Software Practice 1 - OOP (3) – API

- Access Control Review
- Packages
- Java API
- Documentation

Prof. Joonwon Lee
T.A. Jaehyun Song
Jongseok Kim (42)
T.A. Sujin Oh
Junseong Lee (43)
ACCESS CONTROL REVIEW
Access Control - Bad

```java
public class CreditCard {
    String cardNumber;
    double expenses;
    double charge (double amount) {
        expenses = expenses + amount;
    }
    String getCardNumber (String password) {
        if (password.equals ("SECRET!3*!")) {
            return cardNumber;
        }
        return "jerkface";
    }
}
```
public class CreditCard {
    private String cardNumber;
    private double expenses;
    public double charge (double amount) {
        expenses = expenses + amount;
    }
    public String getCardNumber (String password) {
        if (password.equals ("SECRET!3*!")) {
            return cardNumber;
        }
        return "jerkface";
    }
}
Public vs. Private

- **Public**
  - Others can use this

- **Private**
  - Only the class can use this

public/private applies to any field or method
PACKAGES
What does Package do?

- Each class belongs to a package
- Classes in the same package serve a similar purpose
- Packages are just directories
- Classes in other packages need to be imported
Packages in Java

▪ Defining Packages

```java
package path.to.package.foo;
public class Foo { ... }
```

▪ Using Packages

```java
// import a public class, ‘Foo’, in ‘foo’ package
import path.to.package.foo.Foo;

// import all public classes in ‘foo’ package
import path.to.package.foo.*;
```
package parenttools;

public class BabyFood {
}

package parenttools;

public class Baby {
    public void feed (BabyFood food);
}

Examples
Examples – cont’d

// what’s wrong with this code?
package adult;

public class Parent {
    public static void main (String[] args) {
        Baby baby = new Baby();
        baby.feed (new BabyFood ());
    }
}


// A-ha!

package adult;

import parenttools.Baby;  // get Baby from parenttools
import parenttools.BabyFood;  // get BabyFood from parenttools

public class Parent {
    public static void main (String[] args) {
        Baby baby = new Baby();
        baby.feed (new BabyFood ());
    }
}
// Also this is possible
package adult;

import parenttools.*;    // get all classes from parenttools

public class Parent {
    public static void main (String[] args) {
        Baby baby = new Baby();
        baby.feed (new BabyFood ());
    }
}
Why Packages?

- **Combine similar functionality**
  - org.university.libraries.Library
  - org.university.libraries.Book

- **Separate similar names**
  - shopping.List
  - wish.List
Special Packages

- All classes “see” classes in the same package (no import needed)

- All classes “see” classes in java.lang
  - Example: java.lang.String; java.lang.System
JAVA API
API

- In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building application software.
- Good API makes it easier to develop a computer program

* Extract from wikipedia
Java API

- Java includes lots of packages/classes
- Reuse classes to avoid extra work

http://docs.oracle.com/javase/8/docs/api/
Arrays with items

- Create the array bigger than you need
- Track the next “available” slot

```java
Book[] books = new Book[10];
int nextIndex = 0;

books[nextIndex ++] = b;
```
Arrays with items

- Create the array bigger than you need
- Track the next “available” slot

```java
Book[] books = new Book[10];
int nextIndex = 0;

books[nextIndex++] = b;
```

What if the library expands?
import java.util.Arrays;

/*
 * Extend the capability of this library by [additional capability].
 */

int cap;
String[] books;

public void extendCapacity (int cap) {
    this.cap += cap;
    this.books = Arrays.copyOf (this.books, this.cap);
}
# Arrays.copyOf()

<table>
<thead>
<tr>
<th>Modifier and Type</th>
<th>Method and Description</th>
</tr>
</thead>
</table>
| static &lt;T&gt; T[] | ```
copyOf (T[] original, int newLength)
``` |

Copies the specified array, truncating or padding with nulls (if necessary) so the copy has the specified length
Digression...

- What is $T$?
  - Generic class or interface, so called *generic type*, that is parameterized over types

```java
public class Box<T> {
    // T stands for “Type”
    private T t;
    public void set (T t) { this.t = t; }
    public T get () { return t; }
}

Box<String> box = new Box<String>();
```
public interface Pair<K, V> {
    public K getKey();
    public V getValue();
}

public class OrderedPair<K, V> implements Pair<K, V> {
    private K key;
    private V value;
    public OrderedPair (K key, V value) {
        this.key = key;
        this.value = value;
    }
    public K getKey () { return this.key; }  
    public V getValue () { return this.value;}
}

Pair<String, Integer> p1 = new OrderedPair<String, Integer> ("Even", 8);
Pair<String, String> p2 = new OrderedPair<String, String> ("hello", "world");
By the way,

- There must be more efficient API, because TA says Java is easy to develop!
By the way,

- There must be more efficient API, because TA says Java is easy to develop!

- Sure it is! (all in `java.util` package)
By the way,

- There must be more efficient API, because TA says Java is easy to develop!

- Sure it is! (all in java.util package)

What the…
By the way,

- There must be more efficient API, because TA says Java is easy to develop!

- Sure it is! (all in `java.util` package)
List\(<E>\)

- **Ordered collection (also known as sequence)**

- **In Java, List**
  - is an interface to precise control over where in the list each element is inserted
  - enables user to access elements by their integer index
  - enables user to search for elements in the list
ArrayList\<E\>  

- Resizable-array implementation of List interface  
- Most popular implementation  
- With this class, you don’t need to implement `extendCap` method!  
  - Because ArrayList already has the functionality!  

- Example: [http://beginnersbook.com/2013/12/java-arraylist/](http://beginnersbook.com/2013/12/java-arraylist/)
Map\textless K,V\textgreater

- An object that maps keys to values.
- A map cannot duplicate keys
  - Each key can map to at most one value

- In Java, Map
  - is also an interface to map between keys and values
  - enables user to access elements by their keys
HashMap\(<K,V>\)

- Hash table based implementation of Map interface
- Most popular implementation
- HashMap has the functionality of checking existence of keys and values

**Iterator<**E**>**

- An iterator over a collection
- Iterator allow the caller to **remove elements** from the underlying collection during the iteration with well-defined semantics

- **Example with ArrayList:**
  
Java API

- As well as these APIs, there are huge number of prebuilt APIs in Java
- Your ability as a Java coder will be determined whether you can apply the proper APIs into the right position or not
- Don’t hesitate to know more APIs
- Furthermore, understand how those APIs run and when to use them
JAVADOCC
Many software engineers, even if some senior engineers, are bothered to make documents of their implementation.

However, high quality documentating skill cannot be too important for everyone to be a good engineer.

Then how can we make well-made documentations for Java implementation?
Javadoc

- Javadoc is a documentation generator created by Sun Microsystems for Java language (currently Oracle Cooperation) for generating API documentation in HTML format from Java source code.

- Most of all, perfectly easy to use
Syntax of Javadoc

- All contents of Javadoc is represented in the comment starting with two star marks
- The comments for Javadoc must be located right above of each class or method
- Each line has to start with a star mark
- A word starting with @ means that it is Javadoc keyword
  - Closed braces({ … }) are usually used for using keyword
Keywords of Javadoc

- **Contents without keyword**
  - The main explanation about the class/method

- **@param [parameter_name] [description]**
  - Explain about the [parameter_name] with [description]

- **@return [description]**
  - Explain about the return value with [description]

- **@link [package.class#member] [label]**
  - Link to [package.class#member] document in Javadoc

- **See more:**
  [http://docs.oracle.com/javase/7/docs/technote/tools/windows/javadoc.html](http://docs.oracle.com/javase/7/docs/technote/tools/windows/javadoc.html)
/**
 * This class is for practice
 * @author SKKU
 */

public class Test {

/**
 * This method prints the String “display”
 * @param str the str is {@link java.lang.String String} which will be printed
 * @param x the x is unused variable
 * @return return true or Failed
 */

    public boolean disp(String str, int x) {
        System.out.println("display");
        return true;
    }
}
[Lab – Practice #4]

- Implement HashArray
  - package HashArray
  - public class HashArray

- Implement 5 methods
  - boolean `addition`(key, value)
  - boolean `search`(key, value)
  - boolean `delete`(key, value)
  - ArrayList `getArray`(key)
  - int `getCount`(key)

- Use two external Library
  - HashMap
  - ArrayList
Upload to i-Campus

- Compress your HashArray.java file to zip file
- File name: studentID_lab04.zip

Due date

- Today 23:59:59
  - Class 42 (4/02 Monday)
  - Class 43 (4/04 Wednesday)
- Penalty: -10% of each lab score per one day