

즐거운 자바

강경미(carami@nate.com), 김성박 (urstory@gmail.com)

JAVA언어의 특징과 JDK 설치

자바 프로그래밍을 공부하는 이유?

- 기업에서 서버 프로그래밍시 가장 많이 사용하는 언어

자바 언어의 특징

- 객체지향 언어
- 자바 언어는 느리지만, 버전업 되면서 다른 언어들의 장점들을 흡수하고 있다. (모던 자바)
 - 람다(Lambda) : 함수형 프로그래밍
 - Stream API : 람다 표현식과 메서드 참조 등의 기능과 결합해서 매우 복잡하고 어려운 데이터 처리 작업을 쉽게 조회하고 필터링하고 변환하고 처리할 수 있도록 한다.
 - 병렬 프로그래밍 : 여러개의 CPU코어에서 작업을 배분해서 동시에 작업을 수행한다.

자바 프로그램 작성과 실행

- JDK(Java Development Kit)이라는 프로그램을 다운로드하고 설치해야한다.
- 여러 종류의 JDK가 존재한다. openjdk, Oracle JDK, Azul Julu JDK, Amazon Corretto OpenJDK, Adoptium Temurin 등
- 이클립스 재단(The Eclipse Foundation)의 어댑티움(Adoptium) 프로젝트가 '이클립스 테무린(Eclipse Temurin) 자바 SE 바이너리'의 첫 번째 릴리즈를 출시했다. 이는 인텔 64비트 프로세서 기반 윈도우, 리눅스, 맥OS용 자바 SE 8, 자바 SE 11, 자바 SE 16의 최신 버전을 다루는 오픈JDK(OpenJDK)의 '프로덕션 레디(production-ready)' 빌드다.



















Git 설치

- Git 설치 후 다음의 내용을 설정한다.

```
git config --global user.name "이름"  
git config --global user.email 이메일  
git config --global core.autocrlf true
```

JDK 21 설치

- 초보자가 공부할 때는 JDK 8도 충분. 서비스 업체들의 경우 11이상을 사용하는 경우가 많다.
- JDK 21을 고려하는 경우
 - Java 17이 2023년 9월 출시.
 - 10부터는 6개월마다 출시. LTS(Long Term Support)버전은 3년마다 출시.
 - JDK8 LTS가 JDK11 LTS보다 유지보수 기간이 더 길다.
 - JDK 17 vs JDK 21 - JDK 21의 유지보수 기간이 더 길다.
- <https://adoptium.net/>

<p>21.0.2+13</p> <p>Temurin  </p> <p>January 19, 2024</p>	macOS	aarch64	<div>JDK - 192 MB Checksum  .pkg</div> <div>JDK - 191 MB Checksum  .tar.gz</div> <div>JRE - 41 MB Checksum  .pkg</div> <div>JRE - 41 MB Checksum  .tar.gz</div>
<p>21.0.2+13</p> <p>Temurin  </p> <p>January 19, 2024</p>	macOS	x64	<div>JDK - 194 MB Checksum  .pkg</div> <div>JDK - 194 MB Checksum  .tar.gz</div> <div>JRE - 42 MB Checksum  .pkg</div> <div>JRE - 42 MB Checksum  .tar.gz</div>
<p>21.0.2+13</p> <p>Temurin  </p> <p>January 20, 2024</p>	Windows	x64	<div>JDK - 179 MB Checksum  .msi</div> <div>JDK - 204 MB Checksum  .zip</div> <div>JRE - 34 MB Checksum  .msi</div> <div>JRE - 48 MB Checksum  .zip</div>

구형 intel 맥

JDK 21 설치

- 윈도우 사용자는 msi파일을, 맥 사용자는 pkg파일을 다운로드한다.
- msi파일 또는 pkg파일을 실행하여 설치한다.

JDK 21 설정/설치 - Mac사용자

- 터미널에서 `echo $0` 이라고 명령한다.
- zsh라고 나올 경우
`code ~/.zshrc` 명령을 수행한다.
- bash라고 나올 경우
`code ~/.bashrc` 명령을 수행한다.
- 파일을 열었으면 마지막 줄에 다음을 추가한다.

```
export JAVA_HOME=$(/usr/libexec/java_home -v 11)
export PATH=$PATH:$JAVA_HOME/bin
```

JDK 21 설정/설치 - Mac사용자

- 터미널을 종료하고 재시작한다. 다음과 같이 명령한다.

```
java --version  
javac -version
```

JDK 21 설치 - 윈도우 사용자

구글에서 "Windows JAVA_HOME PATH 환경변수 설정" 이라고 검색해보자.

<https://vmppo.tistory.com/6>

자바 프로그램 작성과 실행

- 터미널에서 특정 디렉토리로 이동한다. (소스가 저장될 디렉토리)
- `code Hello.java` 을 실행한 후 아래와 같이 저장한다.

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

자바 프로그램 작성과 실행

- java 컴파일러 javac 명령으로 hello.java를 컴파일 한다.

```
javac Hello.java
```

- 컴파일이 성공하면 오류메시지가 없이 Hello.class 파일이 생성된다.
- ls -la 명령으로 파일이 생성되었는지 확인한다.
- JVM(자바 가상 머신)으로 Hello.class 를 실행한다. java 명령이 JVM을 의미한다. 이 때 확장자는 입력하지 않는다.

```
java Hello
```

Hello

Hello.java 파일 분석하기

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


Hello.java의 3가지 중요한 부분

1. 클래스 선언

```
public class Hello{  
.....  
}
```

- public class로 정의된 Hello 클래스
- public class의 클래스 이름과 파일이름은 같아야 한다. (중요! 대소문자 구분함):
Hello.java

Hello.java의 3가지 중요한 부분

2. 메소드 선언

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

- 클래스는 필드(Field)와 메소드(Method)를 가질 수 있다.
- 프로그램이 실행하려면 반드시 가져야 하는 main메소드
- Java로 만든 프로그램이 실행되려면 위의 코드(code)를 가지고 있어야 한다. 프로그램 시작점이라고도 말한다.

Hello.java의 3가지 중요한 부분

3. System클래스의 out. out이 가지고 있는 println()메소드를 이용해 출력하기

```
System.out.println("Hello");
```

- System.out은 System이 가지고 있는 out이라는 의미이다.
- out.println은 out이 가지고 있는 println이라는 의미이다.
- println뒤에 괄호가 붙어 있는데, 이때 println메소드라고 말한다.
- out은 괄호가 붙지 않았는데, 이때 out필드라고 말한다.
- out이 가지고 있는 println메소드의 역할은 괄호안의 내용을 화면에 출력한다. 즉 "Hello"가 출력되게 된다. (이때 큰따옴표 안의 내용이 출력된다.), 큰따옴표까지 포함해서 문자열(String)이라고 말한다.

컴파일하기

- 컴파일을 하려면 반드시 javac라는 프로그램이 필요하다. javac는 자바 컴파일러 (Compiler)를 말한다.

```
javac Hello.java
```

- 터미널에서 위의 명령을 입력하면 Hello.java라는 파일을 읽어들이어서 컴파일을 하게 된다.
- 컴파일을 성공하면 Hello.class파일이 생성되고, 컴파일이 실패하면 오류메시지가 보여진다.
- Hello.class파일을 바이트(byte) 파일 이라고 말한다. Hello.java는 에디터로 열어보면, 사람이 알아들을 수 있는 말로 되어 있지만, Hello.class는 사람이 알아볼 수 없는 말로 되어 있다.

JVM으로 실행하기

- 자바로 작성된 프로그램이라는 것은 컴파일된 클래스, 즉 바이트 파일을 의미한다.
- 작성된 바이트 파일을 실행하려면 JVM(Java Virtual Machine)이 필요하다. 이 JVM 역할을 수행하는 것이 java 명령이다.
- hwp 파일을 읽어들이려면 아래한글 워드프로세서가 필요하고, psd 파일을 읽어들이려면 포토샵 프로그램이 필요한 것과 같은 원리이다.
- JVM은 바이트 코드를 읽어들이어 실행된다. 바이트 코드를 읽어들이어 실행하기 위해서는 다음과 같은 명령을 실행한다. 이때 확장자 이름은 입력하면 안된다.

```
java Hello
```

- java 프로그램은 Hello 클래스를 한 줄 읽고 해석 하고 실행하고, 한 줄 읽고 실행하고를 반복하면서 실행한다. 이렇게 한줄 한줄 읽어들이고 해석하면서 실행하는 방식을 인터프리터(interpreter)방식이라고 한다.

연습

- 본인의 이름, 성별, 이메일주소를 출력하는 클래스를 작성하고, 컴파일하고, 실행한다.

IntelliJ

IDE?

- IDE : 통합 개발 환경(統合開發環境, Integrated Development Environment, IDE)은 코딩, 디버그, 컴파일, 배포 등 프로그램 개발에 관련된 모든 작업을 하나의 프로그램 안에서 처리하는 환경을 제공하는 소프트웨어이다.
- 출처 : https://ko.wikipedia.org/wiki/통합_개발_환경
- 대표적인 Java언어를 위한 IDE : Eclipse, IntelliJ

IntelliJ

- JetBrains사에서 만든 IDE.
- 무료 버전인 Community버전과 유료 버전인 Ultimate버전으로 나뉜다.
- 자바 언어를 공부할 때는 둘 중 무엇이든 상관없다.
- 다음의 URL에서 다운로드 받는다. 본인의 운영체제에 맞는 버전을 선택해서 다운로드 하면 된다. 30일간 무료로 사용할 수 있는 Ultimate버전을 다운로드 받는다. 학생 일 경우 무료로 사용할 수 있다. 앞으로 웹 개발을 하려면 Ultimate버전이 유리하다.
<https://www.jetbrains.com/ko-kr/idea/download/#section=mac>

설치 (Mac)

- m1계열의 cpu 사용자는 Apple Silicon버전을 다운로드 하고, 그렇지 않은 사용자는 Intel버전을 다운로드 한다.

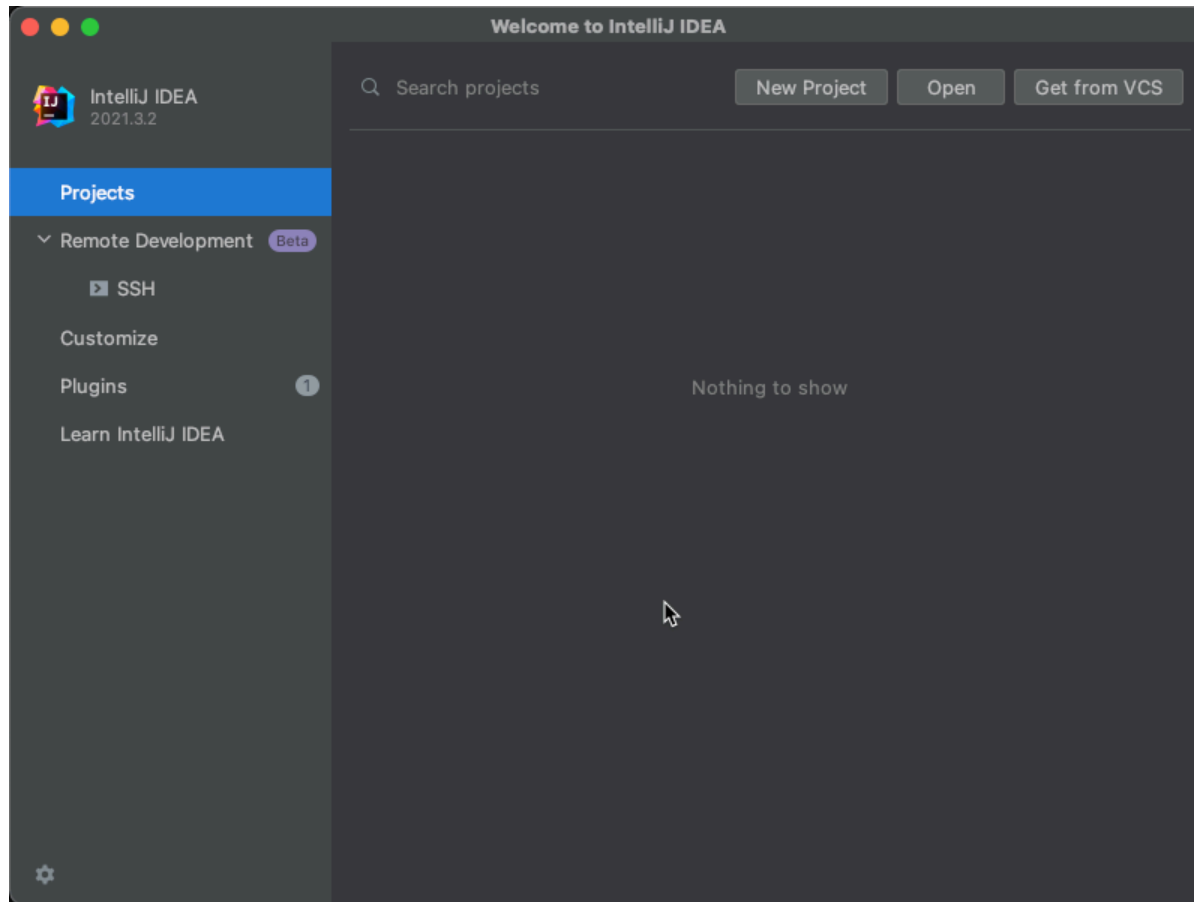


- 다운로드 파일을 클릭하면 아래와 같은 창이 열린다. 좌측의 아이콘을 우측에 드래그하면 설치가 완료된 것이다.



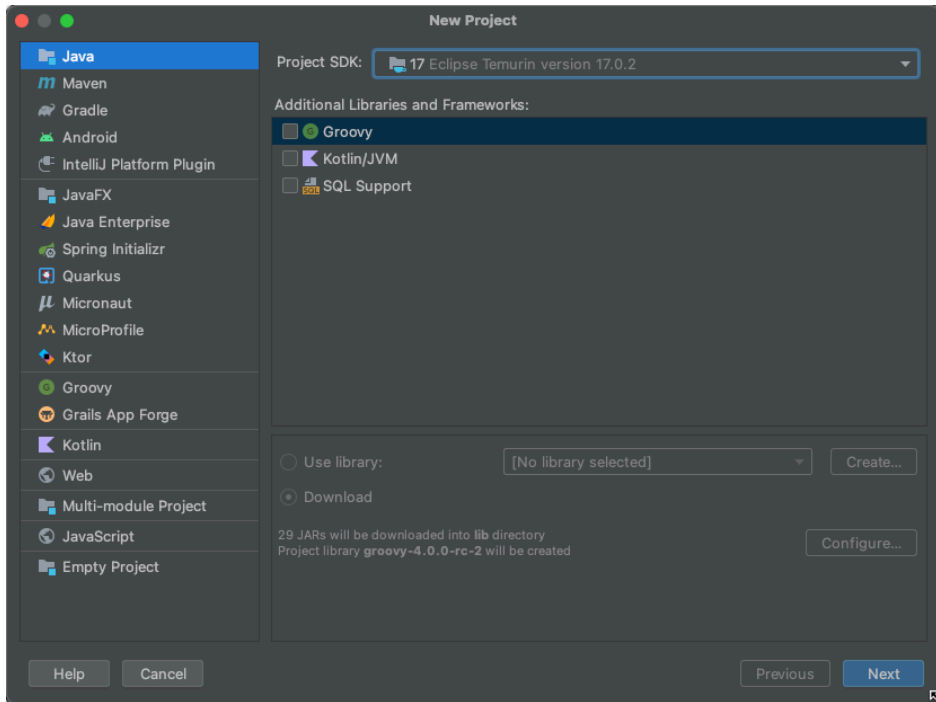
실행

- 어플리케이션 폴더에 있는 IntelliJ IDEA 아이콘을 클릭하여 실행한다. 실행하면, "Welcome to IntelliJ IDEA"라는 제목이 써진 창이 열리게 된다.



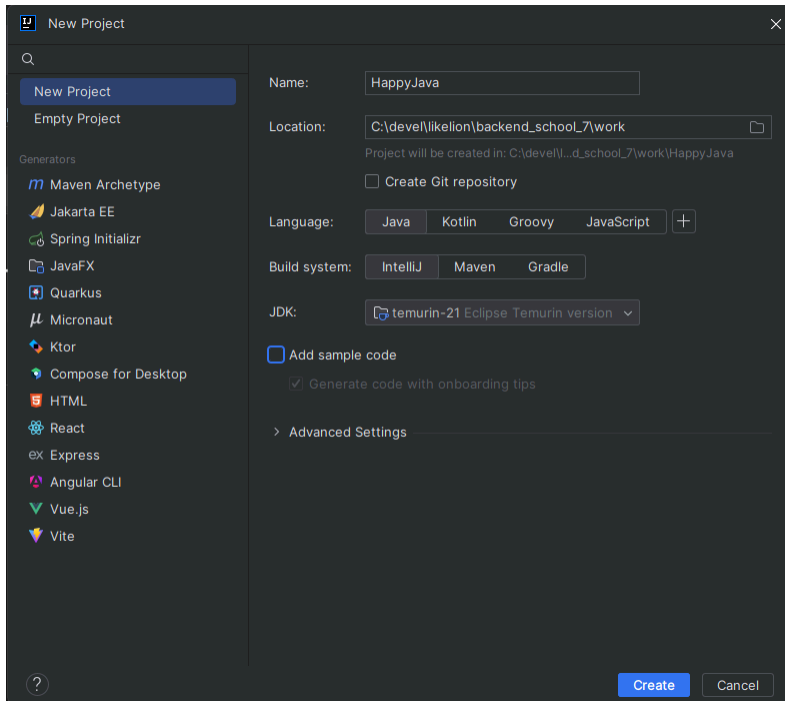
프로젝트 생성하기

- IDEA에서 프로그램 개발을 시작하려면 가장 먼저 해야할 일은 프로젝트를 생성하는 것이다. "New Project"버튼을 클릭한다.
- 라이브러리와 프레임워크를 선택하게 되어있는데 아무것도 선택하지 않고, 우측 하단의 "Next"버튼을 클릭한다.



프로젝트 생성하기

- 우리는 즐겁게 자바를 공부할 목적으로 프로젝트를 만드는 것이니 프로젝트 이름을 "HappyJava"라고 이름지을 것이다.
- 프로젝트 폴더가 생성될 폴더의 위치를 선택한다. 자바 프로젝트만 모아놓는 폴더를 만들어 놓고, 그 폴더안에 프로젝트를 생성하는것이 관리에 좋다.

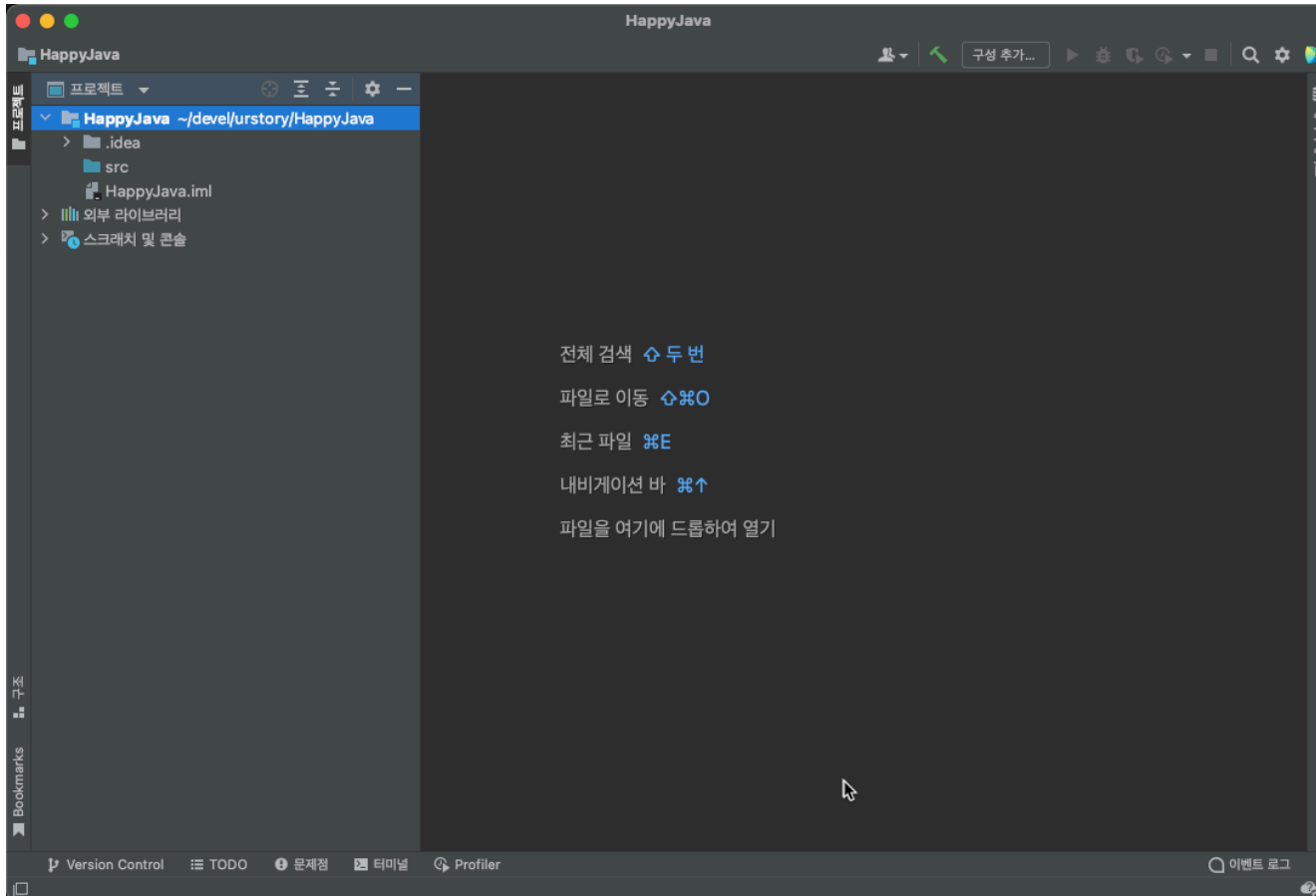


한글 메뉴 사용하기

- 프로젝트를 열면, 창이 열리고 우측 하단에 IntelliJ를 로컬라이즈화 하여 사용할 것이냐는 작은 창이 보인다. 확인을 누르면 IntelliJ가 다시 열리면서 메뉴등이 한글로 표시된다. 영어로 사용하길 원한다면 창을 그냥 닫도록 한다.

IntelliJ IDE 실행 화면

- IntelliJ 가 실행되면 아래와 같이 보여진다.



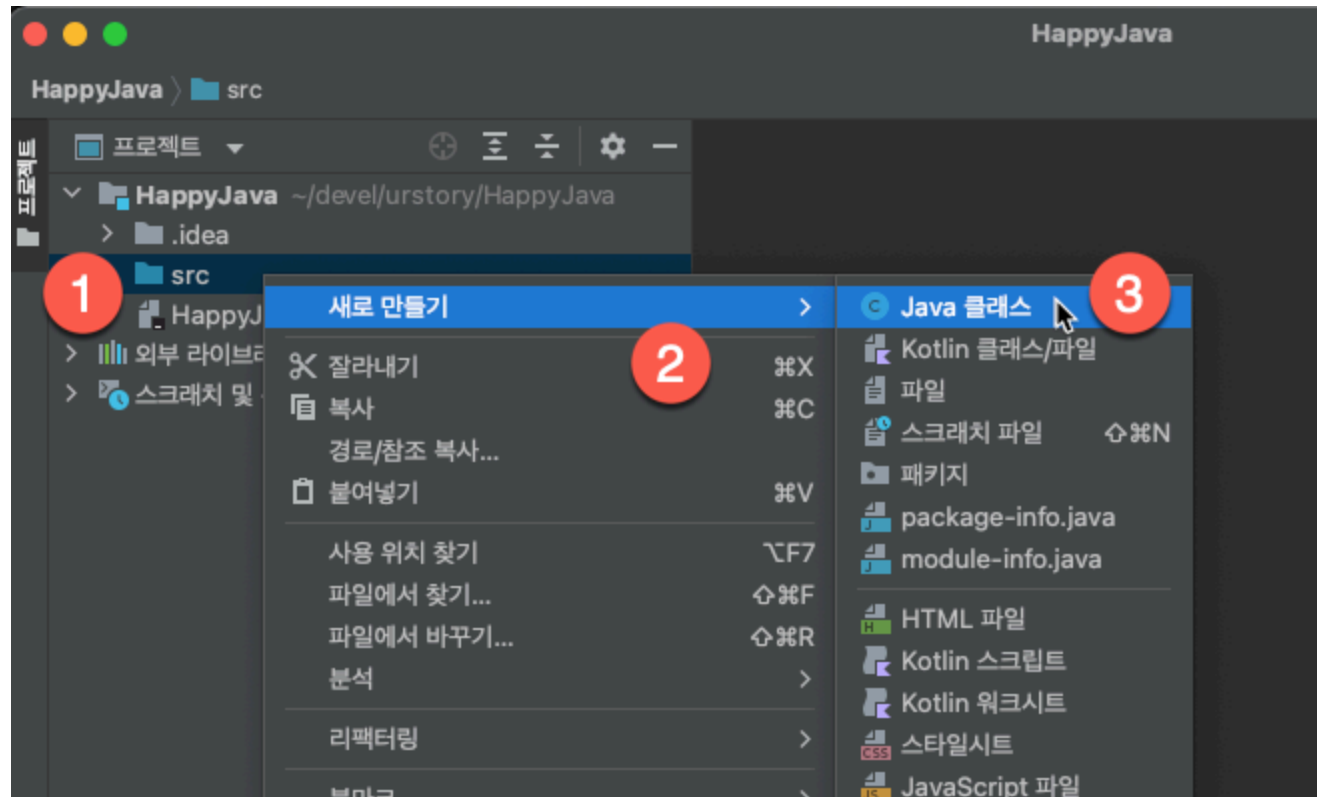
IntelliJ IDE에서 만든 프로젝트 구조

- 프로젝트 폴더 아래의 .idea폴더는 IntelliJ에서 프로젝트를 관리하기 위한 파일이다. 직접 수정하거나 삭제하면 안된다.
- HappyJava.iml파일은 IntelliJ의 설정 파일이다. 직접 수정하거나 삭제하면 안된다.
- 사용자는 src폴더에 Java소스코드를 작성하게 된다.

```
HappyJava
├── .idea
│   ├── .gitignore
│   ├── misc.xml
│   ├── modules.xml
│   └── workspace.xml
├── HappyJava.iml
└── src
```

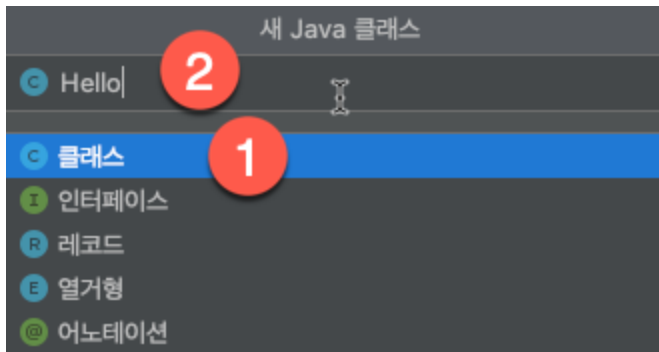

Hello.java 클래스를 작성한다.

1. src폴더를 선택하고 마우스 우측버튼을 클릭한다.
2. 새로 만들기를 클릭한다.
3. Java 클래스를 선택한다.



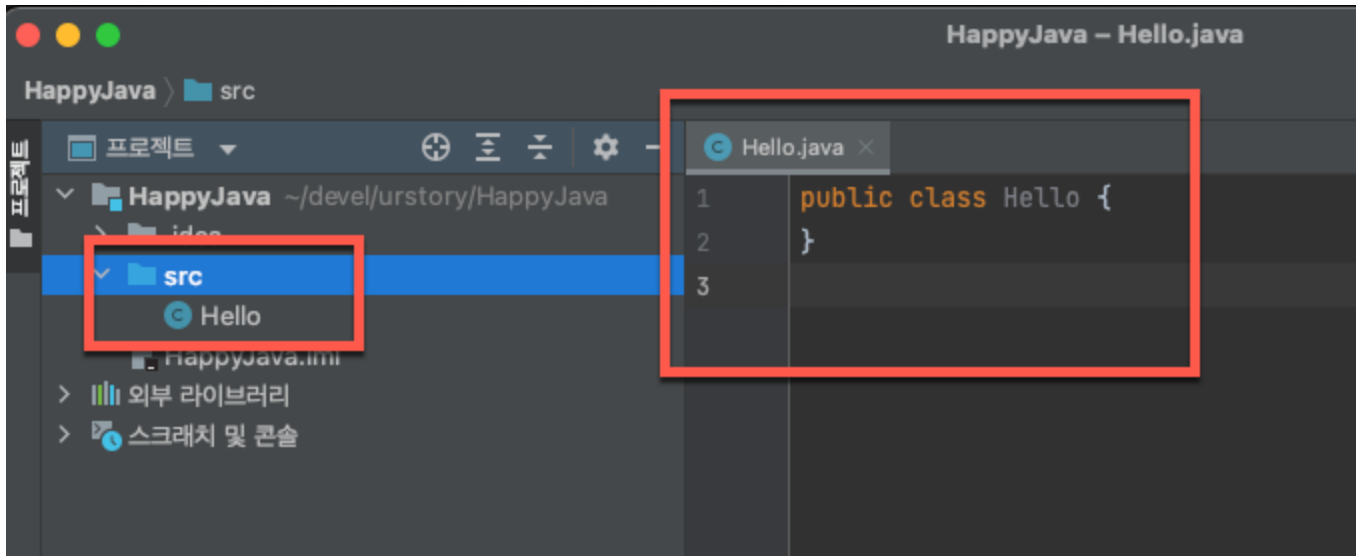
Hello.java 클래스를 작성한다.

1. 창이 열리면 클래스를 선택한다.
2. Hello 라고 클래스 이름을 입력한다. 이때 .java는 작성하지 않는다. 엔터를 입력한다.



Hello.java 클래스를 작성한다.

- src폴더에 Hello클래스가 작성된 것을 알 수 있다. 파일 탐색기로 가보면 Hello.java 파일이 생성된 것을 알 수 있다.

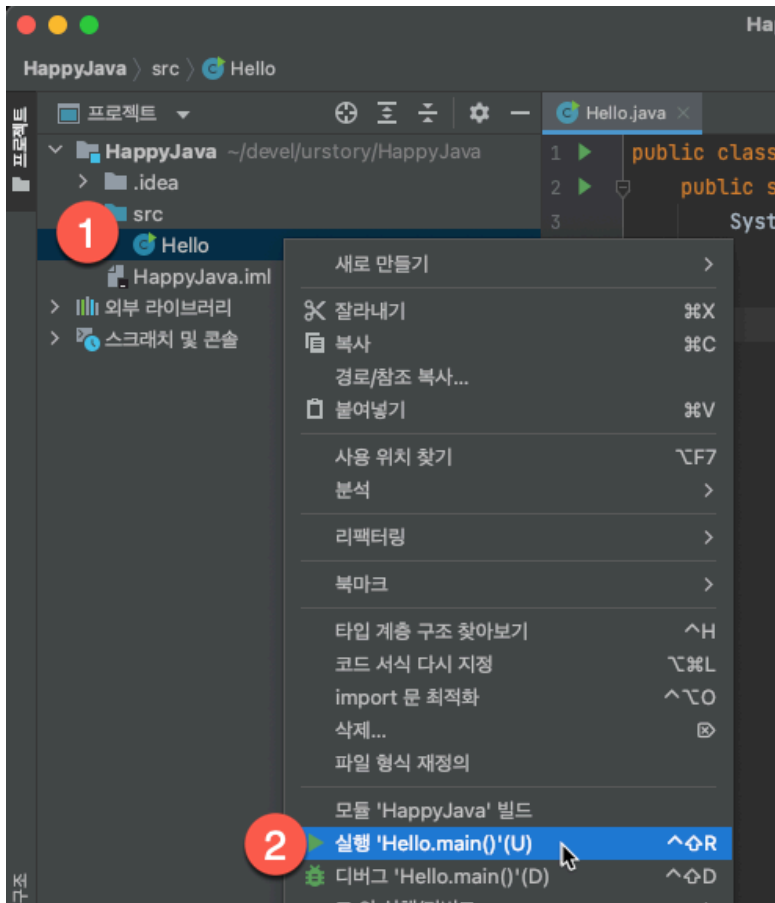


Hello.java 클래스를 다음과 같이 수정한다.

```
public class Hello {  
    public static void main(String[] args){  
        System.out.println("Hello IntelliJ IDEA");  
    }  
}
```

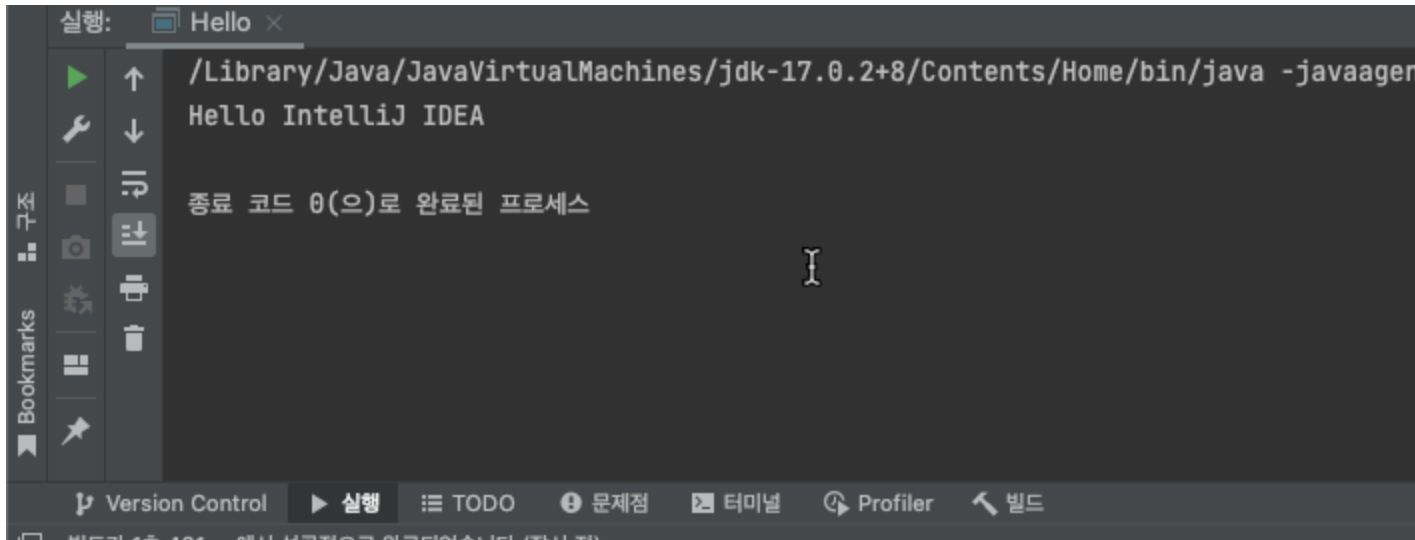
Hello 클래스 실행하기

1. Hello클래스를 선택하고, 우측버튼을 클릭한다.
2. 실행 'Hello.main()'을 선택한다.



Hello 클래스 실행하기

- IntelliJ IDEA의 화면 아래쪽에 Hello클래스를 java명령으로 실행한 결과가 출력된다.
- IntelliJ는 자동으로 컴파일하고, 실행을 하게 된다. 파일을 수정했을 때 컴파일되는 것은 아니다.



프로젝트 구조

- HappyJava 프로젝트 폴더 아래에 out폴더가 생성되고 그 아래에 또 몇가지 폴더가 생성된 후, Hello.class 파일이 생성된 것을 알 수 있다. IntelliJ가 Hello클래스를 실행하면 아래의 경로에 컴파일한 결과물을 생성하고, 해당 클래스를 실행하게 된다.

```
HappyJava
├── HappyJava.iml
├── out
│   └── production
│       ├── HappyJava
│       └── Hello.class
└── src
    └── Hello.java
```

연습

- 본인의 이름, 성별, 이메일주소를 출력하는 클래스를 작성하고, 컴파일하고, 실행한다.

감사합니다.