        集合是我们在Java编程中使用非常广泛的，它就像大海，海纳百川，像万能容器，盛装万物，而且这个大海，万能容器还可以无限变大（如果条件允许）。当这个海、容器的量变得非常大的时候，它的初始容量就会显得很重要了，因为挖海、扩容是需要消耗大量的人力物力财力的。同样的道理，Collection的初始容量也显得异常重要。所以：对于已知的情景，请为集合指定初始容量。

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373833) [copy](http://blog.csdn.net/chenssy/article/details/38373833)

[print?](http://blog.csdn.net/chenssy/article/details/38373833)

1. **public** **static** **void** main(String[] args) {
2. StudentVO student = **null**;
3. **long** begin1 = System.currentTimeMillis();
4. List<StudentVO> list1 = **new** ArrayList<>();
5. **for**(**int** i = 0 ; i < 1000000; i++){
6. student = **new** StudentVO(i,"chenssy\_"+i,i);
7. list1.add(student);
8. }
9. **long** end1 = System.currentTimeMillis();
10. System.out.println("list1 time：" + (end1 - begin1));
12. **long** begin2 = System.currentTimeMillis();
13. List<StudentVO> list2 = **new** ArrayList<>(1000000);
14. **for**(**int** i = 0 ; i < 1000000; i++){
15. student = **new** StudentVO(i,"chenssy\_"+i,i);
16. list2.add(student);
17. }
18. **long** end2 = System.currentTimeMillis();
19. System.out.println("list2 time：" + (end2 - begin2));
20. }

        上面代码两个list都是插入1000000条数据，只不过list1没有没有申请初始化容量，而list2初始化容量1000000。那运行结果如下：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373833) [copy](http://blog.csdn.net/chenssy/article/details/38373833)

[print?](http://blog.csdn.net/chenssy/article/details/38373833)

1. list1 time：1638
2. list2 time：921

        从上面的运行结果我们可以看出list2的速度是list1的两倍左右。在前面LZ就提过，ArrayList的扩容机制是比较消耗资源的。我们先看ArrayList的add方法：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373833) [copy](http://blog.csdn.net/chenssy/article/details/38373833)

[print?](http://blog.csdn.net/chenssy/article/details/38373833)

1. **public** **boolean** add(E e) {
2. ensureCapacity(size + 1);
3. elementData[size++] = e;
4. **return** **true**;
5. }
7. **public** **void** ensureCapacity(**int** minCapacity) {
8. modCount++;         //修改计数器
9. **int** oldCapacity = elementData.length;
10. //当前需要的长度超过了数组长度，进行扩容处理
11. **if** (minCapacity > oldCapacity) {
12. Object oldData[] = elementData;
13. //新的容量 = 旧容量 \* 1.5 + 1
14. **int** newCapacity = (oldCapacity \* 3)/2 + 1;
15. **if** (newCapacity < minCapacity)
16. newCapacity = minCapacity;
17. //数组拷贝，生成新的数组
18. elementData = Arrays.copyOf(elementData, newCapacity);
19. }
20. }

        ArrayList每次新增一个元素，就会检测ArrayList的当前容量是否已经到达临界点，如果到达临界点则会扩容1.5倍。然而ArrayList的扩容以及数组的拷贝生成新的数组是相当耗资源的。所以若我们事先已知集合的使用场景，知道集合的大概范围，我们最好是指定初始化容量，这样对资源的利用会更加好，尤其是大数据量的前提下，效率的提升和资源的利用会显得更加具有优势。

**>>>>>>java集合细节一：请为集合指定初始容量**

-----原文出自:<http://cmsblogs.com/?p=1233>,请尊重作者辛勤劳动成果,转载说明出处.