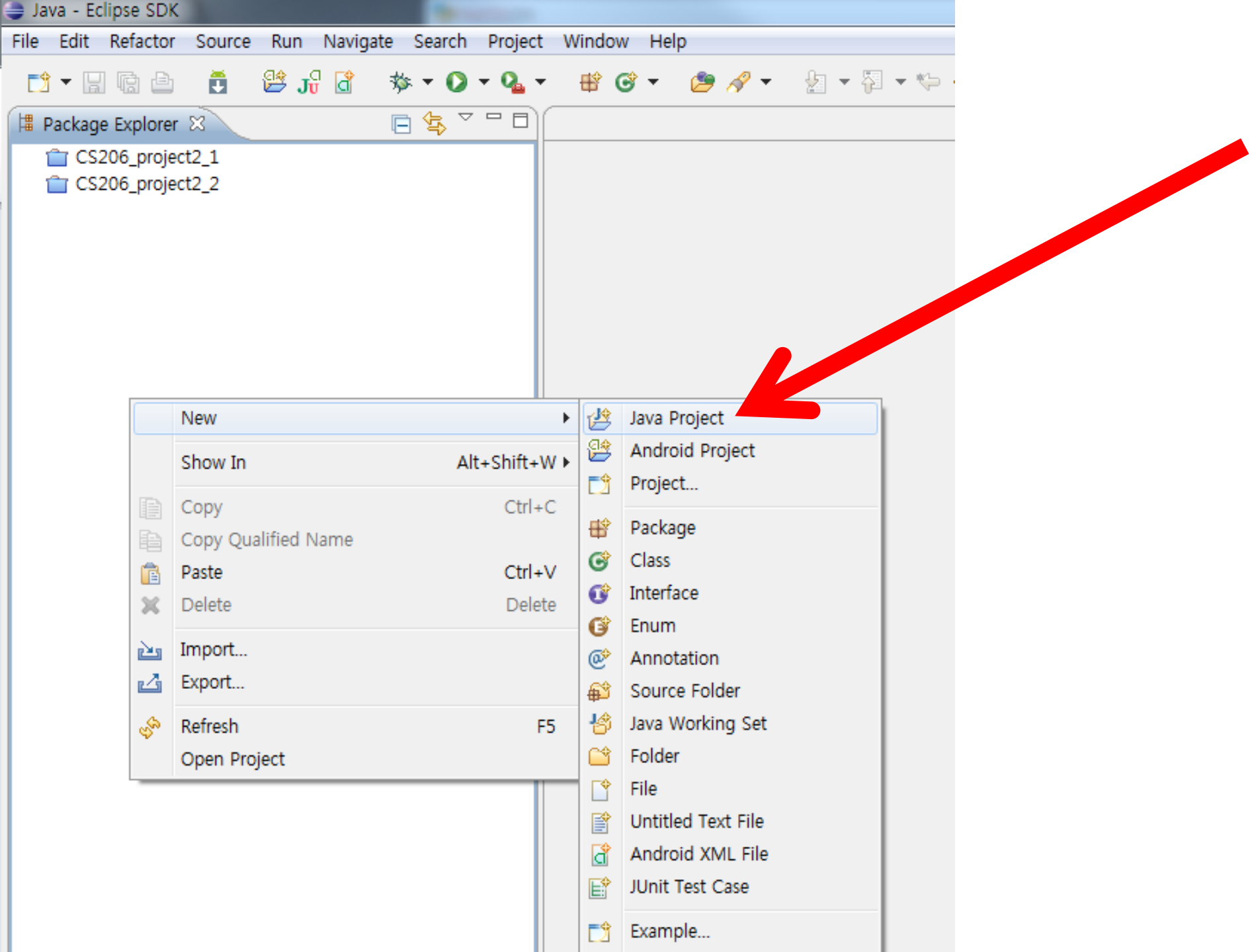
눈이 편안한 자바세미나

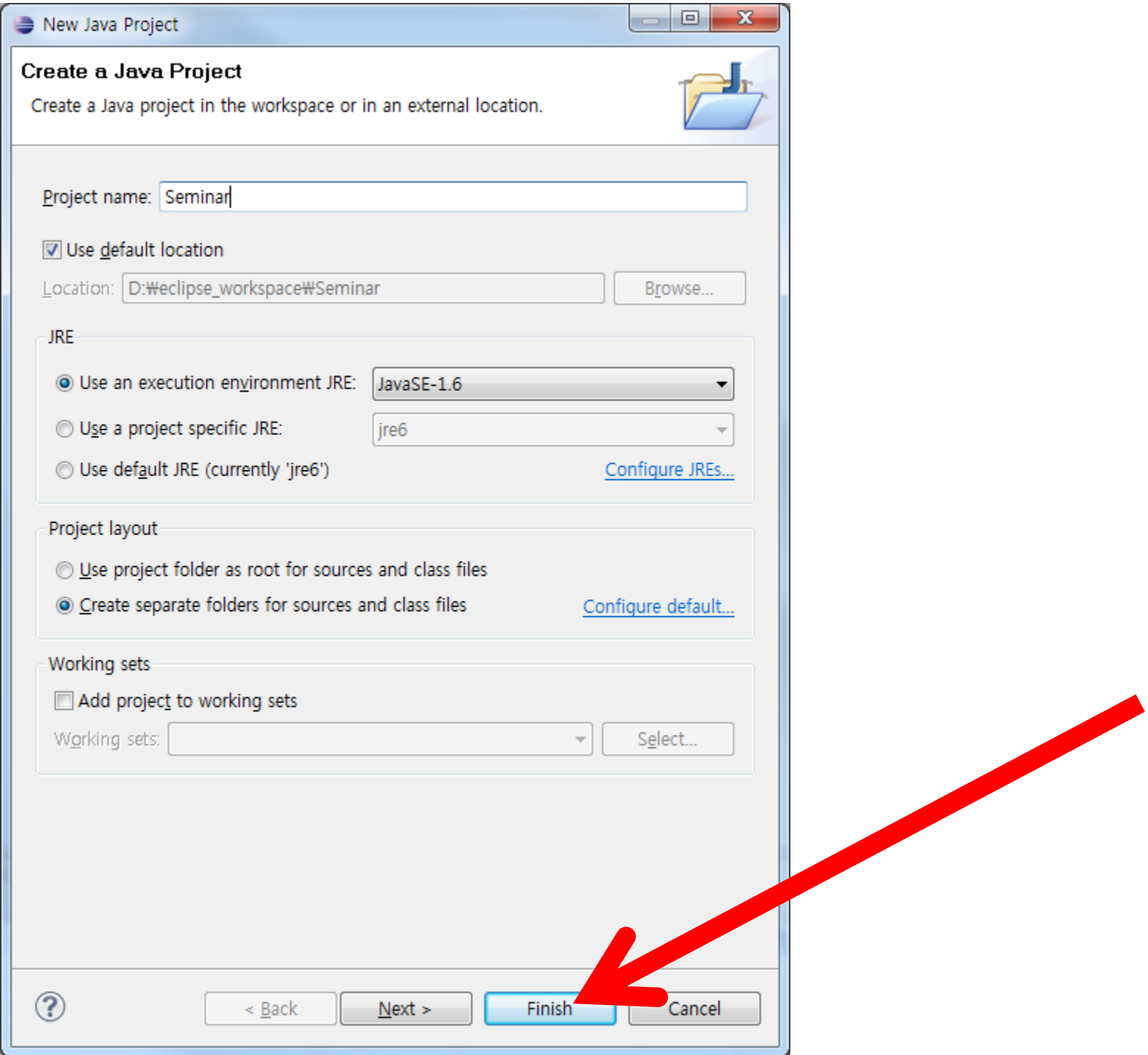
leeopop

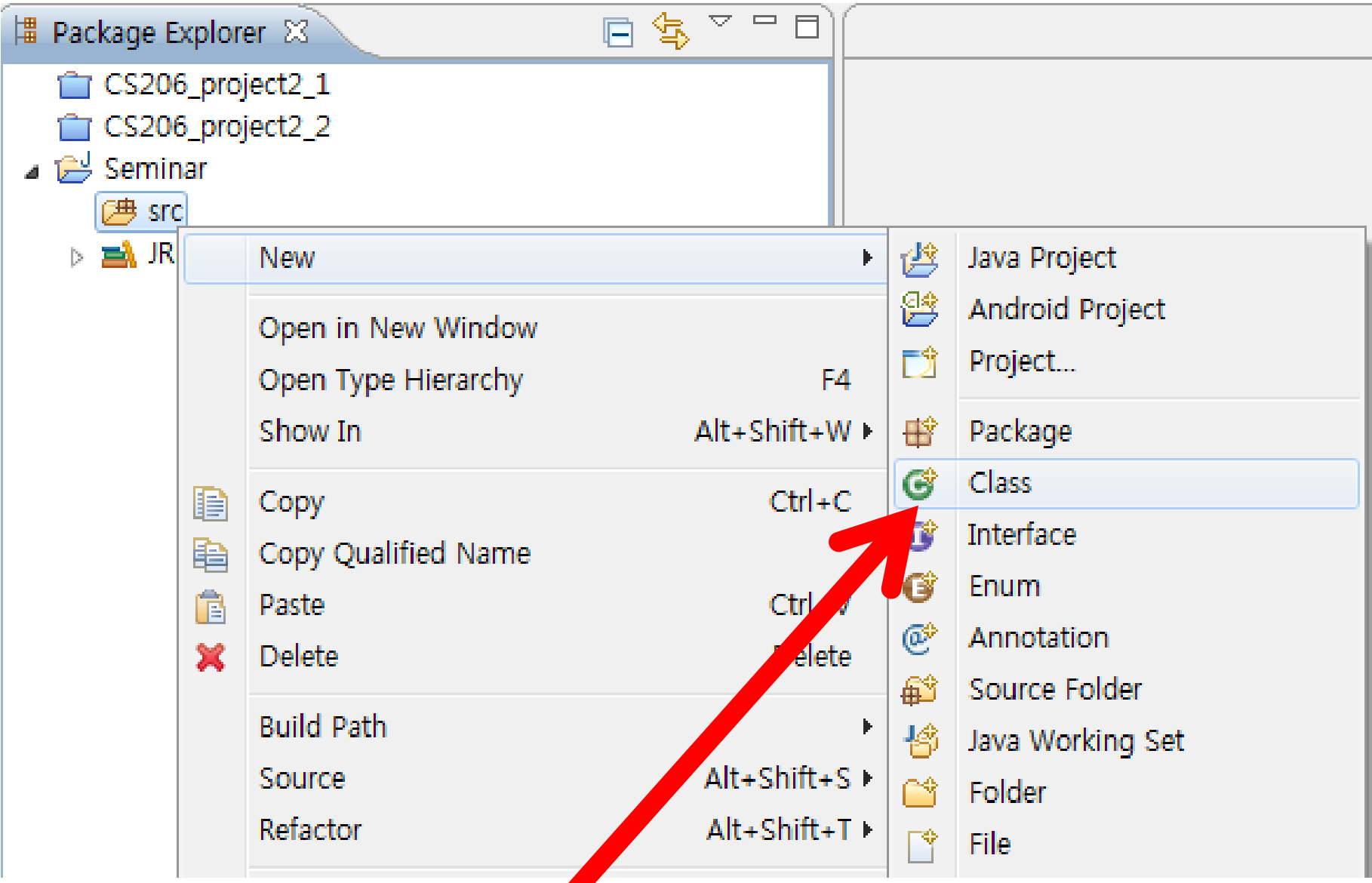
문법을 처음 배울 땐

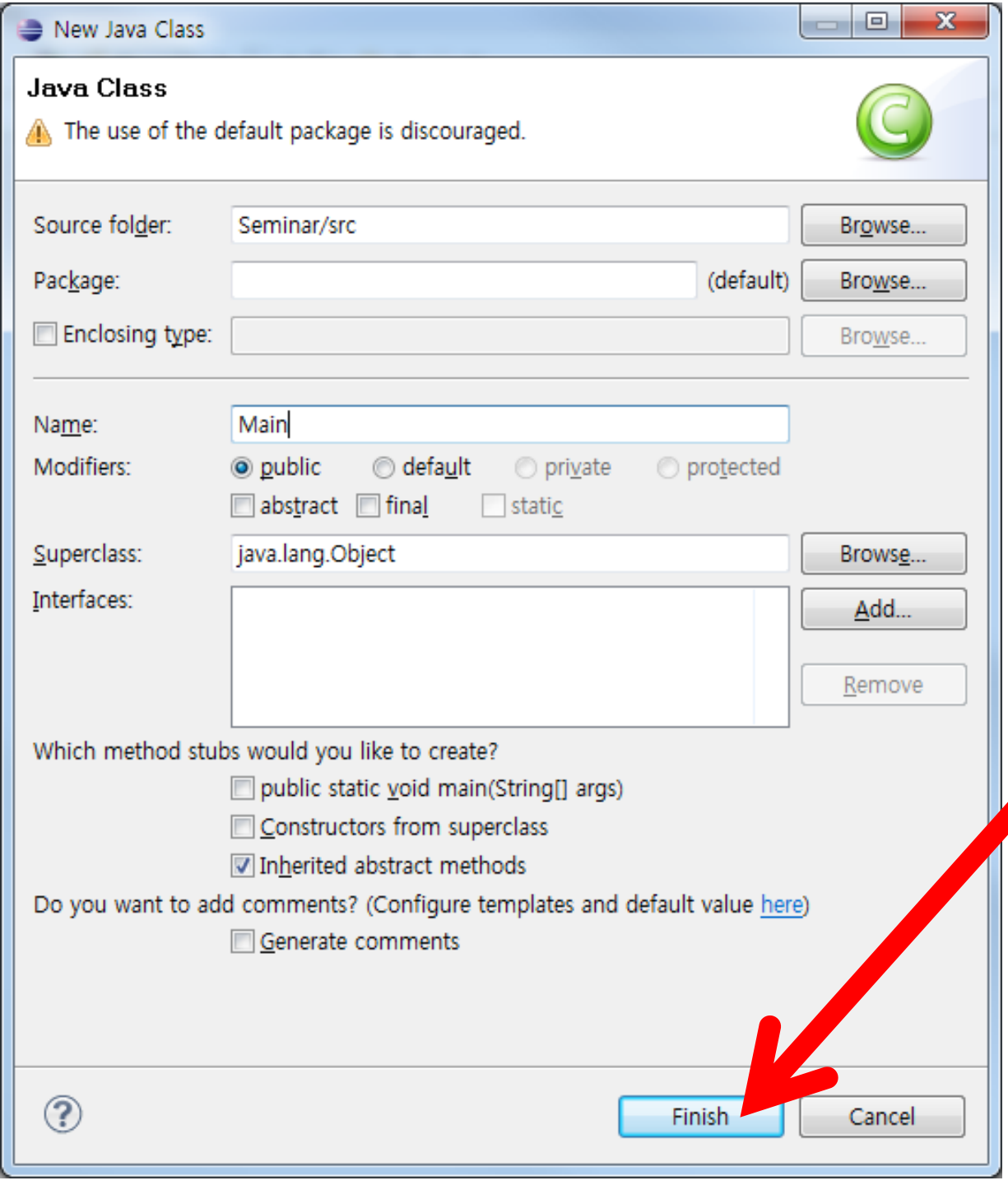
- 우선 무작정 안녕세상아 를 따라합니다.

방법


Java - Eclipse SDK
File Edit Refactor Source Run Navigate Search Project Window Help
Package Explorer
CS206_project2_1
CS206_project2_2
New
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Import...
Export...
Refresh F5
Open Project
Java Project
Android Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
Android XML File
JUnit Test Case
Example...

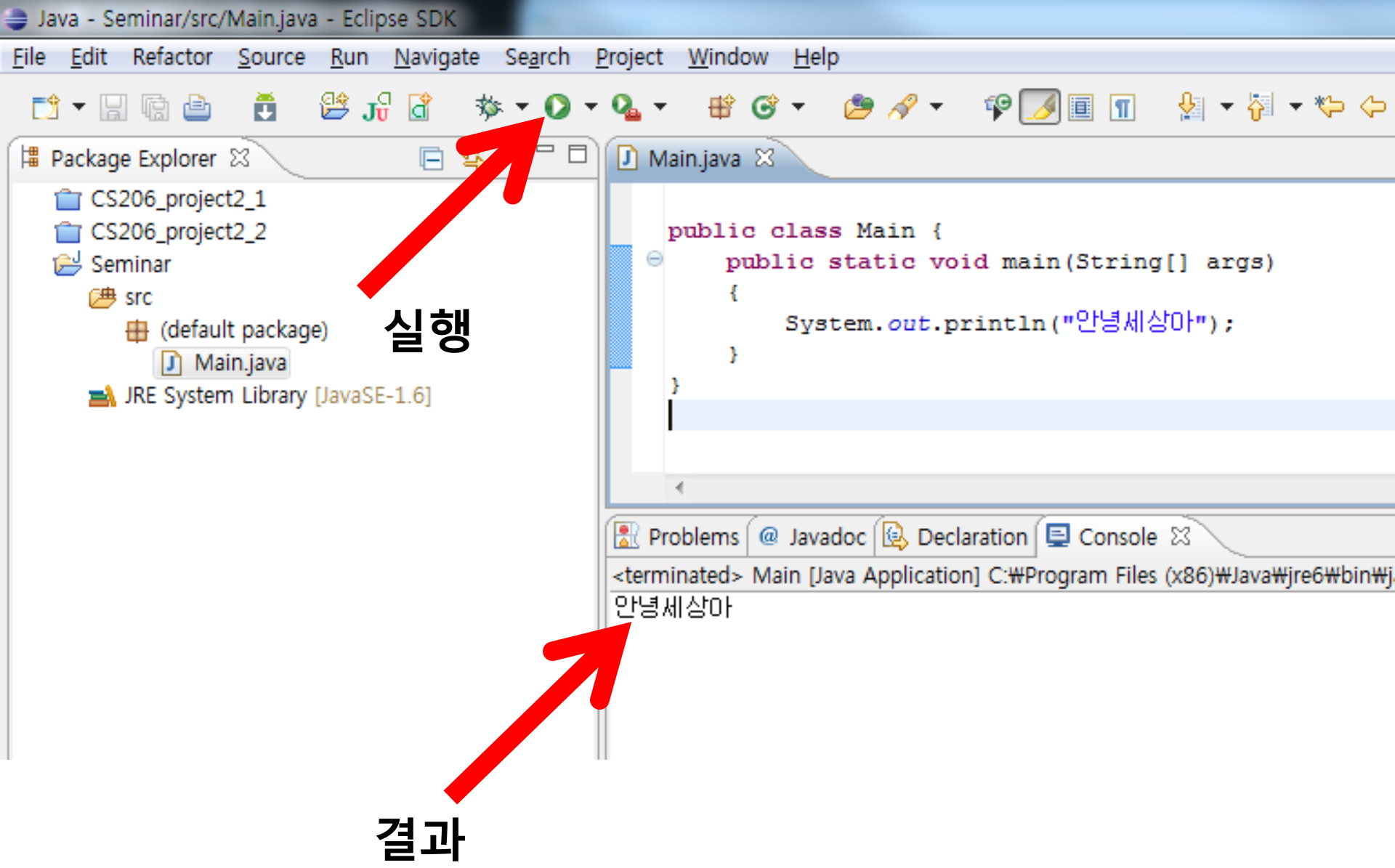

New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: Seminar
Use default location
Location: D:₩eclipse_workspace₩Seminar
Browse...
JRE
Use an execution environment JRE: JavaSE-1.6
Use a project specific JRE: jre6
Use default JRE (currently 'jre6')
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
Working sets:
Select...
< Back
Next >
Finish
Cancel


Package Explorer
CS206_project2_1
CS206_project2_2
Seminar
src
JR
New
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Java Project
Android Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File


New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
Seminar/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
Main
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel

public class Main 안에 다음과 같이 칩니다.

```
public static void main(String[] args)
{
        System.out.println(“안녕세상아”);
}
```


Java - Seminar/src/Main.java - Eclipse SDK
File Edit Refactor Source Run Navigate Search Project Window Help
Package Explorer
CS206_project2_1
CS206_project2_2
Seminar
src
(default package)
Main.java
JRE System Library [JavaSE-1.6]
실행
Main.java
public class Main {
public static void main(String[] args)
{
System.out.println("안녕세상아");
}
}
Problems Javadoc Declaration Console
<terminated> Main [Java Application] C:₩Program Files (x86)₩Java₩jre6₩bin₩j
안녕세상아
결과

참 쉽죠? 그럼 이제 조건문을 보도록 하겠습니다.

아, 그 전에 조건문에 사용하는 변수를 먼저 보져

```
boolean a;
char b;
byte c;
short d;
int e;
float f;
long g;
double h;
```

이 순서대로 대입 가능

d = c;
e = d;
…

절묘한 화살표 길이

위에 두 놈은 대입 불가

단 char는 int->char는
안되지만 char->int는 됨

또한 int를 강제로 char로
형변환 할 수 있음

```
if(a > 3)
{
    //Do something
}
else if(a == 3)
{
    //Do something
}
else
{
    //Do something
}
```

```
if(a > 3)
    //Do something
else if(a == 3)
    //Do something
else
    //Do something
```

한줄짜리 명령은 이렇게도 가능

```
int a = 3;
switch(a)
{
case 3:
     break;
case 4:
     break;
case 5:
     break;
case 6:
     break;
default:
     break;
}
```

크게 다르지 않죠??

for, while, do-while
은 조금 다르게 생겼을 겁니다.

```
for(int k = 0; k < 3; k++)
{
        for(int j = 0; j < k; j++)
        {
            System.out.print("*");
        }
        System.out.println();
}
for(int k = 0; k < 3; k++)
{
        for(int j = 0; j < k; j++)
        {
            System.out.print("*");
        }
        System.out.println();
}
```

충돌안함

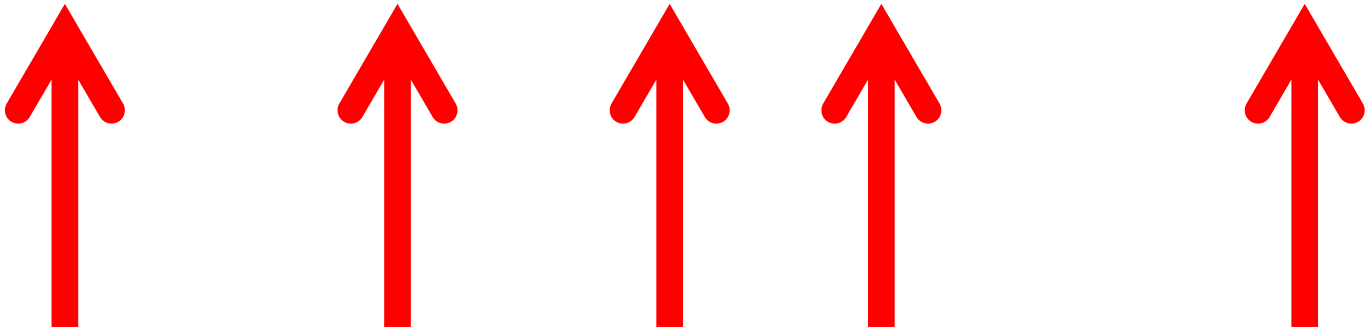
변수 이름 k는
루프 안에서만 살아있고
루프 밖에서는 소멸합니다.
보세요
저렇게 두 개가 같은
소스에 정의되어도
서로 충돌 안 하잖아요.

```
for( 초기화; 조건문; 카운터 증가)


while(조건문)
{
        할일
}

do
{
    할일
}while(조건문);
```

조건문은 반드시 boolean형식의 값을 가지고 있어야 한다.
C나 C++처럼 0 이외의 값을 참으로 받아들이지 않는다.

함수 정의하는법

```
public static void main(String[] args)
```



접근자
accessor

제어자
modifier

반환값
return
value

이름
name,
identifier

인자
parameter

잠깐 과제

“안녕 세상아”를 반환하는 함수를 만들고 그 값을 출력하라

```
Scanner in = new Scanner(System.in);
int a = in.nextInt();
```

이건 키보드로부터 정수를 받아오는 방법입니다.

이것을 바탕으로
0을 입력받으면 “가위”
1을 입력받으면 “바위”
2를 입력받으면 “보”
나머지는 “기권”
을 출력하도록 해 보세요. Switch와 if 둘 다 활용해 보세요

```
Random r = new Random();
int temp = r.nextInt();
```

이를 활용해서 컴퓨터와 이겼는지 졌는지
“기권” 할 때 까지 계속 반복하는 프로그램을 만드세요.

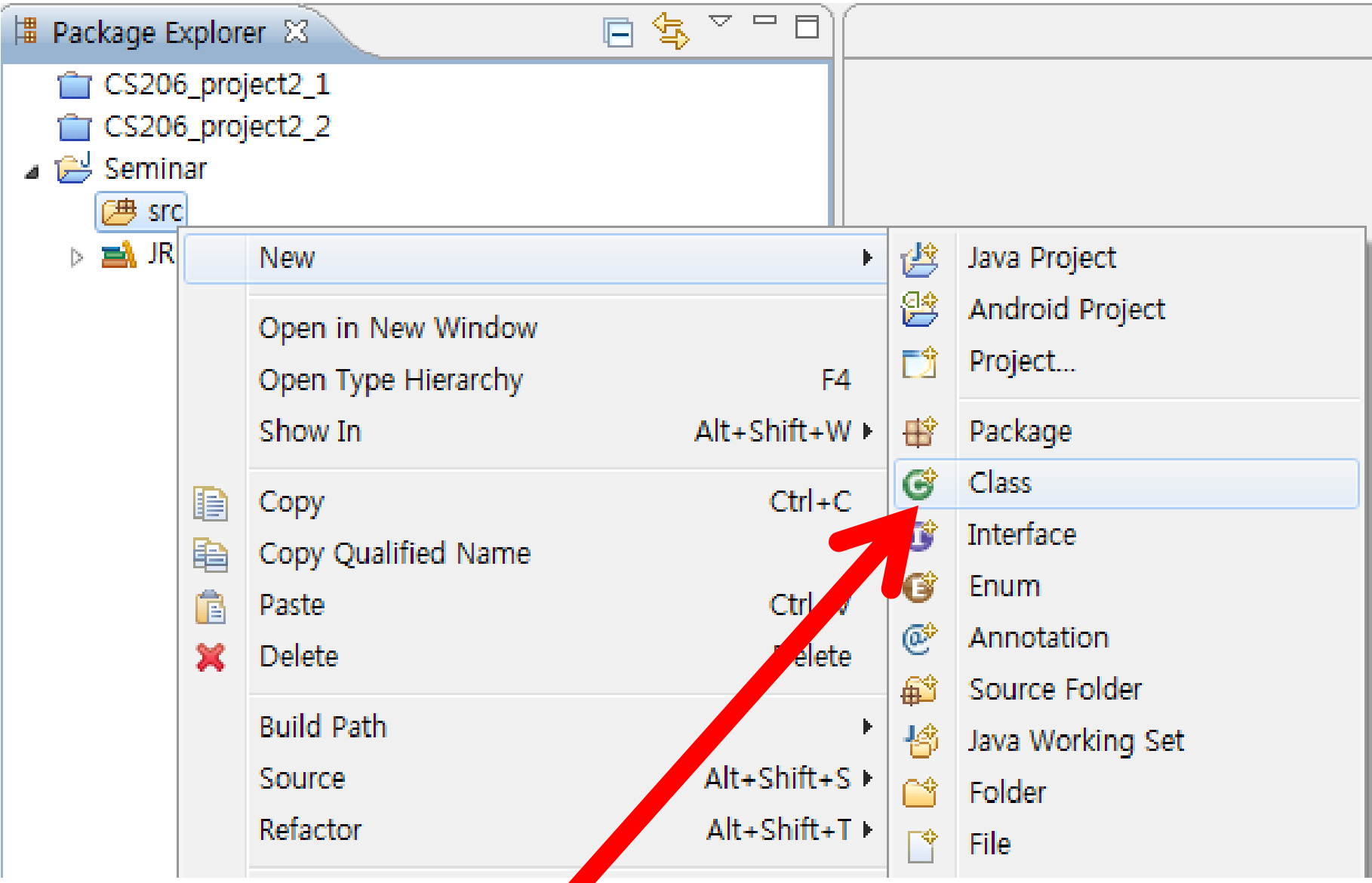
슬라이드 쇼가 끝났습니다. 끝내려면 마우스를 클릭하십시오.

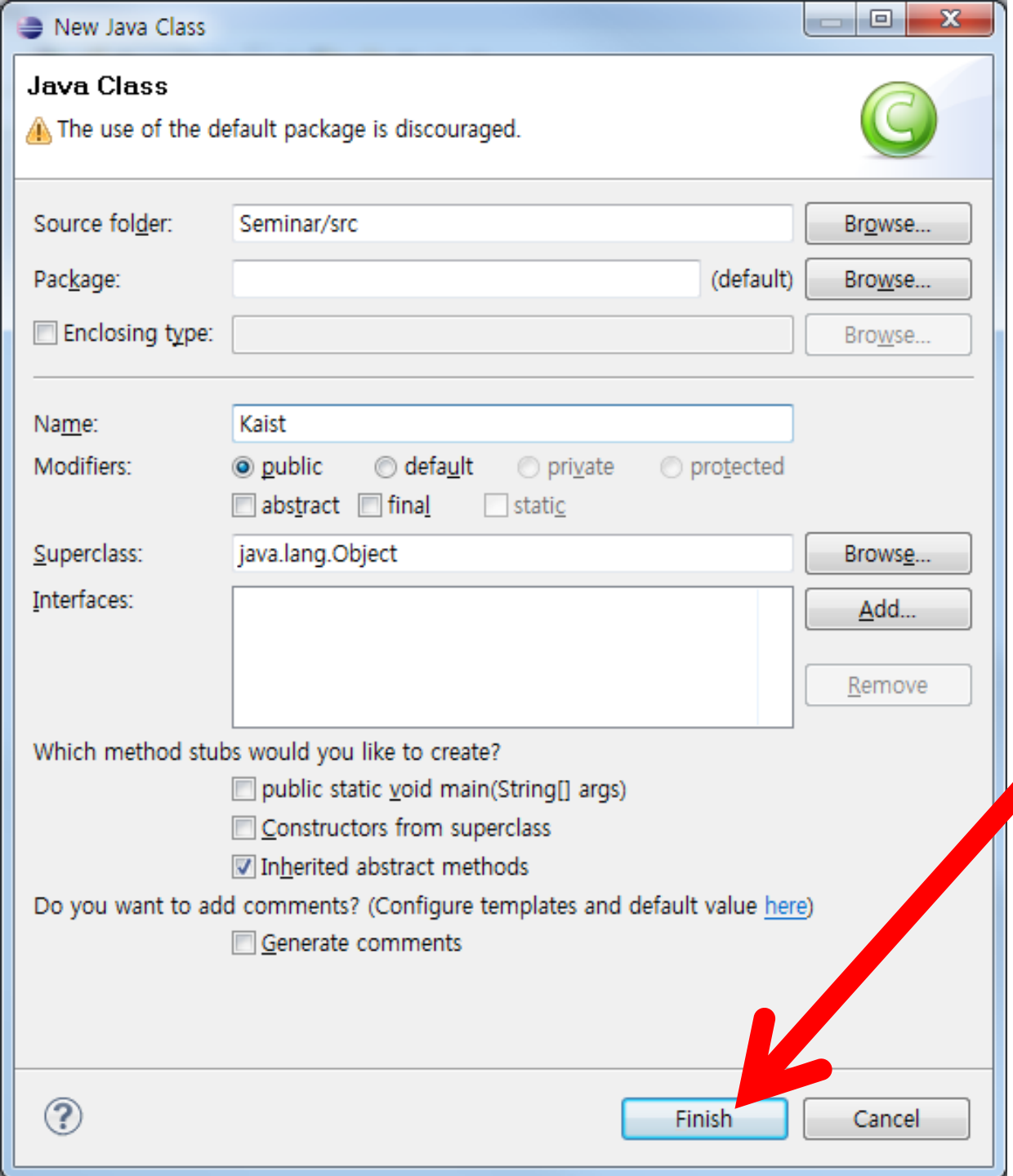
슬라이드 쇼가 끝났습니다. 끝내려면 마우스를 클릭하십시오.

프로그램 구조는 끝났고...
지금부터 객체 지향 프로그래밍을 배워 봅시다.

새로운 클래스를 만들어 봅시다.

용어정리
클래스 : 카이스트 학생
인스턴스(객체) : 20100645

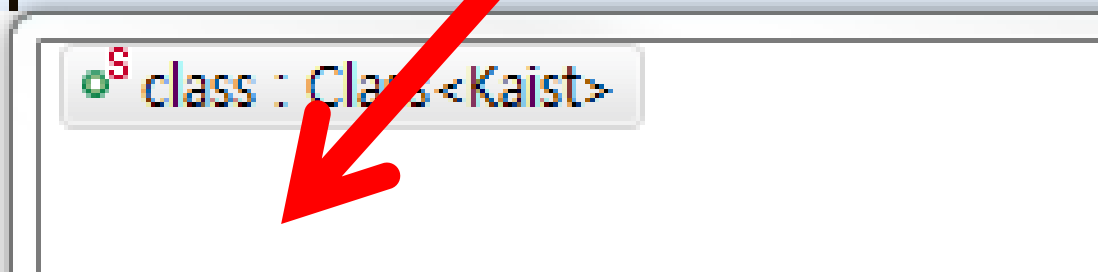

Package Explorer
CS206_project2_1
CS206_project2_2
Seminar
src
JR
New
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Java Project
Android Project
Project...
Package
Class
Interface
Enum
Annotation
Source Folder
Java Working Set
Folder
File


New Java Class
Java Class
The use of the default package is discouraged.
Source folder:
Seminar/src
Browse...
Package:
(default)
Browse...
Enclosing type:
Browse...
Name:
Kaist
Modifiers:
public
default
private
protected
abstract
final
static
Superclass:
java.lang.Object
Browse...
Interfaces:
Add...
Remove
Which method stubs would you like to create?
public static void main(String[] args)
Constructors from superclass
Inherited abstract methods
Do you want to add comments? (Configure templates and default value here)
Generate comments
Finish
Cancel

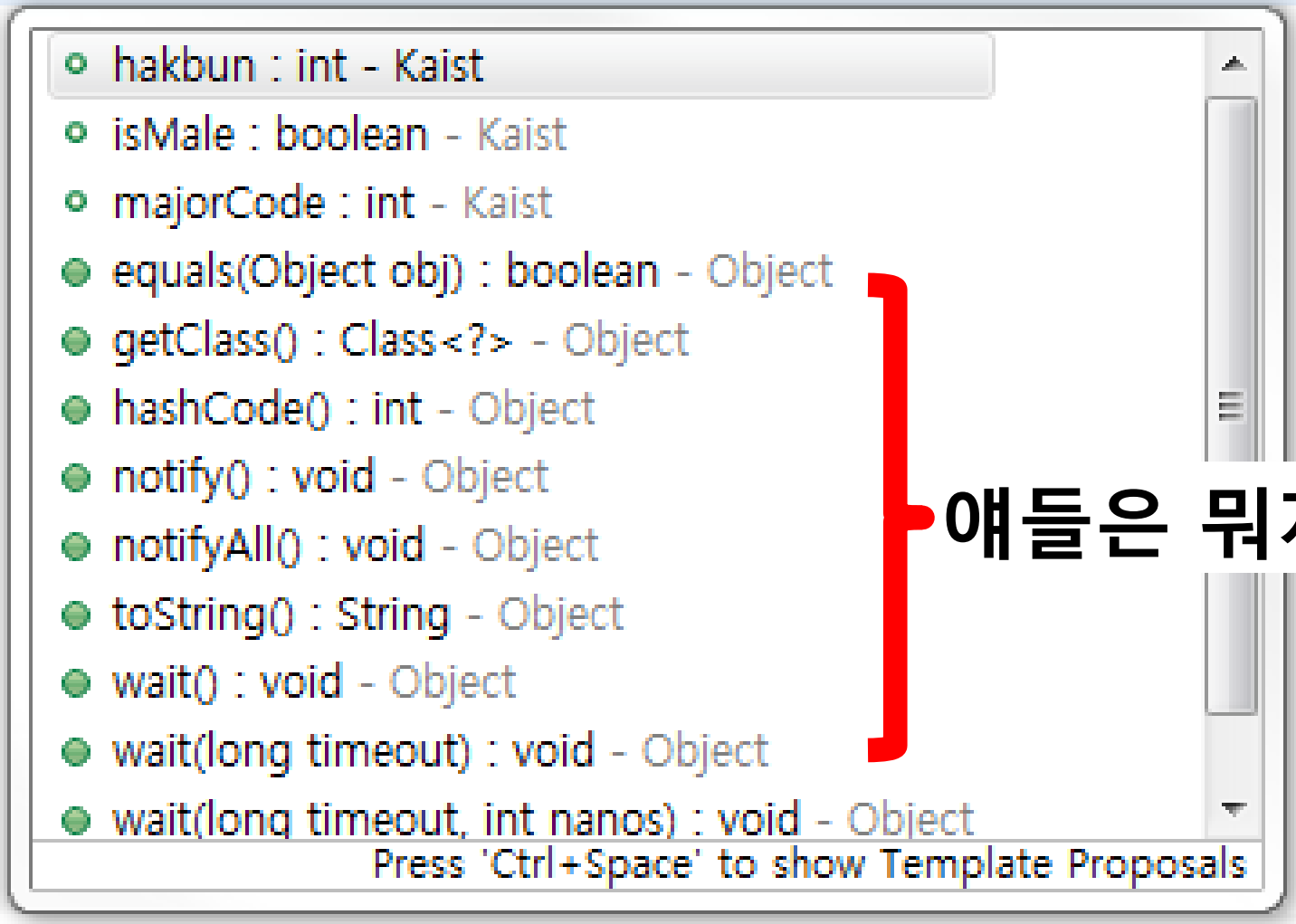
```
public class Kaist {
   public int hakbun;
   public int majorCode;
   public boolean isMale;
}
```

아무것도없네

```
public class Main {
    public static void main(String[] args)
    {
        Kaist.
    }
}
```



```
public class Main {
    public static void main(String[] args)
    {
        Kaist me = new Kaist();
        me.
    }
}
```



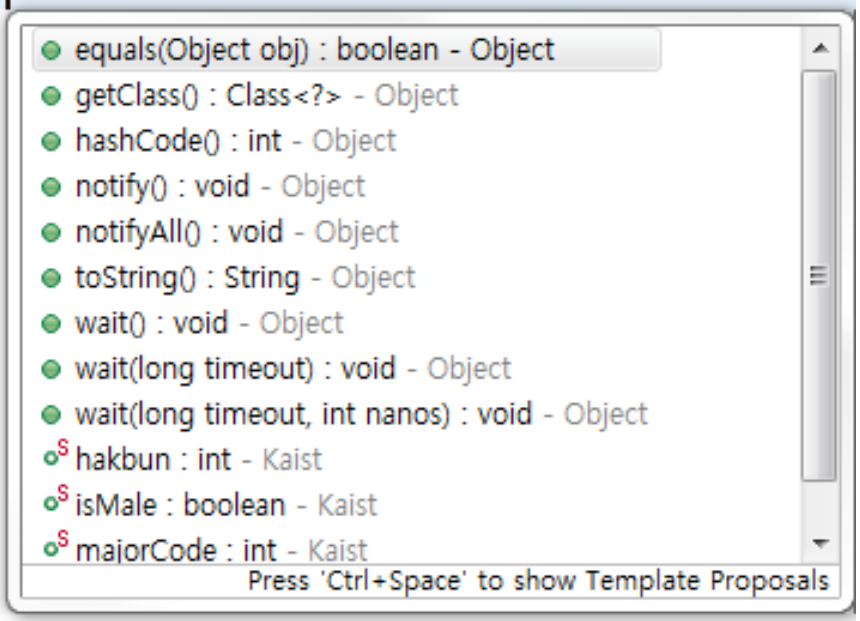
MOAB : Mother Of All Bomb

Object : Mother Of All Object

이번에는 다음과 같이 해 보자

```
public class Kaist {
   public static int hakbun;
   public static int majorCode;
   public static boolean isMale;
}
```

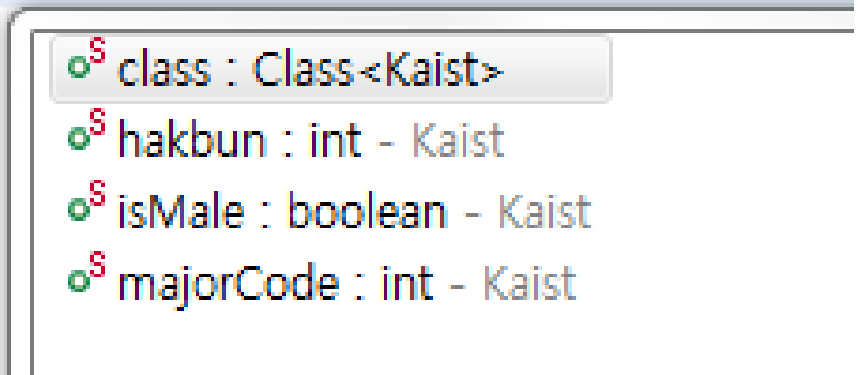
```
public class Main {
    public static void main(String[] args)
    {
        Kaist me = new Kaist();
        me.
    }
}
```



Non-static :
각 객체가 각각의 값을 가짐

Static :
객체과 관계없이 오직 하나만 생성

```
public class Main {
    public static void main(String[] args)
    {
        Kaist.
    }
}
```



모범 활용 사례

```
public class Kaist {
    public int hakbun;
    public int majorCode;
    public boolean sex;

    public static int CS = 1;
    public static int EE = 2;
    public static boolean MALE = true;
    public static boolean FEMALE = false;
}
```

알아두세요 : 자바에서는 하나의 .java 파일에 오직 하나의 클래스 선언 가능, 물론 이름이 같아야 함

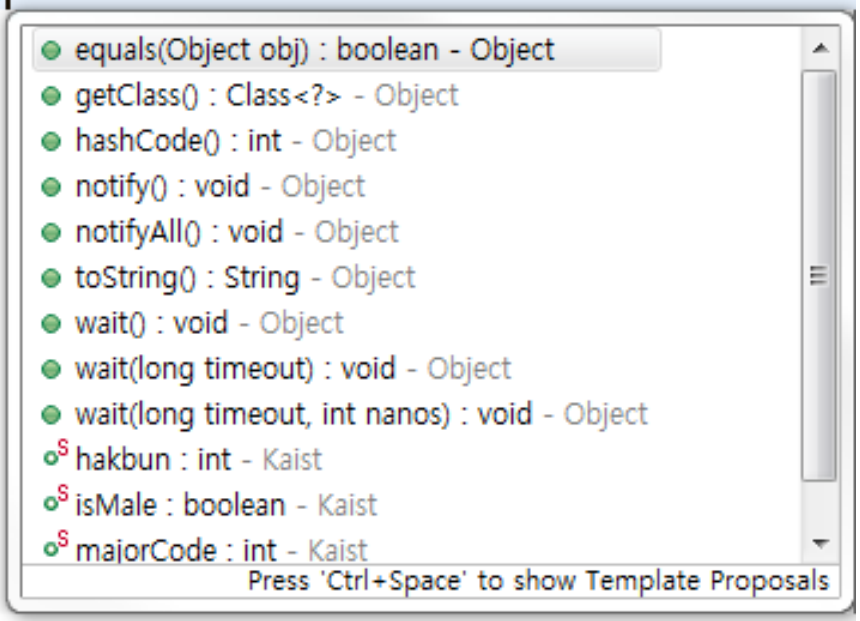
자.
a.Shout();
b.Shout();
를 실행했을 때
I'm CS!
I'm MALE!
I'm EE!
I'm FEMALE!
라고 외치게 해 보세요.

자.
Kaist.Shount2()
를 실행했을 때
I'm Kaist!
라고 외치게 해 보세요.

자꾸자꾸 더 하고 싶어지네....

모든 non-static member variable을 private로 만들어 보세요

```
public class Main {
    public static void main(String[] args)
    {
        Kaist me = new Kaist();
        me.
    }
}
```



아무것도 없네... 어쩌지...

```
public int
getHakbun()
{
    return hakbun;
}
```

=>Getter

```
public void setHakbun(int newHakbun)
{
    hakbun = newHakbun;
}
```

=>Setter

“

멤버 변수가 200개면 200개 다 만들어야 하나요 ㅠㅠ

”

“

네... 그렇습니다...

”

“

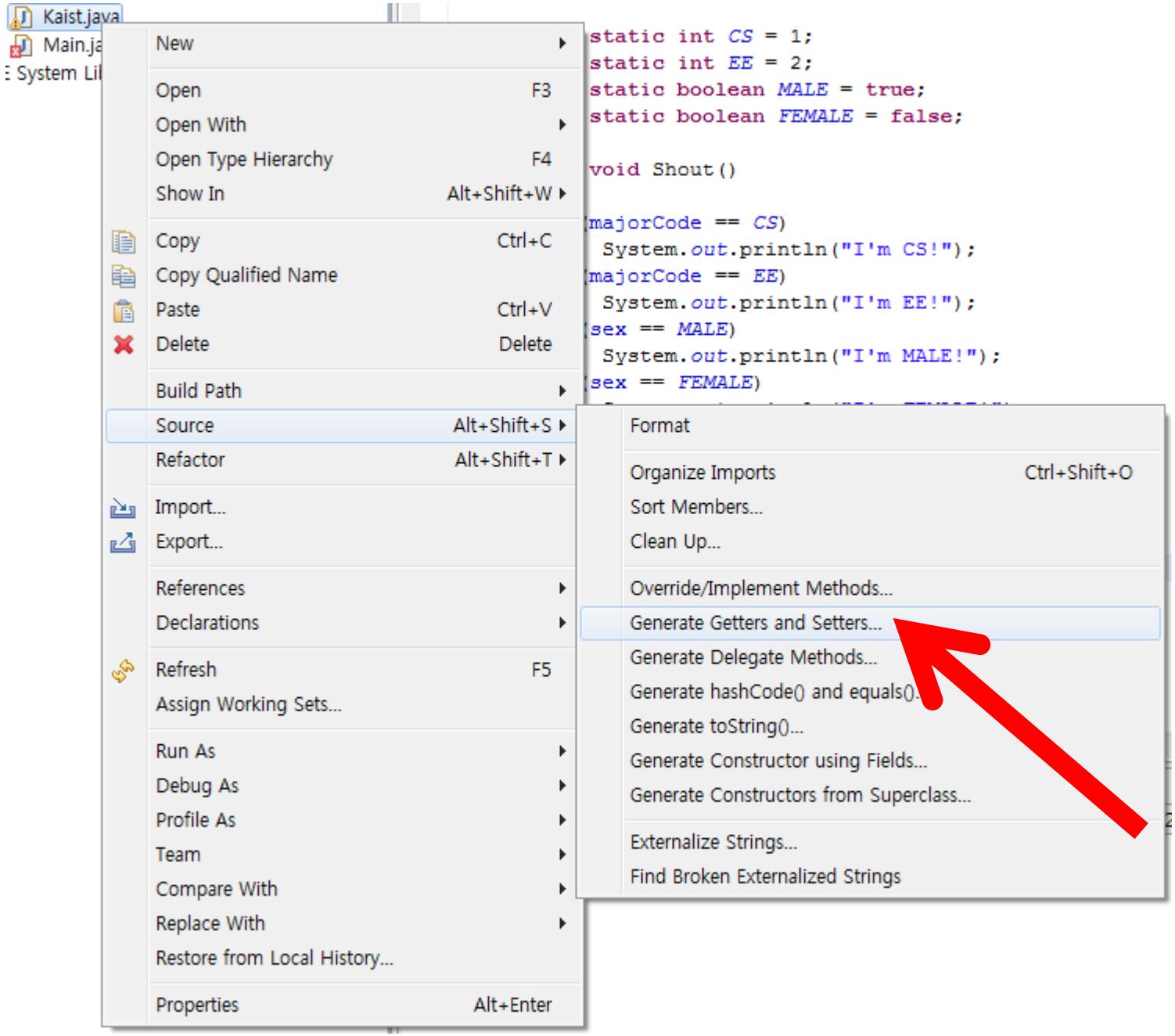
이클립스 너무 안좋아요 자바 때려칠래요...

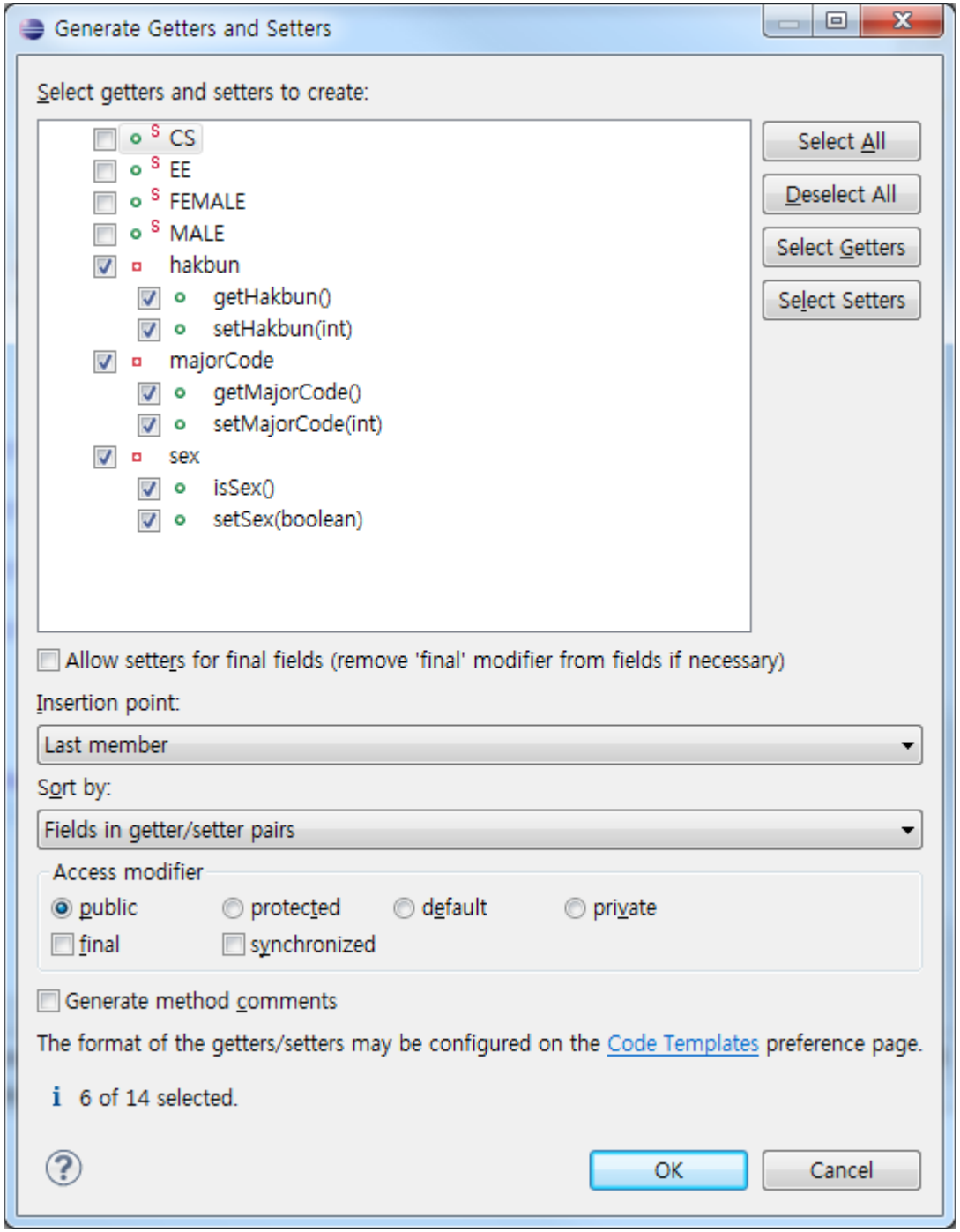
”

“

저런저런 이클립스는 자바 세상에서 우리를 숨쉬고 살아가게 하는 존재입니다.
이클립스로 못하는건 없어요.

”


Kaist.java
Main.ja
System Li
static int CS = 1;
static int EE = 2;
static boolean MALE = true;
static boolean FEMALE = false;
void Shout()
majorCode == CS)
System.out.println("I'm CS!");
majorCode == EE)
System.out.println("I'm EE!");
sex == MALE)
System.out.println("I'm MALE!");
sex == FEMALE)
New
Open F3
Open With
Open Type Hierarchy F4
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
References
Declarations
Refresh F5
Assign Working Sets...
Run As
Debug As
Profile As
Team
Compare With
Replace With
Restore from Local History...
Properties Alt+Enter
Format
Organize Imports Ctrl+Shift+O
Sort Members...
Clean Up...
Override/Implement Methods...
Generate Getters and Setters...
Generate Delegate Methods...
Generate hashCode() and equals()...
Generate toString()...
Generate Constructor using Fields...
Generate Constructors from Superclass...
Externalize Strings...
Find Broken Externalized Strings


Generate Getters and Setters
Select getters and setters to create:
CS
EE
FEMALE
MALE
hakbun
getHakbun()
setHakbun(int)
majorCode
getMajorCode()
setMajorCode(int)
sex
isSex()
setSex(boolean)
Select All
Deselect All
Select Getters
Select Setters
Allow setters for final fields (remove 'final' modifier from fields if necessary)
Insertion point:
Last member
Sort by:
Fields in getter/setter pairs
Access modifier
public
protected
default
private
final
synchronized
Generate method comments
The format of the getters/setters may be configured on the Code Templates preference page.
6 of 14 selected.
OK
Cancel

```
public void setMajorCode(int majorCode) {
        this.majorCode = majorCode;
}

public boolean isSex() {
        return sex;
}

public void setSex(boolean sex) {
        this.sex = sex;
}
```

아름답죠??

this가 뭐냐구요?

자기 자신 을 나타내는 키워드 입니다.

저렇게 내가 가지고 있는 변수 이름 hakbun과 parameter의 hakbun이 겹치는 경우 this로 구분지어 줍니다.

아무것도 없으면 parameter입니다.

```
public int getHakbun() {
        return hakbun;
}

public void setHakbun(int hakbun)
{
        this.hakbun = hakbun;
}

public int getMajorCode() {
        return majorCode;
}
```