



**Histórico de Revisões**

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** |
| 05/11/2019 | 1.0 | Iniciação do documento. | Vinícius Agostini |
| 17/11/2019 | 1.1 | Revisão do documento. | Vinícius Agostini |
|  |  |  |  |

**SUMÁRIO**

[**Introdução**](#_heading=h.harhi1tbw79i) **4**

[Referências](#_heading=h.1fob9te) 4

[**Organização e estilos de código (PHP)**](#_heading=h.veuklflnwcpm) **5**

[**Comentários**](#_heading=h.3as2kigapyw6) **6**

[**Nomeação**](#_heading=h.dy1bqhdgx9hr) **6**

[**Declaração**](#_heading=h.chs3nlq2um3) **6**

[**Reutilização**](#_heading=h.g0c2uhedrhgt) **6**

# **Introdução**

Este guia apresenta os padrões de codificação do Avaliador de trabalhos para eventos do IFSP – Campus Salto, desenvolvido para a disciplina de Desenvolvimento para Web II no curso de Análise e Desenvolvimento de Sistemas do IFSP - Campus Salto.

O Avaliador de trabalhos foi desenvolvido utilizando o framework Javascript React Native, em sua interação front-end e na linguagem de programação PHP para as interações back-end, para tanto este documento estará dividido em duas partes de padronização, sendo elas, a utilizada para criação do aplicativo e sua interface e a de modelagem de negócios.

## Referências

*● JavaScript Guia de estilo e convenções de codificação. Disponível em: <*[*http://www.w3bai.com/pt/js/js\_conventions.html*](http://www.w3bai.com/pt/js/js_conventions.html)*>*

*● Guias de leitura React Navigation. Disponível em: <* [*https://reactnavigation.org/pt-BR/*](https://reactnavigation.org/pt-BR/)*>*

*●Guias React Native. Disponível em: <https://facebook.github.io/react-native/docs/getting-started.html>*

1. Organização e estilos de código (REACT)

● Arquitetura de pastas a ser mantida dentro da pasta src:

o actions (Redux)

o components (Componentes customizados para a aplicação, não os por padrão fornecidos pelo react)

o reducers (Redux)

o screens (telas da aplicação)

▪ styles (estilos gerais do aplicativo, configuráveis)

o images (imagens utilizadas pelo sistema)

▪ logo

▪ mainCard

o services (endpoint de api's)

o store (Redux)

● Os caminhos/rotas das telas deve estar no arquivo routes.js mais externo, na pasta *src.*

*●* Declaração de variáveis sempre no início do bloco que irá utilizá-la.

*●* Importação de bibliotecas sempre no topo do código.

● Padrão de nomeação de métodos: camelCase.

● Padrão de nomeação de variáveis: camelCase.

# Organização e estilos de código (PHP)

● Tamanho de classes php: não mais que 1000 linhas.

*●* Declaração de variáveis sempre que necessário.

*●* Importação de bibliotecas sempre no topo do código.

● Padrão de nomeação de métodos: camelCase ou underscore.

● Padrão de nomeação de variáveis: camelCase.

# Comentários

Métodos (funções)

/\*\*

\* <Descrição da funcionalidade do método>

\*/

// <Descrição das funcionalidades do método>

# Nomeação

Arquivos Javascript

● nomearquivo.js

Exemplo:

Arquivo de tela para leitura de qrcode

avaliar.js

Variáveis

● nomeVariavel

# Declaração

1. Padrão de indentação de declaração de variáveis const:

const nomeVariavel = atribuicao (

{

[...],

[...],

});

2. Padrão de indentação de declaração de classes:

Class NomeClasse {

}

# Reutilização

● Criar arquivos distintos para cada nova tela do sistema

● Manter sempre um arquivo index.js como responsável pela execução da aplicação

● Manter atualizado arquivo de pacotes utilizados pela aplicação