

# SONAECOM

## CodeStyleBook

29 Oct 2010 - JoseCarlosPereira

Copyright © 2010 Sonaecom  
All rights reserved

This document contains Proprietary and Confidential information of Sonaecom, and is protected by copyright, trade secret and other laws. Its receipt or possession does not convey any rights to reproduce, disclose its contents, or to manufacture, use or sell anything it may describe. Reproduction, disclosure or use without specific written authorization of Sonaecom is strictly prohibited.



# Table of Contents

<b>1 Intro.....</b>	<b>Page 1</b>
<b>2 Algumas regras.....</b>	<b>Page 2</b>
2.1 Língua.....	Page 2
2.2 Identação.....	Page 2
2.3 Comentários/Documentação:.....	Page 2
2.4 Práticas comuns PERL.....	Page 2
2.5 Utilitários.....	Page 2
<b>3 Testes.....</b>	<b>Page 3</b>



# 1 Intro

Em PD desenvolvemos ( muito ) e quase sempre na mesma linguagem. Mais do que isso, desenvolvemos todos no mesmo código daí a importancia de ter algumas regras que facilitem a leitura do código escrito.



## 2 Algumas regras

### 2.1 Lingua

Desenvolver o código em inglês ( nomes de variaveis, nomes de funcoes, etc ). As unicas excepções devem ser a mensagens que são passadas aos utilizadores e tudo o que tenha que ver com Marketing/Oferta ( por exemplo as descrições comerciais de produtos ).

### 2.2 Identação

Identar o código com o perltidy ( obviamente se não for PERL prevalece o bom senso ). Uma regra consensual é a utilização de 4 espaços para identar blocos de código.

### 2.3 Comentários/Documentação:

O ideal era documentar o código com base no HowTo de documentação , no entanto nem sempre temos tempo ou se justifica. Mesmo que não se documente totalmente o projecto convém comentar algumas partes do código para facilitar a vida á proxima pessoa que o tiver que ler:


- Um comentário de uma linha por função a explicar o que é que a função faz ( especialmente se o nome não for sugestivo )
- Comentar blocos de código mais cripticos, são 2 minutos que poupam horas ás outras pessoas.

### 2.4 Práticas comuns PERL

não são necessariamente boas práticas mas como temos muito código que as cumpre convém mantê-las

- nomes de funções em minusculas com as palavras separadas por "underscore"s
- funções que recebam argumentos por hash devem declara a hash que contem os argumentos como %a
- \$s é sempre o self-object
- \$c é sempre o nome da class
- O class-method 'new' instancia um objecto
- O class-method 'create' cria um novo object ( não faz retrieve )
- Regras e sugestões de programação: pbp\_regras.pdf

### 2.5 Utilitários

- /servers/libs/novis/Sonaecom/Logger/examples/vimrc
  - ♦ helper para facilitar programação com logging
- ~jcp/.vimrc 



# 3 Testes

Num futuro próximo...

