기본 API 클래스

컴퓨터공학전공
박요한

수업내용

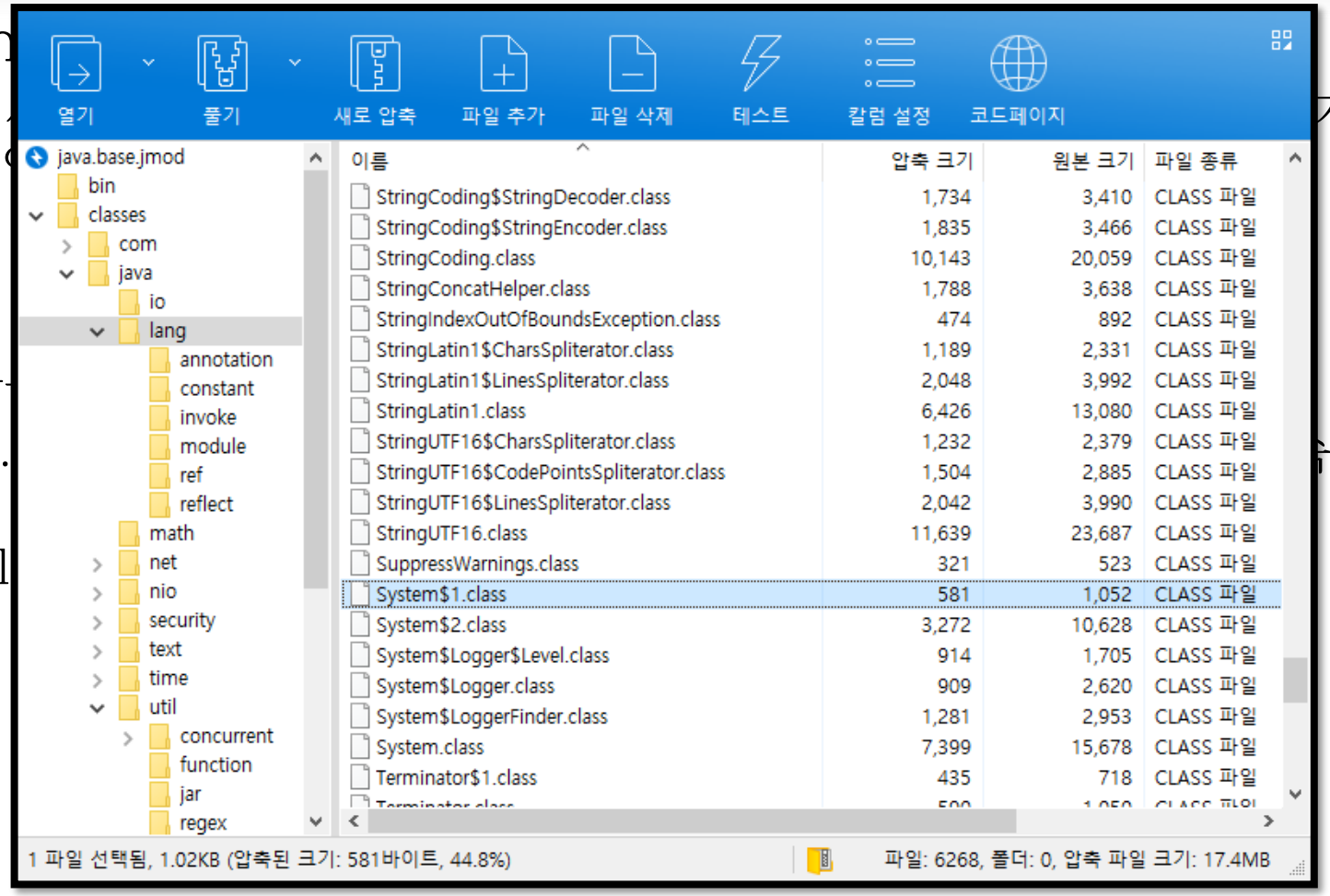
- JDK에서 제공하는 기본 API 클래스
  - ✓Object 클래스
  - ✓String, StringBuffer, StringTokenizer 클래스
  - ✓Wrapper 클래스
  - ✓Math, Random 클래스
  - ✓Arrays 클래스

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



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

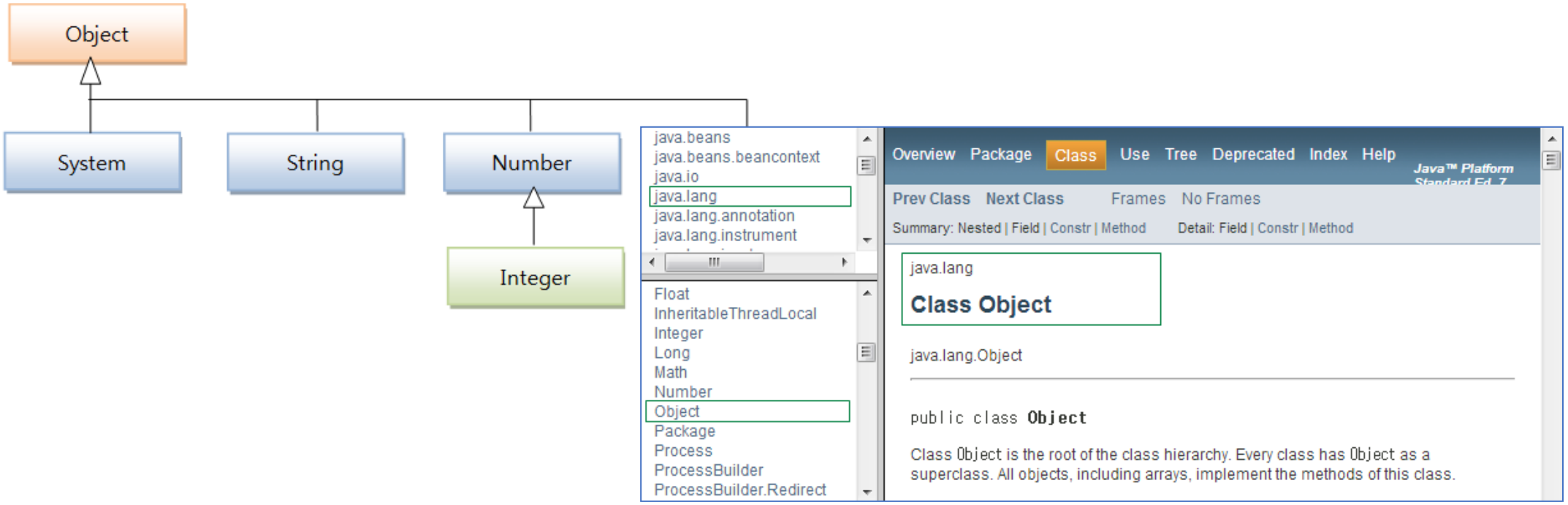
java.lang과 java.util 패키지

- java.lang 패키지
  - ✓자바 프로그램의 기본적인 클래스를 담은 패키지
  - ✓포함된 클래스와 인터페이스는 import 없이 사용
  - ✓주요 클래스

클래스		용도
Object		- 자바 클래스의 최상위 클래스로 사용
System		- 표준 입력장치(키보드)로부터 데이터를 입력 받을 때 사용 - 표준 출력장치(모니터)로 출력하기 위해 사용 - 자바 가상 머신을 종료시킬 때 사용 - 쓰레기 수집기를 실행 요청할 때 사용
Class		- 클래스를 메모리로 로딩할 때 사용
String		- 문자열을 저장하고 여러가지 정보를 얻을 때 사용
StringBuffer, StringBuilder		- 문자열을 저장하고 내부 문자열을 조작할 때 사용
Math		- 수학 함수를 이용할 때 사용
Wrapper	Byte, Short, Character Integer, Float, Double Boolean	- 기본 타입의 데이터를 갖는 객체를 만들 때 사용 - 문자열을 기본 타입으로 변환할 때 사용 - 입력값 검사에 사용

Object 클래스

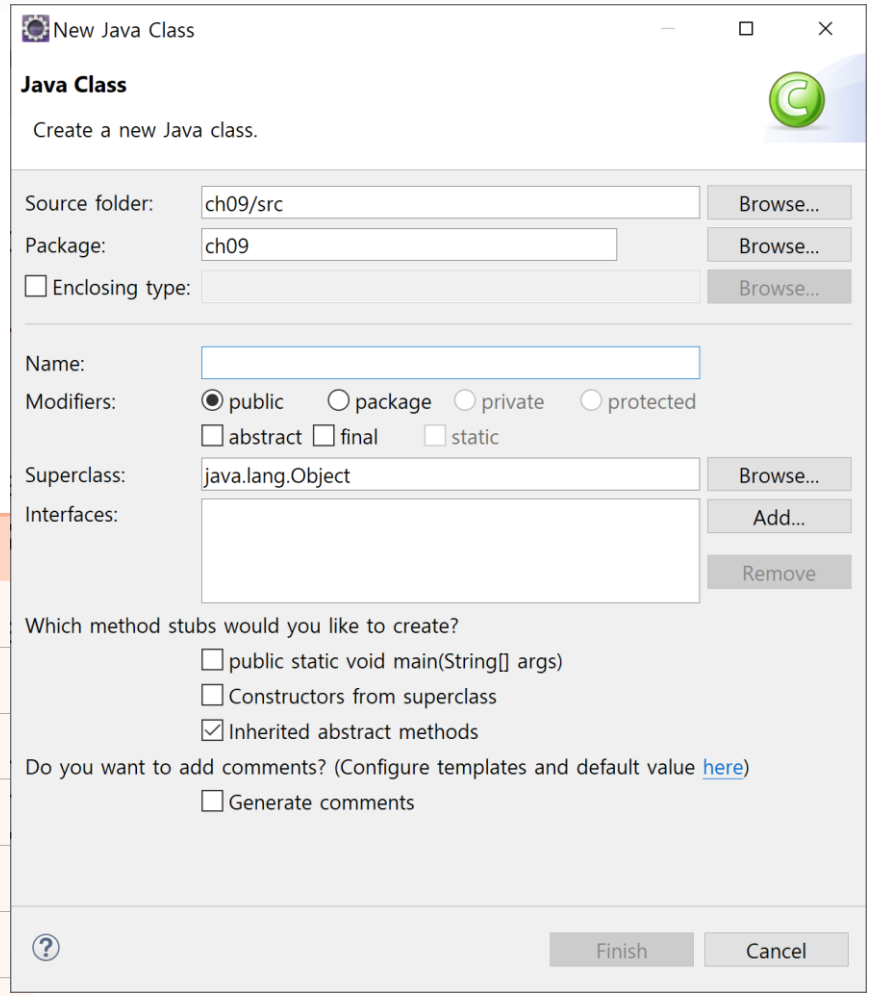
- 자바의 최상위 부모 클래스
  - ✓다른 클래스 상속하지 않으면 java.lang.Object 클래스 상속 암시
  - ✓Object의 메소드는 모든 클래스에서 사용 가능



Object 클래스

- 특징
  - ✓java.lang 패키지에 포함
  - ✓모든 클래스의 수퍼 클래스
    - 모든 클래스에 강제 상속
    - 모든 객체가 공통으로 가지는 객체의 속성을 나타내는 메소드 보유
- 주요 메소드

메소드	설명
boolean equals(Object obj)	obj가 가리키는 객체와 현재 객체를 비교하여 같으면 true 리턴
Class getClass()	현 객체의 런타임 클래스를 리턴
int hashCode()	현 객체에 대한 해시 코드 값 리턴
String toString()	현 객체에 대한 문자열 표현을 리턴
void notify()	현 객체에 대해 대기하고 있는 하나의 스레드를 깨운다.
void notifyAll()	현 객체에 대해 대기하고 있는 모든 스레드를 깨운다.
void wait()	다른 스레드가 깨울 때까지 현재 스레드를 대기하게 한다.



객체를 문자열로 변환

- String toString()
  - ✓ 객체를 문자열로 반환
  - ✓ Object 클래스에 구현된 toString()이 반환하는 문자열

```
public String toString() {
   return getClass().getName() +"@" + Integer.toHexString(hashCode());
}
```

- '객체 + 문자열' -> '객체.toString() + 문자열'로자동변환

```
Point p = new Point(2,3);
System.out.println(p);
String s = p + "점";
```

변환

```
System.out.println(p.toString());
String s = p.toString()+ "점";
```

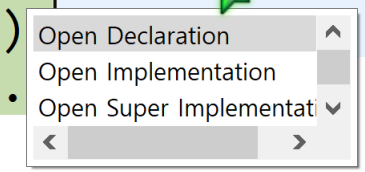
```
Point@15db9742점
```

- 개발자는 자신만의 toString() 작성 필요
  - ✓ Object의 toString() 오버라이딩

```java
package ch09_2;

class Calc{

    public int sum(int a, int b) {
        return a+b;
    }
}

public class StringEx {
    public static void main(String[] args) {

        String str1 = new String("Hello");
        Calc calc = new Calc();
        int a = calc.sum(5, 10);

        System.out.println(str1);
        System.out.println(str1.toString());
        System.out.println(calc);
        System.out.println(calc.toString());
        System.out.println(a);
        System.out.println(a.toString());

    }

}
```

```
Hello
Hello
ch09_2.Calc@53f65459
ch09_2.Calc@53f65459
15
```

```
System.out.println(str1);
System.out.println(str1.toString());
System.out.println(calc);
System.out.println(calc.
System.out.println(a);
System.out.println(a.toString());
```




```
/**
 * This object (which is already a string!) is itself returned.
 *
 * @return  the string itself.
 */
public String toString() {
    return this;
}
```

```
System.out.println(str1);
System.out.println(str1.toString());
System.out.println(calc);
System.out.println(calc.toString());
System.out.println(a);
System.out.println(a.toS
```

Open Declaration
Open Implementati

```
 * @return  a string representation of the object.
 */
public String toString() {
    return getClass().getName() + "@" + Integer.toHexString(hashCode());
}
```

```java
package ch09_2;

class Calc{

    public int sum(int a, int b) {
        return a+b;
    }

    public String toString() {
        return "input sum(a,b)";
    }
}
```

예제 6-2 : Point 클래스에 toString() 작성

Point 클래스에 Point 객체를 문자열로 리턴하는 toString() 메소드를 작성하라.

```
class Point {
   int x, y;
   public Point(int x, int y) {
      this.x = x;
      this.y = y;
   }
   public String toString() {
      return "Point(" + x + "," + y + ")";
   }
}

public class ToStringEx {
   public static void main(String [] args) {
      Point p = new Point(2,3);
      System.out.println(p.toString());
      System.out.println(p); // p는 p.toString()으로 자동 변환
      System.out.println(p + "입니다."); // p.toString() + "입니다"로 자동 변환
   }
}
```

Point 객체를 문자열로 리턴하는 toString() 작성

```
Point(2,3)
Point(2,3)
Point(2,3)입니다.
```

객체 비교와 equals()

- boolean equals(Object obj)
  - ✓ 객체 내용이 같은지 비교
- == 연산자

```java
class Point {
        int x, y;
        public Point(int x, int y) {
            this.x = x; this.y = y;
        }
    }
public class StringEx{
    public static void main(String[] args) {

        Point a = new Point(2,3);
        Point b = new Point(2,3);
        Point c = a;

        boolean res1 = a.equals(b);
        boolean res2 = a.equals(c);
        System.out.println(res1);
        System.out.println(res2);
```

```java
String str1 = new String("Hello");
String str2 = new String("Hello");
boolean res=str1.equals(str2);
System.out.println(res);
```

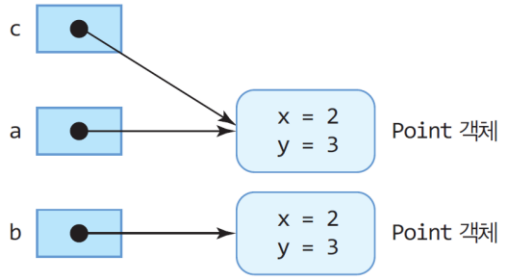
```java
     */
    public boolean equals(Object anObject) {
        if (this == anObject) {
            return true;
        }
        if (anObject instanceof String) {
            String anotherString = (String)anObject;
            int n = value.length;
            if (n == anotherString.value.length) {
                char v1[] = value;
                char v2[] = anotherString.value;
                int i = 0;
                while (n-- != 0) {
                    if (v1[i] != v2[i])
                        return false;
                    i++;
                }
```

객체 비교와 equals()

```
class Point {
    int x, y;
    public Point(int x, int y) {
        this.x = x; this.y = y;
    }
}
```

```
Point a = new Point(2,3);
Point b = new Point(2,3);
Point c = a;
if(a == b) // false
    System.out.println("a==b");
if(a == c) // true
    System.out.println("a==c");
```

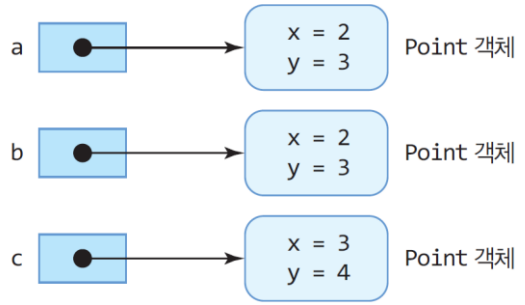
```
a==c
```




```
class Point {
    int x, y;
    public Point(int x, int y) {
        this.x = x; this.y = y;
    }
    public boolean equals(Object p) {
        Point p = (Point)obj;
        if(x == p.x && y == p.y)
            return true;
        else return false;
    }
}
```

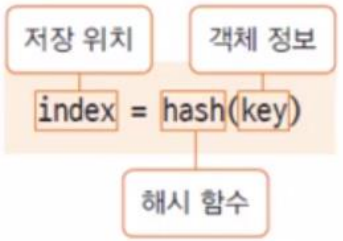
```
Point a = new Point(2,3);
Point b = new Point(2,3);
Point c =  new Point(3,4);

if(a == b) // false
    System.out.println("a==b");
if(a.equals(b)) // true
    System.out.println("a is equal to b");
if(a.equals(c)) // false
    System.out.println("a is equal to c");
```

```
a is equal to b
```



객체 해시코드 hashCode()

- ✓hash : 정보를 저장, 검색하기 위해 사용하는 자료구조
- ✓자료의 특정 값(키값)에 대해 저장 위치를 반환함



- ✓hashCode() 메서드는 인스턴스의 저장 주소를 반환함
- ✓힙 메모리에 인스턴스가 저장되는 방식이 hash, 즉 hashCode()의 반환값은 인스턴스의 주소값
- ✓**만약**, 서로 다른 메모리의 두 인스턴스의 내용이 같다면?

Object 클래스 - hashCode()

✓그런데…

```
Student std1 = new Student(1001, "이상");
Student std2 = new Student(1001, "이상");

System.out.println("std1의 hashCode : " +std1.hashCode());
System.out.println("std2의 hashCode : " +std2.hashCode());

String str1 = new String("test");
String str2 = new String("test");

System.out.println("str1의 hashCode : " +str1.hashCode());
System.out.println("str2의 hashCode : " +str2.hashCode());
```

결과는? 왜?

Object 클래스 - hashCode()

✓그런데…

```
Student std1 = new Student(1001, "이상");
Student std2 = new Student(1001, "이상");

System.out.println("std1의 ha
System.out.println("std2의 ha

String str1 = new String("tes
String str2 = new String("tes

System.out.println("str1의 ha
System.out.println("str2의 ha
```

```
1579572132
917142466
ch09_2.Point@36aa7bc2
69609650
69609650
```

```
     * @return  a hash                    ct.
     */
    public int hashCode() {
        int h = hash;
        if (h == 0 && value.length > 0) {
            char val[] = value;

            for (int i = 0; i < value.length; i++) {
                h = 31 * h + val[i];
            }
            hash = h;
        }
        return h;
    }
```

결과는? 왜?

```java
class Student{

        int studentId;
        String studentName;

        public Student(int studentId, String studentName){
        this.studentId = studentId;
        this.studentName = studentName;
        }

}
```

Object클래스의 equals메소드를 오버라이드 하여 결과값을 도출하시오.

```java
public class EqualsTest {
        public static void main(String[] args) {
        Student std1 = new Student(1001, "이상");
        Student std2 = new Student(1001, "이상");

if(std1 == stu2)
System.out.println("std1와 stu2의 주소는 같습니다.");
else
System.out.println("st1와 std2의 주소는 다릅니다.");

if(std1.equals(std2))
System.out.println("std1와 std2는 동일합니다.");

else
System.out.println("std1와 std2는 동일하지 않습니다.");
```

```
st1와 std2의 주소는 다릅니다.
std1와 std2는 동일하지 않습니다.
```

```
st1와 std2의 주소는 다릅니다.
std1와 std2는 동일합니다.
```