

Module 7 sample

Enhancing Java core libraries with Collections

7.1 A taste of things to come	1
7.2 Collections and more	3
7.3 Summary	31
Index	32

The Commons components we'll discuss in this module and the next have one thing in common: They cover the deficiencies of the core Java classes by building wrappers around them or extending them to provide sets of useful libraries. These components are BeanUtils, which provides wrappers around the Java Reflection and Introspection API; Collections, which enhances the Java Collection API; and Lang, which provides helper utilities for manipulation of the core Lang package.

In this module, we'll study the Collections component; we'll leave the discussion of the other two components to the next module. All three of these components complement the Java core API. These are very hands-on modules and, in that sense, are different from the rest of this book, because we don't need to cover any background material before showcasing the examples. We'll start with a simple example that uses part of each component to break the ice, and then we'll move to the Collections component.

7.1 A taste of things to come

As we said before, this module and the next differ from the other modules in this book because they deal with components that aren't application-oriented. This means the components of these two modules don't lend themselves to complete application development like the Modeler (module 11) or XPath (module 5) Commons components. The Collections component classes and API can be used as utilities and helper routines to build applications and extend the core Java API in everyday functions.

We'll start with a small example that brings all of these components together, as shown in listing 7.1.

Listing 7.1 Using the Collections, BeanUtils, and Lang components together

```
package com.manning.common.chapter07;

import java.util.Map;
import java.util.HashMap;
import java.util.Iterator;
import java.lang.reflect.Method;

import org.apache.commons.collections.Bag;
import org.apache.commons.collections.bag.HashBag;

import org.apache.commons.beanutils.BeanUtils;
import org.apache.commons.beanutils.PropertyUtils;
```

```

import org.apache.commons.lang.StringUtils;

public class TasteOfThingsV1 {

    private static Map testMap;
    private static TestBean testBean;

    public static void main(String args[]) throws Exception {
        prepareData();

        HashBag myBag = new HashBag(testMap.values());

        System.err.println("How many Boxes? " + myBag.getCount("Boxes"));
        myBag.add("Boxes", 5);
        System.err.println("How many Boxes now? " + myBag.getCount("Boxes"));

        Method method =
            testBean.getClass().getDeclaredMethod("getTestMap", new Class[0]);
        HashMap reflectionMap =
            (HashMap)method.invoke(testBean, new Object[0]);
        System.err.println("The value of the 'squ' key using reflection: "
            + reflectionMap.get("squ"));

        String squ = BeanUtils.getMappedProperty(testBean,
            "testMap", "squ");
        squ = StringUtils.capitalize(squ);
        PropertyUtils.setMappedProperty(testBean,
            "testMap", "squ", squ);
        squ = BeanUtils.getMappedProperty(testBean,
            "testMap", "squ");
        System.err.println("The value of the 'squ' key is: " +
            BeanUtils.getMappedProperty(testBean, "testMap", "squ"));
        String box = (String)testMap.get("box");
        String caps =
            Character.toUpperCase(box.charAt(0)) +
            box.substring(1, box.length());
        System.err.println("Capitalizing boxes by Java: " + caps);
    }

    private static void prepareData() {
        testMap = new HashMap();
        testMap.put("box", "boxes");
        testMap.put("squ", "squares");
        testMap.put("rect", "rectangles");
        testMap.put("cir", "circles");

        testBean = new TestBean();
        testBean.setTestMap(testMap);
    }
}

```

Create TestBean with a TestMap as a property

← Create Bag that counts its contents

① Bag working

② Get map value with Java reflection

③ Combine BeanUtils and Lang to change map value

④ Show capitalization with Java

The code listing not only mixes the three components together, but also shows how equivalent actions would be done in native Java code if these components weren't available. The code creates a TestBean JavaBean