public class SparseArray<E> implements Cloneable {

private static final Object DELETED = new Object();

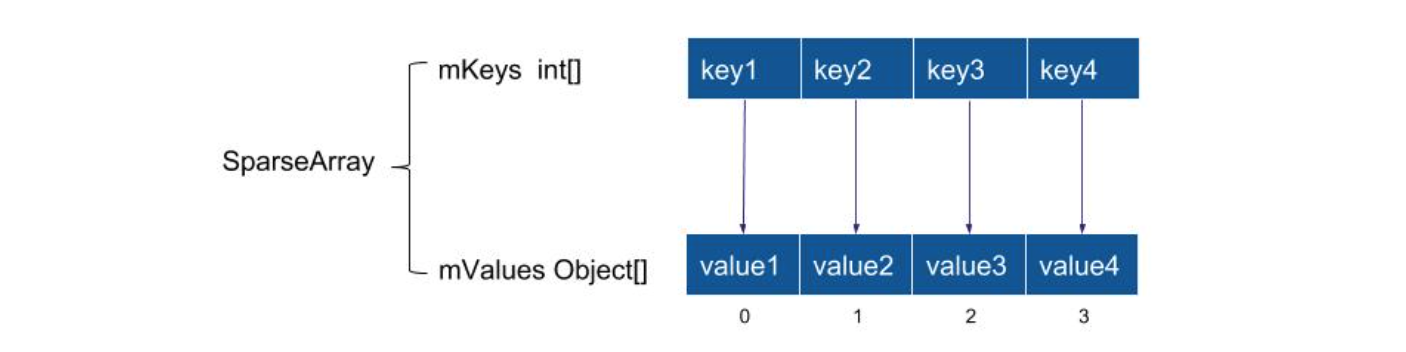
private boolean mGarbage = false; //标记是否存在待回收的键值对

private int[] mKeys;

private Object[] mValues;

private int mSize;

}



SparseArray对应的key只能是int类型，它不会对key进行装箱操作。它使用了两个数组，一个保存key，一个保存value。 从内存使用上来说，SparseArray不需要保存key所对应的哈希值，所以比ArrayMap还能再节省1/3的内存。

SparseArray使用二分查找来找到key对应的插入位置，保证mKeys数组从小到大的排序。

当执行delete()或者removeAt()删除数据的操作，只是将相应位置的数据标记为DELETE，并设置mGarbage=true，而不会直接执行数据拷贝移动的操作。

当执行clear()会清空所有的数据，并设置mGarbage=false；另外有很多时机(比如实际数据大于等于数组容量)都有可能会主动调用gc()方法来清理DELETE数据.

延迟回收机制的好处在于首先删除方法效率更高，同时减少数组数据来回拷贝的次数，比如删除某个数据后被标记删除，接着又需要在相同位置插入数据，则不需要任何数组元素的来回移动操作。可见，对于SparseArray适合频繁删除和插入来回执行的场景，性能很好。