        在实际开发过程中我们经常使用asList讲数组转换为List，这个方法使用起来非常方便，但是asList方法存在几个缺陷：

**一、避免使用基本数据类型数组转换为列表**

        使用8个基本类型数组转换为列表时会存在一个比较有味的缺陷。先看如下程序：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** **void** main(String[] args) {
2. **int**[] ints = {1,2,3,4,5};
3. List list = Arrays.asList(ints);
4. System.out.println("list'size：" + list.size());
5. }
6. ------------------------------------
7. outPut：
8. list'size：1

        程序的运行结果并没有像我们预期的那样是5而是逆天的1，这是什么情况？先看源码：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** <T> List<T> asList(T... a) {
2. **return** **new** ArrayList<>(a);
3. }

        asList接受的参数是一个泛型的变长参数，我们知道基本数据类型是无法发型化的，也就是说8个基本类型是无法作为asList的参数的， 要想作为泛型参数就必须使用其所对应的包装类型。但是这个这个实例中为什么没有出错呢？因为该实例是将int 类型的数组当做其参数，而在Java中数组是一个对象，它是可以泛型化的。所以该例子是不会产生错误的。既然例子是将整个int 类型的数组当做泛型参数，那么经过asList转换就只有一个int 的列表了。如下：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** **void** main(String[] args) {
2. **int**[] ints = {1,2,3,4,5};
3. List list = Arrays.asList(ints);
4. System.out.println("list 的类型:" + list.get(0).getClass());
5. System.out.println("list.get(0) == ints：" + list.get(0).equals(ints));
6. }
7. --------------------------------------------
8. outPut:
9. list 的类型:**class** [I
10. list.get(0) == ints：**true**

        从这个运行结果我们可以充分证明list里面的元素就是int数组。弄清楚这点了，那么修改方法也就一目了然了：将int 改变为Integer。

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** **void** main(String[] args) {
2. Integer[] ints = {1,2,3,4,5};
3. List list = Arrays.asList(ints);
4. System.out.println("list'size：" + list.size());
5. System.out.println("list.get(0) 的类型:" + list.get(0).getClass());
6. System.out.println("list.get(0) == ints[0]：" + list.get(0).equals(ints[0]));
7. }
8. ----------------------------------------
9. outPut:
10. list'size：5
11. list.get(0) 的类型:**class** java.lang.Integer
12. list.get(0) == ints[0]：**true**

**>>>>>>Java细节（2.1）：在使用asList时不要将基本数据类型当做参数。**

**二、asList产生的列表不可操作**

        对于上面的实例我们再做一个小小的修改：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** **void** main(String[] args) {
2. Integer[] ints = {1,2,3,4,5};
3. List list = Arrays.asList(ints);
4. list.add(6);
5. }

        该实例就是讲ints通过asList转换为list 类别，然后再通过add方法加一个元素，这个实例简单的不能再简单了，但是运行结果呢？打出我们所料：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. Exception in thread "main" java.lang.UnsupportedOperationException
2. at java.util.AbstractList.add(Unknown Source)
3. at java.util.AbstractList.add(Unknown Source)
4. at com.chenssy.test.arrayList.AsListTest.main(AsListTest.java:10)

        运行结果尽然抛出UnsupportedOperationException异常，该异常表示list不支持add方法。这就让我们郁闷了，list怎么可能不支持add方法呢？难道jdk脑袋堵塞了？我们再看asList的源码：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **static** <T> List<T> asList(T... a) {
2. **return** **new** ArrayList<>(a);
3. }

        asList接受参数后，直接new 一个ArrayList，到这里看应该是没有错误的啊？别急，再往下看:

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **private** **static** **class** ArrayList<E> **extends** AbstractList<E>
2. **implements** RandomAccess, java.io.Serializable{
3. **private** **static** **final** **long** serialVersionUID = -2764017481108945198L;
4. **private** **final** E[] a;
6. ArrayList(E[] array) {
7. **if** (array==**null**)
8. **throw** **new** NullPointerException();
9. a = array;
10. }
11. //.................
12. }

        这是ArrayList的源码,从这里我们可以看出,此ArrayList不是java.util.ArrayList，他是Arrays的内部类。该内部类提供了size、toArray、get、set、indexOf、contains方法，而像add、remove等改变list结果的方法从AbstractList父类继承过来，同时这些方法也比较奇葩，它直接抛出UnsupportedOperationException异常：

**[java]** [view plain](http://blog.csdn.net/chenssy/article/details/38373877) [copy](http://blog.csdn.net/chenssy/article/details/38373877)

[print?](http://blog.csdn.net/chenssy/article/details/38373877)

1. **public** **boolean** add(E e) {
2. add(size(), e);
3. **return** **true**;
4. }
6. **public** E set(**int** index, E element) {
7. **throw** **new** UnsupportedOperationException();
8. }
10. **public** **void** add(**int** index, E element) {
11. **throw** **new** UnsupportedOperationException();
12. }
14. **public** E remove(**int** index) {
15. **throw** **new** UnsupportedOperationException();
16. }

        通过这些代码可以看出asList返回的列表只不过是一个披着list的外衣，它并没有list的基本特性（变长）。该list是一个长度不可变的列表，传入参数的数组有多长，其返回的列表就只能是多长。所以：

**>>>>>>Java细节（2.2）：不要试图改变asList返回的列表，否则你会自食苦果。**