실습 10

Java 패키지와 HTML 문서화를 위한 주석문

목 적

- Java 패키지 이해
- Java 컴파일러 및 가상기계(virtual machine) 작동 이해
- Java 패키지 사용시 개발환경 설정 이해
- 클래스 경로(class path) 설정 이해
- javadoc을 이용한 HTML 문서 생성 이해

실습결과 응답 링크: https://bit.ly/3FDNP46

Java 패키지와 컴파일 및 실행 (1)

1. 아래 주어진 PPointTest 클래스와 PPoint 클래스를 살펴 본다. 두 클래스 사이는 어떤 관계(relationship)에 있는가? 각 클래스는 어느 패키지에 속하는가?

2. C:\JavaLab 디렉토리와 C:\Temp 디렉토리를 생성한다.

3. C:\JavaLab 디렉토리에 pkgtest 부디렉토리를 생성하고 pkgtest에는 kr\ac\kookmin\cs 하부 디렉토리 구조를 만든다. Java 패키지 실습을 위하여 패키지를 저장할 베이스 디렉토리로 C:\JavaLab\pkgtest 디렉토리를 사용한다. PPointTest.java 파일은 C:\JavaLab\pkgtest 디렉토리에 저장하고, PPoint.java 파일은 C:\JavaLab\pkgtest\kr\ac\kookmin\cs 디렉토리에 저장한다.

4. C:\JavaLab\pkgtest 디렉토리에서 다음 명령을 실행해 본다. 각 명령이 정상적으로 실행되는가? 정상적으로 실행되는 경우 각 명령이 어떤 작업을 수행하는지 설명하라. 에러가 발생하면 그 이유를 설명하라.

   ```
   > javac PPointTest.java
   > java PPointTest
   ```

5. PPoint.java와 PPoint.class 파일을 C:\JavaLab\pkgtest\kr\ac\kookmin\cs 디렉토리에서 C:\Temp 디렉토리로 이동하고 C:\JavaLab\pkgtest\PPointTest.class 파일을 C:\Temp 디렉토리로 이동한다. C:\JavaLab\pkgtest 디렉토리에서 다음 명령을 실행한다. 컴파일이 정상적으로 실행되는가? 컴파일 에러가 발생하는 경우 그 이유를 설명하라.

   ```
   > javac PPointTest.java
   ```

6. PPoint.class 파일을 C:\Temp 디렉토리에서 C:\JavaLab\pkgtest\kr\ac\kookmin\cs 디렉토리로 이동한 후 C:\JavaLab\pkgtest 디렉토리에서 문제 5의 컴파일 명령을 실행한다. 정상적으로 컴파일되는가? 그 이유는 무엇인가?

7. C:\JavaLab\pkgtest\kr\ac\kookmin\cs\PPoint.class 파일과 C:\JavaLab\pkgtest\PPointTest.class 파일을 삭제한다. PPoint.java 파일을 C:\Temp 디렉토리에서 C:\JavaLab\pkgtest\kr\ac\kookmin\cs 디렉토리로 이동한 후 C:\JavaLab\pkgtest 디렉토리에서 다시 문제 5의 컴파일 명령을 실행한다. PPointTest.java를 컴파일할 때 에러가 발생하는가? javac가 어떤 작업을 수행하는지 설명하라.

8. PPointTest.java 파일을 C:\JavaLab\pkgtest\kr 디렉토리로 이동하고, C:\JavaLab\pkgtest\PPointTest.class 파일을 삭제한다.

9. C:\JavaLab\pkgtest 디렉토리에서 다음 명령을 실행해 본다. 각 명령이 어떤 작업을 수행하는지 설명하라. 명령에서 에러가 발생하는 경우 그 이유를 설명하고 어떻게 하여야 정상적으로 실행할 수 있는지 설명하라.

   ```
   > javac kr\PPointTest.java
   > cd kr
   > java PPointTest
   ```

```
// ***********************************************************
//    PPoint.java
// ***********************************************************

package kr.ac.kookmin.cs;

public class PPoint {
   int xA;
   int yA;
   public PPoint(int x, int y) {
      xA = x;
      yA = y;
   };
   public int getX() {
      return xA;
   }
   public int getY() {
      return yA;
   }
}


// ***********************************************************
//    PPointTest.java
// ***********************************************************

import kr.ac.kookmin.cs.*;

class PPointTest {
   public static void main(String args[]) {
      PPoint aObj = new PPoint(10, 20);
      System.out.println("aObj(x, y) = " + aObj.getX() + ", "+ aObj.getY());
   }
}
```

Java 패키지와 컴파일 및 실행 (2)

1. C:\JavaLab\pkgtest의 하부 디렉토리에 있는 모든 클래스 파일(PPointTest.class와 PPoint.class)을 삭제하고, PPointTest.java 파일의 첫 라인에 "package kr.otherpkg;"을 삽입한다. PPointTest 클래스와 PPoint 클래스는 어느 패키지에 속하는가?

2. PPointTest.java 소스파일을 PPointTest 클래스의 패키지에 대응되는 디렉토리로 이동하려면 어떻게 하여야 하는지 기술하라. (힌트: 베이스 디렉토리(C:\JavaLab\pkgtest)를 기준으로 하여 패키지에 대응되는 디렉토리 구조를 만든다.) PPointTest.java 소스파일을 패키지 대응 디렉토리로 이동한다.

3. 베이스 디렉토리(C:\JavaLab\pkgtest)로 이동한다. 베이스 디렉토리에서 PPointTest.java 소스파일을 컴파일하고 실행하려면 javac와 java 명령을 어떻게 주어야 하는지 보이고 그 이유를 설명하라. (힌트: 컴파일할 때는 컴파일할 소스파일에 대한 경로 정보를 주어야 하며, 프로그램 내에서 다른 클래스를 사용하거나 java 명령으로 클래스를 로드하여 실행하려면 클래스가 속한 패키지 정보를 주어야 한다.) PPointTest.java 소스파일을 컴파일하고 실행하여 본다.

4. C:\JavaLab 디렉토리 밖의 아무 디렉토리로 이동한다. 이 디렉토리에서 다음 명령을 각각 실행해 본다. 각 명령이 정상적으로 실행되는가 아니면 에러가 발생하는가? 그 이유는 무엇인가?

   ```
   > java kr.otherpkg.PPointTest

   > java -classpath C:\JavaLab\pkgtest kr.otherpkg.PPointTest
   ```

5. C:\JavaLab\pkgtest의 하부 디렉토리에 있는 모든 클래스 파일(PPointTest.class와 PPoint.class)을 삭제하고, C:\JavaLab 디렉토리 밖의 아무 디렉토리로 이동한다. 이 디렉토리에서 다음 컴파일 명령을 실행해 본다. 각 명령이 정상적으로 실행되는가 아니면 에러가 발생하는가? 그 이유는 무엇인가?

   ```
   > javac C:\JavaLab\pkgtest\kr\otherpkg\PPointTest.java
   > javac -classpath C:\JavaLab\pkgtest C:\JavaLab\pkgtest\kr\otherpkg\PPointTest.java
   ```

6. C:\JavaLab\pkgtest의 하부 디렉토리에 있는 모든 클래스 파일(PPointTest.class와 PPoint.class)을 삭제한 후, set 명령을 사용하여 다음과 같이 CLASSPATH 환경변수를 설정하고 어떤 베이스 디렉토리가 설정되어 있는지 살펴 본다.

   ```
   > set CLASSPATH
   > set CLASSPATH=C:\JavaLab\pkgtest;%CLASSPATH%
   > set CLASSPATH
   ```

   C:\JavaLab 디렉토리 밖의 아무 디렉토리로 이동한다. 이 디렉토리에서 다음 두 명령을 실행해 본다. 각 명령이 정상적으로 실행되는가 아니면 에러가 발생하는가? 그 이유는 무엇인가?

   ```
   > javac C:\JavaLab\pkgtest\kr\otherpkg\PPointTest.java
   > java kr.otherpkg.PPointTest
   ```

Java 패키지와 컴파일 및 실행 (3)

1. 현재의 명령창을 닫고 새로운 명령창을 연다. C:\JavaLab\pkgtest 디렉토리의 모든 파일과 디렉토리를 삭제한 후, 처음에 준 PPointTest.java와 PPoint.java 소스파일을 pkgtest 디렉토리에 저장한다. C:\JavaLab\pkgtest 디렉토리에서 다음 명령을 실행한다.

   ```
   > javac PPoint.java
   ```

   컴파일이 정상적으로 실행되는가? dir 명령으로 현재 디렉토리에 무엇이 있는가 살펴 본다. PPoint.class 파일이 보이는가? 이 것은 무엇을 말해 주는가?

2. C:\JavaLab\pkgtest 디렉토리에서 PPoint.class 파일을 삭제한 후 다음 명령을 실행한다.

   ```
   > javac –d . PPoint.java (또는 javac -d C:\JavaLab\pkgtest PPoint.java)
   ```

   컴파일이 정상적으로 실행되는가? dir 명령으로 현재 디렉토리에 생성된 부디렉토리를 살펴 본다. PPoint.class 파일이 어디에 있는가? 이 것은 무엇을 말해 주는가?

3. C:\JavaLab\pkgtest 디렉토리에서 다음 명령을 실행한다.

   ```
   > javac PPointTest.java
   ```

   컴파일이 정상적으로 실행되는가? PPoint.class 파일이 패키지에 대응되는 디렉토리에 있는지 디렉토리를 살펴 본다. PPoint.java 파일을 임의의 다른 디렉토리로 이동한다. 다시 위 명령을 실행한다. 컴파일이 정상적으로 실행되는가? 정상적으로 실행되면 다음 명령으로 프로그램이 정상적으로 실행되는지 확인한다.

   ```
   > java PPointTest
   ```

   본 문제에서 PPoint.java 소스파일이 어디에 저장되었는지 살펴 본다. 본 문제에서 처음 PPoint.java를 컴파일할 때는 PPoint.java가 PPointTest.java와 같은 디렉토리(즉, PPointTest.java가 속한 디폴트 패키지)에 저장되어 있었으며 컴파일 에러가 발생하였으나 PPoint.java를 다른 디렉토리로 이동한 후 정상적으로 컴파일 되었다. 이 것은 무엇을 말해 주는가?

4. PPoint.java를 C:\JavaLab\pkgtest 디렉토리로 원위치 시키고 PPointTest.class 파일을 삭제한다. PPointTest.java 파일의 첫 라인에 “package kr.otherpkg;”를 삽입한다. 다음 명령을 실행한다.

```
> javac PPointTest.java
```

명령이 정상적으로 실행되는가? `PPointTest.class` 파일이 어느 디렉토리에 있는지 살펴 본다. 다음 명령으로 프로그램을 실행한다. 에러가 발생하는가? 그 이유는 무엇인가?

```
> java PPointTest
```

5. `C:\JavaLab\pkgtest` 디렉토리에서 다음 명령을 실행한다.

```
> javac -d . PPointTest.java
  (또는 javac -d C:\JavaLab\pkgtest PPointTest.java)
> java kr.otherpkg.PPointTest
```

명령이 정상적으로 실행되는가 아니면 에러가 발생하는가? 그 이유는 무엇인가?

6. `C:\JavaLab` 디렉토리와 이 디렉토리에 포함되어 있는 모든 부디렉토리와 파일을 삭제한다.

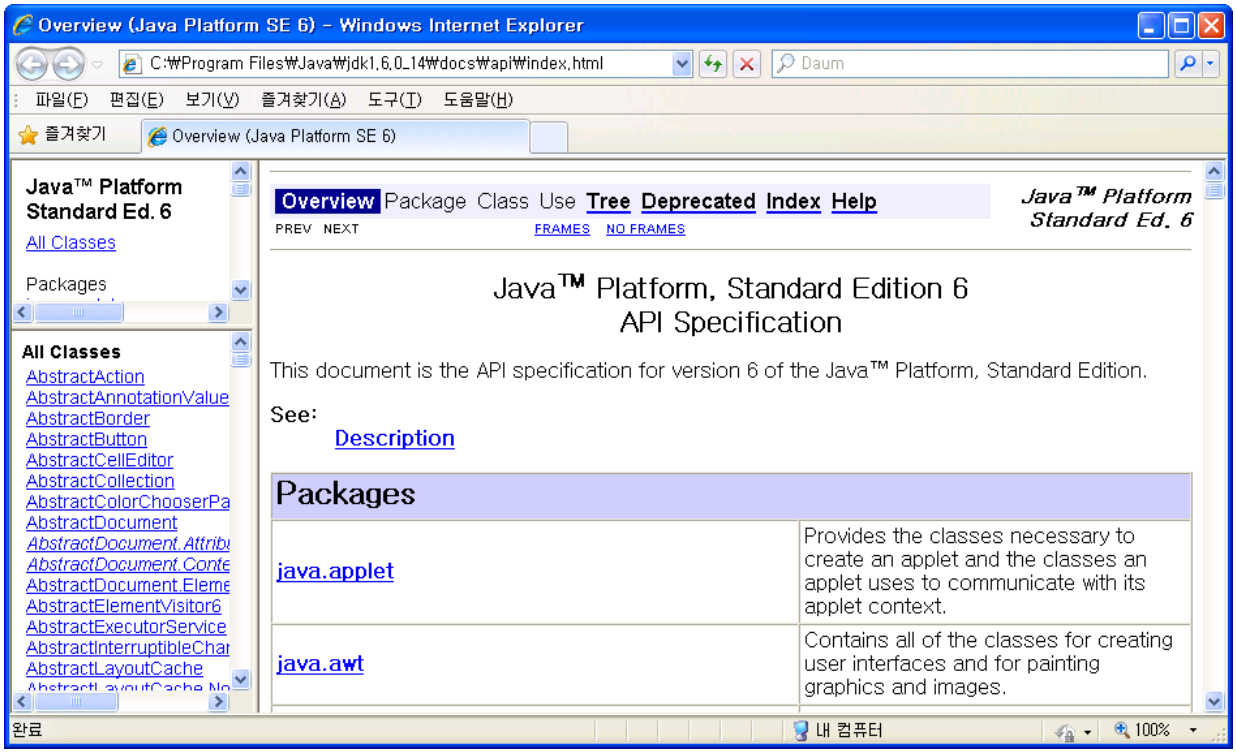
Eclipse와 Java 패키지

1. Eclipse를 실행한다. Package Explorer view에서 JavaLab 프로젝트를 만들고 Windows 탐색기로부터 패키지의 소스파일(예, PPoint.java)을 찾아서 왼클릭하여 프로젝트의 아무 패키지(예, 디폴트 패키지)에 끌어서(drag-and-drop) 파일을 패키지에 복사한다. 다음 Package Explorer view에서 PPoint.java를 더블클릭하여 Java editor로 불러 온다. 소스파일이 자신의 패키지와 다른 패키지에 있으면 편집기(editor)에 파일이 나타나면서 에러를 나타내는 전구모양의 팁이 나타난다. 전구를 왼클릭하면 맞는 패키지로 이동할 것인지 등을 물어 온다. 맞는 항목을 찾아서 더블클릭하면 Package Explorer 뷰에 자동으로 맞는 패키지가 생성되면서 소스파일이 이동하는 것을 볼 수 있으며, Resource Perspective를 보면 이 패키지에 대응되는 디렉토리 구조가 생성된 것을 알 수 있다. 이와 같은 방법으로 PPointTest.java를 복사하여 Eclipse에서 얼마나 쉽게 패키지를 생성하고 다룰 수 있도록 하는지 실습 한다.

2. 패키지를 만드는 또 다른 방법은 다음과 같다. Java 프로젝트 이름 위에서 오른 클릭하여 New → Package를 선택한다. New Java Package 생성 다이어로그가 나타나면 Name 항목에 생성하고자 하는 패키지 이름(예, kr.ac.kookmin.cs)을 입력한다. 패키지가 Package Explorer 뷰에 나타날 것이며, Resource Perspective를 보면 이 패키지에 대응되는 디렉토리 구조가 생성된 것을 알 수 있다.

HTML 문서 생성을 위한 주석문 (Documentation Comments)

JDK에서 제공하는 javadoc은 소스코드로부터 바로 HTML 문서를 생성해주는 매우 유용한 도구(tool)이다. Java 개발자가 참고하는 다음 API 문서도 javadoc을 이용해 Java 라이브러리의 소스코드로부터 생성한 것이다.



ArrayGrowTest 클래스는 원소 개수가 증가하여 보다 큰 크기의 새로운 배열 객체를 할당하고 이전 정보를 복사해야 할 때 올바른 방법과 잘못된 방법을 보여주는 클래스이며, 메쏘드가 javadoc 주석문으로 문서화되어 있다.

1. ArrayGrowTest.java가 저장되어 있는 디렉토리로 이동한다.

2. 소스코드로부터 생성된 HTML 파일을 저장할 디렉토리 이름을 docDir이라 하자. 다음 javadoc 명령은 현재 디렉토리의 docDir 부디렉토리에 ArrayGrowTest.java로부터 HTML 파일을 생성하여 저장한다. docDir 부디렉토리를 살펴 보면 여러 파일이 생성된 것을 볼 수 있다. docDir 부디렉토리의 index.html 파일을 두번 클릭하여 열어 본다. 생성된 HTML 문서에 소스코드에 주어진 메쏘드 주석이 보이는가? 보이지 않는다면 그 이유는 무엇인가? (힌트: 가시성을 생각하라.)

```
> javadoc -d docDir  ArrayGrowTest.java
```

3. 소스코드에 주어진 메쏘드 주석문을 HTML 파일에 포함되게 하려면 javadoc 명령에 어떤 옵션을 추가해야 하는가? 추가된 옵션과 함께 HTML 파일을 생성하기 위한 javadoc 명령을 기술하라.

4. 기술한 javadoc 명령을 실행한 후, docDir 부디렉토리의 index.html 파일을 두번 클릭하여 열어 본다. 생성된 HTML 문서에 소스코드에 주어진 메쏘드 주석이 보이는가? 보인다면 웹브라우저에서 HTML 문서의 여러 링크를 클릭하여 어떤 문서가 보이는지 살펴 본다. 또한 docDir 부디렉토리에 어떤 파일들이 생성되었는지 살펴 본다. 보이지 않는다면 메쏘드 주석이 보일 때까지 적절한 옵션을 찾아 javadoc 명령을 재시도해 본다.

5. 앞에 주어진 PPointTest.java, PPoint.java 파일을 현재 디렉토리에 복사한 후, PPointTest 클래스와 PPoint 클래스의 소스코드에 패키지, 클래스, 메쏘드에 대한 javadoc 주석문을 추가한다. javadoc 명령으로 PPointTest 클래스와 PPoint 클래스의 소스코드로부터 HTML 파일을 생성한다. 생성된 파일은 ppointDocDir 부디렉토리에 저장한다. 패키지, 클래스, 메쏘드에 대한 주석문 설명이 브라우저에 나타나는가?

6. 현재 디렉토리에 PPointTest.java, PPoint.java의 최종 버전, ppointDocDir 부디렉토리, docDir 부디렉토리만 남겨두고, 다른 파일이나 부디렉토리를 삭제한다. 다음 현재 디렉토리를 압축(zip)하여 GitHub 저장소에 저장한 후, GitHub 저장소의 URL을 제출한다.

```
/**
   @version 1.01 2004-02-21
   @author C H
*/

import java.lang.reflect.*;
import java.util.*;

public class ArrayGrowTest {
   public static void main(String[] args) {
      int[] a = { 1, 2, 3 };
      a = (int[]) goodArrayGrow(a);
      arrayPrint(a);

      String[] b = { "Tom", "Dick", "Harry" };
      b = (String[]) goodArrayGrow(b);
      arrayPrint(b);

      System.out.println("The following call will generate an exception.");
      b = (String[]) badArrayGrow(b);
   }

   /**
      This method attempts to grow an array by allocating a
      new array and copying all elements.
      @param a the array to grow
      @return a larger array that contains all elements of a.
      However, the returned array has type Object[], not
      the same type as a
   */
```

```
   static Object[] badArrayGrow(Object[] a) {
      int newLength = a.length * 11 / 10 + 10;
      Object[] newArray = new Object[newLength];
      System.arraycopy(a, 0, newArray, 0, a.length);
      return newArray;
   }

   /**
      This method grows an array by allocating a
      new array of the same type and copying all elements.
      @param a the array to grow. This can be an object array
      or a fundamental type array
      @return a larger array that contains all elements of a.

   */
   static Object goodArrayGrow(Object a) {
      Class cl = a.getClass();
      if (!cl.isArray()) return null;
      Class componentType = cl.getComponentType();
      int length = Array.getLength(a);
      int newLength = length * 11 / 10 + 10;

      Object newArray = Array.newInstance(componentType, newLength);
      System.arraycopy(a, 0, newArray, 0, length);
      return newArray;
   }


   /**
      A convenience method to print all elements in an array
      @param a the array to print. can be an object array
      or a fundamental type array
   */
   static void arrayPrint(Object a) {
      Class cl = a.getClass();
      if (!cl.isArray()) return;
      Class componentType = cl.getComponentType();
      int length = Array.getLength(a);
      System.out.print(componentType.getName()
         + "[" + length + "] = { ");
      for (int i = 0; i < Array.getLength(a); i++)
         System.out.print(Array.get(a, i) + " ");
      System.out.println("}");
   }

}
```