패키지

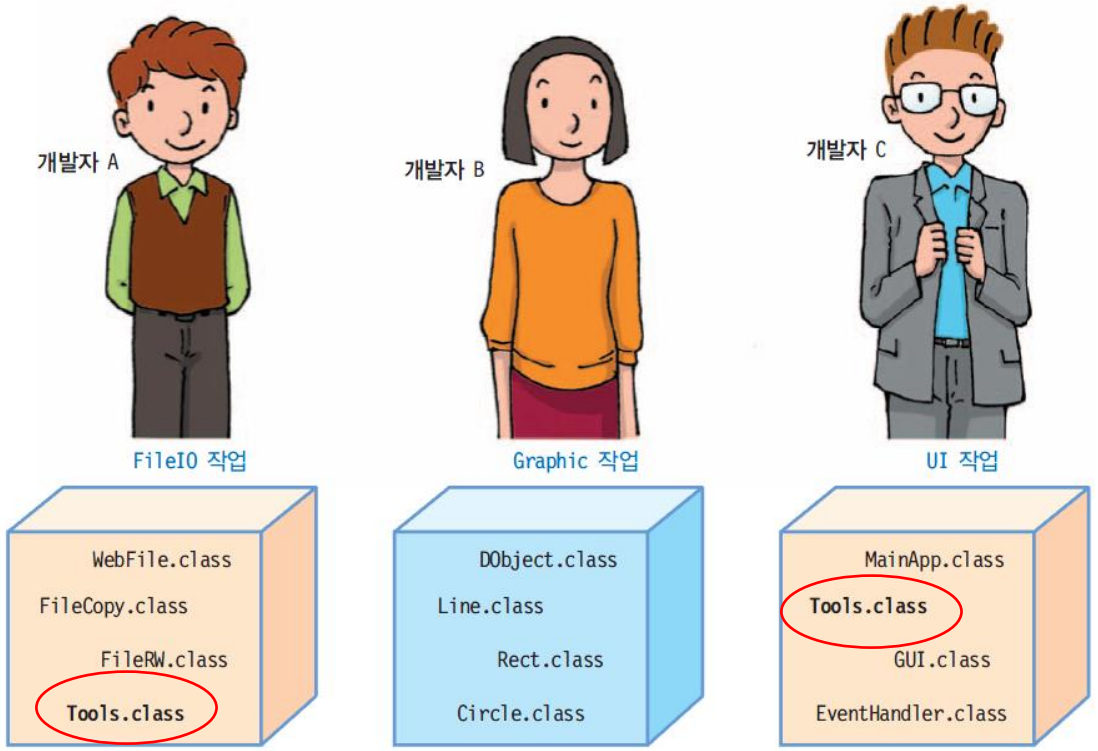
컴퓨터공학전공
박요한

수업내용

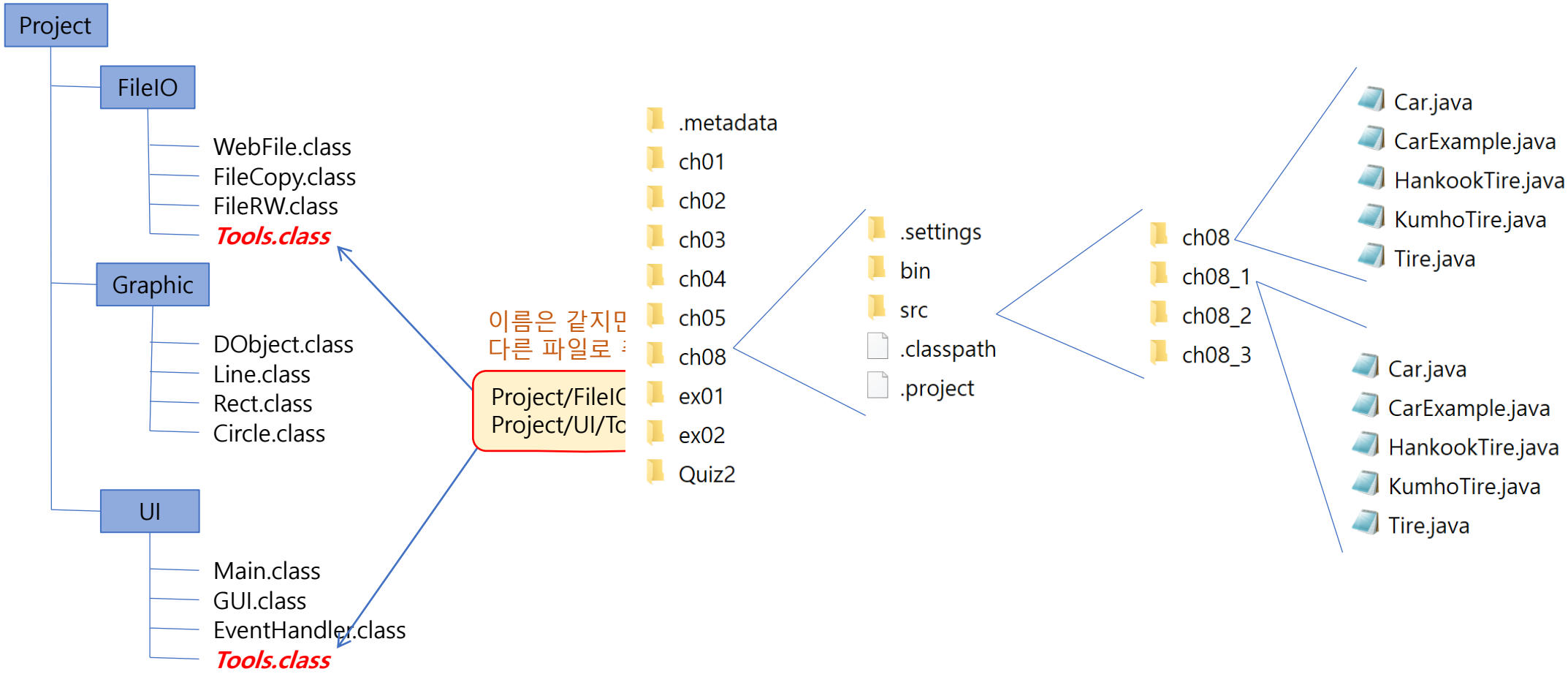
- 패키지의 개념
- 기본 API 클래스
  - ✓Object
  - ✓Wrapper
  - ✓String, StringBuffer, StringTokenizer
  - ✓Math

패키지 개념과 필요성

- 3명이 분담하여 자바 응용프로그램을 개발하는 경우, 동일한 이름의 클래스가 존재할 가능성 있음 -> 합칠 때 오류발생



디렉터리로 각 개발자의 코드 관리(패키지)

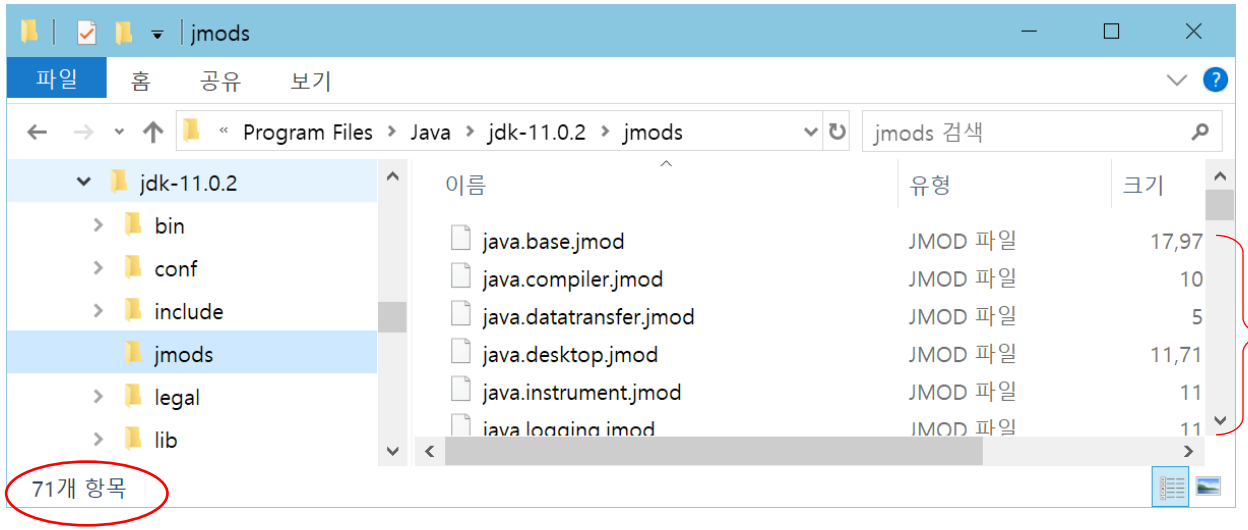


자바의 모듈(module)과 패키지(package)

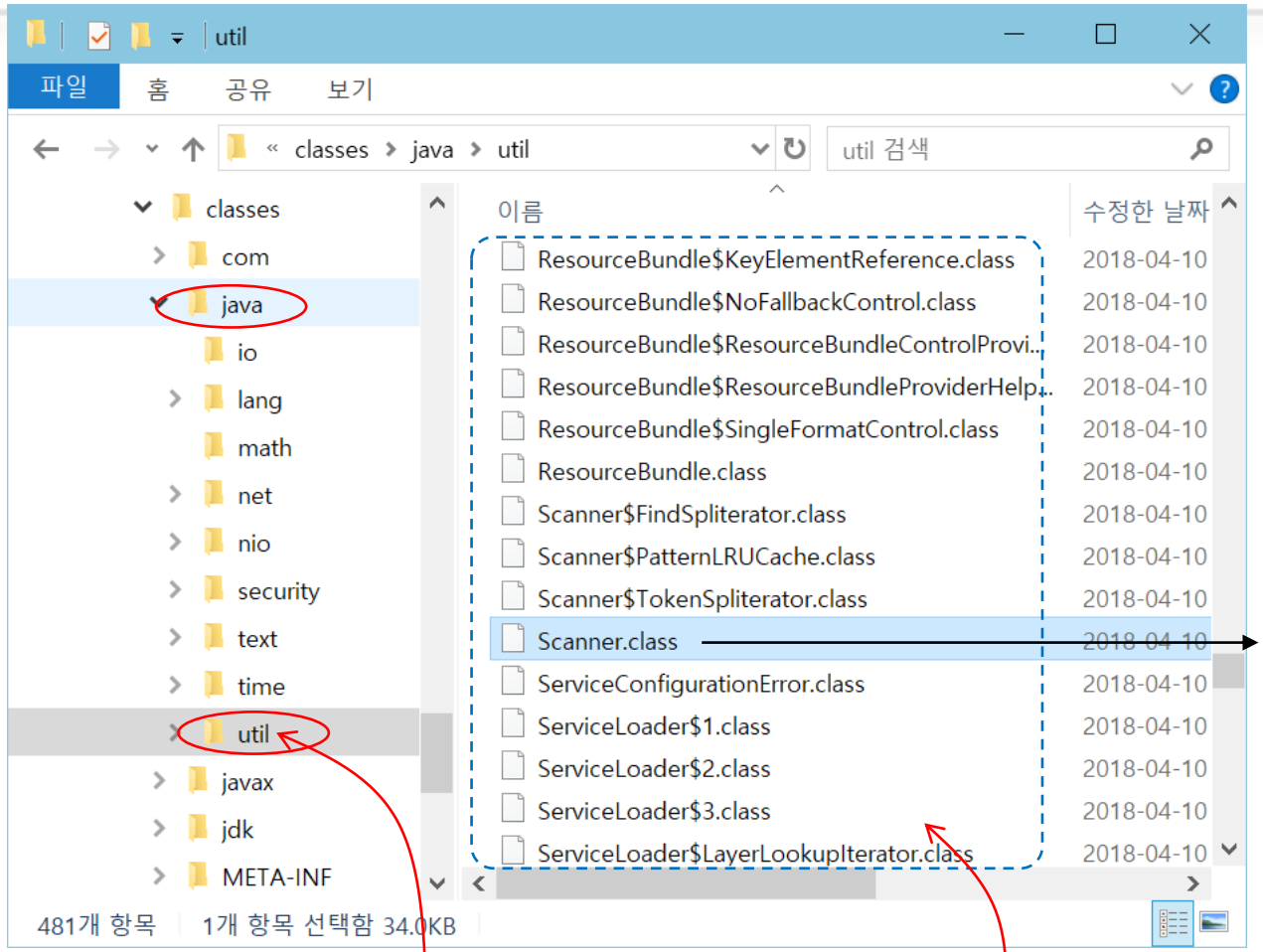
- 패키지
  - ✓ 서로 관련된 클래스와 인터페이스의 컴파일 된 클래스 파일들을 하나의 디렉터리에 묶어 놓은 것
- 모듈
  - ✓ 여러 패키지와 이미지 등의 자원을 모아 놓은 컨테이너
  - ✓ JDK 9부터 자바 API의 모든 클래스들(자바 실행 환경)을 패키지 기반에서 모듈들로 완전히 재구성
  - ✓ 응용프로그램 역시 여러 개의 모듈로 분할하여 작성 가능
    - 클래스들은 패키지로 만들고, 다시 패키지를 모듈로 만듦
  - ✓ 목적
    - 자바 API를 여러 모듈(99개)로 분할하여 응용프로그램의 실행에 적합한 모듈들로만 실행 환경을 구축할 수 있도록 함
    - 메모리 등의 자원이 열악한 작은 소형 기기에 꼭 필요한 모듈로 구성된 작은 크기의 실행 이미지를 만들기 위함
  - ✓ 모듈의 현실
    - Java 9부터 전면적으로 도입, 복잡한 개념, 큰 자바 응용프로그램에는 개발, 유지보수 등에 적합
- 패키지명과 클래스의 경로명
  - ✓ 점(.)으로 연결
    - Project.FileIO.Tools.class
    - Project.UI.Tools.class

자바 API의 모듈 파일들

- 자바 JDK에 제공되는 모듈 파일들
  - ✓자바가 설치된 jmods 디렉터리에 모듈 파일 존재
    - jdk 10의 경우 99개 모듈 파일
    - 모듈 파일은 ZIP 포맷으로 압축된 파일
  - ✓모듈 파일에는 자바 API의 패키지와 클래스들이 들어 있음
  - ✓jmod 명령을 이용하여 모듈 파일에 들어 있는 패키지를 풀어 낼 수 있음



JDK의 java.base 모듈에 들어 있는 패키지들과 클래스들



클래스의 이름(경로명)

java.util.Scanner

패키지명

패키지 명 : java.util

java.util 패키지에 속한 클래스

패키지 사용하기, import문

- 다른 패키지에 작성된 클래스 사용
  - ✓import를 이용하지 않는 경우
    - 소스 내에서 패키지 이름과 클래스 이름의 전체 경로명을 써주어야 함

```
public class ImportExample {
    public static void main(String[] args) {
        java.util.Scanner scanner =
                new java.util.Scanner(System.in);
        System.out.println(scanner.next());
    }
}
```

  - ✓Import를 이용하는 경우
    - 소스의 시작 부분에 사용하려는 패키지 명시
      - 소스에는 클래스 명만 명시하면 됨
    - 특정 클래스의 경로명만 포함
      - import java.util.Scanner;
    - 패키지 내의 모든 클래스 포함
      - import java.util.*;
      - *는 현재 패키지 내의 클래스만을 의미하며 하위 패키지의 클래스까지 포함하지 않는다.

```
import java.util.Scanner;
public class ImportExample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
    }
}
```

```
import java.util.*;
public class ImportExample {
    public static void main(String[] args) {
        Scanner scanner = new Scanner(System.in);
    }
}
```

패키지 만들기

- 패키지 선언
  - ✓ package 패키지명;
    - 컴파일한 클래스 파일을 패키지명의 디렉터리에 저장하라는 지시
    - 소스 파일의 첫 줄에 선언
- 사례

```
package UI; // Tools 클래스를 컴파일하여 UI 패키지에 저장할 것을 지시

public class Tools {          // 이제 이 클래스의 경로명은 UI.Tools가 된다.
.....
}
```

  - Tools 클래스의 경로명은 UI.Tools
  - 다른 클래스에서 Tools 클래스를 사용하기 위해서는 import UI.Tools

```
package Graphic; // Line 클래스를 Graphic 패키지에 저장

import UI.Tools; // Tools 클래스의 경로명 알림

public class Line {
    public void draw() {
        Tools t = new Tools();
    }
}
```

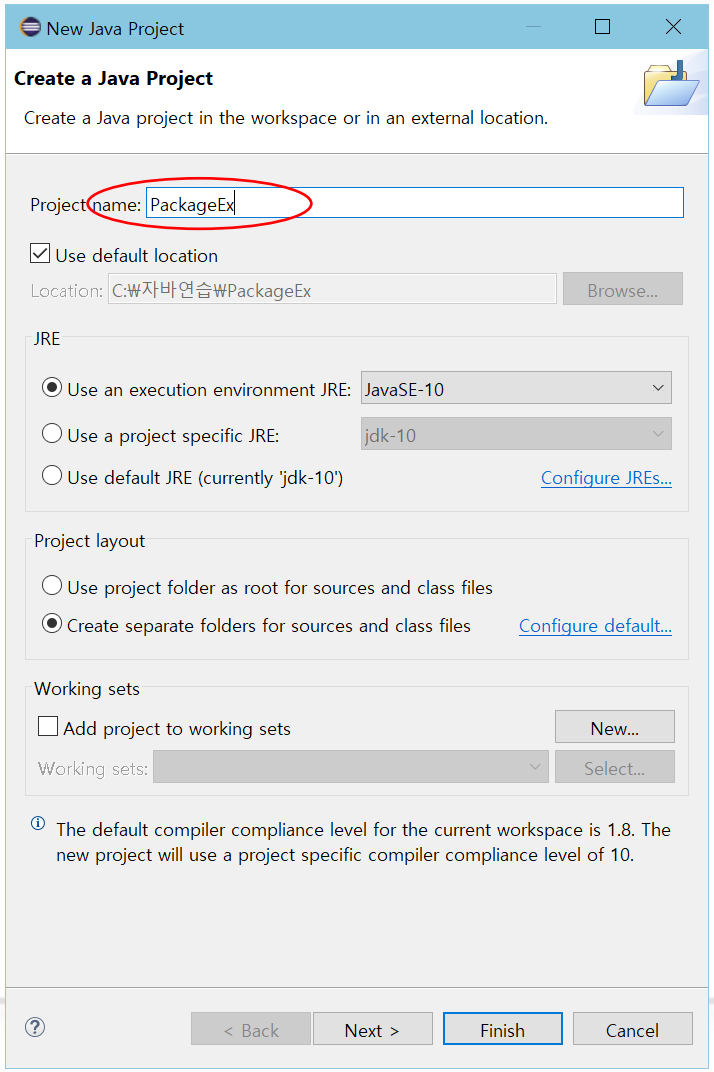
이클립스로 쉽게 패키지 만들기

✓ 예제로 사용할 샘플 소스(5장의 예제 5-7)

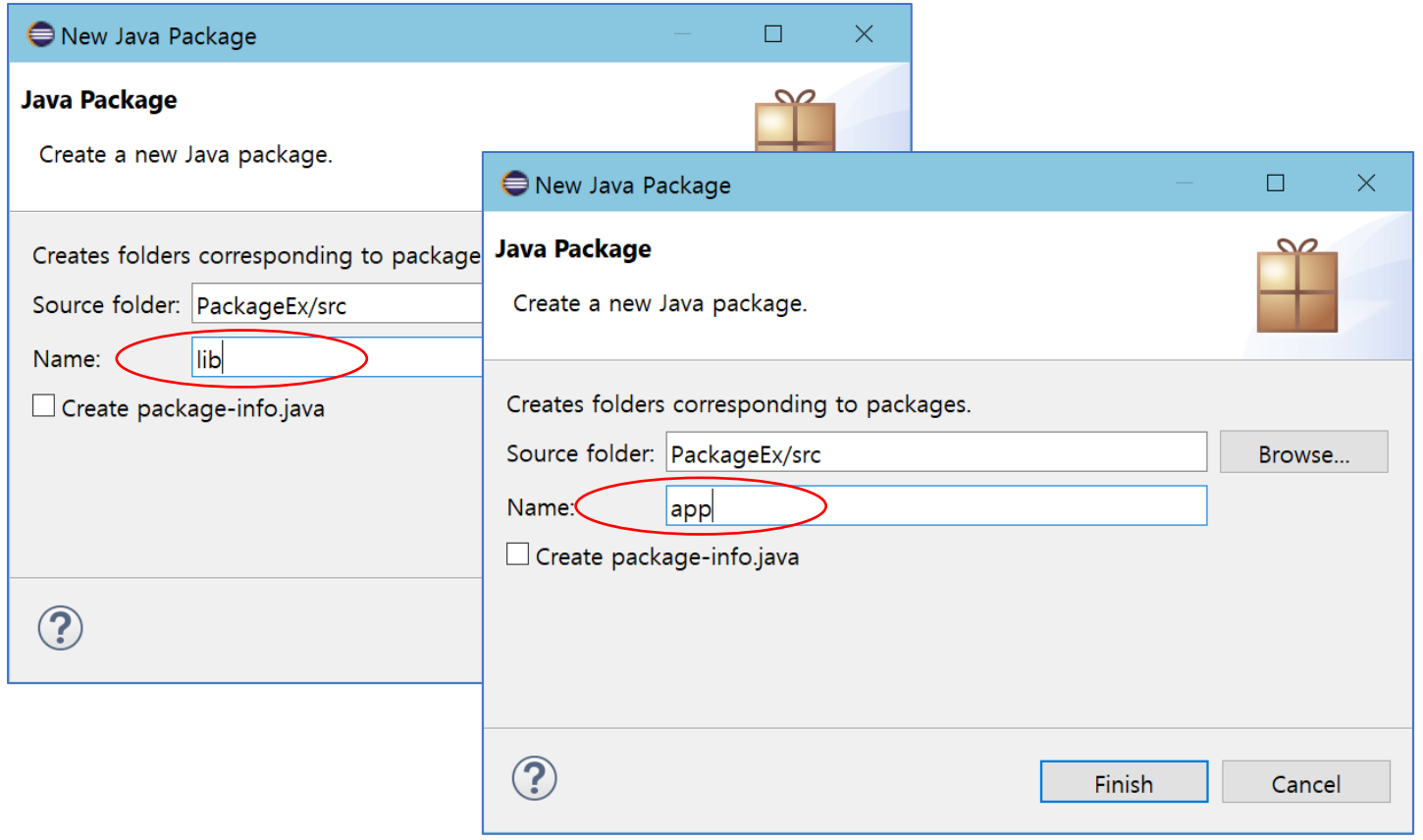
```
abstract class Calculator {
    public abstract int add(int a, int b);
    public abstract int subtract(int a, int b);
    public abstract double average(int[] a);
}

public class GoodCalc extends Calculator {
    public int add(int a, int b) {
        return a+b;
    }
    public int subtract(int a, int b) {
        return a - b;
    }
    public double average(int[] a) {
        double sum = 0;
        for (int i = 0; i < a.length; i++)
            sum += a[i];
        return sum/a.length;
    }
    public static void main(String [] args) {
        Calculator c = new GoodCalc();
        System.out.println(c.add(2,3));
        System.out.println(c.subtract(2,3));
        System.out.println(c.average(new int [] {2,3,4 }));
    }
}
```

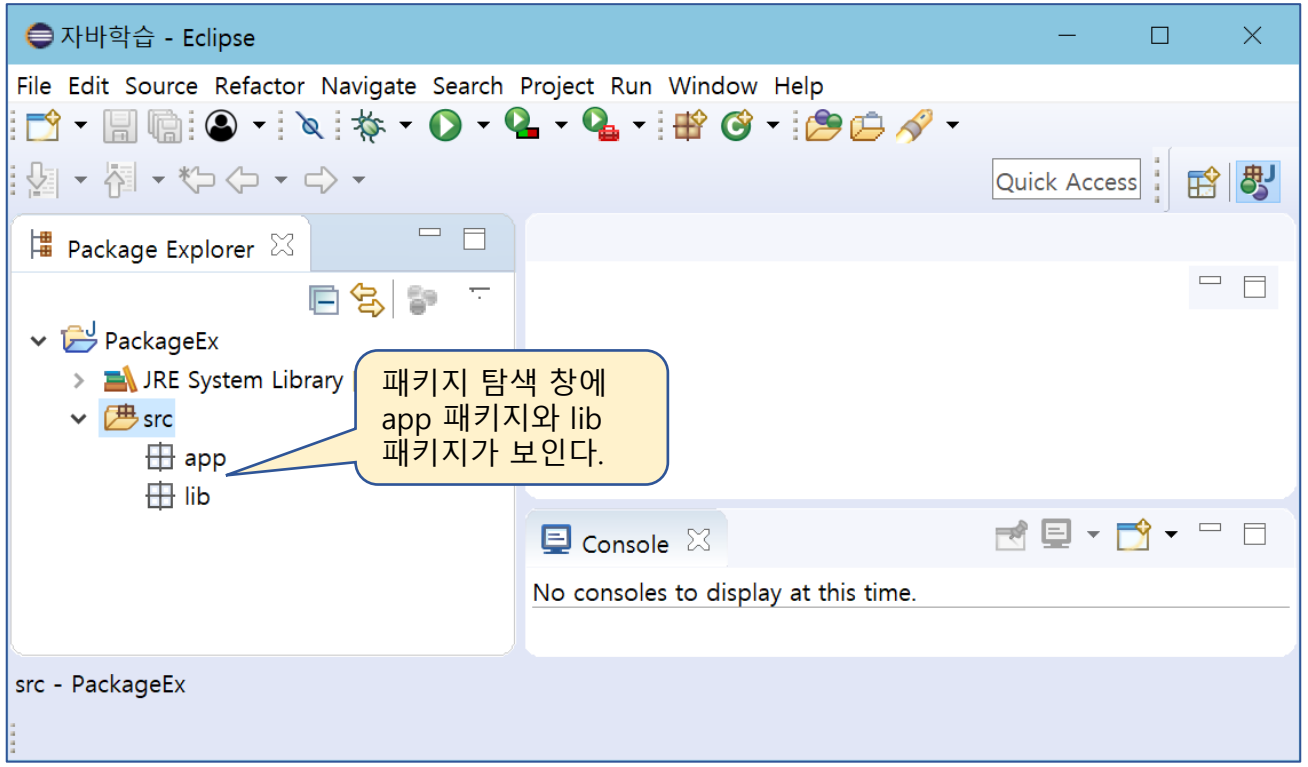
프로젝트 작성(프로젝트 이름 : PackageEx)



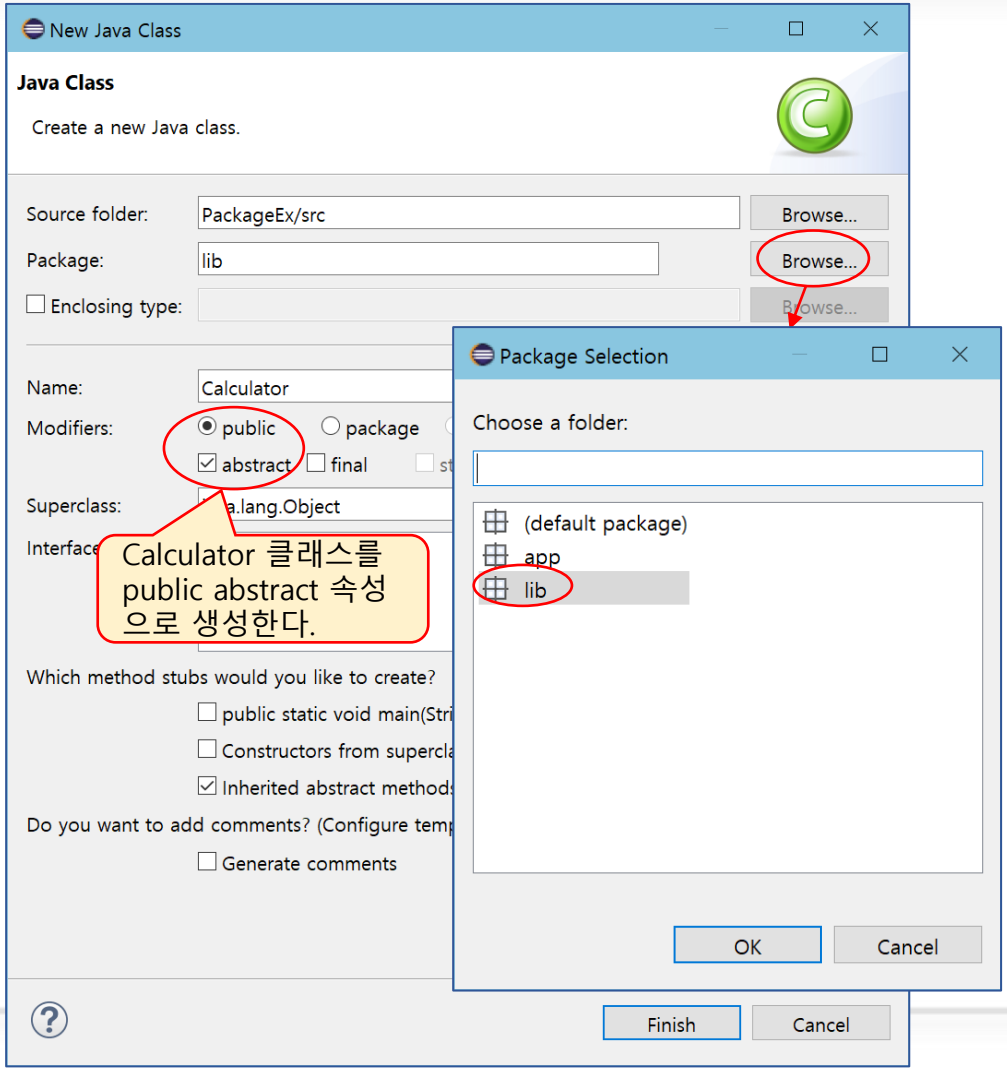
패키지 lib, app 작성



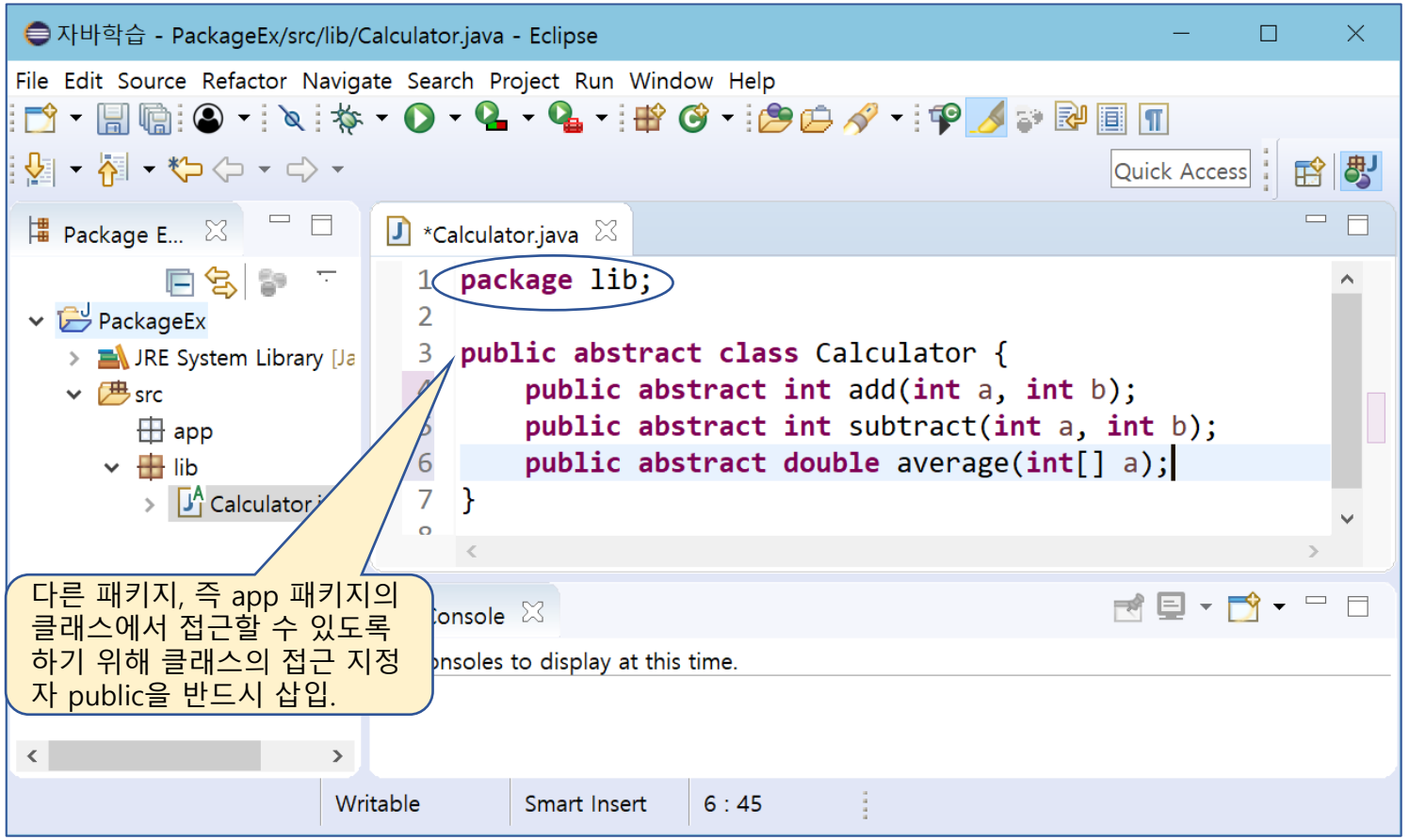
패키지 작성이 완료된 결과



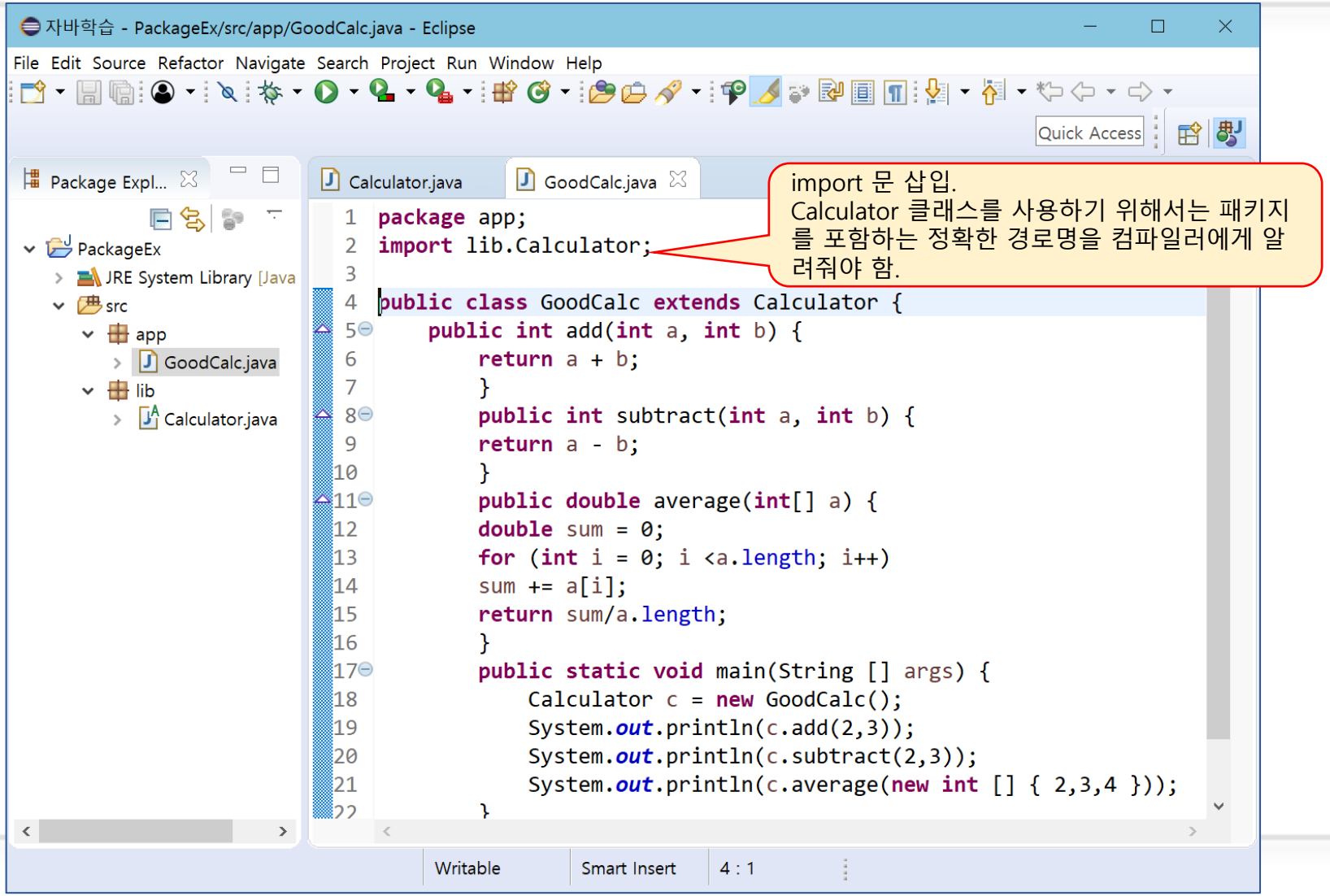
클래스 Calculator 만들기



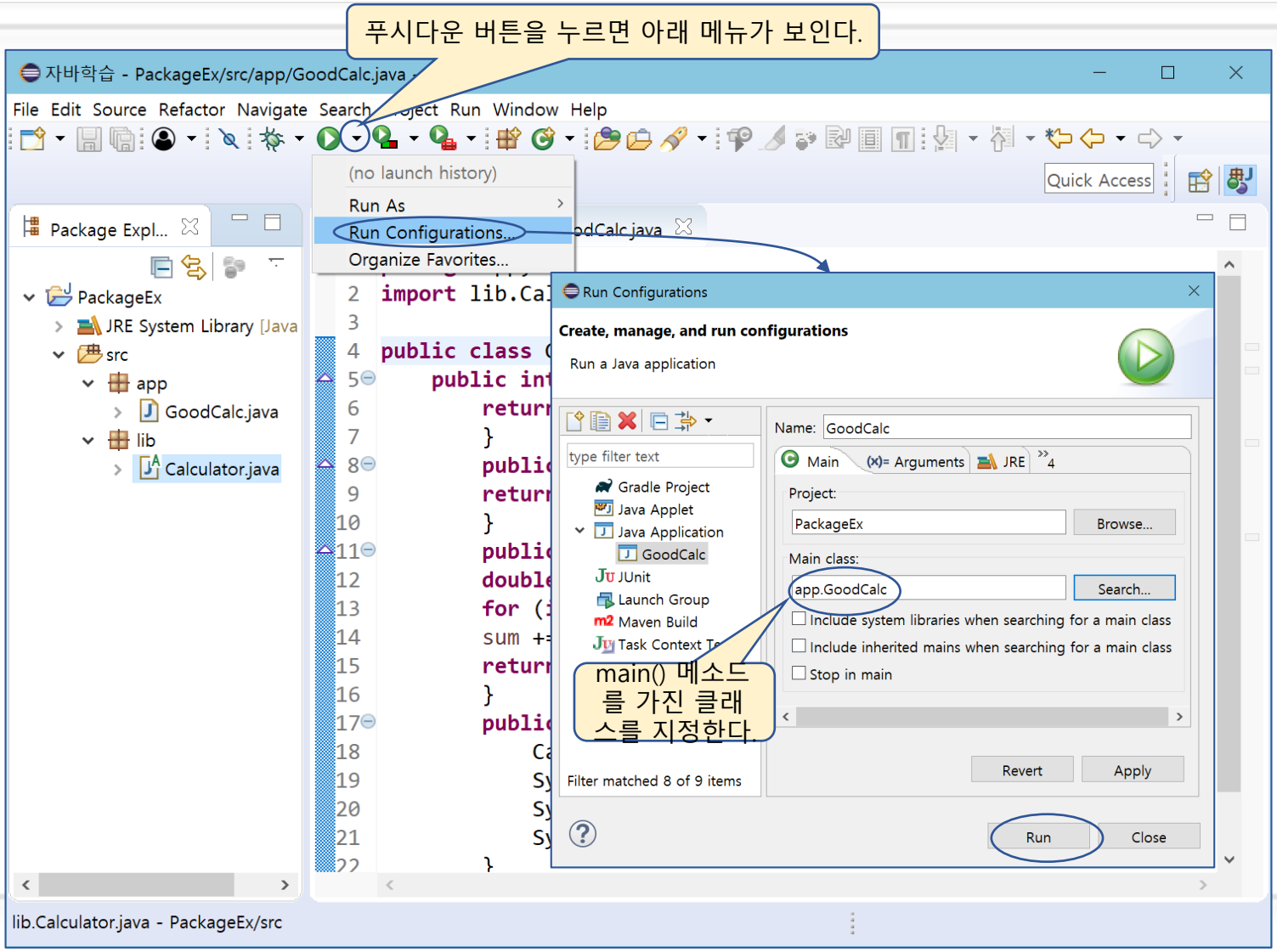
Calculator 소스 수정



GoodCalc.java 작성 후 소스 수정



실행을 위한 Run Configurations 작성

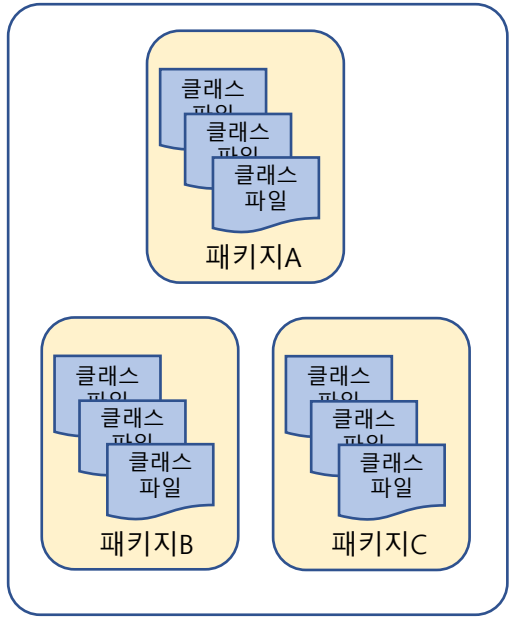


디폴트 패키지와 패키지의 특징

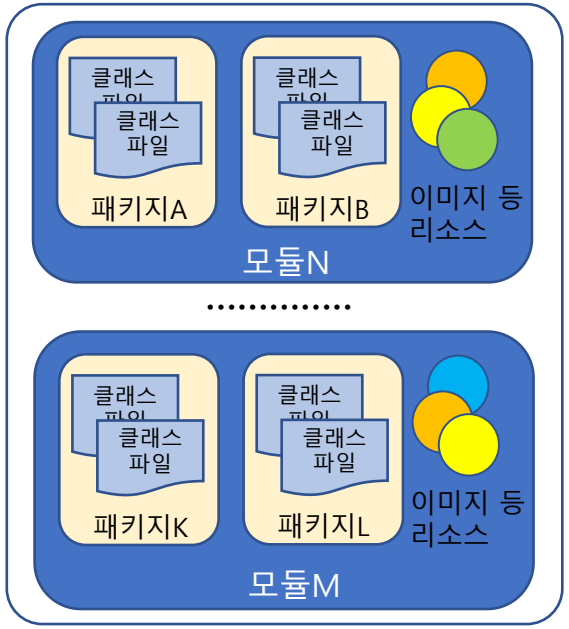
- 디폴트 패키지
  - ✓ package 선언문이 없이 만들어진 클래스의 패키지
  - ✓ 디폴트 패키지는 현재 디렉터리
- 패키지의 특징
  - ✓ 패키지 계층구조
    - 관련된 클래스 파일을 하나의 패키지로 계층화하여 관리 용이
  - ✓ 패키지별 접근 제한
    - 패키지 별로 접근 권한 가능
  - ✓ 동일한 이름의 클래스와 인터페이스의 사용 가능
    - 서로 다른 패키지에 이름이 같은 클래스와 인터페이스 존재 가능
  - ✓ 높은 소프트웨어 재사용성

모듈 개념

- 모듈
  - ✓Java 9에서 도입된 개념
  - ✓패키지와 이미지 등의 리소스를 담은 컨테이너
  - ✓모듈 파일(.jmod)로 저장



Java 8에서 클래스와 패키지



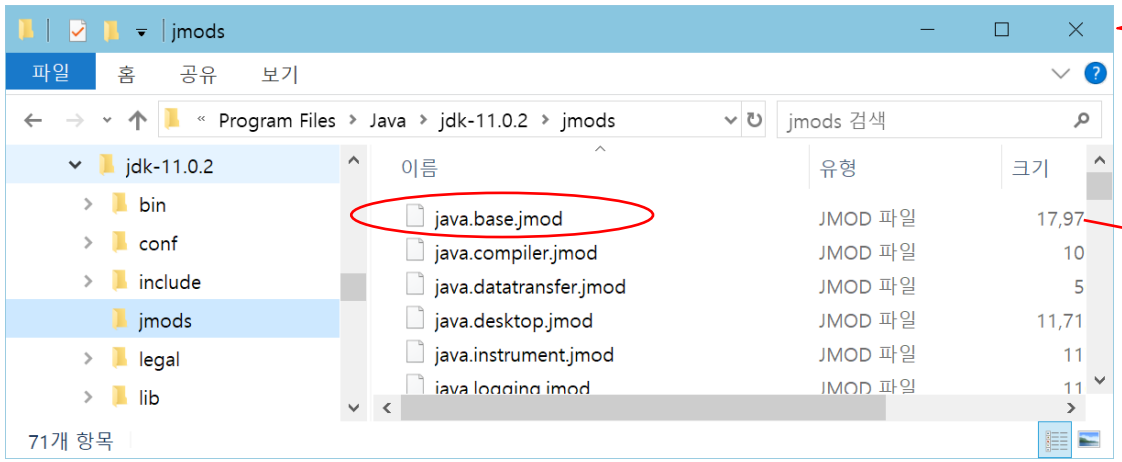
Java 9 이후 클래스와 패키지, 그리고 모듈

자바 플랫폼의 모듈화

- 자바 플랫폼
  - ✓자바의 개발 환경(JDK)과 자바의 실행 환경(JRE)을 지칭
    - Java SE(자바 API) 포함
  - ✓자바 API의 모든 클래스가 여러 개의 모듈로 재구성됨
  - ✓모듈 파일은 JDK의 jmods 디렉터리에 저장하여 배포
- 모듈 파일로부터 모듈을 푸는 명령

  jmod extract "C:\Program Files\Java\jdk-10\jmods\java.base.jmod“

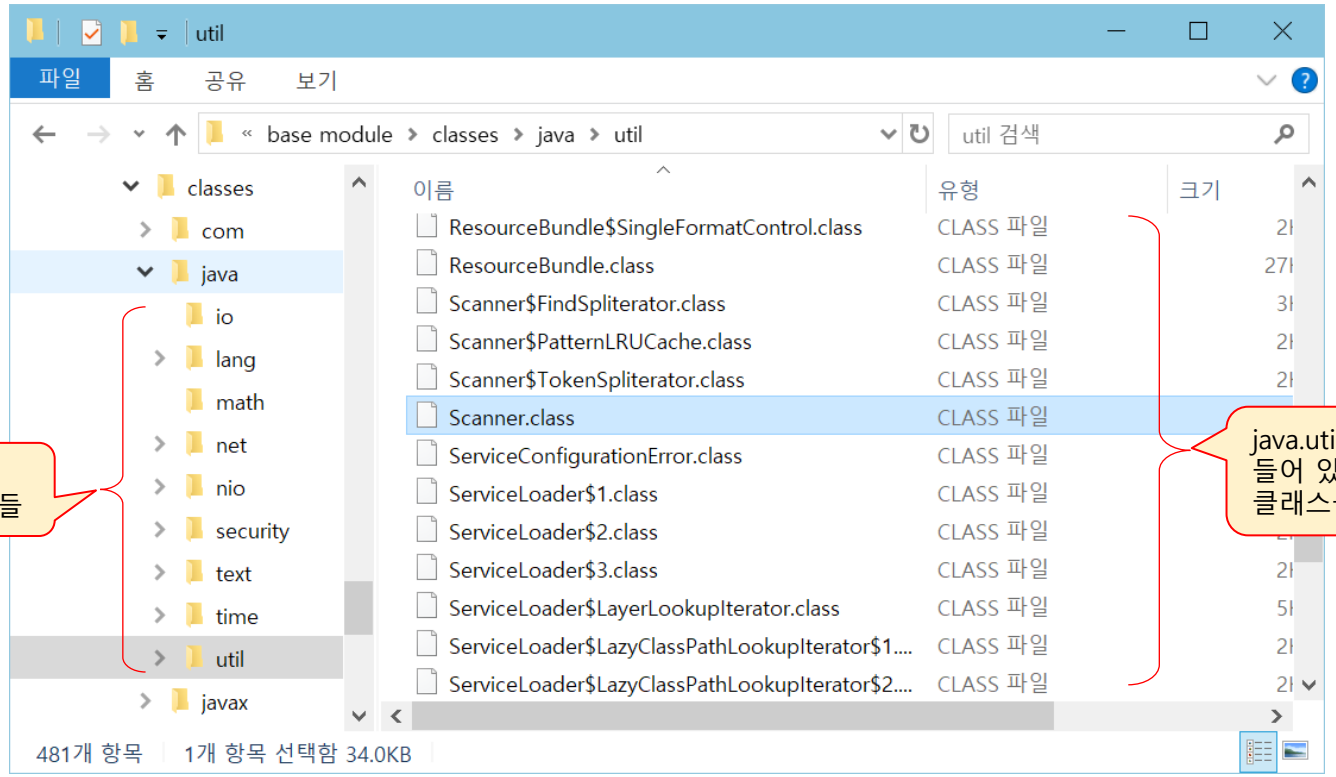
  - 현재 디렉터리에 java.base 모듈이 풀림



jmods 디렉터리에 들어 있는 자바 API의 모듈 파일들

java.base.jmod 모듈 파일 풀기

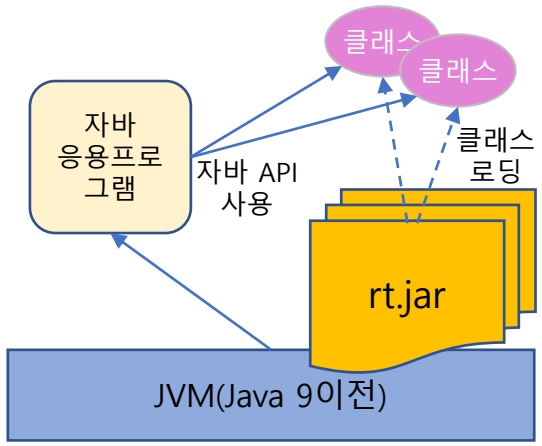
jmod extract "C:₩Program Files₩Java₩jdk-11.0.2₩jmods₩java.base.jmod"



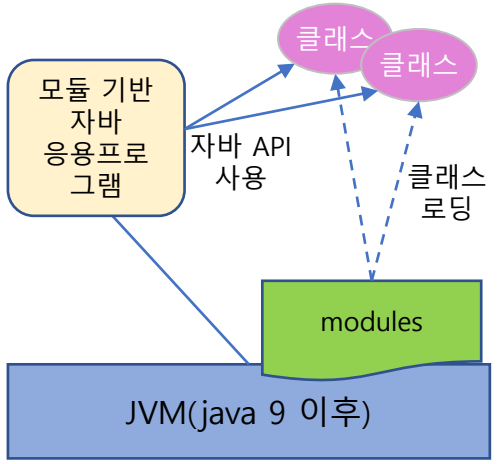
java.base 모듈에 들어 있는 패키지들

java.util 패키지에 들어 있는 자바 API 클래스들

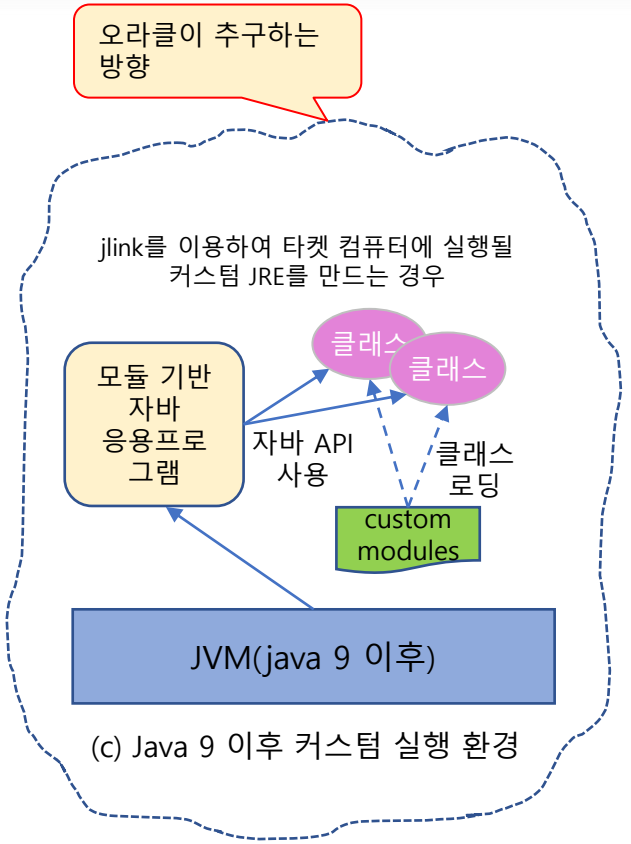
자바 실행 환경 비교



(a) Java 9 이전 실행 환경



(b) Java 9 이후 실행 환경



(c) Java 9 이후 커스텀 실행 환경

* 홈페이지의 자료에, jlink로 맞춤형 실행 환경을 만드는 예제 있음.
실제로 만들어보면 커스텀 JRE가 생기고, lib 디렉터리 밑에 modules 파일의 크기가 현저히 줄어든 것을 볼 수 있고, 실행 중에 차지하는 메모리의 량도 줄어든 것을 확인할 수 있음

자바 모듈화의 목적

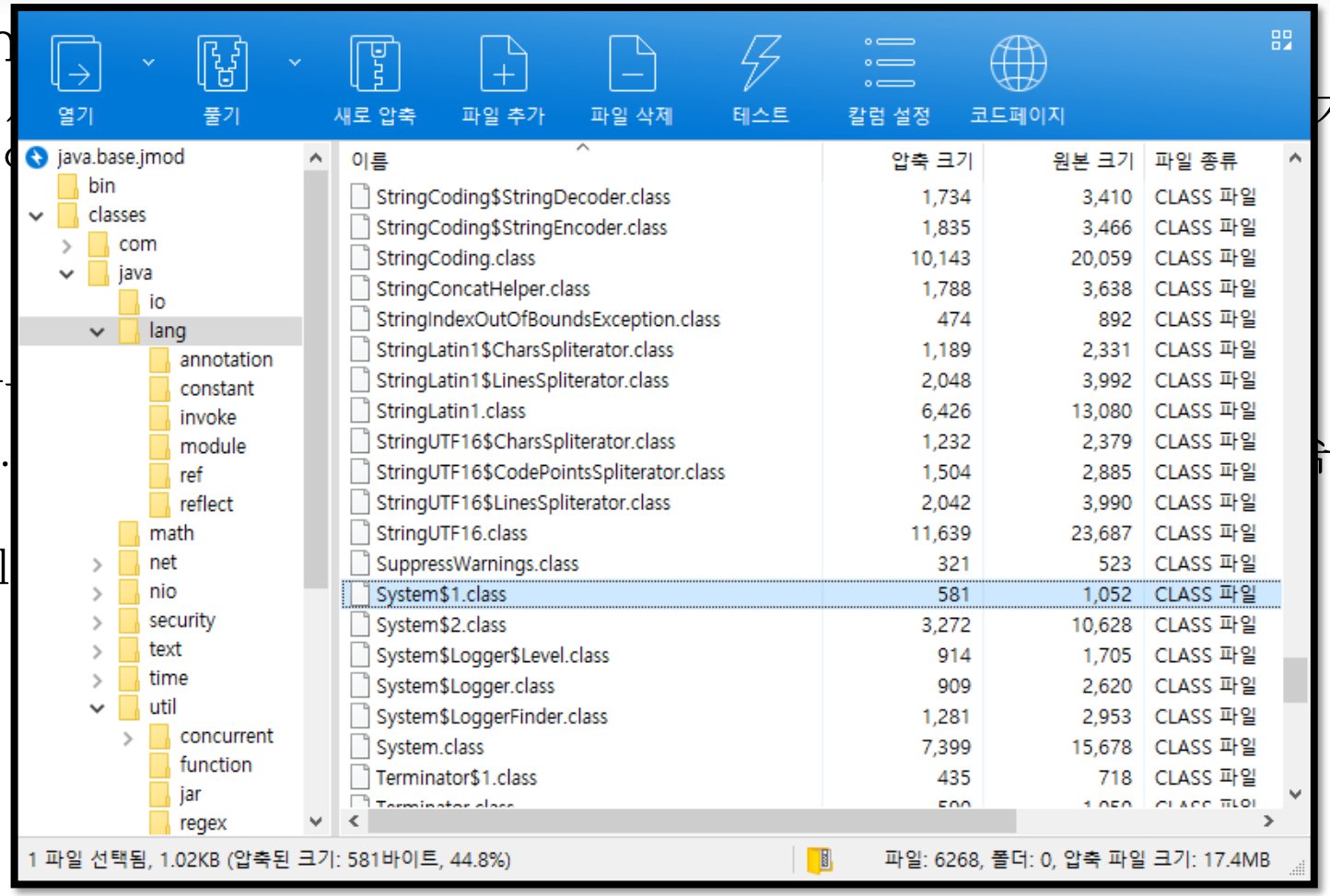
- 가장 큰 목적
  - ✓자바 컴포넌트들을 필요에 따라 조립하여 사용하기 위함
  - ✓컴퓨터 시스템의 불필요한 부담 감소
    - 세밀한 모듈화를 통해 필요 없는 모듈이 로드되지 않게 함
    - 소형 IoT 장치에도 자바 응용프로그램이 실행되고 성능을 유지하게 함

6.5 자바에서 제공하는 패키지

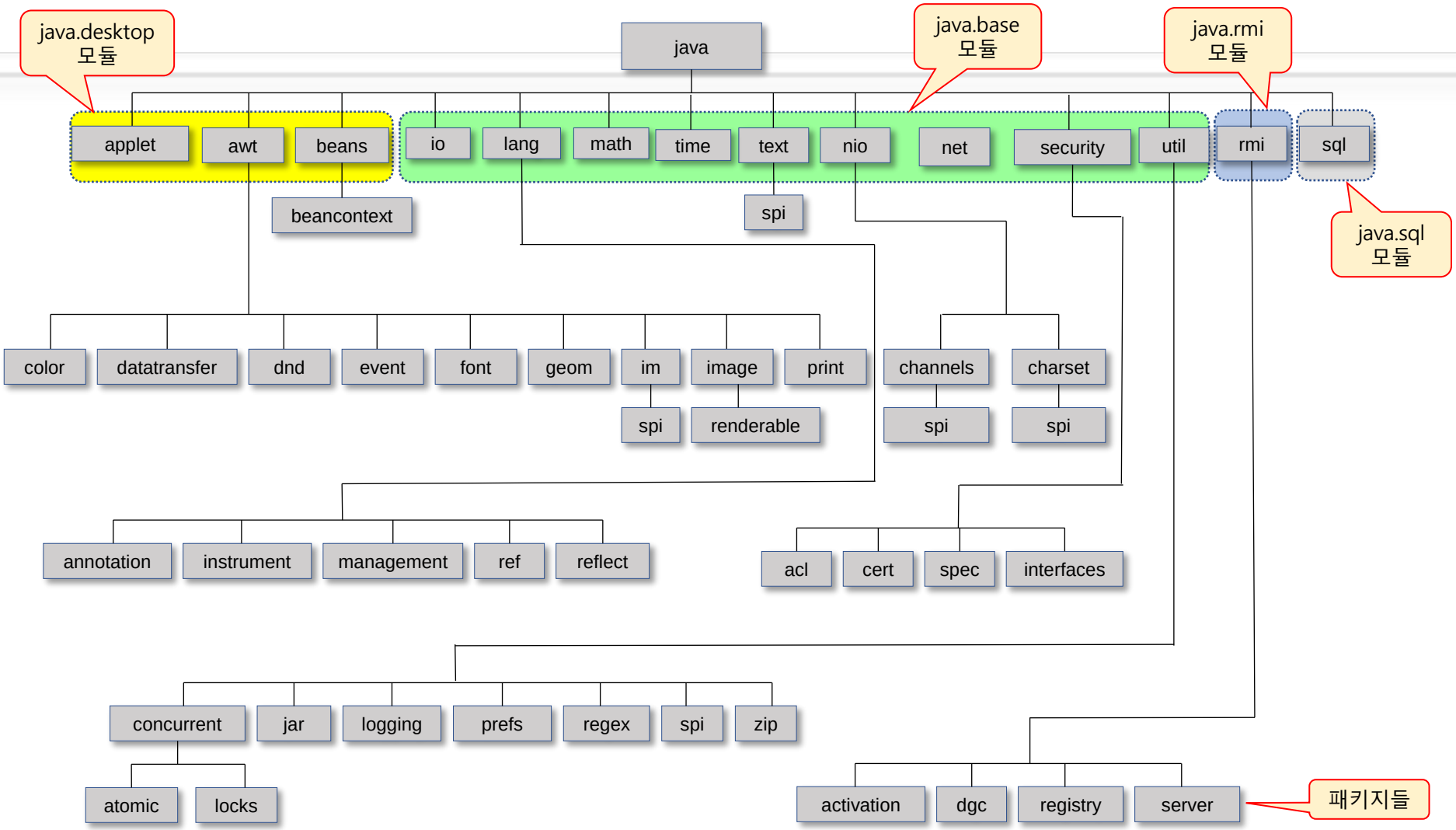
- API (Application
  - ✓응용 프로그램에
    할 수 있게 만든
- 자바 API
  - ✓자바에서 개발자
  - ✓예) System.out.
    (API)
  - ✓C:\Program Fil

기능을 제어

한 클래스



자바 모듈과 패키지 구조


java.desktop
모듈
java
java.base
모듈
java.rmi
모듈
applet
awt
beans
io
lang
math
time
text
nio
net
security
util
rmi
sql
beancontext
spi
java.sql
모듈
color
datatransfer
dnd
event
font
geom
im
image
print
channels
charset
spi
renderable
spi
spi
annotation
instrument
management
ref
reflect
acl
cert
spec
interfaces
concurrent
jar
logging
prefs
regex
spi
zip
atomic
locks
activation
dgc
registry
server
패키지들

주요 패키지

- java.lang
  - ✓ 자바 language 패키지
    - 스트링, 수학 함수, 입출력 등 자바 프로그래밍에 필요한 기본적인 클래스와 인터페이스
  - ✓ 자동으로 컴파일러가 import java.lang.* 문장 추가함
- java.util
  - ✓ 자바 유틸리티 패키지
    - 날짜, 시간, 벡터, 해시맵 등과 같은 다양한 유틸리티 클래스와 인터페이스 제공
- java.io
  - ✓ 키보드, 모니터, 프린터, 디스크 등에 입출력을 할 수 있는 클래스와 인터페이스 제공
- java.awt
  - ✓ 자바 GUI 프로그래밍을 위한 클래스와 인터페이스 제공
- javax.swing
  - ✓ 자바 GUI 프로그래밍을 위한 스윙 패키지

자바 API 참조

- 자바 API의 상세 정보
  - ✓Oracle Technology Network(https://docs.oracle.com/javase/8/docs/api/)에서 온라인제공