

Let's Start Java

01-1.

자바의 세계로 오신 여러분을 환영합니다.

프로그래밍의 시작을 위한 최소한의 준비

▶ JDK 다운로드

- www.oracle.com에서 JDK를 다운로드 한다.
- [그림 01-1] ~ [그림 01-4]의 과정 참조

▶ JDK의 설치

- JDK 다운로드 이후 설치: [그림 01-5] ~ [그림 01-7]의 과정 참조
- 자바 문서 확인 및 다운로드: [그림 01-8] ~ [그림 01-13]의 과정 참조

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정

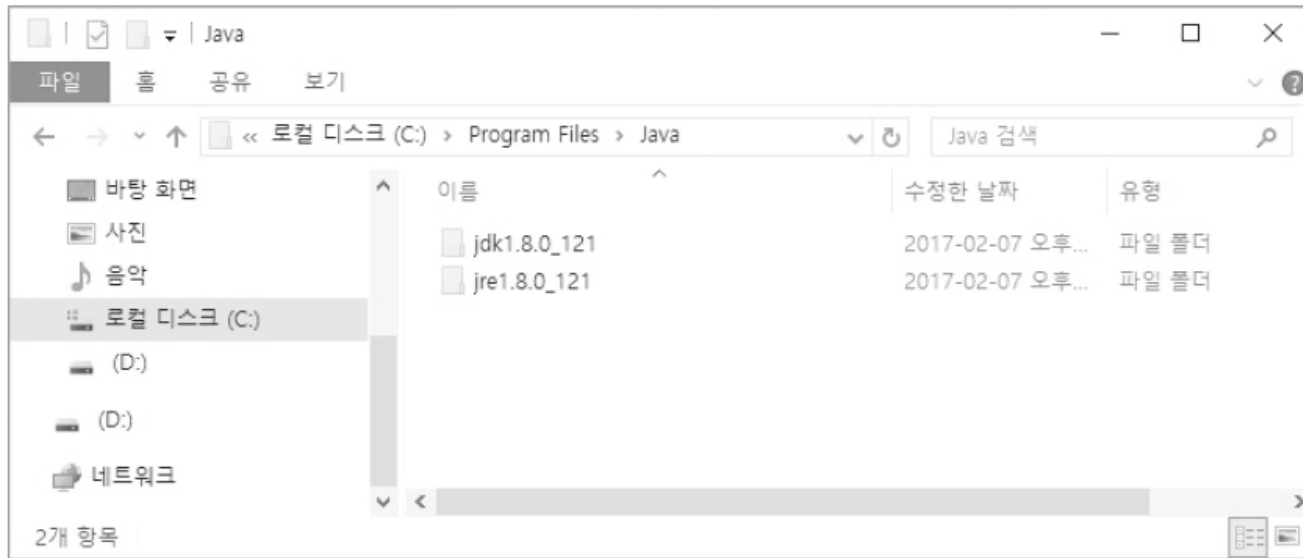
▶ 환경 변수 설정

- [그림 01-14] ~ [그림 01-22]의 과정 참조
- 환경 변수란? 환경 변수 Path란?

▶ 자바의 기본 도구

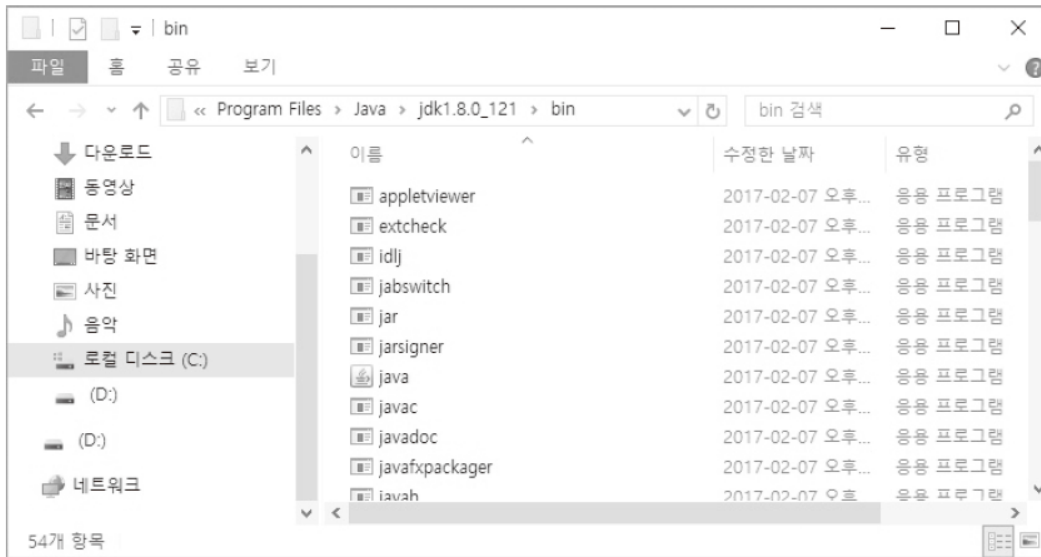
- javac.exe 자바 컴파일러 (Java Compiler)
- java.exe 자바 런처 (Java Launcher)

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



[그림 01-14: 자바 설치 경로]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



[그림 01-15: 자바 개발에 필요한 여러가지 도구들]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



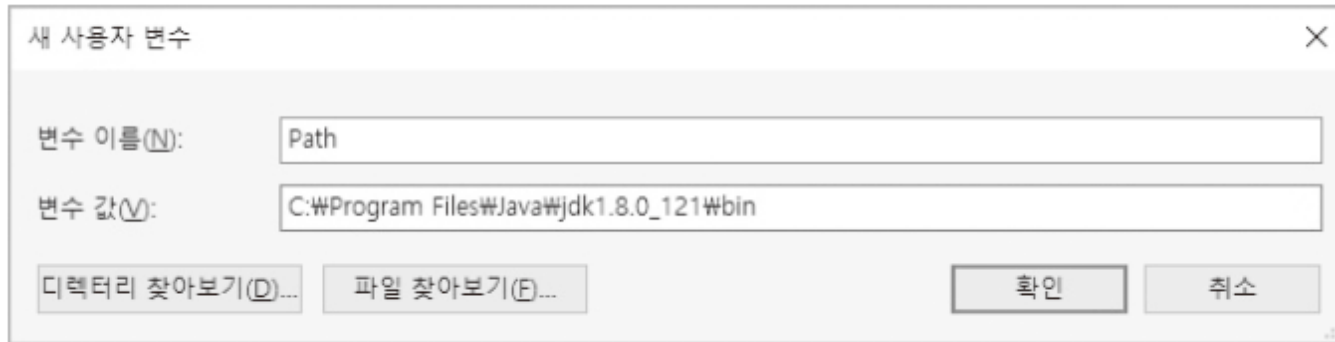
[그림 01-16: 시스템 속성 창]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



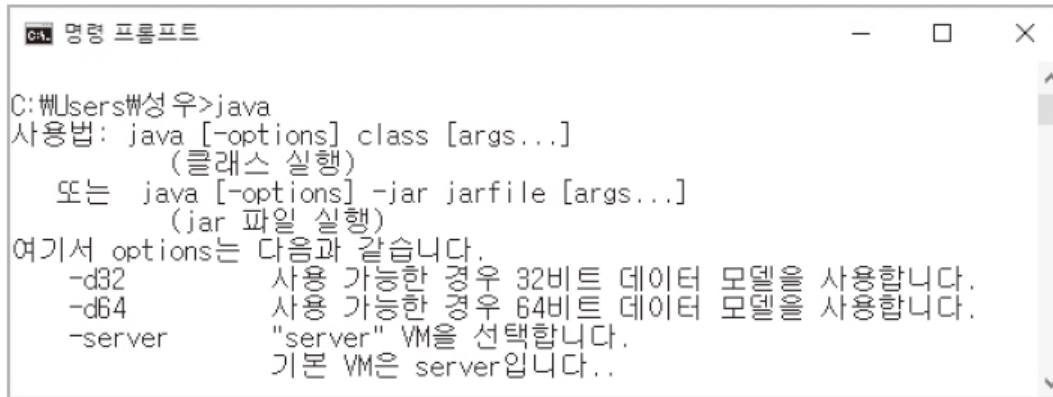
[그림 01-18: 환경 변수 설정 창]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



[그림 01-19: 환경 변수의 등록]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



```
C:\Users\성우>java
사용법: java [-options] class [args...]
        (클래스 실행)
또는 java [-options] -jar jarfile [args...]
        (jar 파일 실행)
여기서 options는 다음과 같습니다.
-d32      사용 가능한 경우 32비트 데이터 모델을 사용합니다.
-d64      사용 가능한 경우 64비트 데이터 모델을 사용합니다.
-server   "server" VM을 선택합니다.
          기본 VM은 server입니다..
```

[그림 01-21: Java.exe의 실행 결과]

설치 이후에 해야 할 추가적인 설정: 환경 변수의 설정



```
C:\Users\성우>javac
Usage: javac <options> <source files>
where possible options include:
  -g               Generate all debugging info
  -g:none          Generate no debugging info
  -g:{lines,vars,source}  Generate only some debugging info
  -nowarn          Generate no warnings
  -verbose         Output messages about what the compiler is
  doing
  -deprecation     Output source locations where deprecated A
  PIs are used
```

[그림 01-22: javac.exe의 실행 결과]

첫 번째 자바 프로그램의 작성과 실행

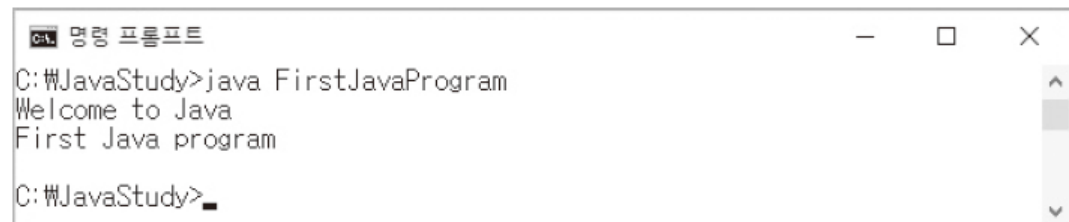
◆ FirstJavaProgram.java

```
1. class FirstJavaProgram
2. {
3.     public static void main(String[] args)
4.     {
5.         System.out.println("Welcome to Java");
6.         System.out.println("First Java program");
7.     }
8. }
```



A Windows Command Prompt window titled "명령 프롬프트" (Command Prompt). The command prompt shows the directory "C:\JavaStudy" and the command "javac FirstJavaProgram.java" being executed. The output is empty, indicating successful compilation.

```
C:\JavaStudy>javac FirstJavaProgram.java
C:\JavaStudy>_
```



A Windows Command Prompt window titled "명령 프롬프트" (Command Prompt). The command prompt shows the directory "C:\JavaStudy" and the command "java FirstJavaProgram" being executed. The output shows the two lines of text printed by the program: "Welcome to Java" and "First Java program".

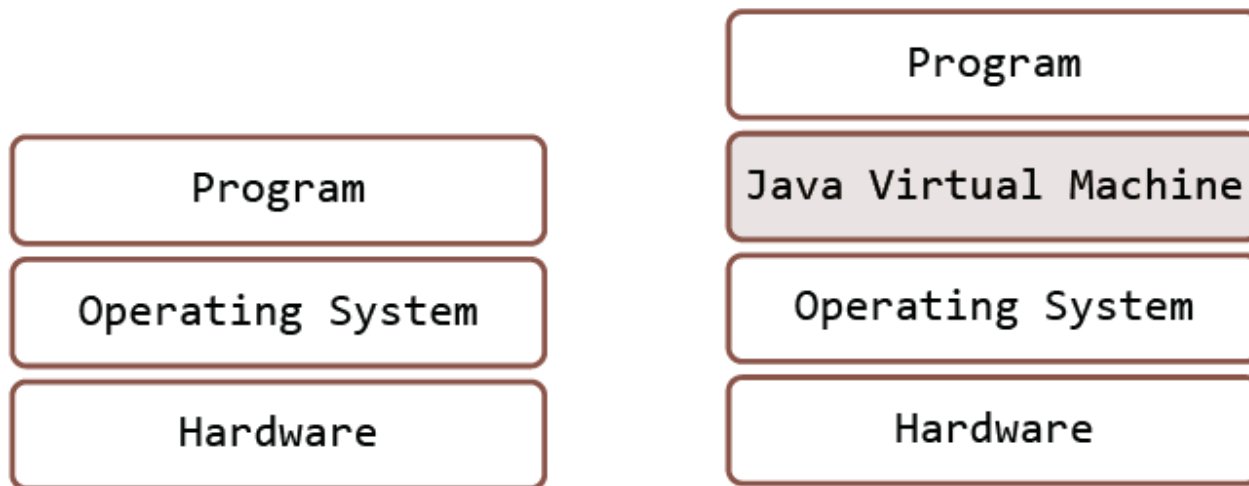
```
C:\JavaStudy>java FirstJavaProgram
Welcome to Java
First Java program
C:\JavaStudy>_
```

01-2.

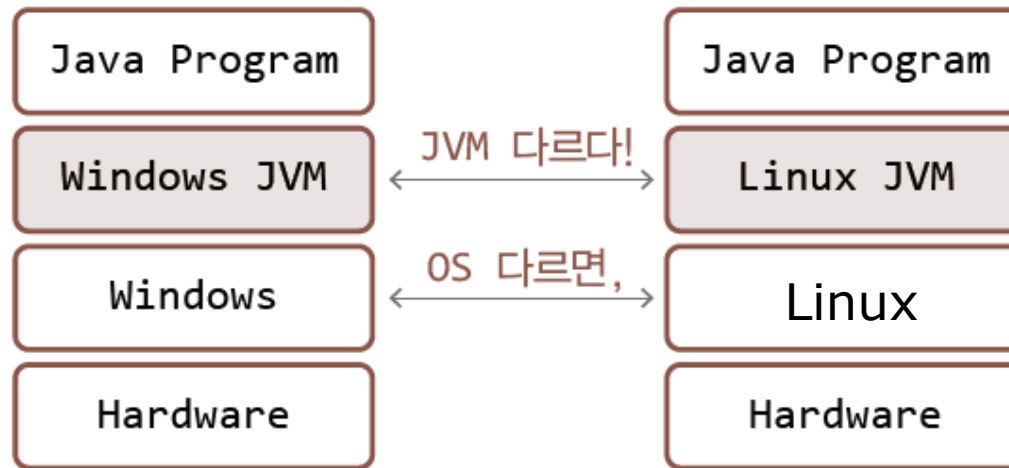
자바 프로그램과

실행의 원리에 대한 이해

일반적인 프로그램과 자바 프로그램의 차이

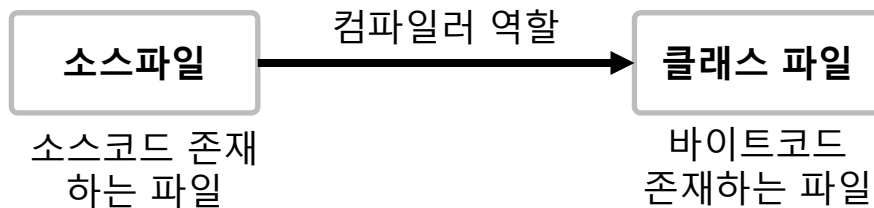


운영체제에 따른 자바 가상머신의 차이



자바 컴파일러와 자바 바이트코드

▶ 자바 컴파일러 (javac.exe)



▶ 자바 런처 (java.exe)

- 자바 프로그램과 자바 가상머신을 처음 구동하는 소프트웨어
- 클래스 파일을 대상으로 구동을 시작한다.

01-3.

첫 번째 자바 프로그램의 관찰과 응용

프로그램의 골격과 구성

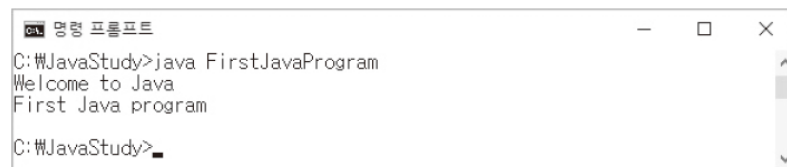
클래스

```
class FirstJavaProgram
{
    메소드
    {
        public static void main(String[] args)
        {
            System.out.println("Welcome to Java");
            System.out.println("First Java program");
        }
    }
}
```

클래스 이름

메소드 이름

- 중괄호를 이용해서 클래스와 메소드의 영역을 구분
- 문장의 끝에는 세미콜론을 붙여서 문장의 끝 표시



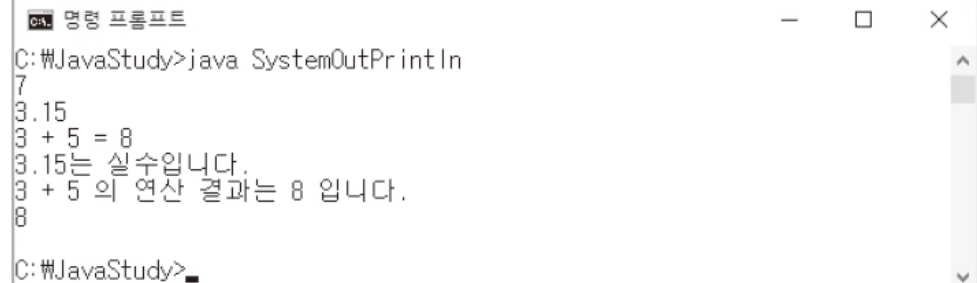
```
C:\JavaStudy>java FirstJavaProgram
Welcome to Java
First Java program
C:\JavaStudy>
```

- 프로그램 실행 시 main 메소드 안 문장들 순차적 실행
- System.out.println의 괄호 안에 출력 내용 큰따옴표로 묶어서 표시
- System.out.println 실행 이후 자동 개행

System.out.println에 대한 다양한 활용

◆ SystemOutPrintIn.java

```
1. class SystemOutPrintIn
2. {
3.     public static void main(String[] args)
4.     {
5.         System.out.println(7);
6.         System.out.println(3.15);
7.         System.out.println("3 + 5 = " + 8);
8.         System.out.println(3.15 + "는 실수입니다.");
9.         System.out.println("3 + 5" + "의 연산 결과는 8입니다.");
10.        System.out.println(3 + 5);
11.    }
12. }
```



C:\명령 프롬프트

C:\JavaStudy>java SystemOutPrintIn

7
3.15
3 + 5 = 8
3.15는 실수입니다.
3 + 5의 연산 결과는 8 입니다.
8

C:\JavaStudy>_

01-4.

들여쓰기와 컴파일의 대상에서 제외되는 주석!

블록 단위 주석

◆ BlockComment.java

```
1.  /*
2.   파일이름: BlockComment.java
3.   작 성 자: 홍길동
4.   작 성 일: 2019년 7월 25일
5.   목 적: System.out.println 메소드의 기능 테스트   주석 처리 영역
6.  */
7.
8.  class BlockComment
9.  {
10.     public static void main(String[] args)
11.     {   주석 처리 영역
12.         /* 다음은 단순한 정수의 출력 */
13.         System.out.println(7);
14.
15.         System.out.println(3.15);
16.         System.out.println("3 + 5 = " + 8);
17.         System.out.println(3.15 + "는 실수입니다.");
18.         System.out.println("3 + 5" + "의 연산 결과는 8입니다.");
19.         주석 처리 영역
20.         /* 다음은 덧셈 결과의 출력 */
21.         System.out.println(3 + 5);
22.     }
23. }
```

블록 단위 주석의 다른 사례

◆ BlockComment2.java

```
1.  /*
2.   * 파일이름: BlockComment2Java
3.   * 작 성 자: 홍길동
4.   * 작 성 일: 2019년 7월 25일
5.   * 목   적: System.out.println 메소드의 기능 테스트
6.   */
7.
8.  class BlockComment2
9.  {
```

행 단위 주석

◆ LineComment.java

```
1.  // 파일이름: LineComment.java
2.  // 작 성 자: 홍길동
3.  // 작 성 일: 2019년 7월 25일
4.  // 목   적: System.out.println 메소드의 기능 테스트
5.
6.  class SystemOutPrintln
7.  {
8.      public static void main(String[] args)
9.      {
10.         System.out.println(7);    // 다음은 단순한 정수의 출력
11.
12.         System.out.println(3.15);
13.         System.out.println("3 + 5 = " + 8);
14.         System.out.println(3.15 + "는 실수입니다.");
15.         System.out.println("3 + 5" + " 의 연산 결과는 8입니다.");
16.
17.         System.out.println(3+5);   // 덧셈 결과의 출력
18.     }
19. }
```

들여 쓰기

```
1. class SystemOutPrintln
2. {
3.     // 4칸 정도 오른쪽으로 들여쓰기 되었다.
4.     public static void main(String[] args)
5.     {
6.         // main 메소드 내에서 4칸 정도 오른쪽으로 들여쓰기 되었다.
7.         System.out.println(7);
8.         ....
9.     }
10. }
```


자바에서 권고 및 추천되는 방식

```
1. class SystemOutPrintln {  
2.     public static void main(String[] args) {  
3.         System.out.println(7);  
4.         ....  
5.     }  
6. }
```

자바에서 권고 및 추천되는
방식

```
1. class SystemOutPrintln  
2. {  
3.     public static void main(String[] args)  
4.     {  
5.         System.out.println(7);  
6.         ....  
7.     }  
8. }
```