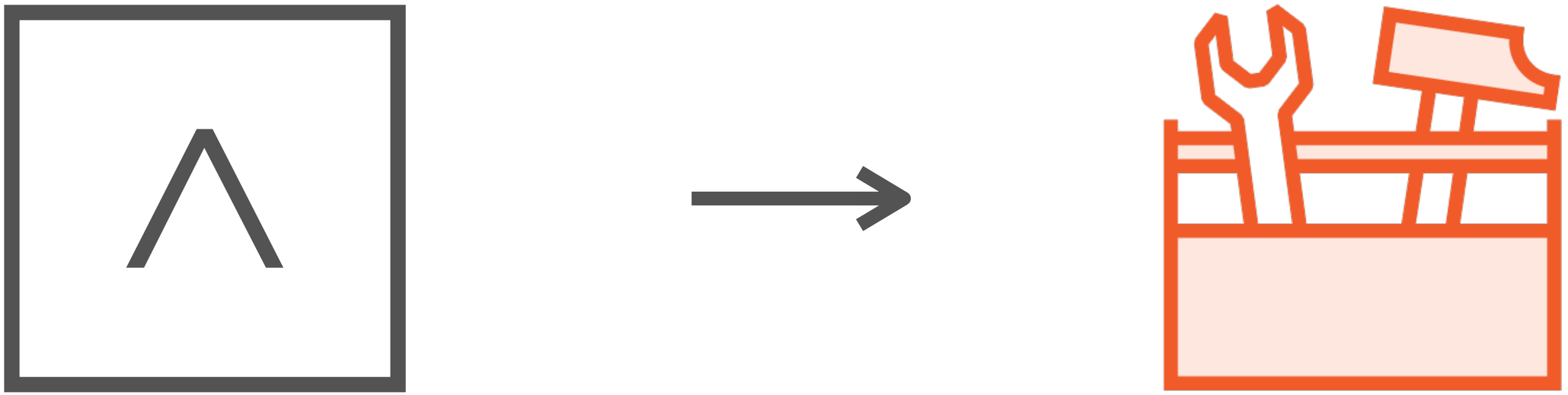
{b}

Code Reuse



Full React Native Project

Full React Native Project



Demo

Setting up your environment for full-blown React Native development

```
source ~/.bash_profile
```

Reloading Environment Variables

```
touch ~/.bash_profile
```

Creating an Environment File

Demo

Generate a new React Native project

Start the project on both iOS and Android environments

Copy app structure from Exponent project into a full-blown project

Get app to run in full-blown environment on iOS and Android

Summary

react-native init FullPluralTodo

react-native run-ios

react-native run-android

Easy to copy from an Exponent project into a full-blown React Native project

Demo

Enable linter for our new project structure

Refactor TaskRow component to get render logic from platform specific files

Summary

Installed linter modules for our new project to get linter working again

Moved TaskRow.js to TaskRow folder and renamed TaskRow.js to Component.js

Extracted render function to platform specific files (Render.ios.js and Render.android.js)

Import Render without specifying platform

Demo

Install and use custom native module for iOS swipe out effect

Reuse style from base component, but adapt style for iOS row

Summary

Installed react-native-swipeout for swipe out effect

Configured buttons for swipeout component

Combined style from base component with local iOS specific style

Demo

Adding an image to the TaskRow component on Android

Summary

Used require with a relative path to set the source on the Image component

Made all the changes in the Render.android.js file, to target the Android platform

Demo

Use animation to add an effect when the Todo is marked as done

Summary

Referenced React.Animated to set up animation

Prefix view with Animated. to allow it to be targeted by animation

Created a new animation using Animated.ValueXY

Used a transform style rule to use the animation

Used Animated.spring to start the animation

Recap

Learn once, write anywhere

Default to using cross platform components, and platform targeting only where you need it

React Native hooks into require, using a convention of x.ios.js and x.android.js to find the correct file for the platform

Still have one component, but get rendering logic from platform specific files

Redux