

# CodingStyle - Escrevendo programas legíveis

2º Seminário de Software Livre Tchelinix FAE Erechim

Douglas Schilling Landgraf  
<dougslan@gmail.com>

Junho 14, 2008.

# CodingStyle

- ▶ Qual é o objetivo desta palestra ?
- ▶ CodingStyle no passado ?
- ▶ Quem cria o CodingStyle ?
- ▶ Como esta palestra foi montada ?
- ▶ O estilo apresentado será melhor que o seu ?
- ▶ "Time que esta ganhando não se mexe.."

## ▶ Princípios:

- Código simples e claro
- Lógico
- Expressões simples
- Comentários que *\*realmente\** façam sentido

- ▶ O que deve ser evitado ?
  - "Códigos que resolvem todos os problemas"
  - Construções não lógicas
  - Comentários em excesso
  - ...

## ▶ Benefícios:

- O código ficará claro a "todos" desenvolvedores envolvidos
- Bugs serão encontrados com mais facilidade
- A "produtividade" de todos aumentará :)

## ▶ Identação:

- TABS ou Espaços ?
- 8, 4 ou 2 Caracteres = 1 TAB ?
- 80 colunas

80 Colunas ? Oooops... =:o(

Espaços

nos lugares certos...



Exemplo 1:

```
for(
    ret=codigo;
    ret<=ant+128;
    ret++)
{
    codigo;
}
```

Exemplo

2:

```
for(x=0;x<=12;x++){
    codigo;
}
```

## ▶ Espaços nos lugares certos...

- Exemplo 1:

```
for(
    ret=codigo;
    ret<=ant+128;
    ret++)
    {
        codigo;
    }
```

- Exemplo 2:

```
for(x=0;x<=12;x++){
    codigo;
}
```

## ▶ Espaços nos lugares certos...

- Exemplo 1:

```
for(  
    ret=codigo;  
    ret<=ant+128;  
    ret++  
)  
    codigo;
```

ou

```
for (ret = codigo; ret <= ant+128; ret++) {  
    codigo;  
}
```

## ▶ Espaços nos lugares certos...

- Exemplo 2:

```
for(x=0;x<=12;x++){  
    codigo;  
}
```

ou

```
for (x = 0; x <= 12; x++) {  
    codigo;  
}
```

## ▶ Expressões e Validações:

- Exemplo 1:

```
if (variavel < 0) x = variavel * 3.69;
```

- Exemplo 2:

```
for (n = 0; n < 100; n++)  
i = var[n]; return( 0 );
```

- Exemplo 3:

```
switch (code){  
case 10: arg = 'V'; break;  
case 30: arg = 'Z'; break;  
}
```

## ▶ Expressões e Validações:

- Exemplo 1:

```
if (variavel < 0) x = ant * 3.69;
```

ou

```
if (variavel < 0) {  
    x = ant * 3.69;  
}
```

## ▶ Expressões e Validações:

- Exemplo 2:

```
for (n = 0; n < 100 ; n++)  
i = var[n]; return( 0 );
```

ou

```
for (n=0; n < 100; n++) {  
    i = var[n];  
}  
return 0;
```

## ▶ Expressões e Validações:

- Exemplo 3:

```
switch(code)
{
case 10: arg = 'V'; break;
case 30: arg = 'Z'; break;
}
```

ou

```
switch(code) {
case 10:
    arg = 'V';
    break;
case 30:
    arg = 'Z';
    break;
}
```

- ▶ Forma natural e clara:

- Exemplo:

```
if(!(variavel < tamanho) || !(variavel >= tipo))
```

ou

```
if((variavel >= tamanho) || (variavel < tipo))
```

## ▶ Funções

- Nome descritivo para cada função
- Use apenas letras, números e sublinhado
- Deixar claro quais são as palavras que compõem o nome da função  
crash, crash\_maquina, crashMaquina
- Use somente uma lingua (Port. ou inglês)

## ▶ Variáveis:

- Nome descritivo para cada variável
- Use apenas letras, números e sublinhado
- Evite variáveis globais, quando usa-las de um nome bem claro
- MACROS (define) - com letras maiúsculas

## ▶ Redundância:

- Exemplo:

```
class UserQueue {  
    int noOfItemsInQ, frontOfTheQueue, queueCapacity;  
    public int noOfUsersInQueue() ..  
}
```

ou

```
class UserQueue {  
    int nItems, front, capacity;  
    public int nUsers() ..  
}
```

## ▶ Chaves:

- Onde iniciar e finalizar as chaves ?

```
if (variavel == true)
{
    foo();
}
else
{
    bar();
}
```

- Em funções:

```
int funcao(void) {
    ...
}
```

## ▶ Chaves:

- Uma nova proposta:

```
int funcao(void)
{
    if (variavel == true) {
        foo();
    } else {
        bar();
    }
}
```

## ▶ Expressões complexas ?

- Exemplo 1:

```
int x = 17;  
int y = 2;
```

```
ret = (((x-y)+2)*((8+y)+2))*((x+3)+666));
```

- Exemplo 2:

```
x = 3;  
y = 6;
```

```
ret = ~(((x<<4)&(y^2))>>5);
```

## ▶ Expressões complexas ?

- Exemplo 1:

```
int x = 17;  
int y = 2;
```

```
ret = (((x - y) + 2) * ((8 + y) + 2)) * ((x + 3) + 666);
```

ou

```
int x = 17;  
int y = 2;
```

```
val1 = ((x - y) + 2);  
val2 = ((8 + y) + 2);  
val3 = (val1 * val2);
```

```
val4 = ((x + 3) + 666);  
ret = (val3 * val4);
```

## ▶ Expressões complexas ?

- Exemplo 2:

```
x = 3;
```

```
y = 6;
```

```
ret = ~(~((x>>4)&(y<<2))>>2);
```

ou

```
val1 = x>>4;
```

```
val2 = y<<2;
```

```
val3 = val1 & val2;
```

```
val4 = val3 >> 2;
```

```
val5 = ~val4;
```

```
ret = ~val5;
```

## ▶ Chaves:

- Como N-Ã-O deve ser feito:

```
if (variavel == true)
  for ( x = 0; x < 5; x++ )
  {
    if ( z != 2 && x < 5 )
    {
      if (variavel2 <= false)
        zx = 5;
        y = 10;
      }
    }
  }
else
{
  x = codigo << 2;
}
```

## ▶ Chaves:

```
if (variavel == true) {  
    for ( x = 0; x < 5; x++ ) {  
        if ( z != 2 && x < 5 ) {  
            if (variavel2 <= false) {  
                zx = 5;  
            }  
            y = 10;  
        }  
    }  
} else {  
    x = codigo << 2;  
}
```

## ▶ Retornos:

- Se função retornar, validar.

```
if (funcao(12, abc, dsc) == 0) {  
    codigo = 2;  
  
} else {  
    codigo = 4;  
}
```

## ▶ Retornos:

```
if (funcao(12, abc, dsc) == 0) {  
    codigo = 2;  
} else {  
    codigo = 4;  
}  
    ou
```

```
int ret;
```

```
ret = funcao(12, abc, dsc);  
if (ret == 0) {  
    codigo = 2;  
} else {  
    codigo = 4;  
}
```

## ▶ Ferramentas para indentação:

- indent

- IDEs

## ▶ Comentários:

- É necessário explicar o que o código já informa ?

```
contador++; /* A variavel será incrementada */
```

```
/* Irá retornar var */  
return var;
```

```
/* contador recebe 10 */  
contador = 10;
```

- Data da última alteração ?
- Quem escreveu determinada função ?
- Comentários podem poluir o código ?

## ▶ Comentários:

- Manter comentários atualizados

```
if ( (carro == FIESTA) || (carro == UNO) || (carro == KA) ||  
    (carro == FOCUS) ) {  
    ....
```

```
/* Os carros FIESTA UNO KA modelo 3000 devem receber revisão  
grátis durante 1 ano */
```

## ► Comentários:

```
if ( (carro == FIESTA) || (carro == UNO) || (carro == KA) ||  
     (carro == FOCUS) ) {  
    ....
```

```
/* Os carros FIESTA UNO KA e FOCUS modelo 3000 devem receber  
revisão grátis durante 1 ano, porque tivemos problemas  
mecânicos com os primeiros clientes */
```

## ► Comentários:

```
/*  
 * funcao - faz alguma coisa..  
 * @parametro - argumento  
 *  
 * Explicacao em detalhes  
 */  
  
int funcao( void ) {  
    ...  
}
```

- Ferramentas para gerar a documentação ?
  - Doxygen
  - Javadoc

## ▶ Números Mágicos:

```
cmd = status_impresora();
switch (cmd) {
case 0x23:
    faca_algo_aqui();
    break;
case 0x45:
    talvez_aqui();
    break;
case 0x77:
    outro();
    break;
...
}
```

O que significa ? 0x23 ?? 0x45 ?? 0x77 ??

## ▶ Números Mágicos:

```
cmd = status_impresora();
switch (cmd) {
CORTA_PAPEL:
    faca_algo_aqui();
    break;
PAPEL_NO_BOCAL:
    talvez_aqui();
    break;
IMPRESSORA_OCUPADA:
    outro();
    break;
...
}
```

- ▶ O que podemos fazer para manter o código em ordem ?
  - Scripts de validação de CodingStyle
  - Mensagens de aviso / Formatação automática (commits)
  - ...

## ▶ Documentação:

- The Practice of Programming  
by Brian W. Kernighan and Rob Pike.  
Addison-Wesley, Inc., 1999.  
ISBN 0-201-61586-X.  
URL: <http://cm.bell-labs.com/cm/cs/tpop/>
- `/usr/src/linux/Documentation/CodingStyle`

## Dúvidas e Sugestões ?

Douglas Schilling Landgraf  
<dougslan@gmail.com>

<http://dougslan.livejournal.com>

<http://people.tchelinux.org>

Junho 14, 2008.