

# 패키지와 클래스 경로

## 08-1. 클래스 패스

# 현재 디렉토리에 대한 이해

현재 디렉토리: 실행 중인 프로그램의 작업 디렉토리

C:\PackageStudy>

현재 디렉토리는 C:\PackageStudy

C:\PackageStudy>javac WhatYourName.java

현재 디렉토리 C:\PackageStudy를 기준으로 자바 파일을 찾는다.

WhatYourName.java

```
class AAA { ... }  
class ZZZ { ... }  
class WhatYourName {  
    public static void main(String  
args[]) {  
        AAA aaa = new AAA();  
        aaa.showName();  
  
        ZZZ zzz = new ZZZ();  
        zzz.showName();  
    }  
}
```

# 디렉토리 구조 변경 및 컴파일

WhatYourName.class

C:\PackageStudy에 위치시킨다.

AAA.class

C:\PackageStudy\MyClass에 위치시킨다.

ZZZ.class

C:\PackageStudy\MyClass에 위치시킨다.

C:\PackageStudy>java WhatYourName

실행이 가능하게 하려면?

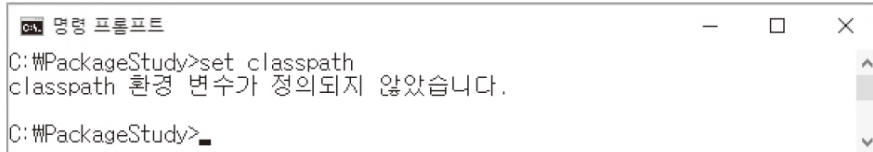
WhatYourName.java

```
class AAA { ... }  
class ZZZ { ... }  
class WhatYourName {  
    public static void main(String  
args[]) {  
        AAA aaa = new AAA();  
        aaa.showName();  
  
        ZZZ zzz = new ZZZ();  
        zzz.showName();  
    }  
}
```

# 클래스 패스란?

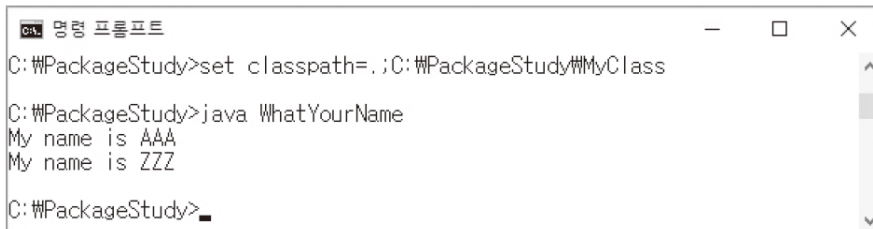
클래스 패스: 자바 가상머신의 클래스 탐색 경로

C:\PackageStudy>set classpath



```
명령 프롬프트
C:\PackageStudy>set classpath
classpath 환경 변수가 정의되지 않았습니다.
C:\PackageStudy>
```

C:\PackageStudy>set classpath=.;C:\PackageStudy\MyClass



```
명령 프롬프트
C:\PackageStudy>set classpath=.;C:\PackageStudy\MyClass
C:\PackageStudy>java WhatYourName
My name is AAA
My name is ZZZ
C:\PackageStudy>
```

# 절대 경로 vs 상대 경로

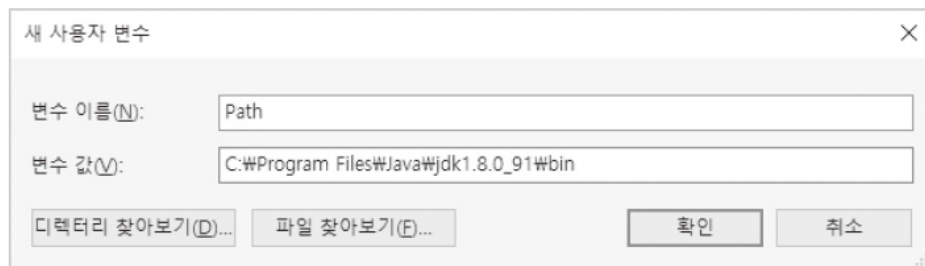
```
C:\PackageStudy>set classpath=.;C:\PackageStudy\MyClass
```

루트 디렉토리를 시작으로 지정한 '절대 경로'

```
C:\PackageStudy>set classpath=.;.\MyClass
```

현재 디렉토리를 기준으로 지정한 '상대 경로'

# 클래스 패스를 고정시키는 방법



변수의 이름으로 classpath, 변수 값으로 경로 정보 전달하면 클래스 패스가 시스템 전체에 적용이 된다.

But! 좋은 방법 아니므로, 이렇듯 클래스 패스를 고정시키는 일이 가능하다는 사실 정도만 알고 있자.

## 08-2. 패키지의 이해



# 패키지 선언이 필요한 상황의 연출

```
public class Circle {  
    double rad;  
    final double PI;  
  
    public Circle(double r) {  
        rad = r;  
        PI = 3.14;  
    }  
    public double getArea() {  
        return (rad * rad) * PI;  
    }  
}
```

[www.wxfox.com](http://www.wxfox.com)의 Circle.java

```
public class Circle {  
    double rad;  
    final double PI;  
  
    public Circle(double r) {  
        rad = r;  
        PI = 3.14;  
    }  
    public double getPerimeter() {  
        return (rad * 2) * PI;  
    }  
}
```

[www.fxmx.com](http://www.fxmx.com)의 Circle.java

## 공간에서의 충돌

동일 이름의 클래스 파일을 같은 위치에 둘 수 없다.

## 접근 방법에서의 충돌

인스턴스 생성 방법에서 두 클래스에 차이가 없다.

# 공간적, 접근적 충돌 해결을 위한 패키지 선언

## 클래스 접근 방법의 구분

- 서로 다른 패키지의 두 클래스는 인스턴스 생성 시 사용하는 이름이 다르다.

## 클래스의 공간적인 구분

- 서로 다른 패키지의 두 클래스 파일은 저장되는 위치가 다르다.

컴파일 과정에서, 클래스 파일이 저장되어야 하는 위치를 상대적으로 결정이 된다.

그리고 이렇게 결정된 위치는 컴파일 이후에 바꿀 수 없다.

# 패키지 선언에 따른 문제 해결

```
package com.wxfx.smart;
```

```
public class Circle {  
    double rad;  
    final double PI;
```

```
    public Circle(double r)  
{ ... }
```

```
    public double getArea()  
{ ... }  
} www.wxfx.com의 Circle.java
```

```
package com.fxmx.simple;
```

```
public class Circle {  
    double rad;  
    final double PI;
```

```
    public Circle(double r) { ... }  
    public double getPerimeter()
```

```
{ ... }  
} www.fxmx.com의 Circle.java
```

패키지 이름은 모두 소문자로 구성

인터넷 도메인 이름의 역순으로 이름을 구성

이름 끝에 클래스를 정의한 주체 또는 팀의 이름 추가

```
com.wxfx.smart.Circle c1 = new com.wxfx.smart.Circle(3.5);
```

```
com.fxmx.simple.Circle c2 = new com.fxmx.simple.Circle(5.5);
```

-d 옵션을 주고 컴파일 하면 패키지 디렉토리도 자동 생성

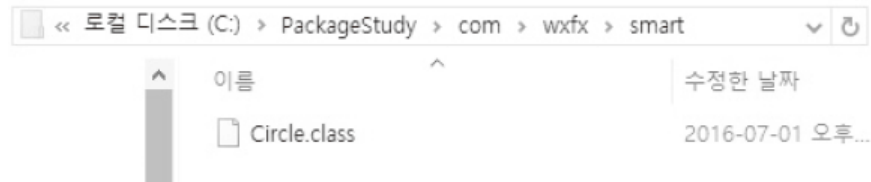
# 패키지 선언이 된 소스파일 컴파일 방법

```
C:\PackageStudy>javac -d <directory> <filename>
```

<directory> 패키지를 생성할 위치 정보

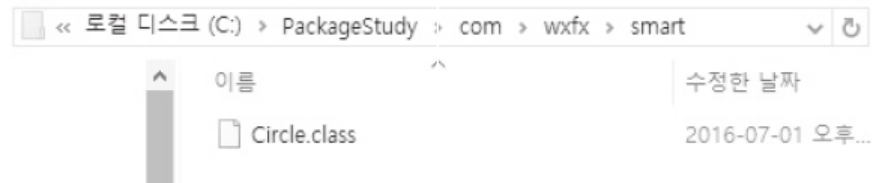
<filename> 컴파일할 파일의 이름

```
C:\PackageStudy>javac -d . src\circle1\Circle.java
```



# 패키지로 묶인 클래스의 접근

```
com.wxfx.smart.Circle c1 = new com.wxfx.smart.Circle(3.5);
```



클래스 패스로 . . .

패키지 지정으  
로 . . .

# 클래스 하나에 대한 import 선언

## ◆ ImportCircle.java

```
1. import com.wxfx.smart.Circle;
2.
3. class ImportCircle {
4.     public static void main(String args[]) {
5.         Circle c1 = new Circle(3.5);
6.         System.out.println("반지름 3.5 원 넓이: " + c1.getArea());
7.         Circle c2 = new Circle(5.5);
8.         System.out.println("반지름 5.5 원 넓이: " + c2.getArea());
9.     }
10. }
```

```
import com.wxfx.smart.Circle;
import com.fxmx.simple.Circle;
```

동일 이름의 두 클래스에 대한 import 선언은 컴파일 오류

# 패키지 전체에 대한 import 선언

```
import com.wxfx.smart.*;
```

com.wxfx.smart 패키지로 묶인 전체 클래스에 대한 패키지 선언