



**Centro Universitário de Desenvolvimento do Centro Oeste - UNIDESC**

**Acadêmica: Luciane Rodrigues da Silva**

Atividade de Pesquisa e Desenvolvimento de Aplicativos Mobile com React Native

Luziânia - GO

2025



**Centro Universitário de Desenvolvimento do Centro Oeste - UNIDESC**

**Acadêmica: Luciane Rodrigues da Silva**

**Turma: 5º Período Análise e Desenvolvimento de Sistemas**

Trabalho realizado na Disciplina de Desenvolvimento de Aplicativos Móveis ministrada pelo Prof.Karython Gomes de Oliveira.

## Sumário

### I. Pesquisa e Fundação Teórica

1. Introdução ao React e React Native
2. Sintaxe e Estrutura de Componentes
3. Pré-requisitos e Arquitetura

# I. Pesquisa e Fundamentação Teórica

Este artigo técnico analisa os conceitos essenciais do React Native, destacando sua arquitetura, vantagens e aplicações práticas. O objetivo é apresentar fundamentos sólidos para criação de aplicativos móveis híbridos, explicando não apenas o que usar, mas o porquê de cada conceito. As informações foram obtidas em fontes como documentação oficial do React Native, literatura especializada como *React Native in Action*, e artigos do MDN Web Docs, garantindo confiabilidade e atualidade.

## 1. Introdução ao React e React Native

### Definição e Propósito

O React Native é um framework open-source criado pelo Facebook para desenvolvimento de aplicativos móveis nativos utilizando JavaScript e os princípios do React. Enquanto o React tradicional é voltado ao desenvolvimento web, o React Native adapta esses conceitos para o ambiente mobile, permitindo criar aplicativos reais para iOS (Apple) e Android (Google) sem exigir linguagens como Swift ou Kotlin.

Sua principal vantagem está no fato de que o código JavaScript é convertido para elementos nativos das plataformas, evitando o uso de WebViews estratégia comum em frameworks híbridos como Ionic e Cordova e proporcionando desempenho próximo ao de um aplicativo totalmente nativo.

Ferramentas como o **Expo**, também open-source, atuam como uma camada adicional que simplifica o ciclo de desenvolvimento. Ele facilita testes, compilação (build), execução e publicação de apps sem necessidade inicial de configurar Xcode ou Android Studio, além de fornecer suporte a preview via QR Code.

### Bridge (Ponte) e Desempenho

O React Native opera por meio de um mecanismo denominado **Bridge** (ponte), responsável por fazer a comunicação entre o código JavaScript (lógica da aplicação) e APIs nativas do dispositivo escritas em Objective-C/Swift no iOS e Java/Kotlin no Android. É essa ponte que permite a execução rápida e interação direta com recursos como câmera, GPS e armazenamento, garantindo experiência fluida.

## 2. Sintaxe e Estrutura de Componentes

### Componentização

A filosofia central do React Native é a componentização: a interface é construída a partir de pequenos blocos independentes e reutilizáveis que encapsulam estilo, estrutura e comportamento. Isso favorece organização do código, manutenção e escalabilidade. Exemplos básicos incluem componentes como `<View>` e `<Text>`, que servem de blocos estruturais para criar interfaces completas.

## JSX

JSX é uma extensão sintática do JavaScript que permite escrever estruturas visuais semelhantes a HTML dentro do código. Ele integra estrutura, lógica e estilo em um único arquivo, tornando o desenvolvimento mais intuitivo. No React Native, JSX é convertido pelo Babel em JavaScript puro antes de ser interpretado pelo framework.

## Props (Propriedades)

As **props** correspondem a dados recebidos de um componente pai, funcionando como parâmetros que permitem personalizar e comunicar-se entre componentes. São imutáveis e de leitura única, garantindo previsibilidade. Exemplos: `<Card titulo="Promoções" />` ou `<Image source={props.imagem} />`.

## State (Estado)

O **estado** representa informações mutáveis pertencentes exclusivamente ao componente variáveis que mudam ao longo do uso da aplicação, refletindo diferentes momentos do ciclo de vida da interface. Ele é inicializado internamente (`this.state = { valor: '' }`) e atualizado com `setState()`, método que dispara re-renderização automática da interface para refletir mudanças visuais, como digitação em campos ou alterações de listas.

## 3. Pré-requisitos e Arquitetura

### Funcionalidades ES6+

O React Native depende de funcionalidades do ECMAScript 2015 (ES6) e posteriores, incluindo:

- **class** — para definição de componentes baseados em classe;
- **import/export** — para modularização de código;
- **arrow functions** — sintaxe enxuta ideal para callbacks e eventos;

- **funções de array (map, filter, reduce)** — essenciais para manipulação e renderização dinâmica de listas de dados.

Esses recursos tornam o código mais expressivo, organizado e eficiente.

## Ferramentas e Ambiente

O **Expo** atua como um pacote de ferramentas que simplifica desenvolvimento, testes e compilação, acelerando o início do projeto e evitando configurações complexas até que o app necessite de integração nativa avançada.

## Bridge e Integrações Nativas

A Bridge realiza a comunicação entre threads JavaScript e nativas, serializando mensagens e ativando APIs do dispositivo conforme necessário. Esse mecanismo permite alta performance e flexibilidade, tornando o React Native uma solução forte para aplicativos móveis que exigem recursos nativos.

## 4. CONCLUSÃO

O estudo evidenciou que o React e o React Native representam tecnologias essenciais para o desenvolvimento moderno de aplicações, permitindo maior produtividade, reutilização de componentes e uma padronização na construção de interfaces. Conceitos como **hooks**, **props**, **estado** e **ciclo de vida** são fundamentais para compreender o funcionamento dessas ferramentas e construir aplicações mais performáticas e organizadas.

Além disso, boas práticas como tratamentos assíncronos, uso correto do **useEffect**, controle de estados e modularização garantem melhor experiência ao usuário e evitam erros comuns em aplicações reais. Assim, o domínio desses princípios fortalece a capacidade de desenvolver soluções escaláveis, eficientes e alinhadas ao mercado atual.

## CONSIDERAÇÕES FINAIS

A disciplina proporciona uma visão prática e conceitual sobre o desenvolvimento com React e React Native, preparando o estudante para os desafios da construção de interfaces modernas. Observou-se que, mais do que aprender sintaxe, é necessário compreender **a lógica de funcionamento dos componentes**, a importância da reutilização e a gestão de efeitos colaterais. Esses conhecimentos traduzem-se em maior clareza, organização e qualidade no código produzido, facilitando tanto a evolução profissional quanto o trabalho em equipe.

## REFERÊNCIAS

- Facebook Open Source — **React Documentation**. Disponível em: <https://react.dev>
- Meta Platforms — **React Native Documentation**. Disponível em: <https://reactnative.dev>
- Wazlawick, R. **Engenharia de Software Moderna**. Elsevier, 2021.
- Flanagan, D. **JavaScript: O Guia Definitivo**. Bookman, 2020.
- Mitchell, A. **Learning React**. O'Reilly Media, 2022.