◆ C++ Web 编程

C++ 标准库 →

C++ STL 教程

在前面的章节中,我们已经学习了 C++ 模板的概念。C++ STL(标准模板库)是一套功能强大的 C++ 模板类,提供了通用的模板类和函数,这些模板类和函数可以实现多种流行和常用的算法和数据结构,如向量、链表、队列、栈。

C++ 标准模板库的核心包括以下三个组件:

组件	描述
容器(Containers)	容器是用来管理某一类对象的集合。C++ 提供了各种不同类型的容器,比如 deque、list、vector、map 等。
算法 (Algorithms)	算法作用于容器。它们提供了执行各种操作的方式,包括对容器内容执行初始化、排序、搜索和转换等操作。
迭代器(iterators)	迭代器用于遍历对象集合的元素。这些集合可能是容器,也可能是容器的子集。

这三个组件都带有丰富的预定义函数,帮助我们通过简单的方式处理复杂的任务。

下面的程序演示了向量容器(一个 C++ 标准的模板),它与数组十分相似,唯一不同的是,向量在需要扩展大小的时候,会自动处理它自己的存储需求:

实例

```
#include <iostream>
#include <vector>
using namespace std;
int main()
// 创建一个向量存储 int
vector<int> vec;
int i;
// 显示 vec 的原始大小
cout << "vector size = " << vec.size() << endl;</pre>
// 推入 5 个值到向量中
for(i = 0; i < 5; i++){
vec.push_back(i);
// 显示 vec 扩展后的大小
cout << "extended vector size = " << vec.size() << endl;</pre>
// 访问向量中的 5 个值
for(i = 0; i < 5; i++){
cout << "value of vec [" << i << "] = " << vec[i] << endl;</pre>
// 使用迭代器 iterator 访问值
vector<int>::iterator v = vec.begin();
while( v != vec.end()) {
cout << "value of v = " << *v << endl;</pre>
V++;
```

```
return 0;
}
```

当上面的代码被编译和执行时,它会产生下列结果:

```
vector size = 0
extended vector size = 5
value of vec [0] = 0
value of vec [1] = 1
value of vec [2] = 2
value of vec [3] = 3
value of vec [4] = 4
value of v = 0
value of v = 1
value of v = 2
value of v = 3
value of v = 4
```

关于上面实例中所使用的各种函数,有几点要注意:

- push_back()成员函数在向量的末尾插入值,如果有必要会扩展向量的大小。
- size()函数显示向量的大小。
- begin()函数返回一个指向向量开头的迭代器。
- end()函数返回一个指向向量末尾的迭代器。

◆ C++ Web 编程

C++ 标准库 →



1篇笔记

🕑 写笔记



C++ STL 之 vector 的 capacity 和 size 属性区别

size 是当前 vector 容器真实占用的大小,也就是容器当前拥有多少个容器。

capacity 是指在发生 realloc 前能允许的最大元素数,即预分配的内存空间。

当然,这两个属性分别对应两个方法: resize()和 reserve()。

使用 resize() 容器内的对象内存空间是真正存在的。

使用 **reserve()** 仅仅只是修改了 capacity 的值,容器内的对象并没有真实的内存空间(空间是"野"的)。

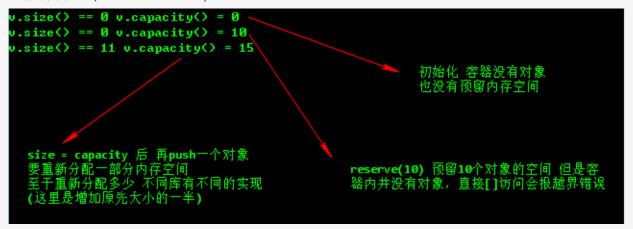
此时切记使用[]操作符访问容器内的对象,很可能出现数组越界的问题。

下面用例子进行说明:

```
#include <iostream>
#include <vector>
```

```
using std::vector;
int main(void)
{
    vector<int> v;
    std::cout<<"v.size() == " << v.size() << " v.capacity() = " << v.capacity() << std::e
ndl;
    v.reserve(10);
    std::cout<<"v.size() == " << v.size() << " v.capacity() = " << v.capacity() << std::e
ndl;
    v.resize(10);
    v.push_back(0);
    std::cout<<"v.size() == " << v.size() << " v.capacity() = " << v.capacity() << std::e
ndl;
    return 0;
}</pre>
```

运行结果为: (win 10 + VS2010)



注: 对于 reserve(10) 后接着直接使用 [] 访问越界报错(内存是野的), 大家可以加一行代码试一下, 我这里没有贴出来。

这里直接用[]访问, vector 退化为数组, 不会进行越界的判断。此时推荐使用 at(), 会先进行越界检查。

相关引申:

针对 capacity 这个属性,STL 中的其他容器,如 list map set deque,由于这些容器的内存是散列分布的,因此不会发生类似 realloc() 的调用情况,因此我们可以认为 capacity 属性针对这些容器是没有意义的,因此设计时这些容器没有该属性。

在 STL 中, 拥有 capacity 属性的容器只有 vector 和 string。

Jacob 10个月前(05-17)