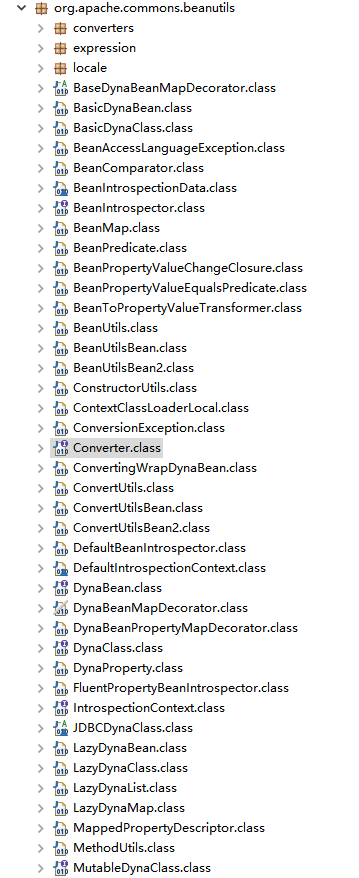
Apache commons

# beanUtils



## Converters包

包下的类都是和转换有关的类。这些类最终实现Converter接口。

### ArrayConverter

数组转换

### StringConverter

String转换

### FileConverter

文件转换

### DateConverter

日期转换

### BigDecimalConverter

BigDecimal转换

## ConverterUtils

## BeanUtils

### BeanUtils.copyProperties(dest, orig);

使用前提是，两个对象的属性名称相同。

dest：目标对象，orig：原对象。

就是将orig的值赋给dest。

### BeanUtils.setProperty

给对象属性设置值，

|  |
| --- |
| public class Test {  public static void main(String[] args) throws ClassNotFoundException, InstantiationException, IllegalAccessException, InvocationTargetException {  String className = "apachecommons.beanUtils.test3.Person";  Class<?> clazz = Class.forName(className);  Object bean = clazz.newInstance();  BeanUtils.setProperty(bean, "name", "hhh");  System.out.println(bean);    Person p = new Person();  BeanUtils.setProperty(p, "name", "hhh");  System.out.println(p);  }  } |

### BeanUtils.populate(p, map);

将map封装到bean中。

# Collections

## Buffer包

## List包

## Map包

### CaseInsensitiveMap

是一个对键的大小不敏感的Map，向该map中put时，会将其转换为小写字符串（底层的转换过程是先转成大写再转成小写）。所以向该map中put元素时，主键为name和主键为NAME是一样的。get元素时也是不区分大小写的。

而hashMap对大小写敏感。

### FixedSizeMap

FixedSizeMap修饰另一个Map防止添加/删除并且固定Map大小。仅允许对已有的键进行值的修改，影响到Map大小的操作都是不被支持的

|  |
| --- |
| HashMap<String, String> src = new HashMap<String, String>();  src.put("name", "jianggujin");  FixedSizeMap fixedSizeMap = (FixedSizeMap) FixedSizeMap.decorate(src);  fixedSizeMap.put("name", "121");  // 不支持  //fixedSizeMap.put("name1", "121");  System.out.println(fixedSizeMap); |

### LinkedMap

LinkedMap实现了一个可维护顺序的Map，键的顺序为插入的顺序。

### MultiKeyMap

多个值作为主键，

### MultiValueMap

值为list类型。List中保存的是多个值。

## Set包

## CollectionUtils

## ListUtils

### ListUtils.union(list1, list2)

将list1和list2拼接在一起放到一个新的list中。

底层用ArrayList，然后addAll(list1)和addAll(list2)。返回这个ArrayList。

### ListUtils.intersection(list1, list2)

Intersection--交叉

将list1和list2中长度短的那个放到HashSet（list1中就算有两个相同的元素也只能保存一次）中，然后遍历长的那一个list，如果HashSet中contains这个元素就将这个元素放到新的list中。最后返回这个新的list。

### ListUtils.subtract(list2, list1)

从list2中去掉list1后剩下放到一个新的list中。先将list1放在HashBag中，遍历list2获取元素e，调用HashBag的remove方法，如果HashBag中有e就移除并返回true，否则就返回false。如果返回false就将这个e放到新的list中，最后返回这个list。

假如list2中有两个相同的e，只有当list1中有两个相同的e时才会将list2中的两个e剪切掉。List1中出现几次，就减去几次。

### ListUtils.sum(list1, list2)

return subtract(union(list1, list2), intersection(list1, list2))

## MapUtils

### MapUtils.getString(map1, "1")

### MapUtils.getDouble(map2, "1")

### MapUtils.isEmpty(map3)

return map == null || map.isEmpty()

# IO

## Comparator包

## Filefilter包

## Input包

## Output包

## CopyUtils

### CopyUtils.copy

和IOUtils.copy是一样的功能，但是CopyUtils.copy不建议使用了。

## FileUtils

### copyFile(final File srcFile, final File destFile);

### copyDirectory(final File srcDir, final File destDir)

### copyInputStreamToFile(final InputStream source, final File destination)

### copyToDirectory(final File src, final File destDir)

将文件或者文件夹copy到文件夹下。

## FilenameUtils

String filename = "E:\\test\\test.txt";

### getPrefix

获取根目录，例如E:\

### getExtension

获取文件后缀，例如txt

### getBaseName

Test

### getPath

test\

### separatorsToWindows

E:\test\test.txt

全路径-windows下的路径

### separatorsToUnix

E:/test/test.txt

全路径-unix下的路径

## IOUtils

### IOUtils.toByteArray

|  |
| --- |
| input = new FileInputStream(new File(pathname));  byteArray = IOUtils.toByteArray(input);  new String(byteArray); |

### IOUtils.buffer

|  |
| --- |
| input = new FileInputStream(new File(pathname));  BufferedInputStream buffer = IOUtils.buffer(input);  String string = IOUtils.toString(buffer); |

### IOUtils.copy

|  |
| --- |
| String pathname = "E:\\test\\test.txt";  FileInputStream inputStream = new FileInputStream(new File(pathname));  FileOutputStream outputStream = new FileOutputStream(new File("E:\\test\\test5.txt"));  IOUtils.copy(inputStream, outputStream);    IOUtils.closeQuietly(inputStream);  IOUtils.closeQuietly(outputStream); |

将test1中的内容复制到test5中。

# Betwixt

# Compress

# Email

# Lang

## ArrayUtils

合并两个数组

|  |
| --- |
| String[] s1 = new String[] { "1", "2", "3" };  String[] s2 = new String[] { "a", "b", "c" };  String[] s = (String[]) ArrayUtils.addAll(s1, s2);  for (int i = 0; i < s.length; i++) {  System.out.println(s[i]);  }  String str = ArrayUtils.toString(s);  str = str.substring(1, str.length() - 1);  System.out.println(str + ">>" + str.length()); |

ArrayUtils.remove(array, 3);

删除数组元素。

## StringUtils

StringUtils.join(list, ",");将list转成String并且用,分隔开。

## NumberUtils

## DateUtils