32 How Strings Work in C++ (and how to use them)

需要*pointer*和*array*的前置知识，因为你会发现字符串和数组关系是很紧密的。
首先，字符串是什么？本质上来说字符串就是一组*characters*字符，字符是指字母，数字和符号这些东西，本质上就是文本，对我们来说当然会需要用某种方式把文本的形状和格式在电脑上表示出来。

单一字符、整个段落、一个单词、一堆单词.....所有这些被称为字符串，*string of text*(文本字符串)

C++中有一种数字类型叫*char*，它是*character*的缩写，占用1个字节内存。它很有用，因为能把指针转换为char类型指针，让你根据字节进行指针运算。它对分配内存缓冲区也很有用，如果你想分配1k的内存，你分配1024个char就行了。
它对字符串和文本也很有用，因为C++默认处理的字符方式就是*ASCII*(aski)字符.
字符也可以是大于一个字节的(比如中文、日文)，可以有2-4个字节的字符。

如果我们只用1字节来表示字符，1字节=8bit，这意味着我们有$2^8 = 256$ 种可能，显然数量不够，没法适配所有语言。
所以我们有了*utf-16*，也就是16位的字符编码，这意味着我们有$2^{16} = 65536$ 种可能.

但是在C++基础语言中不适用任何库只是原始数据类型的话，*char*就是1字节。

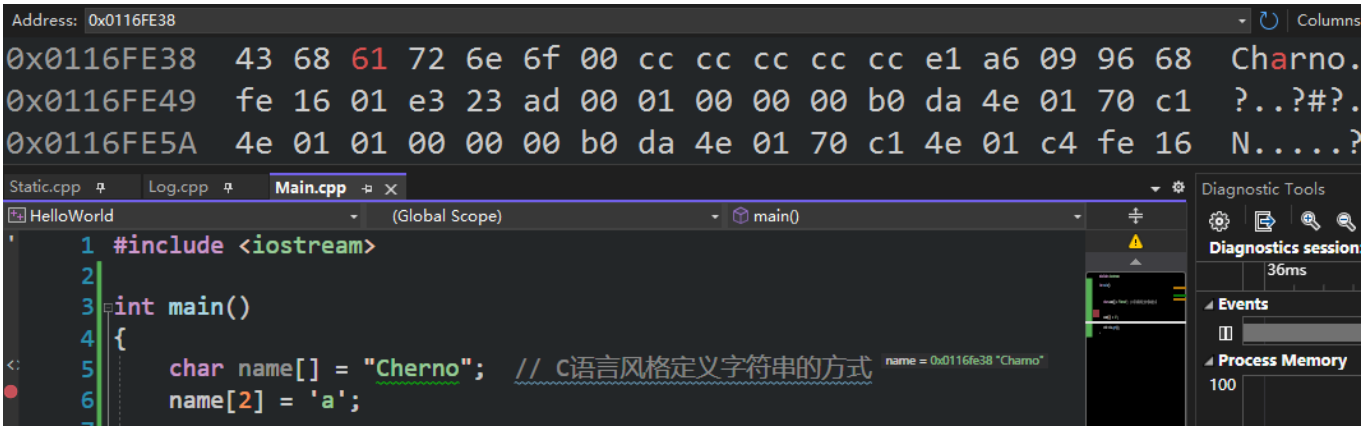
1. const* char

经验法则：如果你不用*new*关键字，那就不要用*delete*

```C++
const char* name = "Cherno";  // C语言风格定义字符串的方式
name[2] = 'a';  // 无法实现，如果你知道不会去修改字符串就加上const，否则就去掉
```

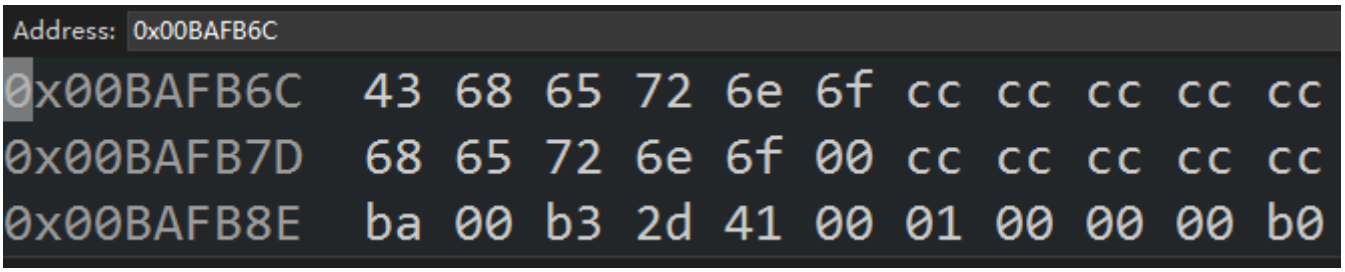
C++11开始，将字符串常量直接赋值给非`const`的`char*`指针被视为不安全的行为。
故用

```C++
char name[] = "Cherno";
```



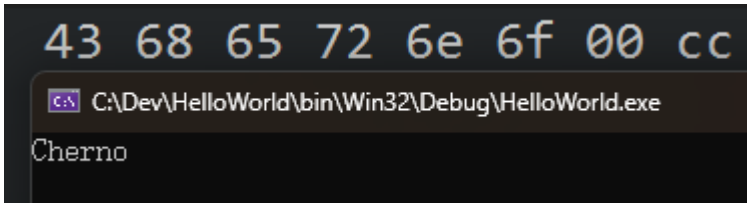
图中给的`00`字符被称为*the null termination character*(空终止符)，这样我们就知道字符串在哪里结束，也便于知道字符串大小

```C++
char name2[6] = { 'C','h','e','r','n','o' };
std::cout << name2 << std::endl;   // Cherno烫烫烫烫烫.....
```



分配了*array guard*，但是没有空终止符，所以cout不知道打印到哪里结束

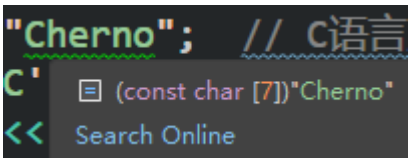
```C++
char name2[7] = { 'C','h','e','r','n','o','\0'}; // 直接写成0也行
```



正常输出

2. std::string

std::string有个接受参数为char指针或者const char指针的构造函数



还有很多内置方法

```C++
std::string name = "Cherno";
std::cout << name << std::endl;          // Cherno
std::cout << name.size() << std::endl;   // 6
```

字符串拼接

```C++
std::string name = "Cherno" + "hello";  //'+': cannot add two pointers
```

上面提到了它们是const char数组，不是真正的string

```C++
std::string name = "Cherno";
name += "hello";   // Chernohello
```

或者

```C++
std::string name = std::string("Cherno") + "hello!";
```

3. 字符串传给其他函数

```C++
void PrintString(const std::string& string) //引用，而不会拷贝,并保证不修改
```