**C#高级特性-不安全的指针**

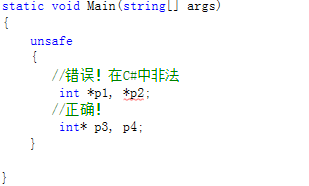
**C#是强类型的、类型安全的语言。在CRL通用运行库的中，不允许用户通过指针直接操作内存，除非添加unsafe关键字，同时在项目配置中选中配置项Properties -> Build->Allow unsafe code，来绕开CRL管理机制来处理指针。**

* **指针为什么是不安全的？**

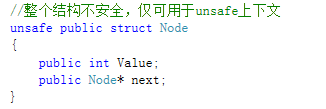
GC回收托管堆上的垃圾内存时，会压缩托管堆，因此可用对象的

内存地址可能重新分配。若已有指针指向该对象，此刻该指针指向的地址内容是未知的，所以指针是不安全的。

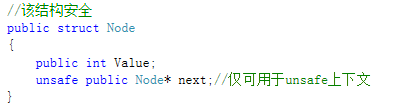
* **关键字unsafe可以修饰不安全的（使用了指针）代码块、结构体、类、类型成员和参数、方法。**
* **与C、C++不同，\*只能用于底层数据类型**

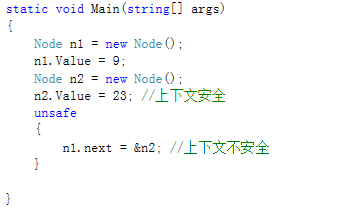
****

* **定义不安全的结构**

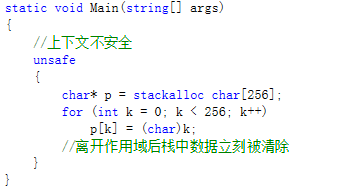
****

* **定义结构中某些不安全的属性**

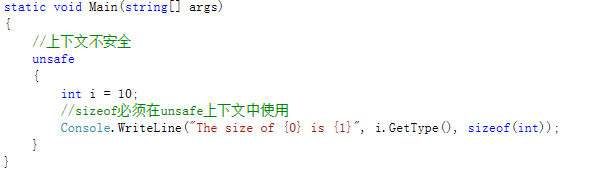
****

****

* **Unsafe上下文里面使用stackalloc关键字动态分配栈内存**

****

* **Unsafe上下文里面使用sizeof关键字获取值类型对象的字节数**

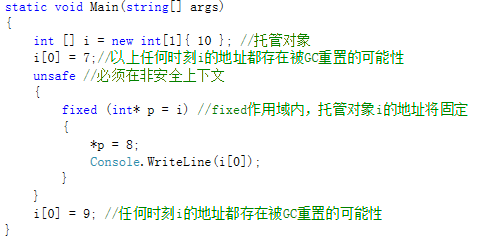
****

* **unsafe上下文里面使用fixed关键字固定托管类型对象的堆地址**

由于引用类型创建的对象都保存在托管堆上，而GC垃圾收集器会

在不确定的时间点回收托管堆上不可用的内存，同时压缩堆内存，存在剩余可用的托管对象的地址被重置的情况。因此在unsafe上下文里面使用指针操作某个托管对象时，存在GC回收机制发生后指针指向的内存数据将是未知的风险。

好在可以用fixed关键字修饰指针操作对象地址的上下文，这样GC将在该上下文里面跳过对此对象内存的地址重置。

****