


本文被 1 个清单收录, 推荐清单

C++那些事之现代C++篇



什么是“依赖于参数的查询”（又名ADL或“ Koenig查找”）？

Koenig查找或参数依赖查找描述了C ++编译器如何查找不合格的名称。简单来说：**如果在函数的名称空间中定义了一种或多种参数类型，则不必为函数限定名称空间。**

例如：

```
1 namespace MyNamespace
2 {
3     class MyClass {};
4     void doSomething(MyClass);
5 }
6
7 MyNamespace::MyClass obj; // global object
8
9
10 int main()
11 {
12     doSomething(obj); // Works Fine - MyNamespace::doSomething() is called.
13 }
```

在上面的示例中，既没有使用 `using` 声明也没有使用 `using` 指令，但是编译器仍然可以通过应用Koenig查找将正确的不合格名称 `doSomething()` 识别为在名称空间 `MyNamespace` 中声明的函数。

它是如何工作的？

该算法告诉编译器不仅要查看本地作用域，还要查看包含参数类型的名称空间。因此，在上面的代码中，编译器发现对象obj（它是函数 `doSomething()` 的参数）属于命名空间 `MyNamespace`。因此，它将查看该命名空间以找到 `doSomething()` 的声明。

Koenig查找的优点是什么？

如上面的简单代码示例所示，Koenig查找为程序员提供了便利和易用性。如果没有Koenig查找，那么程序员将不得不负担重复指定完全限定名称的费用，或者使用大量using声明。

为什么批评Koenig查找？

过度依赖Koenig查找会导致语义问题，有时会使程序员措手不及。

考虑std :: swap的示例，它是交换两个值的标准库算法。使用Koenig查找时，使用此算法时必须谨慎，因为：

```
1 | std::swap(obj1,obj2);
```

可能不会显示以下行为：

```
1 | using std::swap;
2 | swap(obj1, obj2);
```

使用ADL，调用哪个版本的交换函数将取决于传递给它的参数的名称空间。

如果存在命名空间A，并且存在 `A::obj1`，`A::obj2` 和 `A::swap()`，则第二个示例将导致对 `A::swap()` 的调用，这可能不是用户想要的。

此外，如果由于某种原因同时定义了 `A::swap (A :: MyClass&, A :: MyClass&)` 和 `std::swap (A::MyClass&, A::MyClass&)`，则第一个示例将调用 `std::swap (A::MyClass&, A::MyClass&)`，但是第二个将无法编译，因为 `swap(obj1, obj2);` 会模棱两可。

上述完整例子如下：

```
1 #include <iostream>
2
3 namespace A {
4     template<typename T>
5     class smart_ptr {
6     public:
7         smart_ptr() noexcept : ptr_(nullptr) {
8
9         }
10
11         smart_ptr(const T &ptr) noexcept : ptr_(new T(ptr)) {
12
13         }
14
15         smart_ptr(smart_ptr &rhs) noexcept {
16             ptr_ = rhs.release(); // 释放所有权,此时rhs的ptr_指针为nullptr
17         }
18
19         smart_ptr &operator=(smart_ptr rhs) noexcept {
20             swap(rhs);
21             return *this;
22         }
23
24         void swap(smart_ptr &rhs) noexcept { // noexcept == throw() 保证不抛出异常
25             using std::swap;
26             swap(ptr_, rhs.ptr_);
27         }
28
29         T *release() noexcept {
30             T *ptr = ptr_;
31             ptr_ = nullptr;
32             return ptr;
33         }
34
35         T *get() const noexcept {
36             return ptr_;
37         }
38
39     private:
40         T *ptr_;
41     };
42
43     // 提供一个非成员swap函数for ADL(Argument Dependent Lookup)
44     template<typename T>
45     void swap(A::smart_ptr<T> &lhs, A::smart_ptr<T> &rhs) noexcept {
46         lhs.swap(rhs);
47     }
48 }
49
50 // 开启这个注释，会引发ADL冲突
51 //namespace std {
52 //    // 提供一个非成员swap函数for ADL(Argument Dependent Lookup)
53 //    template<typename T>
54 //    void swap(A::smart_ptr<T> &lhs, A::smart_ptr<T> &rhs) noexcept {
55 //        lhs.swap(rhs);
56 //    }
57 //}
58 //}
59
60 int main() {
61
62     using std::swap;
63     A::smart_ptr<std::string> s1("hello"), s2("world");
64     // 交换前
65     std::cout << *s1.get() << " " << *s2.get() << std::endl;
66     swap(s1, s2); // 这里swap 能够通过Koenig搜索或者说ADL根据s1与s2的命名空间来查找swap函数
67     // 交换后
68     std::cout << *s1.get() << " " << *s2.get() << std::endl;
69 }
```

本篇学习自：  
<https://stackoverflow.com/questions/8111677/what-is-argument-dependent-lookup-aka-adl-or-koenig-lookup>



公众号guangcity  
379 篇文章 >

现代C++之ADL

转到我的清单



单。

公众号guangcity

### 让我们行动起来:认识日常生活的活动通过学习模拟生活视频游戏

日常生活活动识别(ADL)是智能辅助机器人的一个重要过程。但收集大量带注释的数据集需要耗时的时间标记,并引起隐私问题。比如说,这些数据是在真实家庭中收集的。在这...

用户8789655

### C++核心准则T.47:避免使用通用名称的高度不受限模板

An unconstrained template argument is a perfect match for anything so such a tem...

面向对象思考

### 激进的ADL(Argument-dependent lookup)

ADL是Argument-dependent lookup的缩写,中文译作参数依赖查找。ADL对于避免冗长的代码很有用处。但是也会造成一些歧义。

高利鹏

### CppQuiz一些有趣的题和分析

CppQuiz is a simple online quiz that you can use to test your knowledge of the C...

高利鹏

### “人脸识别”助力渔业管理

据美国政府计算网站(GCN)报道,为了监测包括北太平洋和东白令湾在内的约三百万平方英里海洋中的鱼类资源,美国阿拉斯加州渔业科学中心开始使用面部识别技术(更准确...

人工智能快报

### 软考分类精讲-软件架构设计(一)

cai\_java

### 现代C++之SFINAE

0.导语1.C++自省72.卷式的C++98方式2.1重载决议2.2 SFINAE2.3 sizeof运算符2.4 结合一切2.5 实现我们的想法2.6 小结3...



公众号guangcity

### 现代C++之constexpr

(1) C++11中的constexpr指定的函数返回值和参数必须保证是字面值,而且必须只有一行return代码,这给函数的设计者带来了更多的限制。比如通常只能...

公众号guangcity

### 软考分类精讲-软件架构设计(三)

cai\_java

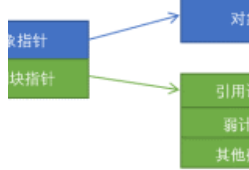
### aprium教程\_4.adb常用命令

一些理论/原理的可以查看官方文档: <https://developer.android.com/studio/command-line/adb?hl=zh-cn...>

千往

### C++避坑指南

导语:如果,将编程语言比作武功秘籍,C++无异于《九阴真经》。《九阴真经》威力强大、博大精深,经中所载内功、轻功、拳、掌、腿、刀法、剑法、杖法、鞭法、指爪、点穴...



腾讯技术工程官方号

### 现代C++之容器

本节将深入学习现代C++实战30讲中的第4节与第5节容器所提到的内容。正文中的一些文字直接引用自上面。

公众号guangcity

### R语言用向量自回归 (VAR) 进行经济数据脉冲响应研究分析

自从Sims (1980) 发表开创性的论文以来,向量自回归模型已经成为宏观经济研究中的关键工具。这篇文章介绍了VAR分析的基本概念,并指导了简单模型的估算过程。

拓展

更多文章 >

#### 社区

专栏文章  
阅读清单  
互动问答  
技术沙龙  
技术快讯  
团队主页  
开发者手册  
腾讯云T+平台

#### 活动

原创分享计划  
自媒体分享计划  
邀请作者入驻  
自荐上首页  
在线直播  
生态合作计划

#### 资源

技术周刊  
社区标榜  
开发者实验室

#### 关于

视频介绍  
社区规范  
免责声明  
联系我们  
友情链接

#### 腾讯云开发者



扫码关注腾讯云开发者  
领取腾讯云代金券

#### 热门产品

域名注册

云服务器

区块链服务

消息队列

网络加速

云数据库

域名解析

云存储

视频直播

#### 热门推荐

人脸识别

腾讯会议

企业云

CDN 加速

视频会议

图像分析

MySQL 数据库

SSL 证书

语音识别

#### 更多推荐

数据安全

负载均衡

短信

文字识别

云点播

商标注册

小程序开发

网站监控

数据迁移