Glossary

Dev Menu

The in-app developer menu (available in development builds) that offers access to various development and debugging actions. Learn more about the Dev Menu in the docs.

Fabric Renderer

React Native executes the same React framework code as React for the web. However, React Native renders to general platform views (host views) instead of DOM nodes (which can be considered web's host views). Rendering to host views is made possible by the Fabric Renderer. Fabric lets React talk to each platform and manage its host view instances. The Fabric Renderer exists in JavaScript and targets interfaces made available by C++ code. Read more about React renderers in this blog post.

Host platform

The platform embedding React Native (e.g., Android, iOS, macOS, Windows).

Host View Tree (and Host View)

Tree representation of views in the host platform (e.g. Android, iOS). On Android, the host views are instances of `android.view.ViewGroup`, `android.widget.TextView`, etc. which are the building blocks of the host view tree. The size and location of each host view are based on `LayoutMetrics` calculated with Yoga, and the style and content of each host view are based on information from the React Shadow Tree.

JavaScript Interfaces (JSI)

A lightweight API to embed a JavaScript engine in a C++ application. Fabric uses it to communicate between Fabric's C++ core and React.

Java Native Interface (JNI)

An API to write Java native methods used to communicate between Fabric's C++ core and Android, written in Java.

React Component

A JavaScript function or class that instructs how to create a React Element. Read more about React components, elements in this blog post.

React Composite Components

React Components with `render` implementations that reduce to other React Composite Components or React Host Components.

React Host Components or Host Components

React Components whose view implementation is provided by a host view (e.g., `<View>`, `<Text>`). On the Web, ReactDOM's Host components would be components like `<p>` and `<div>`.

React Element Tree (and React Element)

A *React Element Tree* is created by React in JavaScript and consists of React Elements. A *React Element* is a plain JavaScript object that describes what should appear on the screen. It includes props, styles, and children. React Elements only exist in JavaScript and can represent instantiations of either React Composite Components or React Host Components. Read more about React components and elements in this blog post.

React Shadow Tree (and React Shadow Node)

A *React Shadow Tree* is created by the Fabric Renderer and consists of React Shadow Nodes. A React Shadow Node is an object that represents a React Host Component to be mounted, and contains props that originate from JavaScript. They also contain layout information (x, y, width, height). In Fabric, React Shadow Node objects exist in C++. Before Fabric, these existed in the mobile runtime heap (e.g. Android JVM).

Yoga Tree (and Yoga Node)

The *Yoga Tree* is used by Yoga to calculate layout information for a React Shadow Tree. Each React Shadow Node typically creates a *Yoga Node* because React Native employs Yoga to calculate layout. However, this is not a hard requirement. Fabric can also create React Shadow Nodes that do not use Yoga; the implementation of each React Shadow Node determines how to calculate layout.

*Last updated on **Jul 28, 2023***