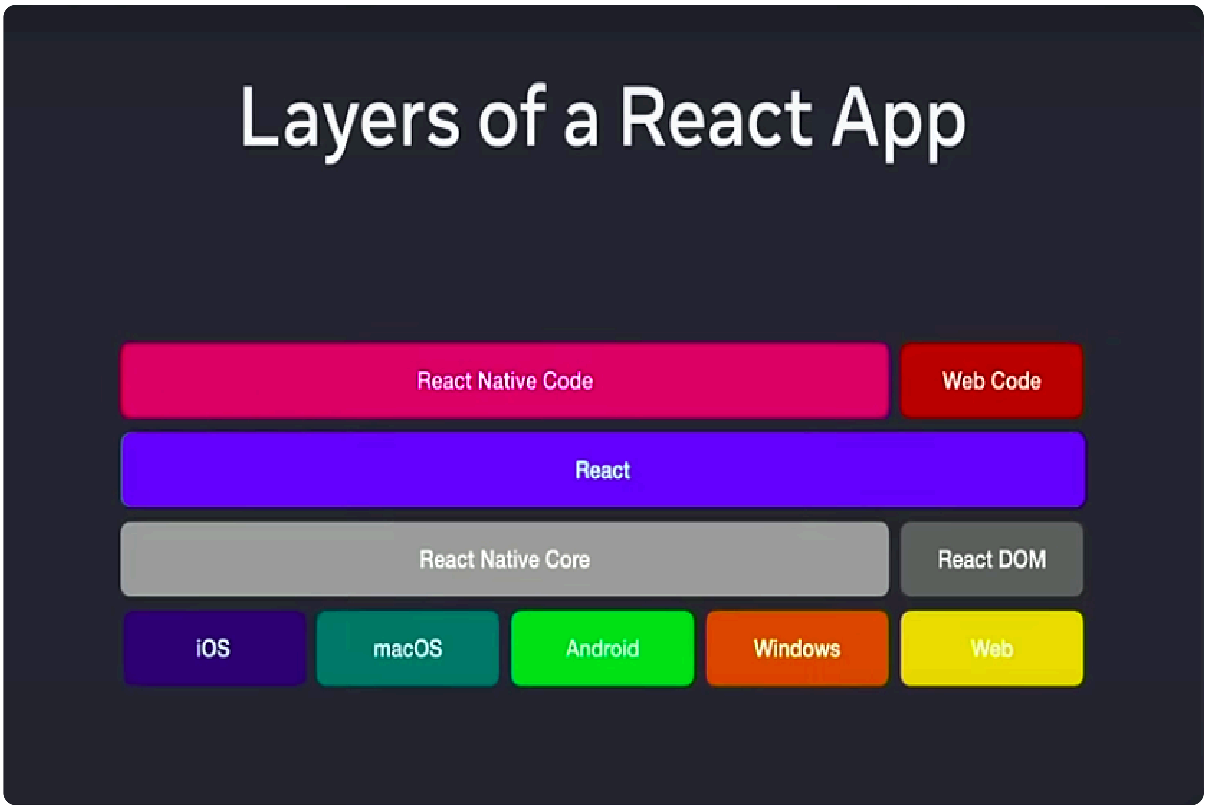
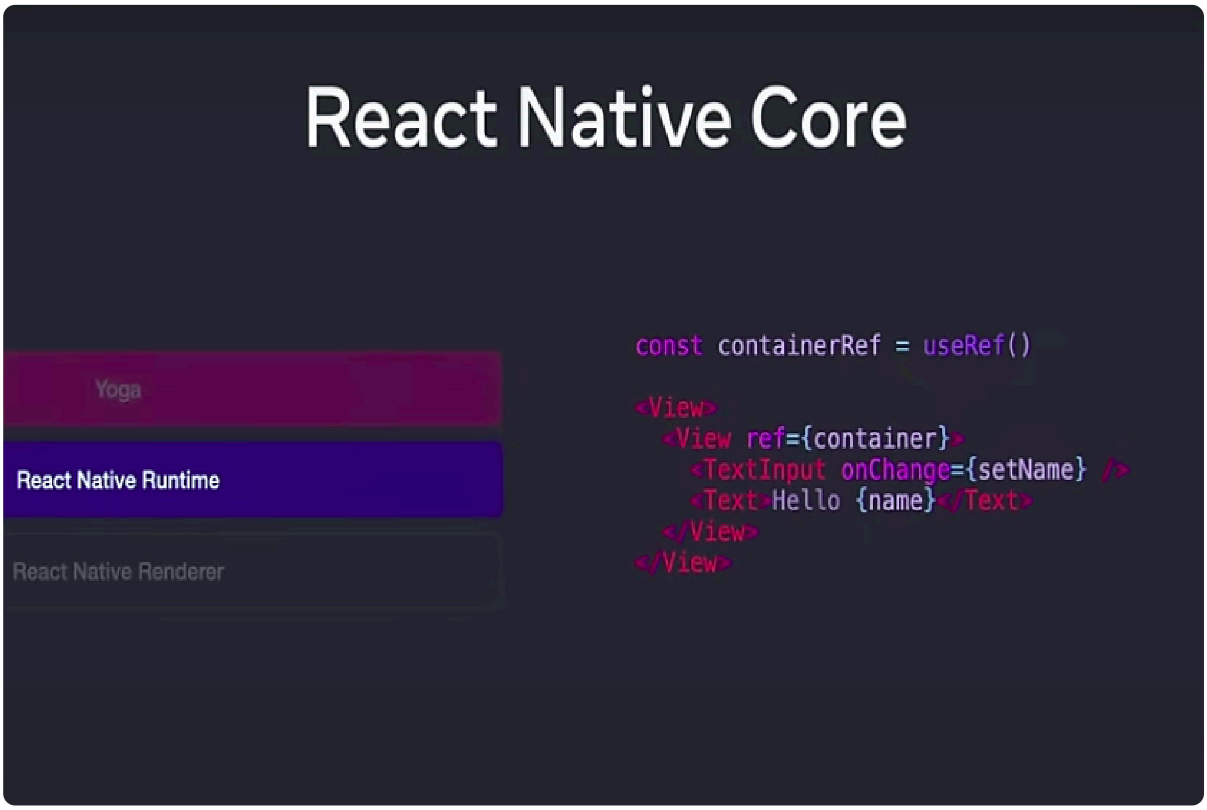
📑 Cross Platform React

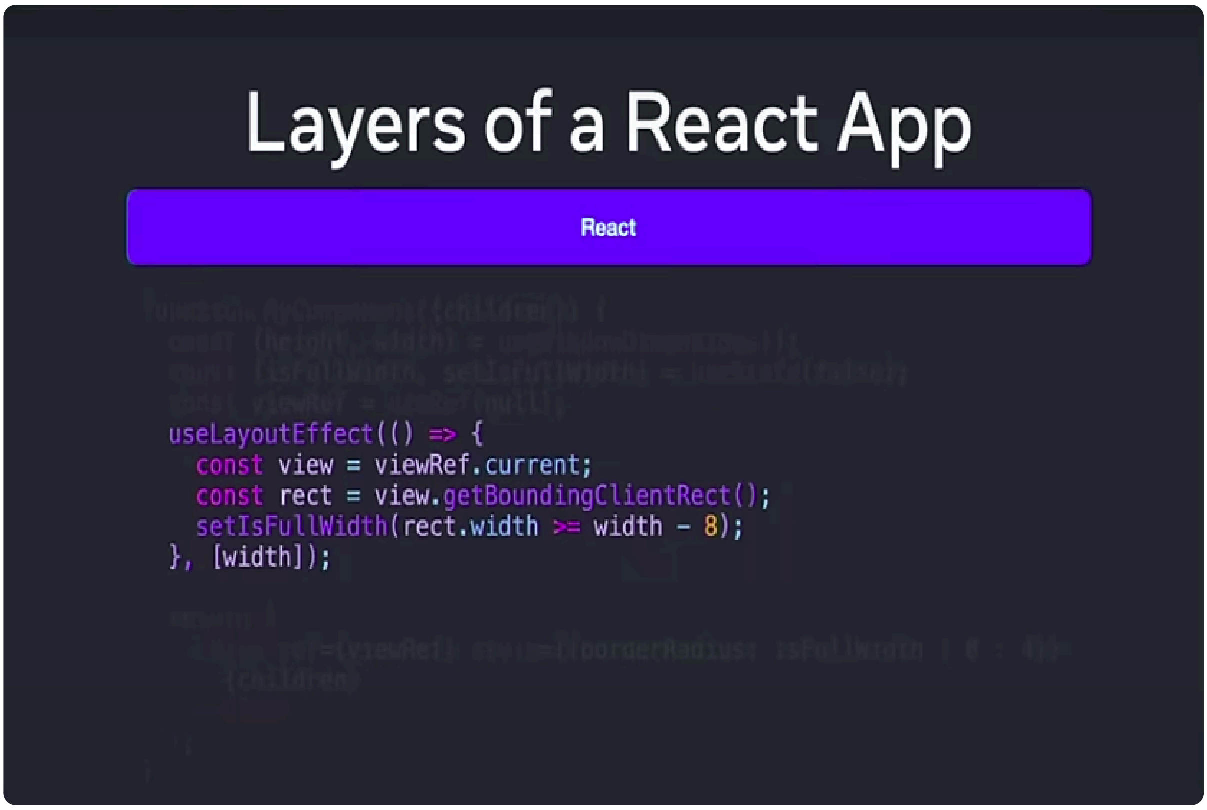


Layers of a React App



React Native Core

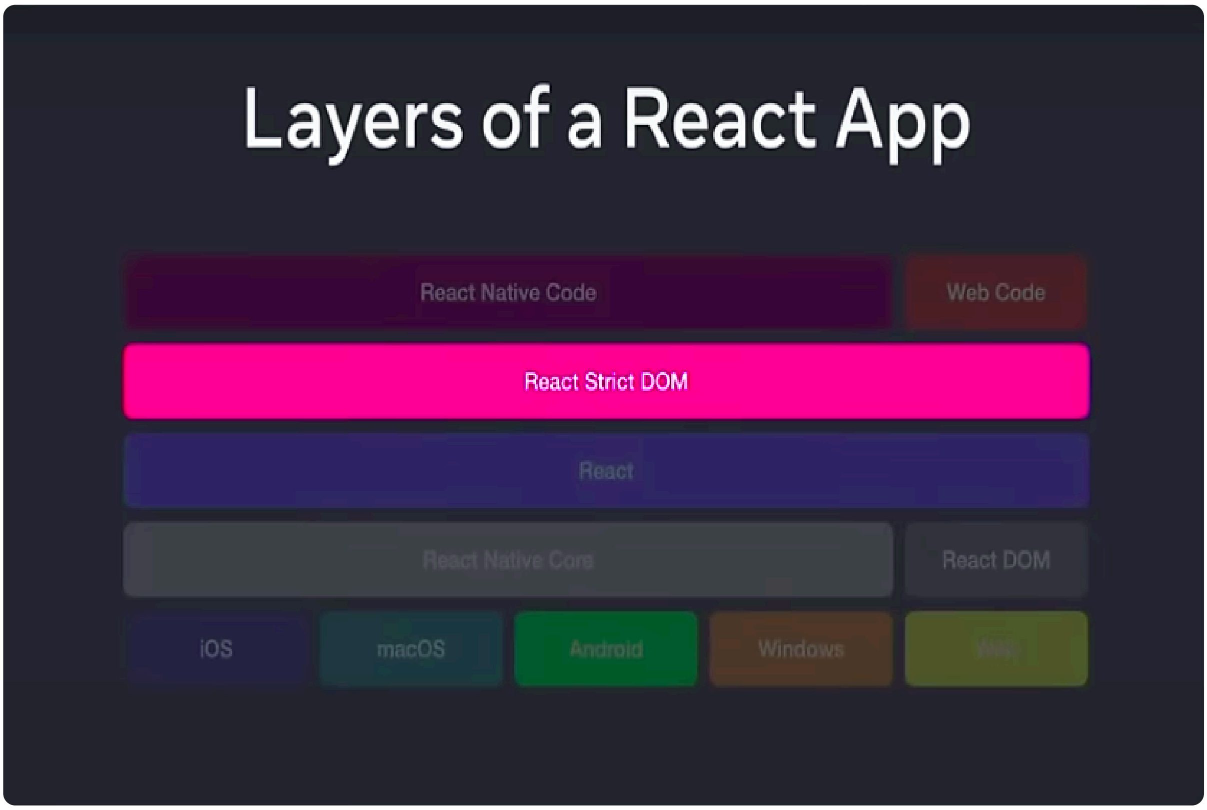
```
const containerRef = useRef()

<View>
  <View ref={container}>
    <TextInput onChange={setName} />
    <Text>Hello {name}</Text>
  </View>
</View>
```



Layers of a React App

React

```
function MyComponent({children}) {
  const {height, width} = useWindowDimensions();
  const [isFullWidth, setIsFullWidth] = useState(false);
  const viewRef = useRef(null);

  useLayoutEffect(() => {
    const view = viewRef.current;
    const rect = view.getBoundingClientRect();
    setIsFullWidth(rect.width >= width - 8);
  }, [width]);

  return (
    <View ref={viewRef} style={{borderRadius: isFullWidth ? 0 : 8}}>
      {children}
    </View>
  );
}
```



Layers of a React App

React Native Code

Web Code

React Strict DOM

React

React Native Core

React DOM

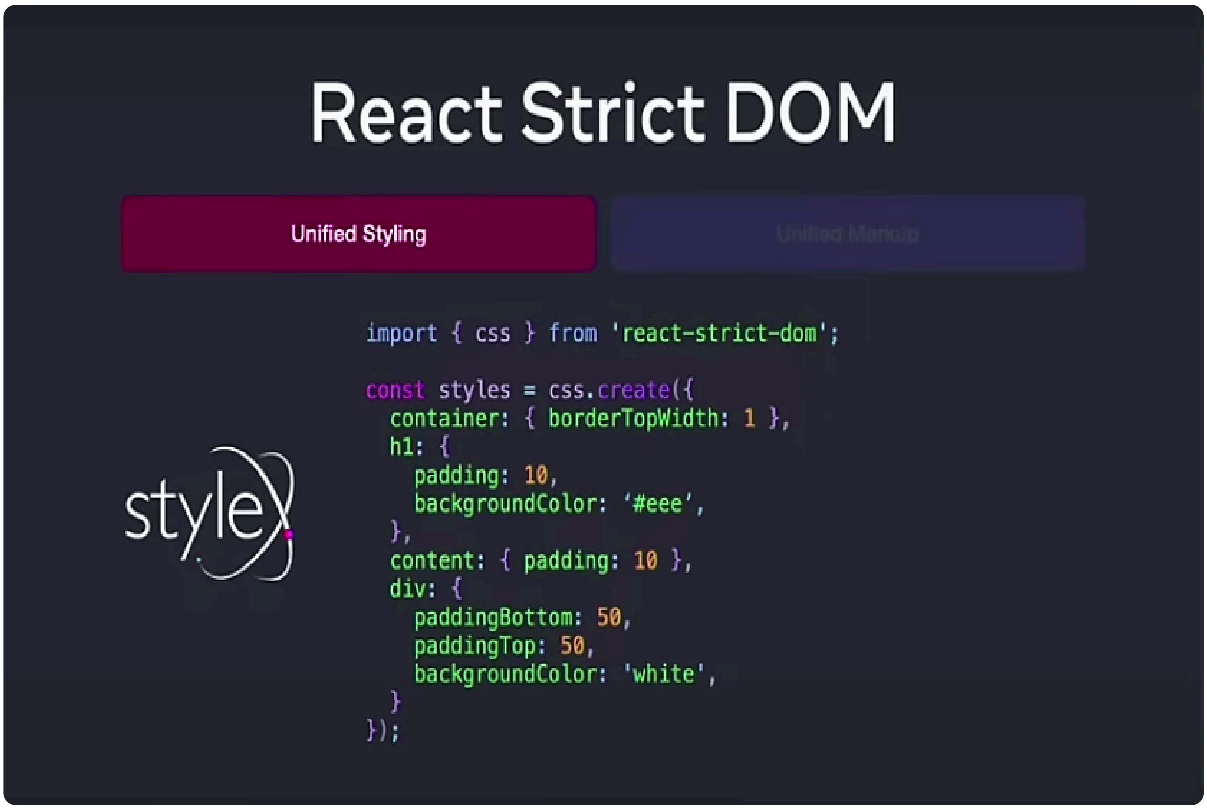
iOS

macOS

Android

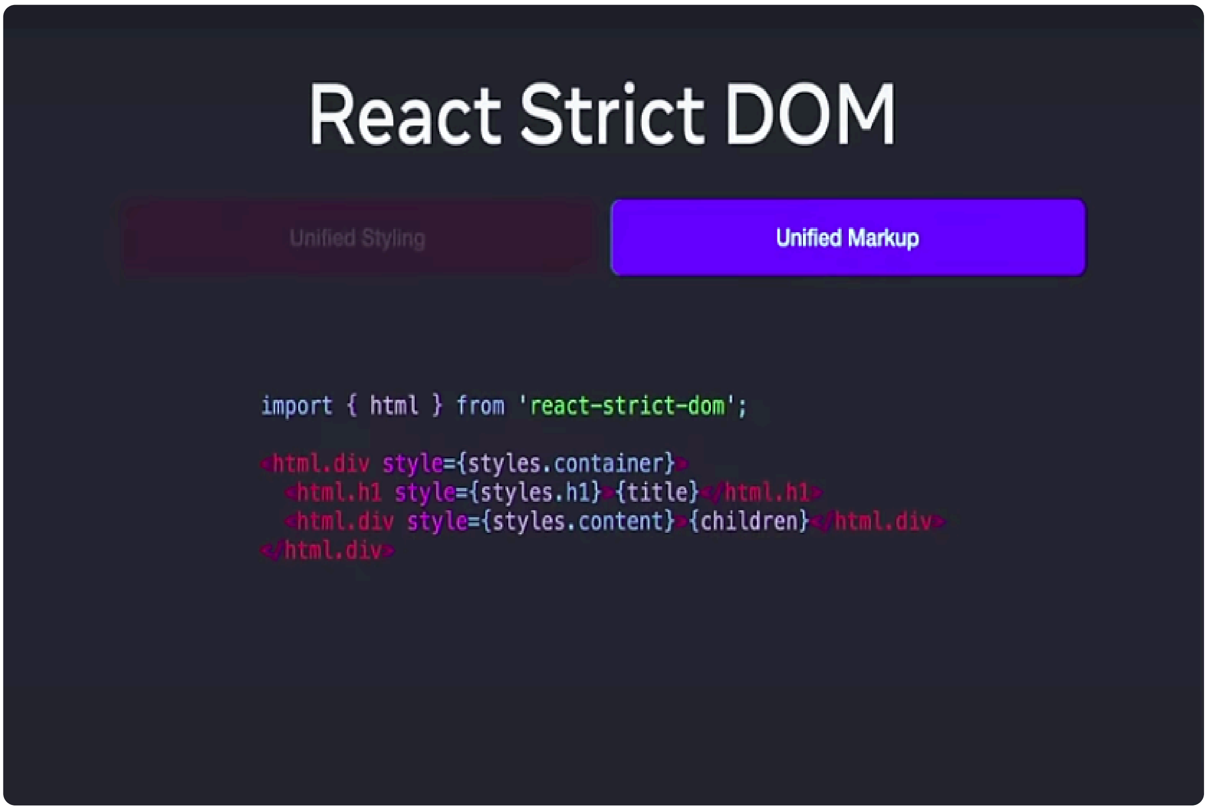
Windows

Web



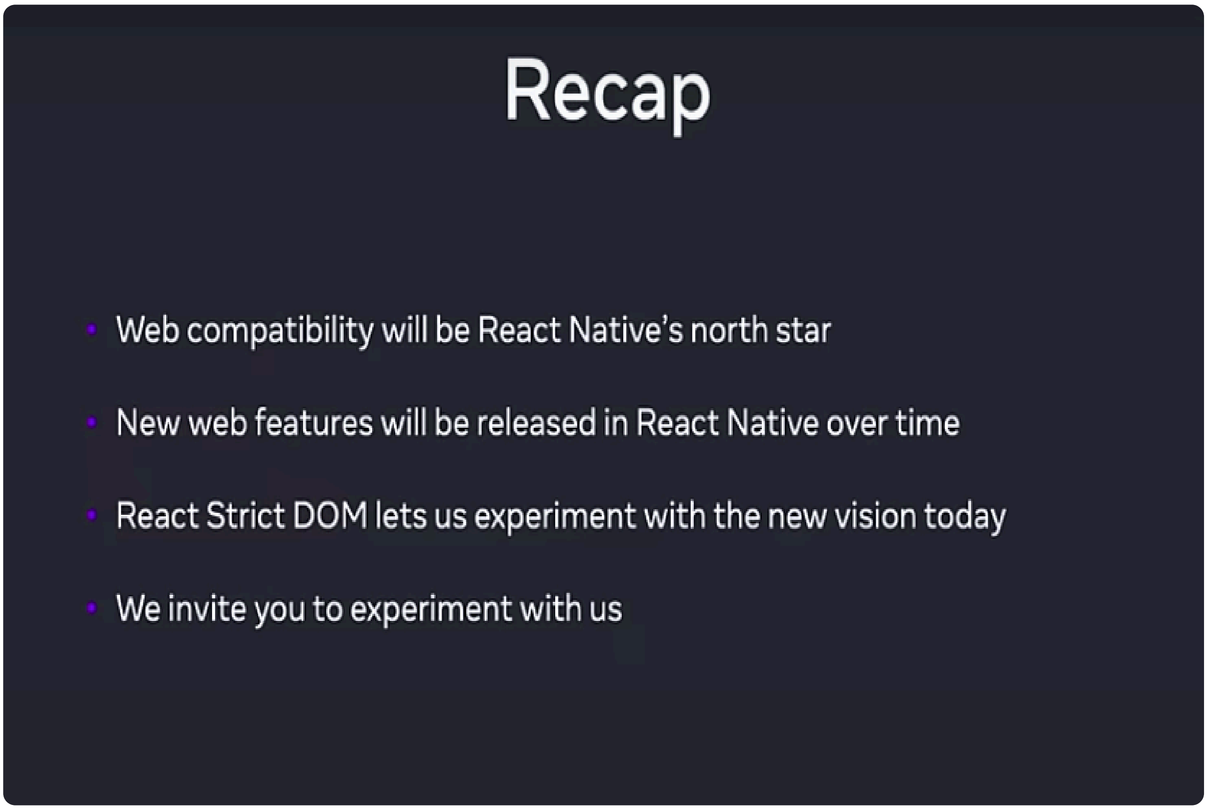
React Strict DOM

```
import { css } from 'react-strict-dom';

const styles = css.create({
  container: { borderTopWidth: 1 },
  h1: {
    padding: 10,
    backgroundColor: ‘#eee’,
  },
  content: { padding: 10 },
  div: {
    paddingBottom: 50,
    paddingTop: 50,
    backgroundColor: 'white',
  }
});
```



React Strict DOM

```
import { html } from 'react-strict-dom';

<html.div style={styles.container}>
  <html.h1 style={styles.h1}>{title}</html.h1>
  <html.div style={styles.content}>{children}</html.div>
</html.div>
```



Recap

- Web compatibility will be React Native’s north star
- New web features will be released in React Native over time
- React Strict DOM lets us experiment with the new vision today
- We invite you to experiment with us