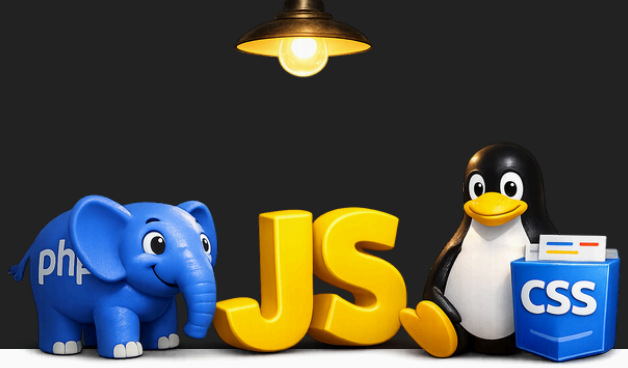
Code Style Guide



Padrões de estilo de programação definem uma forma consistente de escrever códigos, tornando-os mais legíveis, previsíveis e manuteníveis. Ao adotar convenções comuns, a equipe melhora a colaboração, reduz erros e facilita a evolução dos sistemas, independentemente de quem escreveu o código. Esses padrões também permitem a automação por meio de ferramentas, garantindo consistência e qualidade sem depender exclusivamente de boas práticas individuais.

PHP - PSR-1 / PSR-12

- Os arquivos DEVEM usar apenas tags *<?php* e *<?=*.
- Os arquivos DEVEM usar somente UTF-8 sem BOM para código PHP.
- Os arquivos DEVEM declarar símbolos (classes, funções, constantes, etc.) ou causar efeitos colaterais (por exemplo, gerar saída, alterar configurações de arquivos .ini, etc.), mas NÃO DEVEM fazer ambos.
- Os namespaces e as classes DEVEM seguir um PSR de "autoloading": [PSR-0 , PSR-4].
- Os nomes das classes DEVEM ser declarados em PascalCase.
- As constantes de classe DEVEM ser declaradas em UPPER_CASE.
- Os nomes dos métodos DEVEM ser declarados em camelCase.
- Uma declaração por linha.
- Ordem dos elementos dentro de uma classe: 1. Traits (*use*); 2. Constantes; 3. Propriedades; 4. Métodos.
- Visibilidade explícita (public, protected, private).
- Posição inicial obrigatória do declare(strict_types=1).

JS - AIRBNB STYLE GUIDE

- Usar *const* e *let* — nunca *var.*
- Declarar variáveis uma por linha.
- Template strings para concatenação *(`Olá, ${nome}.`)*
- *===* e *!==* no lugar de *==* e *!=.*
- Arrow functions *(() => {})* para funções curtas.
- Usar métodos de array (*map*, *filter*) em vez de loops tradicionais.
- Imports sempre no topo do arquivo.
- PascalCase para nomes de classe.
- camelCase para nomes de variáveis, funções e métodos.
- UPPER_CASE para nomes de constantes.

CSS - GOOGLE CSS GUIDE

- Evitar seletores excessivamente específicos.
- Evitar IDs (*#id*) para estilização.
- Preferir classes a seletores de tag.
- Não depender da estrutura do DOM para estilização.
- Evitar seletores encadeados profundamente.
- Evitar sobrescritas frequentes (!important).
- Usar kebab-case para nomes de classes.
- Evitar abreviações obscuras.
- Nomes refletem função, não aparência.
- Evitar inline styles.

AUTOMAÇÃO DE PADRÕES

A formatação do código pode ser automatizada com ferramentas que garantem consistência visual, enquanto as regras semânticas e estruturais dependem do programador e devem ser aplicadas com critério e entendimento do contexto. Utilize o código abaixo para instalar as ferramentas de automação automaticamente.

```
composer require rcm-cabos-eletricos/dev-standards // Instala a biblioteca
vendor/rcm-cabos-eletricos/dev-standards/scripts/install.sh // Copia os arquivos e instala plugins no VSCODE
```

ARQUIVOS COPIADOS

.vscode
↳ *settings.json*
.php-cs-fixer.php
.prettierrc
.bladeformatterrc

PLUGINS INSTALADOS

Prettier - Code formatter
PHP CS Fixer
Laravel Blade formatter

GITHUB

Acesse o repositório do projeto no github da organização.
Rcm-Cabos-Eletricos/dev-standards

RESULTADO

✨ Menos discussão de estilo
✨ Código mais legível
✨ Menos retrabalho