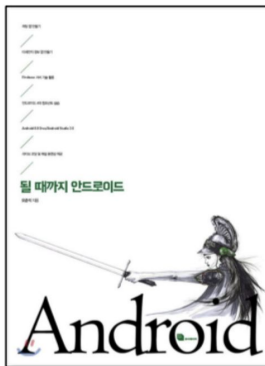


Java 입문

0장. Java를 시작하자

교육하고 책 쓰는 개발자 오준석

- Android App Developer (2009 ~)
- Flutter App Developer (2019 ~)
- 카카오 안드로이드 신입 교육
- 플러터 개발자 양성 교육
- 국내 플러터 1호 책 외 2종 집필
- 오준석의 생존코딩 유튜브 채널 운영
- 인프런에 다수의 강의
- 일본에서 개발자 경력
- LG전자 MC사업부
- 컴퓨터 학원, 직업학교 운영
- 알리미프로 서비스 운영



전반전 목차

- **Java**를 시작하자
- 프로그램의 작성 방법
- 식과 연산자
- 조건분기와 반복
- 배열
- 메소드
- 복수 클래스를 사용한 개발
- 객체 지향 프로그래밍
- 인스턴스와 클래스
- 클래스
- 캡슐화
- 상속
- 추상클래스와 인터페이스
- 다형성
- **Java** 표준 클래스
- 예외
- **Java**로 할 수 있는 것들

Java의 특징

- 배우기 쉬운 표준적인 **기본 문법**
- 대규모 개발을 지원하는 **객체지향 프로그래밍**
- 풍부한 표준 명령어
- 다양한 컴퓨터에서 똑같은 동작을 하는 **범용성**
- 개발자가 메모리 관리에 신경쓰지 않아도 됨

Java로 할 수 있는 것

- PC용 응용프로그램
- 서버
- 안드로이드

프로그래밍의 준비

Java 프로그래밍을 체험 해 봅시다.

- Java 프로그램을 작성하기 위해 비싼 소프트웨어나 최신 컴퓨터가 필요할까요?
- 어려운 설정?

-> 아닙니다. 일반적인 컴퓨터와 인터넷 연결만 있으면 바로 만들 수 있습니다

<https://repl.it>

The screenshot displays the repl.it web interface. The browser's address bar shows the URL `repl.it/@JunsukOh/AcclaimedDarkcyanBlock`. The interface includes a top navigation bar with a menu icon, a profile picture, the username `@JunsukOh/AcclaimedDarkcyanBlock`, and buttons for `invite`, `run`, and `share`. Below this is a file explorer on the left showing a file named `Main.java`. The main area is a code editor with a dark theme, displaying the following Java code:

```
1 class Main {  
2     public static void main(String[] args) {  
3         System.out.println("Hello world!");  
4     }  
5 }
```

The code is labeled as `saved`. On the right side, there is a terminal window showing the output of the program:

```
OpenJDK Runtime Environment (bu  
.04.1)  
javac -classpath ./run_dir/j  
. Main.java  
java -classpath ./run_dir/ju  
n  
Hello world!  
>
```

<https://onlinegdb.com>

The screenshot displays the OnlineGDB web application. The browser's address bar shows the URL `onlinegdb.com`. The interface includes a top navigation bar with various icons and a sidebar on the left with the following links: OnlineGDB beta, online compiler and debugger for c/c++, code. compile. run. debug. share., IDE, My Projects, Learn Programming, Programming Questions, Jobs (marked as new), Sign Up, and Login. At the bottom of the sidebar are social media icons for Facebook and Twitter, and a button indicating 60.7K users. The main area features a code editor with a file named `Main.java`. The editor has a toolbar with buttons for Run, Debug, Stop, Share, Save, Beautify, and Download. The code in the editor is as follows:

```
1- /*****
2
3- Welcome to GDB Online.
4- GDB online is an online compiler and debugger tool for C, C++, Python, Java, PHP, Ruby, Perl,
5- C#, VB, Swift, Pascal, Fortran, Haskell, Objective-C, Assembly, HTML, CSS, JS, SQLite, Prolog.
6- Code, Compile, Run and Debug online from anywhere in world.
7
8- *****/
9- public class Main
10- {
11-     public static void main(String[] args) {
12-         System.out.println("Hello World");
13-     }
14- }
15-
```


프로그래밍의 3가지 스텝

1. 프로그램의 작성
2. 컴파일
3. 실행

Oh/HightechWheatPixel

invite

run

Main.java



saved

```
1 class Main {  
2     public static void main(String[] args) {  
3         System.out.println("Hello world!");  
4     }  
5 }
```

Run 버튼을 눌러 실행 해 봅시다

```
> java -classpath ./run_dir/junit-4.12.jar Main  
Hello world!
```

```
> 
```

Mission : 화면에 자신이 원하는 문자를 표시해 봅시다

Main.java



saved

```
1  class Main {  
2      public static void main(String[] args) {  
3          System.out.println("오준석의 생존코딩");  
4      }  
5  }
```

밑 줄 부분을 변경 하면 됩니다

Main.java



saved

```
1  class Main {  
2      public static void main(String[] args) {  
3          System.out.println("오준석의 생존코딩");  
4      }  
5  }
```

Mission : 오류를 찾아보고 수정 해 봅시다

Main.java



saved

```
1  class Main {
2      public static void main(String[] args) {
3          System.out.println("오준석");
4          System.out.println("40세");
5          System.out.println("프로그래밍은 취미로 할 때가 제일 재미있습니다");
6      }
7  }
```

```
1  class Main {  
2      public static void main(String[] args) {  
3          System.out.println("오준석");  
4          System.out.println("40세");  
5          System.out.println("프로그래밍은 취미로 할 때가 제일 재미있습니다");  
6          System.out.println(10 + 20);  
7      }  
8  }
```

+, -, *(곱하기), /(나누기) 등의 기호를 사용할 수 있음

```
System.out.println(35 - 10);  
System.out.println(-5 * 2);  
System.out.println(6 * 6 * 3.14);  
System.out.println("정답은" + 64);
```

변수

수학에서 x 나 y 같은 문자로 만든 수식과 비슷합니다

```
5      System.out.println("게임을 좋아합니다");
6      System.out.println("35 + 35의 계산을 합니다");
7      System.out.println(35 + 35);
8      int x;   변수 x를 준비
9      x = 6;   x에 6을 대입
10     System.out.println(x * x * 3.14);
11     }
12 }
```


프로그래밍 체험을 끝내며

실제로 **Java**로 개발된 인터넷 쇼핑몰, 금융 시스템, 게임 등이 돌아가고 있습니다.

“언젠가 만들고 싶은 프로그램”을 자유롭게 상상해 보세요.

제1장 프로그램의 작성 방법

프로그램의 작성 방법

Java에서는, 프로그램의 작성 방법에 대해 여러가지 룰이 정해져 있습니다.

만드는 프로그램의 내용의 규모와 상관없이, 반드시 사용 되어지는 기본적인 룰은 굉장히 중요합니다.

우선은 그런 기본적인 문법을 확실히 알고 넘어가는 것 부터 학습을 시작해 봅시다.

1. 소스코드의 작성

- a. 사람이 읽을 수 있는 프로그램을 **소스 코드** (source code) 또는 소스 라고 한다

2. 컴파일

- a. 컴퓨터는 **CPU**가 프로그램을 해석해서 움직인다
- b. 컴퓨터가 이해 할 수 있도록 **기계어** (machine code)로 소스코드를 변환해야 한다
- c. **컴파일**이라는 처리를 하여 **바이트 코드** (byte code)라는 상태로 변환 됨 (1과 0의 조합)
- d. 이 때 소스코드의 문법 체크가 실행 되고 오류가 발견되면 오류 부분을 출력 해 줌
- e. 이런 과정을 **컴파일러** (compiler) 라는 변환 소프트웨어가 담당 함

3. 실행

- a. 컴파일이 완료되면, **인터프리터** (interpreter) 라고 불리는 소프트웨어에게 바이트코드의 실행을 지시
- b. 인터프리터는 **JVM** (Java Virtual Machine) 이라는 장치를 내부에 가지고 있어, 바이트코드를 한줄씩 읽으며 기계어로 변환하고, 컴퓨터의 **CPU**로 보냄
- c. 컴퓨터는 소스코드에서 지시한 대로 동작 함

개발환경 구축

두 가지 방법

1. 자신의 PC에 컴파일러와 인터프리터를 설치
2. repl.it 를 활용

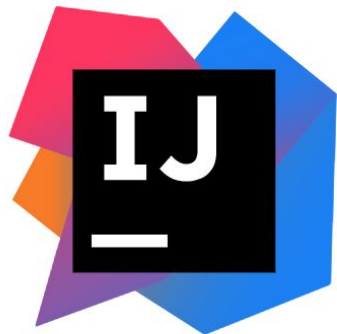
학습용으로는 어떠한 방법을 사용해도 상관 없음

개발환경 구축

자바 개발을 편하게 하기 위한 통합개발환경(Integrated Development Environment, IDE)은 여러가지가 있습니다.

- IntelliJ
- Eclipse
- Visual Studio Code
- 등등

그 전에 JDK (Java Development Kit) 설치 (<https://adoptium.net/>)



버전: 2019.3.4
빌드: 193.6911.18
2020년 3월 17일
[릴리스 노트](#)

다운로드 IntelliJ IDEA

Windows Mac Linux

Ultimate

웹 및 엔터프라이즈 개발용

Download

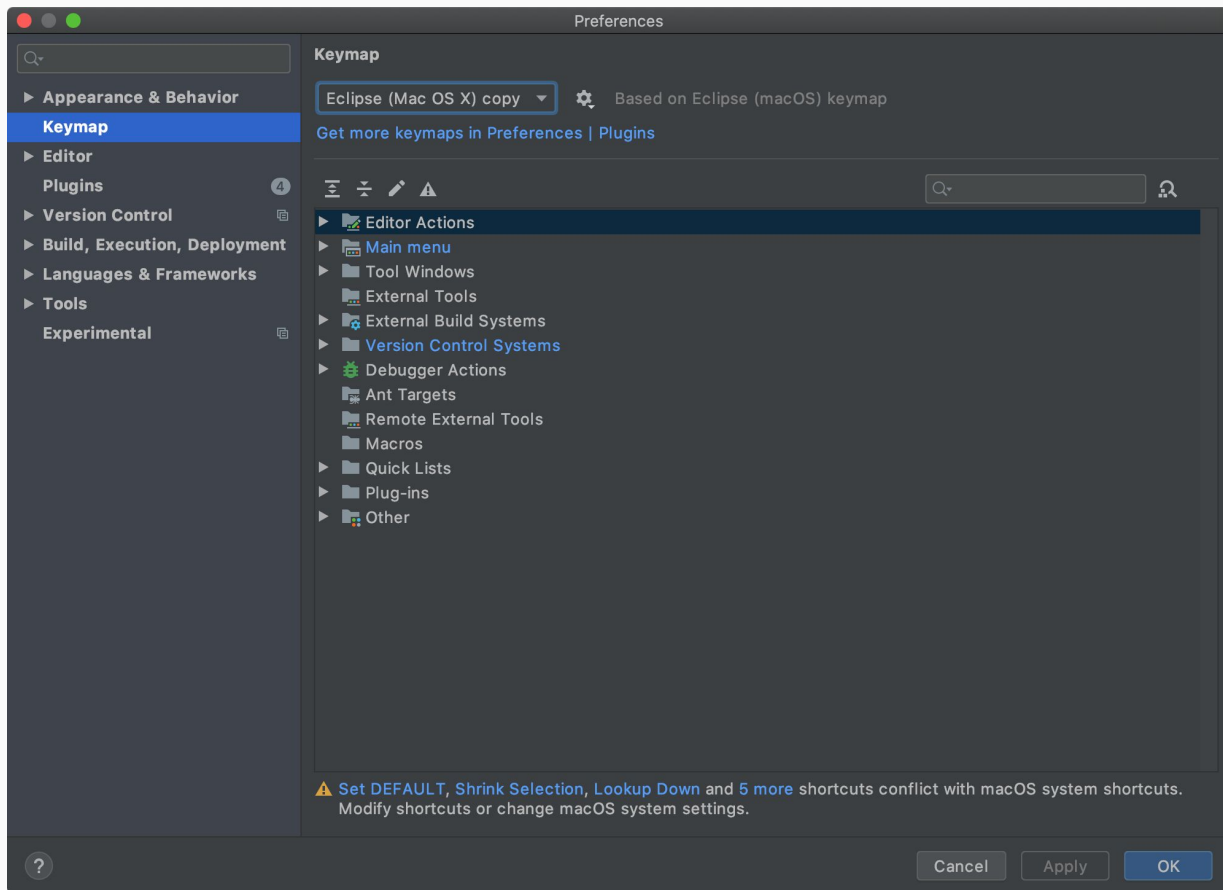
무료 평가판

Community

JVM 및 Android 개발용

Download

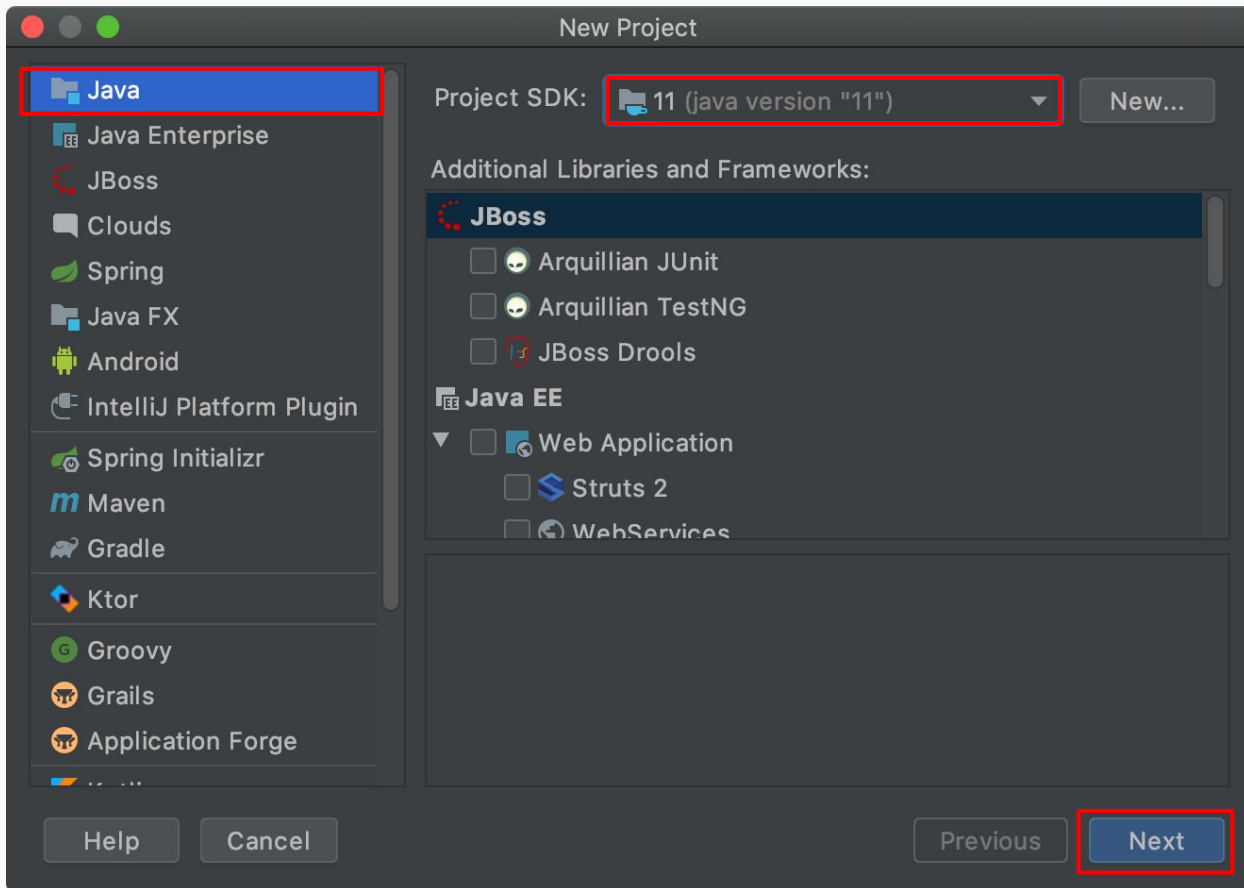
무료, 오픈 소스

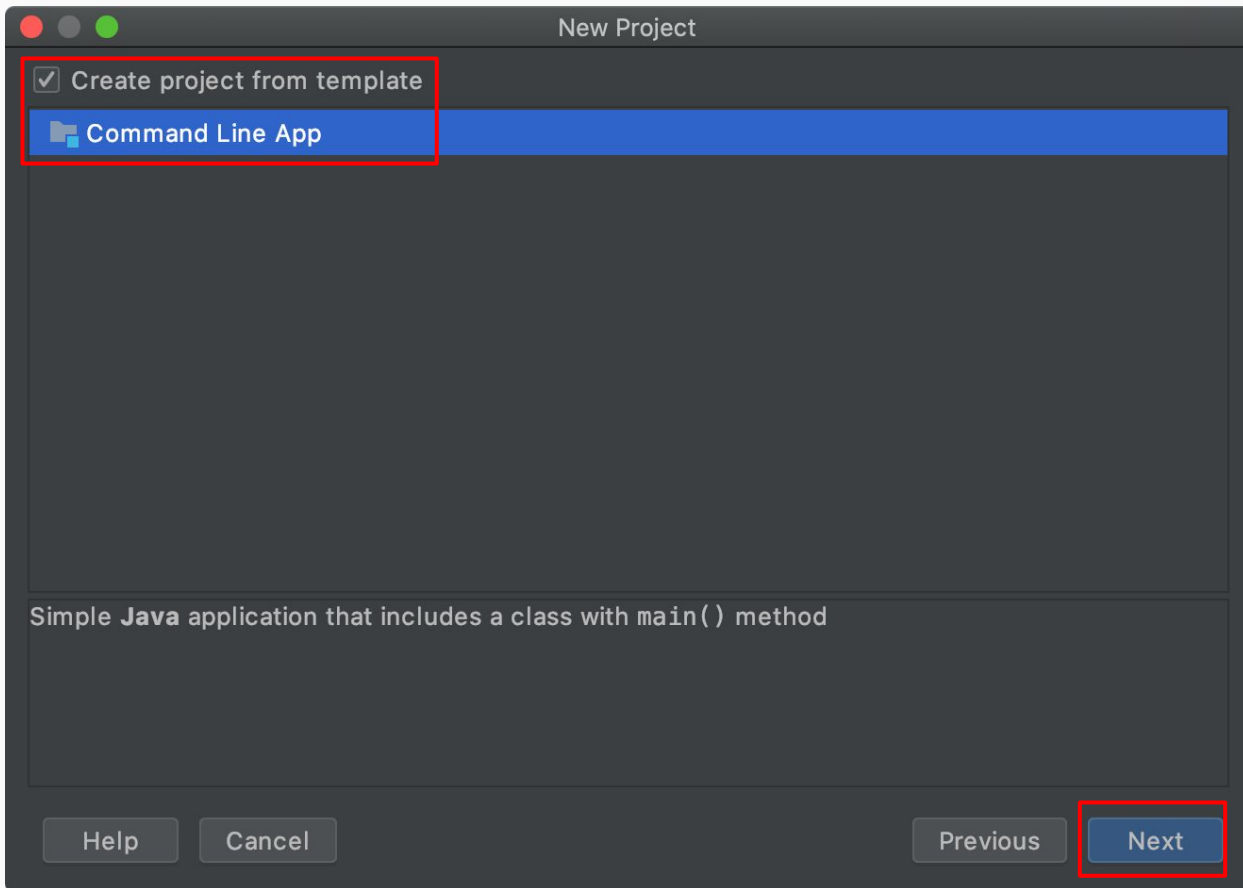


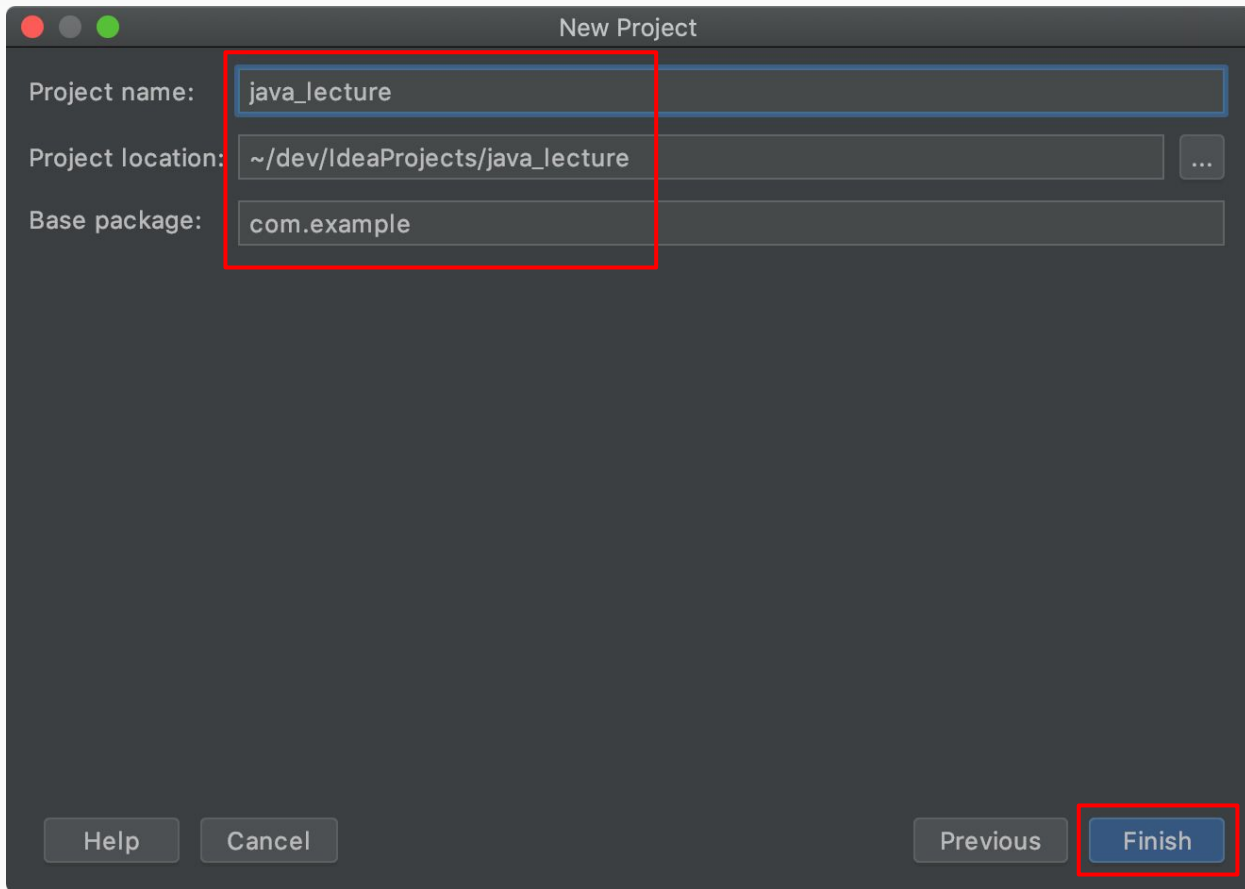
- AdoptOpenJdk 설치 (<https://adoptium.net/>) - 환경변수 안 잡아도 됨
- Formatter 변경 - <https://github.com/google/styleguide/blob/gh-pages/eclipse-java-google-style.xml>
- Tab 키 누르면 스페이스 4가 되도록 변경
- Text Size 변경
- tree 폰트 변경
- encoding UTF-8로 변경
- 1초마다 자동 저장 설정

Java 프로그램의 기본 구조









The image shows a 'New Project' dialog box from the IntelliJ IDEA IDE. The dialog has a dark gray background and a title bar with standard macOS window controls (red, yellow, green buttons). It contains three input fields: 'Project name:', 'Project location:', and 'Base package:'. The 'Project name' field contains 'java_lecture', the 'Project location' field contains '~/dev/IdeaProjects/java_lecture', and the 'Base package' field contains 'com.example'. A red rectangular box highlights the 'Project name' and 'Base package' fields. At the bottom of the dialog, there are four buttons: 'Help', 'Cancel', 'Previous', and 'Finish'. The 'Finish' button is highlighted with a red rectangular box.

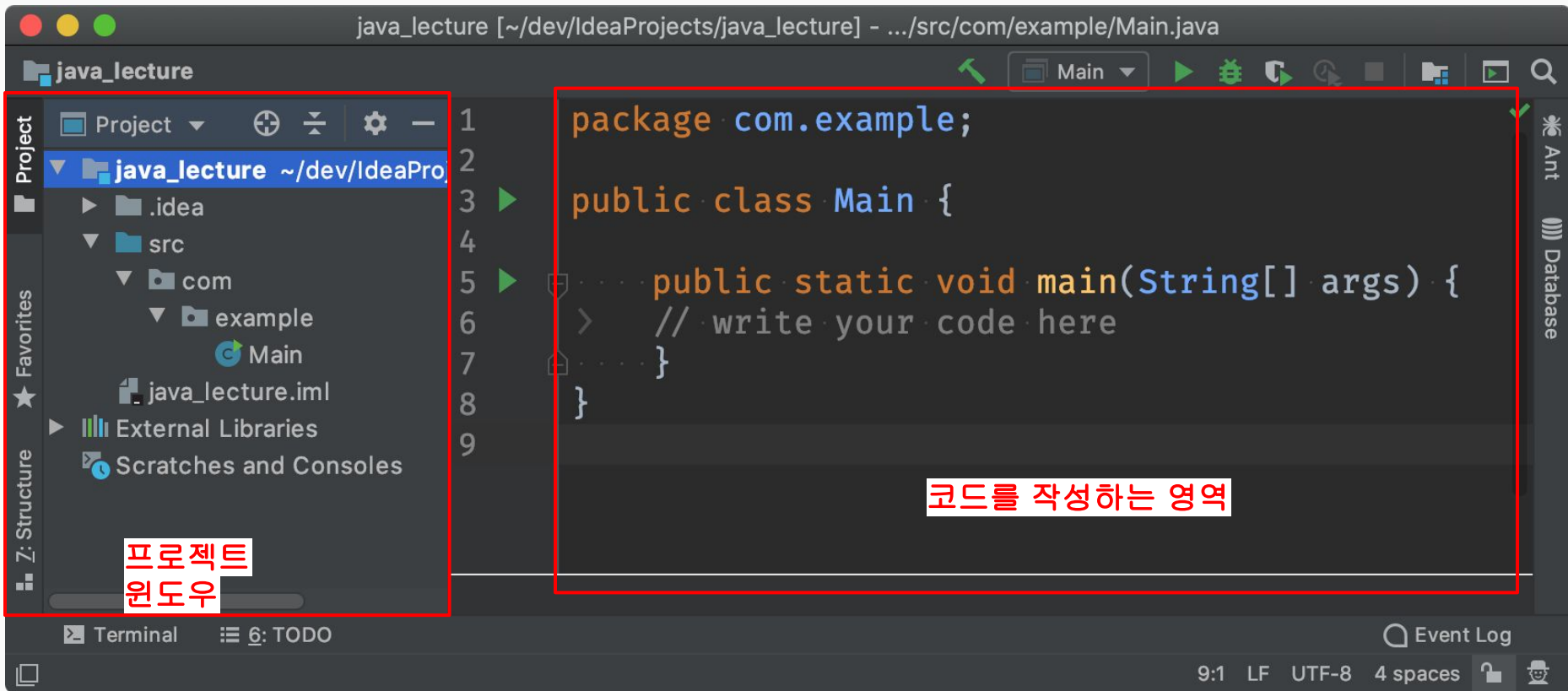
New Project

Project name: java_lecture

Project location: ~/dev/IdeaProjects/java_lecture ...

Base package: com.example

Help Cancel Previous Finish



프로그램의 골격

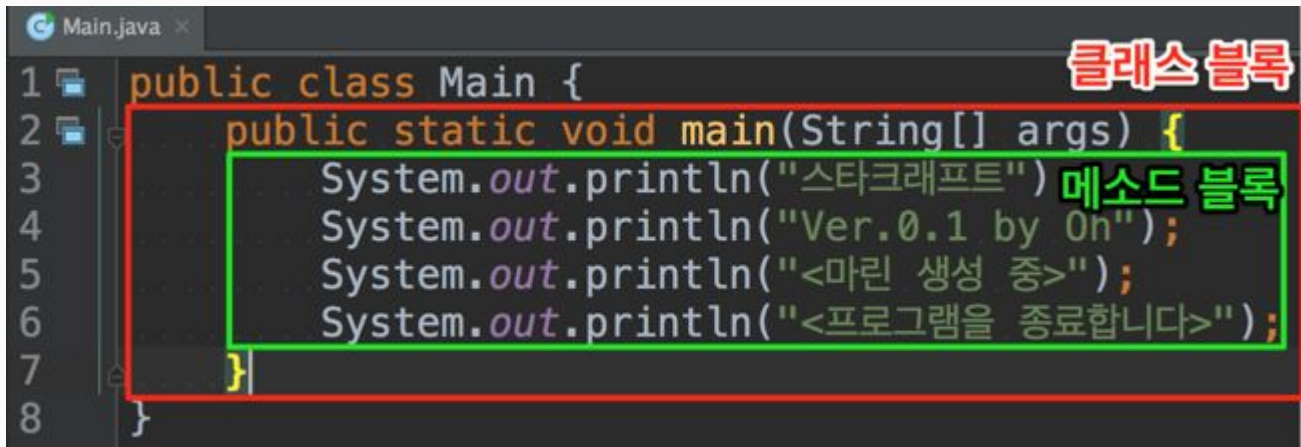
명령을 쓰는 부분은 “중앙 부분” 뿐

```
1 public class Main {
2     public static void main(String[] args) {
3         System.out.println("스타크래프트");
4         System.out.println("Ver.0.1 by Oh");
5         System.out.println("<마린 생성 중>");
6         System.out.println("<프로그램을 종료합니다>");
7     }
8 }
```

처리나 명령 부분

코드 블록 구조

외부 블록은 정해진 형태이므로 수정하지 않습니다.



The screenshot shows a code editor window titled 'Main.java'. The code is as follows:

```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("스타크래프트")  
4         System.out.println("Ver.0.1 by On");  
5         System.out.println("<마린 생성 중>");  
6         System.out.println("<프로그램을 종료합니다>");  
7     }  
8 }
```

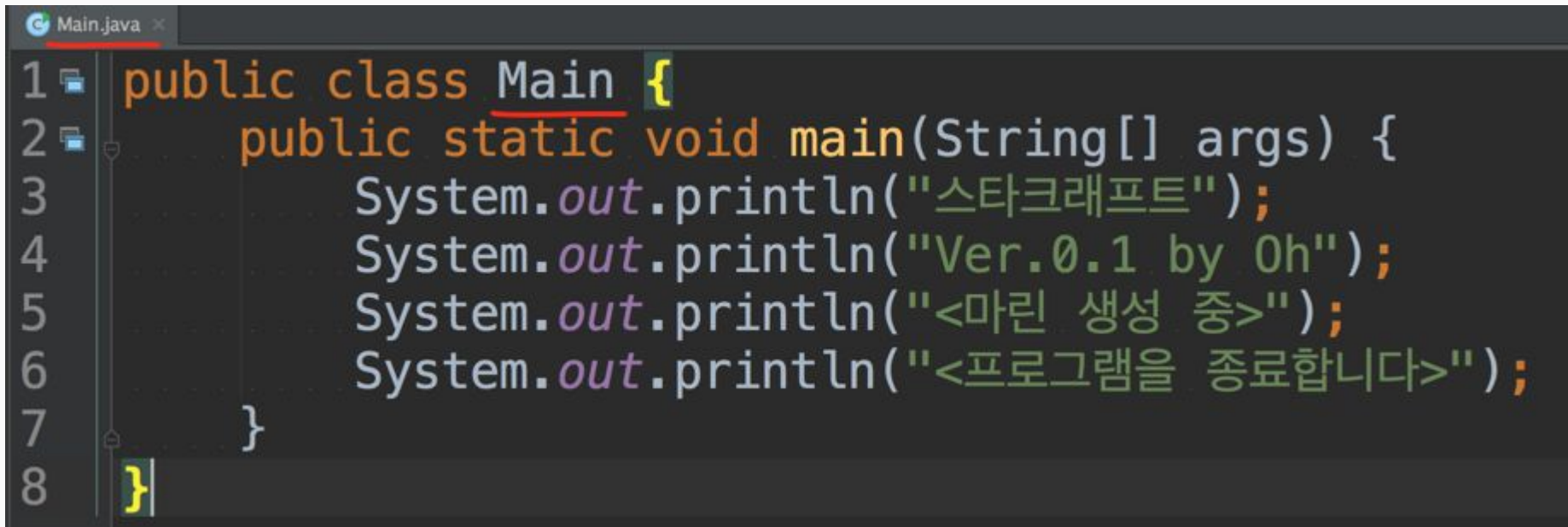
Annotations in the image:

- A red box highlights the entire class structure, with the label **클래스 블록** (Class Block) in red text to its right.
- A green box highlights the method body (lines 3-6), with the label **메소드 블록** (Method Block) in green text to its right.

프로그램의 이름 : 클래스명 (class name)

```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("스타크래프트");  
4         System.out.println("Ver.0.1 by Oh");  
5         System.out.println("<마린 생성 중>");  
6         System.out.println("<프로그램을 종료합니다>");  
7     }  
8 }
```

클래스라는 파일 단위로 코드를 작성하며, 클래스의 이름은 대문자 알파벳으로 시작하는 이름이 일반적 (Camel case)



```
1 public class Main {  
2     public static void main(String[] args) {  
3         System.out.println("스타크래프트");  
4         System.out.println("Ver.0.1 by Oh");  
5         System.out.println("<마린 생성 중>");  
6         System.out.println("<프로그램을 종료합니다>");  
7     }  
8 }
```

- 소스코드를 저장 할 때 파일명은 “클래스명.java” 으로 한다
- 클래스 명은 알파벳 대문자로 시작한다

Java 프로그램의 작성 방법

1. 어떤 프로그램을 만들고 싶은지 생각한다
2. 프로그램 명을 결정한다. (클래스명이 결정 됨)
3. “클래스명.java” 라는 이름으로 파일을 만든다.
4. 소스코드의 외측부분을 기술한다
5. 소스코드의 내측부분에 명령을 써 나간다

의식할 것 1 : 정확하게 기술할 것

- 영어 대, 소문자 구별
- 오 (o, O)와 영 (0), 엘(l) 과 일(1), 세미콜론(;)과 콜론(:), 마침표(.)와 콤마(,)
- 괄호 ((), {}, [])나 인용부호(' ', ") 의 종류
- `public static void main(String[] args)` 가 시작 지점

의식할 것 2: “위에서 아래로”가 아니라 “밖에서 안으로”

블록을 열면 반드시 닫는다

```
public class Main {  
}
```



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

Mission : 오류를 찾아보세요

의식할 것 3: 읽기 쉬운 코드를 기술 할 것

정상 동작 하더라도 “사람이 읽기 어려운 코드”나 “너무 복잡한 코드”는 수정이나 유지보수가 어려워 진다.

특히, 현장에서 동료와 함께 일 할 때 등
누가 봐도 알기 쉽게 기술 해야 함

```
public class Main {public static void main(String[] args) {System.out.println  
    ("자유로운 포맷이 가능");}}}
```

```
public class Main {  
    public static void main(String[] args) {  
        System.out.println("자유로운 포맷이 가능");  
    }  
}
```


1. 복수행 코멘트 : /* 주석 본문 (복수행으로 작성 가능) */
2. 단일 코멘트 : // 주석 본문 (한줄로 작성)

```
/*
    샘플 프로그램 Main
    개발자 : 오준석
    작성일 : 2015년 10월
*/
public class Main {
    // 여기부터 main 메소드
    public static void main(String[] args) {
        int age;    // 나이를 넣을 곳
        age = 20;
        System.out.println("나는 " + age + "살");
    }
}
```

```
public class Main {  
    public static void main(String[] args) {  
1. 변수 선언    int version;  
2. 계산        version = 1 + 5;  
3. 명령 실행    System.out.println("일기 소프트웨어");  
                System.out.println("Ver" + version);  
                System.out.println("아직 개발중입니다");  
                System.out.println("종료합니다");  
    }  
}
```

변수 선언

“새로운 변수를 준비 하자” 라고 컴퓨터에 지시하는 문장 작성

변수란, 데이터를 저장하기 위해 컴퓨터 내부에 준비하는 상자 같은 것으로서, 숫자나 문자열 등 프로그램이 사용할 여러가지 정보를 넣거나 빼는 것이 가능하다.

변수의 실체는 컴퓨터의 메모리이다.

변수에 값을 넣으면 실제로 메모리에 값이 써진다.

```
public class Main {  
    public static void main(String[] args) {  
        int age;  
        age = 30;  
        System.out.println(age);  
    }  
}
```

타입(데이터를 넣는 상자의 종류나 크기) 변수명(데이터를 넣는 상자의 이름);

int : 정수를 나타내는 타입. 소수나 문자열을 담을 수는 없음

변수의 이름

이름을 정하는 것은 프로그래머의 자유이지만,

보통은 알파벳, 숫자, 언더스코어(_) 등의 조합으로 만듭니다.

한글, 한자 등도 가능하나 비추천.

변수 명명 습관 1 : 금지 되어 있는 단어를 사용하면 안 됨

자바에서는 “변수명으로 사용하면 안 되는 단어”로 되어 있는 **예약어(keyword)**가 약 50개 존재 하고 있음.

int, void, public, static, class 등이 예약어로 사용 되고 있음.

변수 명명 습관 2:

이미 사용되고 있는 변수명을 다시 사용하면
안 됨

이미 사용하고 있는 변수명을 다시 변수명으로 선언하면 에러가 발생

2개의 변수를 구별 할 수가 없음

변수 명명 습관 3 : 대소문자 구별

대소문자 구별 해야 함.

예를 들어 **name** 과 **Name** 은 다른 변수임

변수 명명 습관 4: 소문자로 시작하는 알기쉬운 이름을 사용

변수명은 습관적으로 소문자로 시작하도록 한다

단 그 이후에 나오는 단어의 첫 문자는 대문자로 한다. (Camel case)

예를 들어 `myAge`

Java의 예약어 일람

abstract, assert, boolean, break. byte, case, catch, char, class, const, continue, default, do, double, else, enum, extends, final, finally, float, for, if, goto, implements, import, instanceof, int, interface, long, native, new, package, private, protected, public, return, short, static, strictfp, super, switch, synchronized, this, throw, throws, transient, try, void, volatile, while 등

https://en.wikipedia.org/wiki/List_of_Java_keywords

분류	형 이름	데이터 타입	변수선언의 예	이용빈도
정수	long	큰 정수	long worldPeople;	적음
	int	보통 정수	int salary;	많음
	short	작은 정수	short age;	적음
	byte	더 작은 정수	byte glasses;	적음
소수 (실수)	double	보통 소수	double pi;	보통
	float	정밀한 소수	float weight;	적음
참, 거짓	boolean	true 또는 false	boolean isMarried;	보통
문자	char	1개의 문자	char initial;	적음
문자열	String	여러개의 문자	String name;	많음

```
long worldPeople = 6900000000L;  
int salary = 152000;  
short age = 18;  
byte glasses = 2;
```

소비 메모리	상자의 크기	담을 수 있는 정수의 범위
8 byte	long	-9223372036854775808 ~ 9223372036854775808 (+-약 900경)
4 byte	int	-2147483648 ~ 2147483647 (+-약 21억)
2 byte	short	-32768 ~ 32767 (+-약 3.2만)
1 byte	byte	-128 ~ 127 (+-약 128)

소수를 담을 수 있는 2가지 타입 (double, float)

```
double height = 171.2;  
float weight = 67.5F;
```

통칭 부동소수점형 (floating point type) 이라고 함.

double이 float 보다 많은 메모리를 소비하지만, 보다 정밀한 계산을 할 수 있음.
특별한 사정이 없는 한 double 형을 사용합니다.

부동소수점방식에는 “**정말로 정밀한 계산을 할 수 없다**” 는 약점이 있음.
아주 작은 오차가 발생함.

오차가 허락되지 않는 계산에 **double** 이나 **float** 을 사용하는 것에 주의해야 함

```
boolean isMarried = true;  
boolean result = false;
```

boolean 형은 “YES 인지 NO 인지”

“진짜 인지 거짓인지”

“앞 인지 뒤 인지”

“성공 인지 실패 인지”

등의 **2자 택1**의 정보를 대입하기 위한 형 입니다.

긍정의 의미를 하는 **true**, 부정의 의미를 하는 **false** 를 대입 할 수 있습니다.

한 문자를 담을 수 있는 char 형, 문자열을 담을 수 있는 String 형

```
char gender = '남';  
String name = "오준석";
```

char 형은 영문, 한글 관계없이 “한 문자”를 대입할 수 있는 형입니다.

한편, String 형은 문자열(0 문자 이상의 문자의 집합)을 대입할 수 있는 형입니다.

“문자” 데이터를 기술 할 때에는 따옴표 (') 로 감쌉니다.

“문자열” 데이터를 기술 할 때에는 이중따옴표 (") 로 감쌉니다.

변수의 초기화

무엇을 위해 변수 선언을 하는가?

-> "변수에 데이터를 넣고 싶으니까"

```
int age;  
age = 22;
```

```
int age = 22;
```


변수의 이용

변수에는 다른 값을 몇 번이라도 대입 가능합니다.

변수에 값을 넣은 후, 다른 값을 다시 넣으면 어떻게 될까요?

```
int age = 20;  
System.out.println("나의 나이는 " + age);  
age = 31;  
System.out.println("아니, 사실은 " + age);
```

```
double tax = 1.1;  
int fax = 5;  
System.out.println("5만원에서 4만원으로 할인합니다");  
tax = 4;  
System.out.println("FAX의 새로운 가격(세금포함) ");  
System.out.println(fax * tax + "만원");
```

Mission : 위 코드를 실행 해 보고, 잘못된 부분을 찾아 봅시다.

값을 변경할 필요가 없는 수

final 키워드를 붙여 변수의 변경을 방지 할 수 있다

final 을 붙여 선언된 것은 **상수 (constant variable)** 라고 하고,
선언과 동시에 초기값을 대입해야 하고, 이 후에는 값을 변경 할 수 없다.

```
final double TAX = 1.1;
int fax = 5;
System.out.println("5만원에서 4만원으로 할인합니다");
TAX = 4;
System.out.println("FAX의 새로운 가격(세금포함) ");
System.out.println(fax * TAX + "만원");
```

개발과 실행의 흐름

- **Java**의 문법에 따라 소스 코드를 작성
- 소스 코드를 컴파일러로 컴파일하고 바이트 코드로 변환한다
- 인터프리터는 바이트코드를 기계어로 변환하면서 **CPU**에게 일을 시킨다

개발의 흐름과 기본 구조

- 소스 코드는 블록에 의한 이중구조를 가지고 있다
- 외측부분은 형식적인 기술을 하고, 내측에 코드를 기술한다
- 읽기 쉬운 소스를 쓰기 위해서는 코멘트와 들여쓰기를 활용한다

변수선언

- 변수는 “[타입] [변수명];” 으로 선언하여 사용한다
- 변수명은 기본적으로 자유지만, 암묵적인 규칙이 있다.
- 자바의 변수에는 대표적인 9개의 타입이 있어, 용도에 맞게 사용한다
- **final** 을 붙여서 선언된 변수의 값은 변경 불가능하다. 이를 상수라고 한다

연습문제 1-1

Java로 프로그램을 개발하기 위해, (A) 와 (B) 라는 소프트웨어가 필요하다. (A) 는 Java 문법으로 작성한 (C) 를 (D) 로 변환 해 줍니다. (B) 는 내부에 가지고 있는 (E) 을 사용하여 이것을 해석하고 기계어로 변환하여 CPU 가 실행합니다.

연습문제 1-2

화면에 다음과 같은 결과를 표시하는 소스 코드를 작성 해 주세요.

1. 소스 코드 안에 3을 변수 **a** 에, 5 를 변수 **b** 에 대입합니다.
2. 그 곱셈의 결과를 변수 **c** 에 대입합니다.
3. 변수 **a, b, c** 를 이용하여 다음과 같이 화면에 출력합니다.

출력 예)

가로 3, 세로 5의 직사각형의 면적은 15

연습문제 1-3

아래에 표시된 5개의 값을 담기 위한 적절한 타입(형) 을 생각 해, 선언과 동시에 “초기화” 하는 소스 코드를 작성합니다. (출력하지 않아도 됨)

또한, 변수명은 자유롭게 생각해도 되지만, **Java** 의 룰을 지켜 주세요.

2개 이상의 타입(형)이 생각될 경우에는, 어느 것을 사용해도 괜찮습니다.

1. true
2. '남'
3. 3.14
4. 314159265853979L
5. “항구를 공격! 적에게 15포인트의 데미지를 주었다”