参考STL allocator进行实现，采用模板编程，支持任意类型。

private:  
 union Slot\_ {  
 value\_type element;  
 Slot\_\* next;  
 };  
  
 typedef char\* data\_pointer\_; //

typedef Slot\_ slot\_type\_;

//Slot要么是value\_type，要么是Slot\*，大小为两个的较大值  
 typedef Slot\_\* slot\_pointer\_;  
  
 slot\_pointer\_ currentBlock\_; //block list头指针  
 slot\_pointer\_ currentSlot\_;

//当前可用的第一个slot的位置，即可以存放value\_type数据的位置  
 slot\_pointer\_ lastSlot\_; //最后一个slot的位置  
 slot\_pointer\_ freeSlots\_; //空闲链表头指针

**将对象的构造分为两个部分：1，分配内存 2，对象构造。分别对应于类的成员函数allocate和construct。**

内存池由一块一块的block组成，block之间通过链表相连。当内存池的内存不够时，通过allocateBlock()申请新的block加入。将每个block按照内存池中的存储类型value\_type划分成一个一个的slot。每个对象内存占用block中的一个slot

内存池创建时，并不分配内存，当有内存需求，即调用allocate时，如果发现内存池中没有内存，则调用allocateBlock分配一个block。

当对象析构之后，并不释放占用的内存空间，而是将其加入freeslot空闲链表中。表明可以在这部分内存构造对象。

有内存请求时，首先判断freeslot是否有空闲内存块存在，如果没有的话，就看分配的block中是否还有空闲块，如果还没有，就新建一个block。

**在内存池运行期间，内存池的大小只会增长不会减小。直到把内存池销毁之后，才会把所有的block释放。**

实现的时候用到的重要的知识：

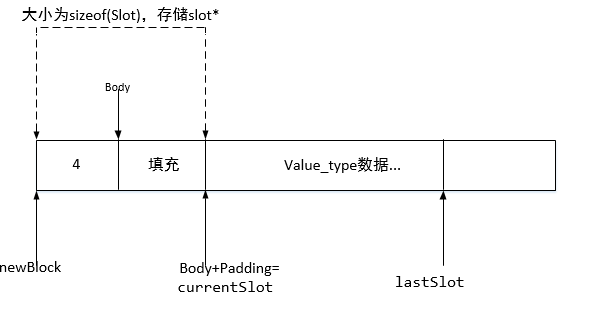
**1、基于STL allocater的架构**

**2、union结构体**

**3、placement new**

**4、reinterpret\_cast进行底层的转换，不同类型的指针之间等。**

每个block的结构



每个block的第一个slot存储一个slot\*指针，指向下一个block

每个slot要么是value，要么是next指针，当空闲时，存储next指针，组成freeslot链表

当存储对象时，存储value。

**关于union中指针的转换**

newBlock指针

每次新建一个block之后，将其指针类型转换成data\_pointer，也就是char\*。

reinterpret\_cast<slot\_pointer\_>(newBlock)->next = currentBlock\_;

然后把char\*转换成slot\_pointer，也就是slot\*，表明每个block的头指针指向一个slot，并且这个slot中存储的元素是一个slot\* next，指向另外一个slot。

currentSlot\_指针

currentSlot\_ = reinterpret\_cast<slot\_pointer\_>(body + bodyPadding)

定义为Slot\*，也就是currentSlot指向slot类型。初始时设为上图位置，之后每使用一个slot，currentSlot指针就后移一个。也比较好理解，每个Slot的大小都是一样的，移动一个Slot的长度，就指向了下一个Slot，也就是currentSlot++。

freeSlots\_指针

最开始的时候，freeslot链表为空，当内存中的一个Slot释放后，就将其加入freeSlot链表。

当内存中这块Slot使用时，指针T\*指向这块内存，释放时：

reinterpret\_cast<slot\_pointer\_>(p)->next = freeSlots\_

将T\*指针p直接转换成slot\_pointer，也就是现在这块内存存储slot类型的数据，并且这个slot现在是slot\_\* next而不是value，直接将原来的值覆盖，加入空闲链表freelist。

使用Union来定义Slot相比于Struct更能节省内存空间，并且不会造成使用上的困难。内存池中维护两个链表，一个是Block链表，一个是空闲块链表。Block链表用于串联各个Block，建立Block链表的这部分内存（每个Block中的第一个Slot）不会改变。

而对于其他Slot，使用的时候只需要存储value\_type类型的数据，只有在空间释放之后才需要串联到空闲块链表，这时候将这块内存转换成Slot\*，并且这块内存存储的是Slot\* next，而不是value\_type。

将Slot定义成

union Slot\_ {  
 value\_type element;  
 Slot\_\* next;  
 };

的目的在于：

1）每个Slot都有足够大的空间存储value\_type的element。因为指针的大小都是4字节。并且只有维护Block链表和空闲块链表时才会用到指针。而其他情况下都需要存储value\_type类型的数据，并且value\_type类型数据很有可能大于4字节。

2）定义成这种形式的话，每个Slot的空间可以存储value\_type，也可以在Slot不用时将其转换成Slot\*组成空闲块链表。这也就是用reinterpret\_cast<>()转换slot\_pointer\_和pointer可行的原因。根据定义，每个Slot就是可以存储这两种类型的数据。