**ArrayList**

1. ArrayList的特性 及 属性



1. 特性
   * 1. RandomAccess : 标记该类的实例 具备下标访问的功能
     2. Serializable : 标记该类的实例对象可以被流化传输
     3. Cloneable : 标记该类的实例对象可以被拷贝，从而创建新的对象(深)
        1. **浅拷贝** ：拷贝后内存中的数据不复制 ， 引用复制一份(数据2份，引用1份)
           1. 引用数据类型

在堆中不备份数据，只是引用拷贝份

* + - * 1. 实现方式

直接引用 传递

A a = new A ();

A b = a;

* + - 1. **深拷贝** ：拷贝后内存中的数据也复制一份(数据2份，引用2份)
         1. 基本数据类型的 ， String

基本数据类型 ：在栈中直接 备份数据

String ： 拷贝后引用虽然指向的是同一份数据， 但是改变值后 并不会 影响之前的数据 ，而是重新创建一个String对象

* + - * 1. 实现方式

对象反序列后，会成为独立的对象

让引用类型，实现cloneable ，然后调clone 方法

1. 属性
   * 1. defaultCapacity = 10

默认构造器list 初始容量 ，如果size > 10 内部会进行扩容,

为什么list要扩容 ？ 数组长度大小不能改变

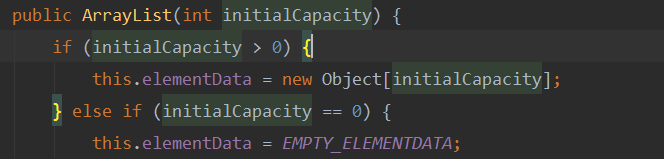
* + 1. empty\_elementData = {}

创建空的new ArrayList(0)时 ， 引用指向的数组。那么无论创建多少个list都是指向这个 数组。从而节省内存

* + 1. defaultCapacity\_empty\_elementData

创建list默认构造器指向的数据

* + 1. elementData 正在存放数据的地方 长度等于数组容量

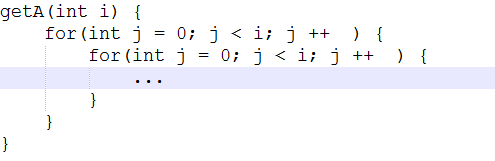


* + 1. 为什么 创建list指定大小跟不指定大小 ，指向的 数组不是同一个

因为list扩容时 ，是在原来容量的大小上扩容 1.5倍 ， 在首次扩容时要根据初始大小

1. 什么是时间复杂度 ： 参数 跟 方法执行时间的 关系
   1. 常量时间复杂度 O(1) : 如list.get(1) 直接取一次
   2. 线性时间复杂度 O(n) : 跟循环有关 如迭代器取数据

指数相关时间复杂度 ：O(n^2) 如 循环嵌套



* 1. 对数相关时间复杂度 ： 二分查找

1. Arraylist 添加元素
2. .
3. .