

Qt Project

[Downloads](#) [Documentation](#) [Forums](#) [Wiki](#) [Groups](#) [Blogs](#) [Showroom](#)
[Qt.io](#)

[Wiki Home](#)

[Developing Qt](#)

[Guidelines](#)

[Qt_Coding_Style_SimplifiedChinese](#)

Last edit: September 16, 2011

简体中文 [English](#) [Spanish](#) [日本語](#)

Qt 编码风格

这是我们编写Qt代码时所使用的编码惯例的一个概述。数据是通过挖掘Qt源码、论坛、邮件列表以及与开发者的协作而收集起来的。

缩进

采用4个空格

空格，不要用TAB！

变量声明

每行一个变量

尽可能避免短的变量名(比如“a”，“rbarr”，“nughdeget”)

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

```
1. // Wrong
2. int a, b;
3. char *c, *d;
4.
5. // Correct
6. int height;
7. int width;
8. char *nameOfThis;
9. char *nameOfThat;
```

以小写字符开头，后续单词以大写开头

避免使用缩写

```
1. // Wrong
2. short Cntr;
3. char ITEM_DELIM = '\t';
4.
5. // Correct
6. short counter;
7. char itemDelimiter = '\t';
```

类名总是以大写开头。公有类以Q开头(QRgb)，公有函数通常以q开头(qRgb)。

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)

Table of Content

- Qt 编码风格
 - 缩进
 - 变量声明
 - 空白
 - 大括号
 - 圆括号
 - Switch 语句
 - 断行
 - 继承与关键字`virtual`
 - 通用例外

总是在每个关键字和大括号前使用一个空格

```
1. // Wrong
2. if(foo) {
3. }
4.
5. // Correct
6. if (foo) {
7. }
```

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

```
1. char *x;
2. const QString &myString;
3. 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

```
1. // Wrong
2. char* blockOfMemory = (char* ) malloc(data.size());
3.
4. // Correct
5. char *blockOfMemory = reinterpret_cast<char *>(malloc(data.size()));
```

大括号

基本原则：左大括号和语句保持在同一行：

```
1. // Wrong
2. if (codec)
3. {
4. }
5.
6. // Correct
7. if (codec) {
8. }
```

例外：函数定义和类定义中，左大括号总是单独占一行：

```
1. static void foo(int g)
2. {
3.     qDebug("foo: %i", g);
4. }
5.
6. class Moo
7. {
8. };
```

控制语句的body中只有一行时不使用大括号

```
1. // Wrong
2. if (address.isEmpty()) {
3.     return false;
4. }
5.
6. for (int i = 0; i < 10; ++i) {
7.     qDebug("%i", i);
8. }
9.
10. // Correct
11. if (address.isEmpty())
12.     return false;
13.
14. for (int i = 0; i < 10; ++i)
```

15. `qDebug("%i", i);`

例外1: 如果父语句跨多行, 则使用大括号

```
1. // Correct
2. if (address.isEmpty() || !isValid()
3.    || !codec) {
4.     return false;
5. }
```

例外2: 在if-else结构中, 有一处跨多行, 则使用大括号

```
1. // Wrong
2. if (address.isEmpty())
3.     return false;
4. else {
5.     qDebug("%s", qPrintable(address));
6.     ++it;
7. }
8.
9. // Correct
10. if (address.isEmpty()) {
11.     return false;
12. } else {
13.     qDebug("%s", qPrintable(address));
14.     ++it;
15. }
16.
17. // Wrong
18. if (a)
19.     if (b)
20.         ...
21.     else
22.         ...
23.
24. // Correct
25. if (a) {
26.     if (b)
27.         ...
28.     else
29.         ...
30. }
```

如果控制语句的body为空, 则使用大括号

```
1. // Wrong
2. while (a);
3.
4. // Correct
5. while (a) {}
```

圆括号

使用圆括号将表达式分组

```
1. // Wrong
2. if (a && b || c)
3.
4. // Correct
5. if ((a && b) || c)
6.
7. // Wrong
8. a + b & c
9.
10. // Correct
11. (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return)语句，或者用注释说明无需break

```
1.      switch (myEnum) {  
2.      case Value1:  
3.          doSomething();  
4.          break;  
5.      case Value2:  
6.          doSomethingElse();  
7.          // fall through  
8.      default:  
9.          defaultHandling();  
10.         break;  
11.     }
```

断行

保持每行短于100 个字符，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。如果你的编辑器太窄，一个放在行尾的操作符不容易被看到。

```
1.      // Correct  
2.      if (longExpression  
3.          + otherLongExpression  
4.          + otherOtherLongExpression) {  
5.      }  
6.  
7.      // Wrong  
8.      if (longExpression +  
9.          otherLongExpression +  
10.         otherOtherLongExpression) {  
11.     }
```

继承与关键字 `virtual`

重新实现一个虚函数时，头文件中 不 放置 virtual 关键字。

通用例外

如果它使你的代码看起来不好，你可以打破任何一个规则 。

注：译文一开始发表在 [CSDN博客](#) [blog.csdn.net] ，有问题欢迎讨论

Categories:

- Developing_Qt
- Guidelines
- SimplifiedChinese
- Developing_Qt



