**缩进**

* 采用4个空格
* 空格，不要用TAB！

**变量**

* 每行一个变量
* 尽可能避免短的变量名
* 单字符的变量只在临时变量或循环的计数中使用
* 等到真正需要使用时再定义变量

// Wrong

int a, b;

char \*c, \*d;

// Correct

int height;

int width;

char \*nameOfThis;

char \*nameOfThat;

* 以小写字符开头，后续单词以大写开头
* 避免使用缩写

// Wrong

short Cntr;

char ITEM\_DELIM = '/t';

// Correct

short counter;

char itemDelimiter = '/t';

* 类名总是以大写开头

**补充**

在Qt例子编写中，对变量名有如下建议：

* 成员变量前不要加 "m\_" 或其他什么东西
* 如果参数名和成员变量名发生冲突，使用 "this->" 解决

void MyClass::setColor(const QColor &color;)

{

this->color = color;

}

或

void MyClass::setColor(const QColor &newColor;)

{

color = newColor;

}

**避免使用** (意义不明确的字符)：

void MyClass::setColor(const QColor &c)

{

color = c;

}

注意：在构造函数中，会遇到同样的问题。但无论你信与不信，下面的可以工作

MyClass::MyClass(const QColor &color;)

: color(color)

{

}

**空白**

* 利用空行将语句恰当地分组
* 总是使用一个空行(不要空多行)
* 总是在每个关键字和大括号前使用一个空格

// Wrong

if(foo){

}

// Correct

if (foo) {

}

* 对指针和引用，在类型和\*、&之间加一个空格，但在\*、&与变量之间不加空格

char \*x;

const QString &myString;

const char \* const y = "hello";

* 二元操作符前后加空白
* 类型转换后不加空白
* 尽量避免C风格的类型转换

// Wrong

char\* blockOfMemory = (char\* ) malloc(data.size());

// Correct

char \*blockOfMemory = reinterpret\_cast<char \*>(malloc(data.size()));

* 除了断行外，不要尝试用空白对齐一些东西

//Wrong

x = rect.x();

y = rect.y();

width = rect.width();

height = rect.height();

**大括号**

* 基本原则：左大括号和语句保持在同一行

// Wrong

if (codec)

{

}

// Correct

if (codec) {

}

* 例外：函数定义和类定义中，左大括号总是单独占一行

class Moo

{

};

* 控制语句的body中只有一行时不使用大括号

// Wrong

if (address.isEmpty()) {

return false;

}

// Correct

if (address.isEmpty())

return false;

if (x) {

// do something strange

yyyyyyyyy = yyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyyy +

zzzzzzzzzzzzzzzzzzzzzz;

}

* 例外1：如果父语句跨多行，则使用大括号

// Correct

if (address.isEmpty() || !isValid()

|| !codec) {

return false;

}

* 例外2：在if-else结构中，有一处跨多行，则使用大括号

// Wrong

if (address.isEmpty())

return false;

else {

qDebug("%s", qPrintable(address));

++it;

}

// Correct

if (address.isEmpty()) {

return false;

} else {

qDebug("%s", qPrintable(address));

++it;

}

// Wrong

if (a)

if (b)

...

else

...

// Correct

if (a) {

if (b)

...

else

...

}

* 如果控制语句的body为空，则使用大括号

// Wrong

while (a);

// Correct

while (a) {}

**圆括号**

* 使用圆括号将表达式分组

// Wrong

if (a && b || c)

// Correct

if ((a && b) || c)

// Wrong

a + b & c

// Correct

(a + b) & c

**switch语句**

* case 和 switch 位于同一列
* 每一个case必须有一个break或renturn语句，或者用注释说明无需break

switch (myEnum) {

case Value1:

doSomething();

break;

case Value2:

doSomethingElse();

// fall through

default:

defaultHandling();

break;

}

**断行**

* 保持每行短于**100** 个字符（注意补充部分），需要时进行断行
* 逗号放一行的结束，操作符放到一行的开头（操作符和同级的前一行对齐）

// Correct

if (longExpression

+ otherLongExpression

+ otherOtherLongExpression) {

}

* 例外：如果使用的 if 语句 和 && 或者 ||，对齐需要一点调整(否则控制语句和body会较难以分辨)

//Wrong

if (dsfljfsfskjldsjkljklsjdk

&& fdsljsjdsdljklsjsjkdfs

&& dsfljkdfjkldksdfjdjkfdksfdkjld) {

sadjdjddadhsad;

}

//Correct

if (dsfljfsfskjldsjkljklsjdk

&& fdsljsjdsdljklsjsjkdfs

&& dsfljkdfjkldksdfjdjkfdksfdkjld) {

sadjdjddadhsad;

}

对 whle 或else if，不存在这个问题：

while (dsfljfsfskjldsjkljklsjdk

&& fdsljsjdsdljklsjsjkdfs

&& dsfljkdfjkldksdfjdjkfdksfdkjld) {

sadjdjddadhsad;

}

**补充**

* Qt 例子中比前面的标准100严格一点，采用的是80.
* Qt Quarterly 中采用的是 58
* Qt 书籍中采用的是 68 (?)

**继承与virtual**

* 重新实现一个虚函数时，头文件中不放置 virtual 关键字