Android为什么要设计出Bundle而不是直接使用HashMap来进行数据传递？

转自凯子哥的 [AndroidDifficultAnalysis](https://github.com/ZhaoKaiQiang/AndroidDifficultAnalysis/blob/master/2.Android%E4%B8%BA%E4%BB%80%E4%B9%88%E8%A6%81%E8%AE%BE%E8%AE%A1%E5%87%BABundle%E8%80%8C%E4%B8%8D%E6%98%AF%E7%9B%B4%E6%8E%A5%E4%BD%BF%E7%94%A8HashMap%E6%9D%A5%E8%BF%9B%E8%A1%8C%E6%95%B0%E6%8D%AE%E4%BC%A0%E9%80%92.md) 。

1.Bundle内部是由ArrayMap实现的，ArrayMap的内部实现是两个数组，一个int数组是存储对象数据对应下标，一个对象数组保存key和value，内部使用二分法对key进行排序，所以在添加、删除、查找数据的时候，都会使用二分法查找，只适合于小数据量操作，如果在数据量比较大的情况下，那么它的性能将退化。而HashMap内部则是数组+链表结构，所以在数据量较少的时候，HashMap的Entry Array比ArrayMap占用更多的内存。因为使用Bundle的场景大多数为小数据量，我没见过在两个Activity之间传递10个以上数据的场景，所以相比之下，在这种情况下使用ArrayMap保存数据，在操作速度和内存占用上都具有优势，因此使用Bundle来传递数据，可以保证更快的速度和更少的内存占用。

2.另外一个原因，则是在Android中如果使用Intent来携带数据的话，需要数据是基本类型或者是可序列化类型，HashMap使用Serializable进行序列化，而Bundle则是使用Parcelable进行序列化。而在Android平台中，更推荐使用Parcelable实现序列化，虽然写法复杂，但是开销更小，所以为了更加快速的进行数据的序列化和反序列化，系统封装了Bundle类，方便我们进行数据的传输。

以上内容来自于对[这个问题](https://github.com/android-cn/android-discuss/issues/142)的思考和总结。