C++11标准简介

一、C++标准的历史

1.1990年，The Annotated C++ Reference Manual，B.Stroustrup

2.1998年，ISO/IEC 15882:1998，C++98 \ C++98/03

3.2003年，ISO/IEC 15882:2003，C++03 /

4.2011年，ISO/IEC 15882:2011，C++11

二、C++标准的目标

1.C++98/03

1)比C语言更适合系统编程，且对C完全兼容；

2)支持数据抽象；

3)支持面向对象编程；

4)支持泛型编程。

2.C++11

1)更适合系统开发及库开发；

2)更加易学易用；

3)保持稳定，对C和C++98/03全面兼容；

4)扩充标准库，提供更多与时代需求相一致的功能。

三、编译选项

GCC编译器缺省语言标准是C++98/03，如果需要使用C++11标准，那么就需要显式指明：

GCC < 4.8: -std=c++0x

GCC >= 4.8: -std=c++11

语法部分

一、类型推导

在程序代码中不显式指明变量的类型，而是由编译器根据代码的上下文隐式推导出该变量的类型，其推导规则类似于旧标准对函数模板类型参数的隐式推断。

1.auto

旧标准中auto关键字的作用和C语言一样，表示自动变量，是关于变量存储位置的类型饰词，通常不写，因为局部变量的默认存储就是auto。

void foo(void){

int a;

auto int b; // 表示放在栈里，一般可以省略不写

static int c; // 数据区，静态变量

register int d; // 寄存器，寄存器变量

}

register：寄存器变量，建议编译器放在寄存器里。

补充：建议表示是否放入寄存器不一定，因为寄存器空间有限(通常可以放20多个)；但是现在很多编译器有寄存器优化，会把一些频繁访问的变量自动放入寄存器。

------------------

新标准中auto关键字已经不再表示变量的存储属性，而是用于类型推导。

1. auto的基本用法

参见：auto1.cpp

1. auto同指针或引用结合使用

参见：auto2.cpp

1. auto使用的限制

参见：auto3.cpp

1. auto试用的场景

参见：auto4.cpp