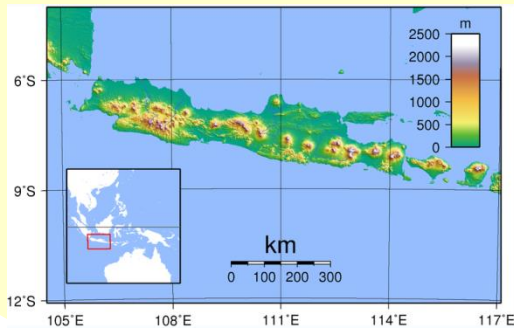


자바 소개

Instructor: Park, JoonSeok (pjs50@pusan.ac.kr)

자바란?

자바 섬



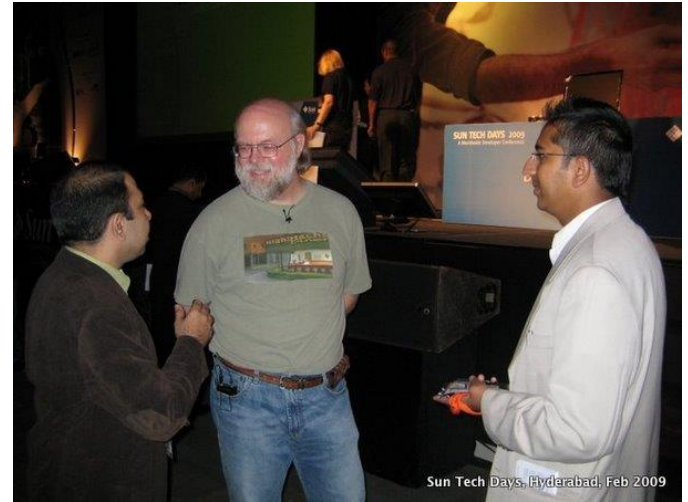
자바에서 만들어진 커피



프로그래밍 언어

```
/*  
 * Outputs "Hello, world!" and then exits  
 */  
  
public class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello, world!");  
    }  
}
```

자바란?



James Gosling



자바란?

- ▶ 1991년에 Sun에서는 제임스 고슬링(James Gosling)를 비롯한 Green 연구팀에서는 가정용 전자 제품에 사용할 수 있는 작은 컴퓨터 언어를 설계
- ▶ 처음에 C++를 사용하여 운영 체제를 만들려고 시도하였는데 C++의 복잡도로 인하여 실패
- ▶ Green 프로젝트를 위한 더 나은 언어를 직접 만들게 되는데 이것이 바로 자바.
- ▶ Green 프로젝트는 Time Warner의 주문형 비디오 시스템을 개발하다가 Time Warner가 경쟁사인 실리콘 그래픽스사를 선택하는 바람에 결국 실패
- ▶ 1993년, 그래픽 기반의 월드 와이드 웹(world wide web)이 발표되고 자바의 개발자들은 곧 이러한 웹 기반의 응용 프로그램에는 자바와 같은 기계 독립적인 언어가 이상적이라는 것을 발견



자바의 발전

」



Java 1.0

- 1996년
- 211개의 클래스
- 속도는 느림
- 애플릿이 가장 주목받음



Java 1.1

- 1997년
- 477개의 클래스
- 내부클래스 개념 도입
- 속도가 약간 빨라짐
- **SWING GUI** 추가



Java 1.2-1.4 (Java 2)

- 1998-2004년
- 2000여개의 클래스
- 3가지 버전 존재 (**ME, SE, EE**)
- 웹과 모바일 기반의 엔터프라이즈 프로그래밍 언어로서 부각



Java 1.5-1.6

- 2004-2006년
- 3200-3700개의 클래스
- 언어 자체에도 변경을 가함: 제네릭, **for-each**, 오토박싱, 열거형

자바의 특징

- ▶ 단순하지만 강력하다
 - ▶ 꼭 필요로 하는 기능만을 포함시키고 복잡하고 많이 쓰이지 않는 기능은 삭제
 - ▶ 포인터 연산, 연산자 오버로딩, 다중 상속 등의 복잡한 기능을 삭제
 - ▶ 자동 메모리 관리 기능, 멀티 스레드, 방대한 라이브러리 제공
- ▶ 객체 지향적이다.
 - ▶ 객체 지향은 프로그램을 설계하는 방법론
 - ▶ 기본 데이터 타입을 제외한 거의 모든 것이 객체로 표현
- ▶ 분산 환경 지원
 - ▶ 네트워크상에서 동작되는 것을 기본으로 설계
 - ▶ 쉽게 네트워크 관련 프로그램을 개발



-
- ▶ 견고하다
 - ▶ 오류를 만들 수 있는 원인들을 제거
 - ▶ 안전하다.
 - ▶ 바이러스, 파일의 삭제나 수정, 데이터 파괴 작업이나 컴퓨터 오류 연산 등을 방지하면서 실행되도록 설계되었다.
 - ▶ 컴퓨터 구조에 독립적이다.
 - ▶ 컴퓨터 구조에 독립적인 바이트 코드로 번역
 - ▶ 이러한 바이트 코드 특성 때문에 인터넷에 연결된 서로 다른 기종의 컴퓨터에서도 자바는 실행될 수 있다.



▶ 멀티스레딩 지원

- ▶ 자바는 언어 수준에서 멀티스레딩(multithreading)을 지원
- ▶ 멀티스레딩이란 많은 작업을 동시에 실행

▶ 동적이다(Dynamic).

- ▶ 라이브러리들은 실행 파일에 영향을 끼치지 않고 자유롭게 새로운 기능들을 추가할 수 있다.
- ▶ 자바는 실행되기 직전에 라이브러리를 동적으로 링크하므로 실행할 때 변경된 라이브러리가 자동적으로 참조된다.



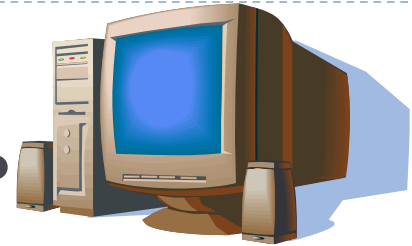
자바 플랫폼

- ▶ 자바로 기술된 프로그램을 개발 및 실행 할 수 있는 소프트웨어 모임
- ▶ 실행환경(JVM)과 개발환경(자바 API)를 제공



자바 플랫폼의 종류

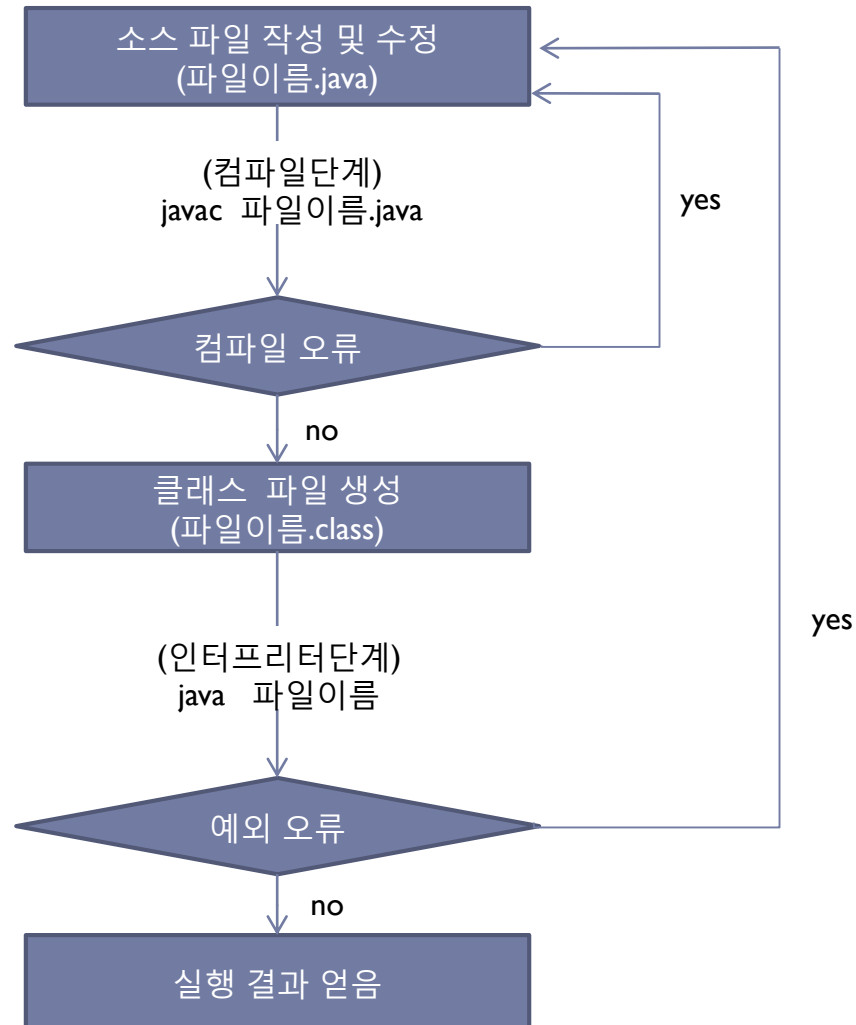
- Java SE(Standard Edition)
- Java EE(Enterprise Edition)
- Java ME(Micro Edition)

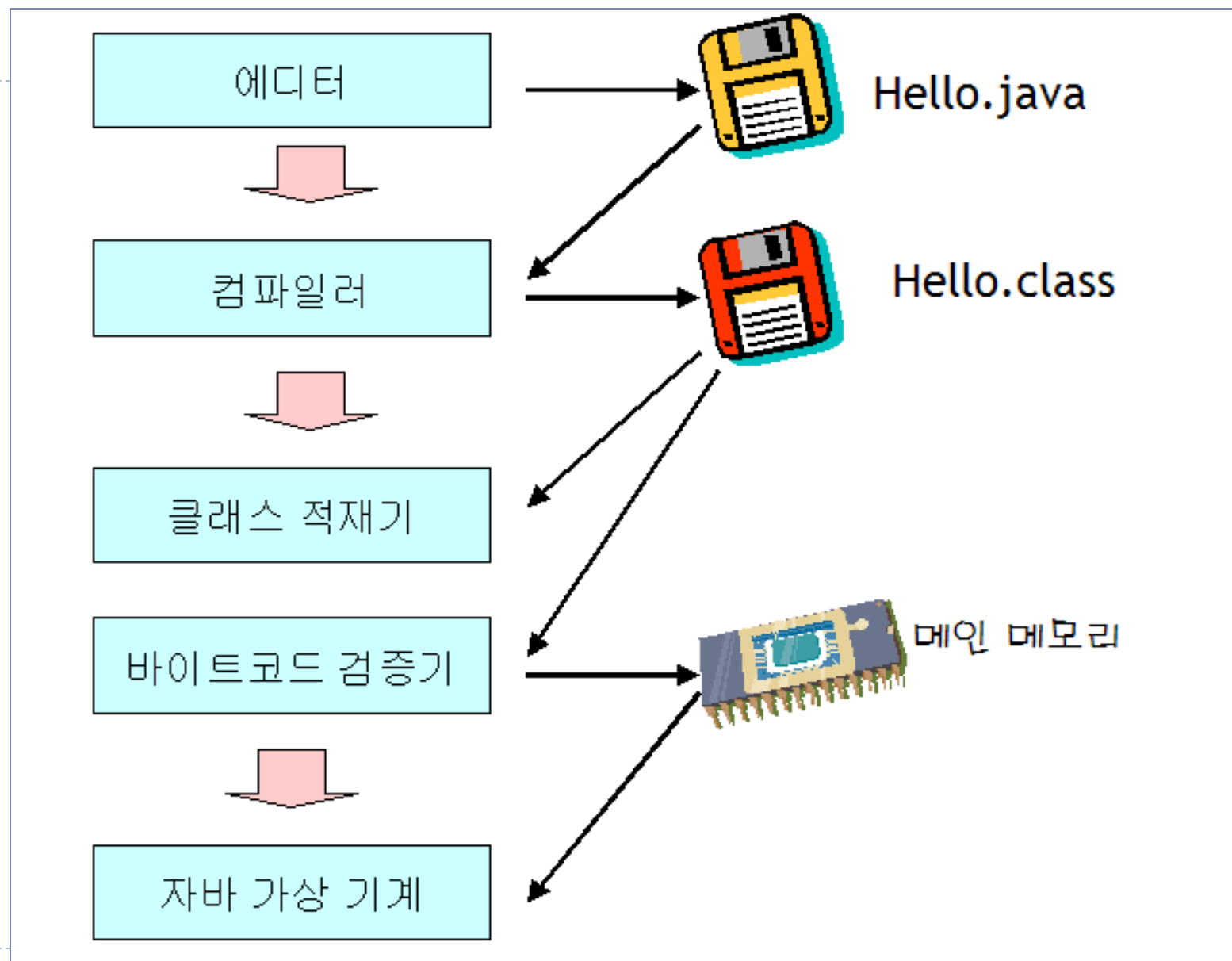


-
- ▶ Java SE는 데스크탑과 서버에서 자바 애플리케이션을 개발하고 실행할 수 있게 해주며 임베디드 환경(embedded environment)과 실시간 환경(real-Time environments)도 지원
 - ▶ Java EE는 기업용 애플리케이션을 개발하는 데 필요한 여러 가지 도구 및 라이브러리들을 모아 놓은 것
 - ▶ Java ME는 핸드폰, PDA, TV 셉톱박스, 프린터와 같은 모바일 기기나 다른 임베디드 장치들에서 실행되는 애플리케이션을 위한 강인하고 유연한 환경을 제공
-



자바 프로그램 개발 단계





자바 프로그램 개발 단계

- ▶ 소스 파일의 생성
 - ▶ 에디터를 사용
 - ▶ 소스 파일은 .java 확장자
 - ▶ 메모장, 이클립스, 넷빈, Jbuilder, Visual J++ 등 ..



▶ 컴파일

- ▶ 컴파일러로 컴파일
- ▶ 컴파일러는 자바 소스 코드를 바이트 코드로 변환
- ▶ 바이트 코드는 확장자가 .class로 끝나는 파일에 저장

▶ 클래스 적재

- ▶ 바이트 코드 파일을 메모리로 적재



▶ 바이트 코드 검증

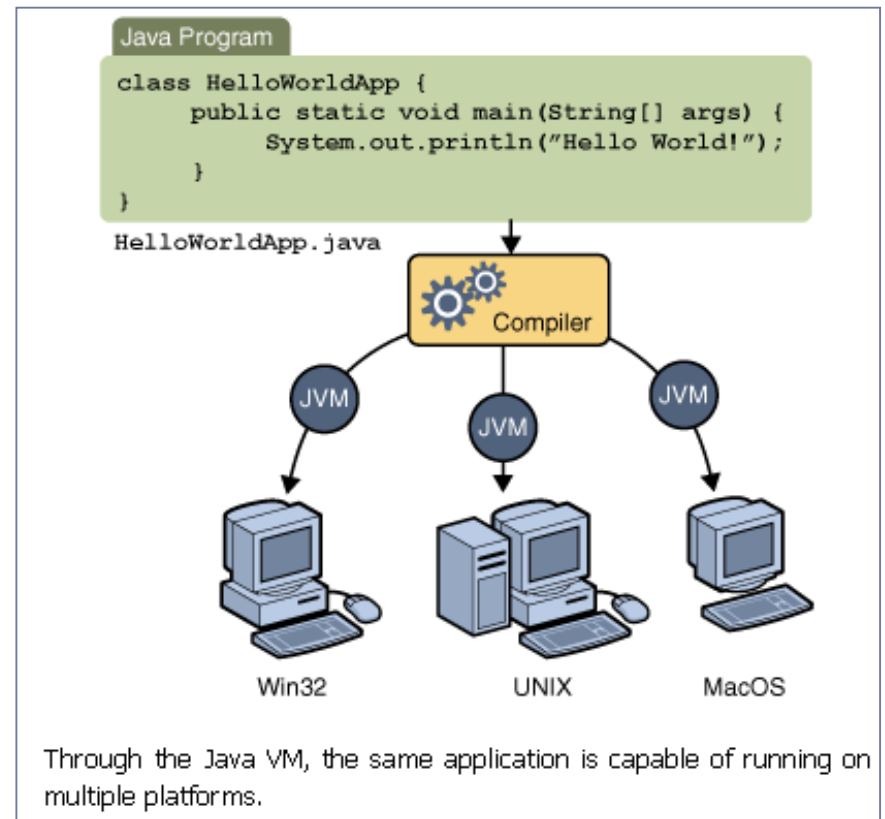
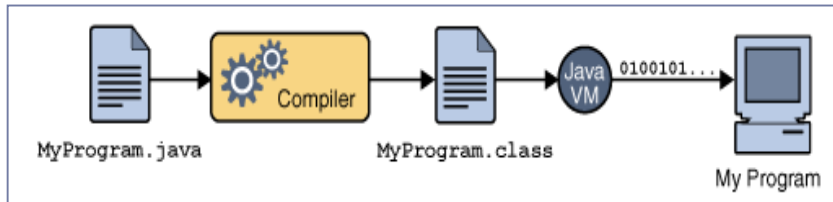
- ▶ 바이트 코드들이 이상이 없으며 자바의 보안 규칙을 위배하지 않는지를 검사

▶ 실행

- ▶ 자바 가상 기계가 바이트 코드를 실행

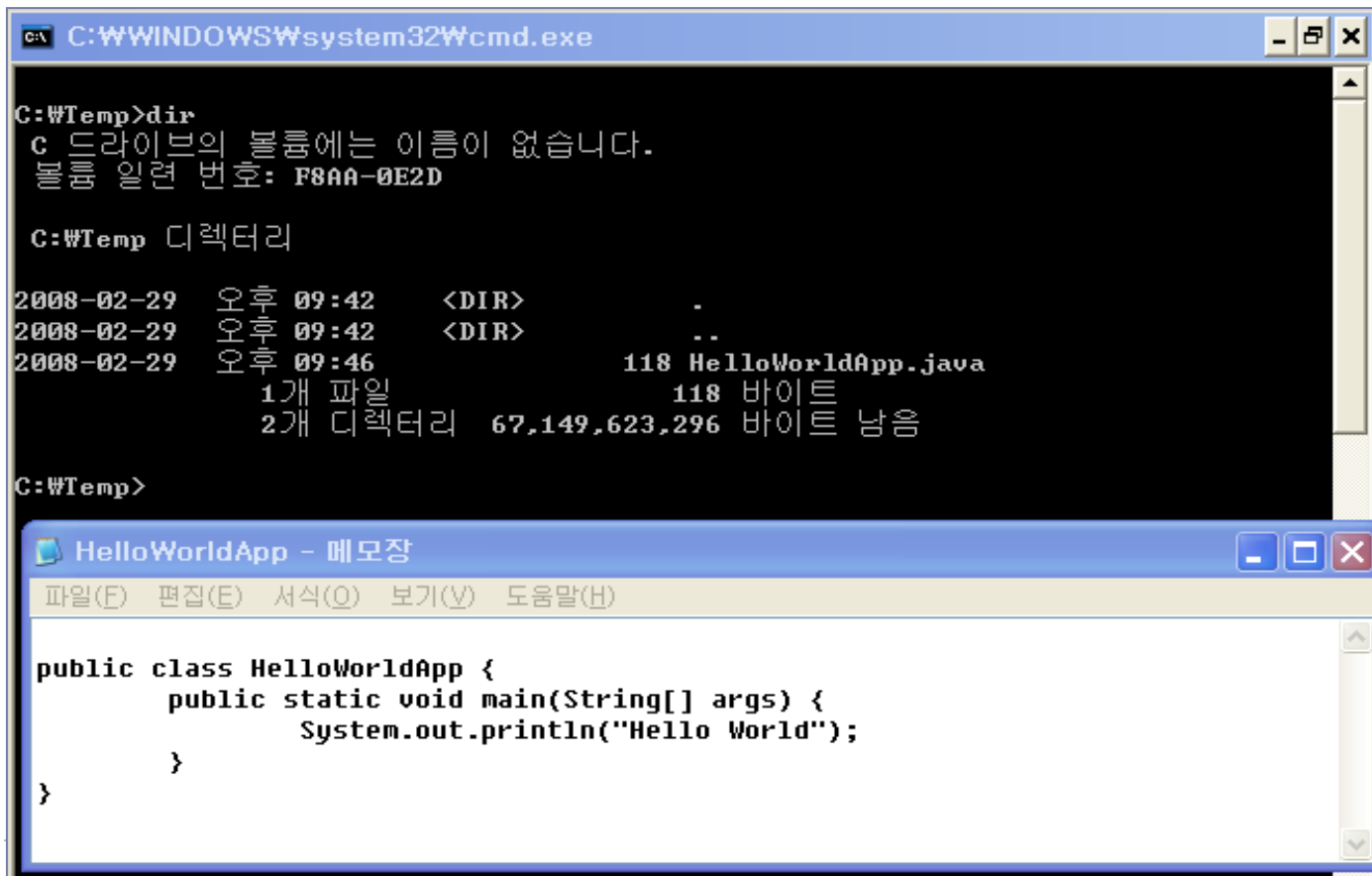


자바의 실행 모델



자바 프로그램 작성 예

- ▶ 프로그램 코드 작성
 - ▶ 메모장, 에디터, Eclipse...



The screenshot shows two overlapping windows. The top window is a command prompt titled "C:\WINDOWS\system32\cmd.exe". It displays the output of the 'dir' command in the C:\Temp directory, showing a file named 'HelloWorldApp.java' with a size of 118 bytes. The bottom window is a Notepad application titled "HelloWorldApp - 메모장". It contains the following Java code:

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

▶ 컴파일

C:\WINDOWS\system32\cmd.exe

C:\Temp>dir

C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: F8AA-0E2D

C:\Temp 디렉터리

2008-02-29	오후 09:42	<DIR>	.
2008-02-29	오후 09:42	<DIR>	..
2008-02-29	오후 09:46		118 HelloWorldApp.java
	1개 파일		118 바이트
	2개 디렉터리	67,149,623,296	바이트 남음

C:\Temp>javac HelloWorldApp.java

C:\Temp>dir

C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: F8AA-0E2D

C:\Temp 디렉터리

2008-02-29	오후 09:47	<DIR>	.
2008-02-29	오후 09:47	<DIR>	..
2008-02-29	오후 09:47		431 HelloWorldApp.class
2008-02-29	오후 09:46		118 HelloWorldApp.java
	2개 파일		549 바이트
	2개 디렉터리	67,149,582,336	바이트 남음

C:\Temp>

▶ 실행

```
C:\WINDOWS\system32\cmd.exe

C:\WTemp>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: F8AA-0E2D

C:\WTemp 디렉터리

2008-02-29 오후 10:17 <DIR> .
2008-02-29 오후 10:17 <DIR> ..
2008-02-29 오후 09:51          431 HelloWorldApp.class
2008-02-29 오후 09:54          123 HelloWorldApp.java
                2개 파일              554 바이트
                2개 디렉터리 67,149,946,880 바이트 남음

C:\WTemp>javac HelloWorldApp.java

C:\WTemp>dir
C 드라이브의 볼륨에는 이름이 없습니다.
볼륨 일련 번호: F8AA-0E2D

C:\WTemp 디렉터리

2008-02-29 오후 10:17 <DIR> .
2008-02-29 오후 10:17 <DIR> ..
2008-02-29 오후 10:17          431 HelloWorldApp.class
2008-02-29 오후 09:54          123 HelloWorldApp.java
                2개 파일              554 바이트
                2개 디렉터리 67,149,955,072 바이트 남음

C:\WTemp>java HelloWorldApp
Hello World

C:\WTemp>
```

자바 프로그램 예제

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

클래스정의

메소드 정의

1. 자바 프로그램은 확장자가 “java”인 소스 파일을 하나 만들어서 자바 문법에 맞는 내용을 기술해야 한다.

2. 자바는 클래스를 하나의 단위로 프로그램을 작성하기에 자바 소스 파일 안에 클래스를 정의해야 한다.

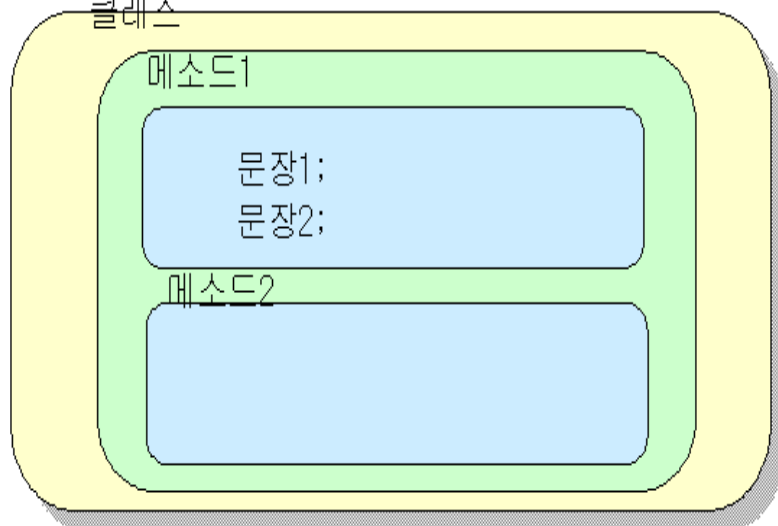
▶ 3. 제대로 동작하려면 자바 소스 파일명이 클래스명과 동일해야 한다.

▶ 클래스

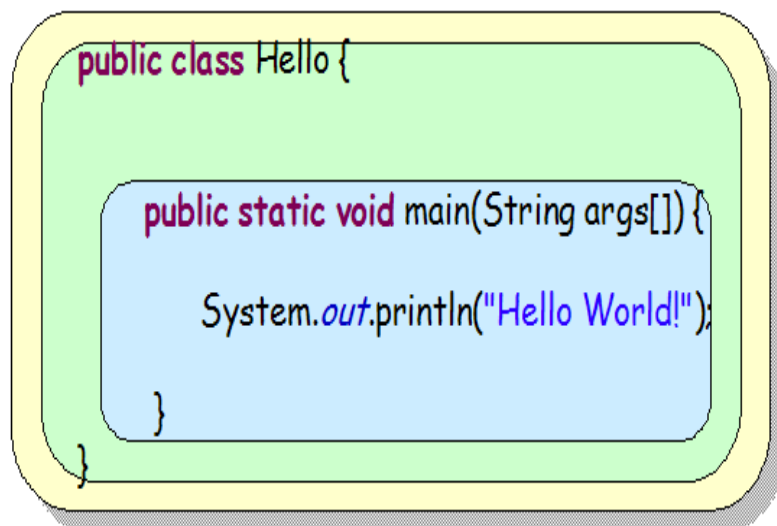
- ▶ 객체를 만드는 설계도: 자바 프로그램은 클래스들로 구성

소스 파일

클래스



Hello.java



누구든지 접근할 수 있도록
public으로 선언합니다.

클래스라는 것을 알려주는 부분

클래스 이름

```
public class Hello{
```

리턴 유형을 지정하는 부분. void는 아무것도 리턴하지 않는다는 것을 의미합니다.

메소드 이름

메소드 인자

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

누구든지 접근할 수 있도록 public으로 선언합니다.

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

표준 출력으로
출력하라는 것을
의미합니다.

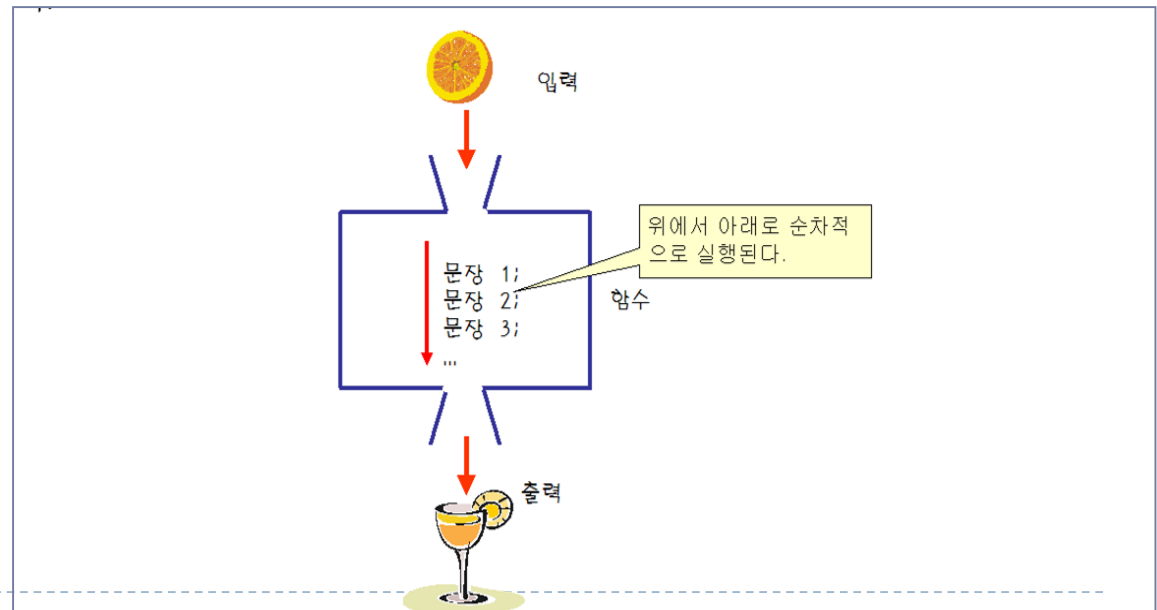
출력할 String 값 (문
자열)

```
}
```

```
}
```

▶ 메소드 (method)

- ▶ 입력을 받아서 작업을 수행하고 결과를 내보내는 작은 기계
- ▶ 메소드 안에 들어 있는 문장들을 차례대로 실행한 후에 작업의 결과를 반환



▶ main 메소드

- ▶ 프로그램의 진입점
- ▶ 프로그램을 실행시키면 main 함수 내부에 기술된 내용들을 순차적으로 수행



▶ 문장

- ▶ **문장(statement)**은 사용자가 컴퓨터에게 작업을 지시하는 단위이다.
- ▶ 문장들은 메소드 안에 들어 있다.
- ▶ 보통 프로그램의 한 줄이 하나의 문장이 된다.
- ▶ 문장의 끝은 항상 세미콜론(;)으로 끝나게 된다.

```
System.out.println( "안녕하세요!" );
```



```
/**
 * 표준 출력으로 "Hello World!"를 표시하는 간단한 자바 애플리케이션의 구현이다.
 */

public class Hello {
    public static void main(String args[]) {
        System.out.println("Hello World!") // 문자열 출력
    }
}
```

주석(comment)

▶ 주석

- ▶ 코드를 이해하기 쉽게 하기 위해서 사용 → 설명을 적어 넣은 것

▶ 주석문의 종류

- ▶ /* ~ */ 주석문
 - ▶ 블록 단위로 주석 처리
- ▶ // 주석문
 - ▶ 한 줄만 주석처리
- ▶ /** ~ */ 주석문
 - ▶ javaDoc를 이용해 소스코드에 대한 도움말을 생성

객체의 개념으로 자바 프로그래밍 작성

- ▶ 프로그래밍 순서
 - ▶ 클래스 이름 정의하기
 - ▶ 클래스 정의하기
 - ▶ 변수와 메소드 정의하기
 - ▶ 프로그램이 시작되는 메인 함수 구현하기
- ▶ 예) FirstProgram.java



FirstProgram.java

- ▶ 1단계: 클래스 이름 정하기

FirstProgram



FirstProgram.java

▶ 2단계: 클래스 정의하기

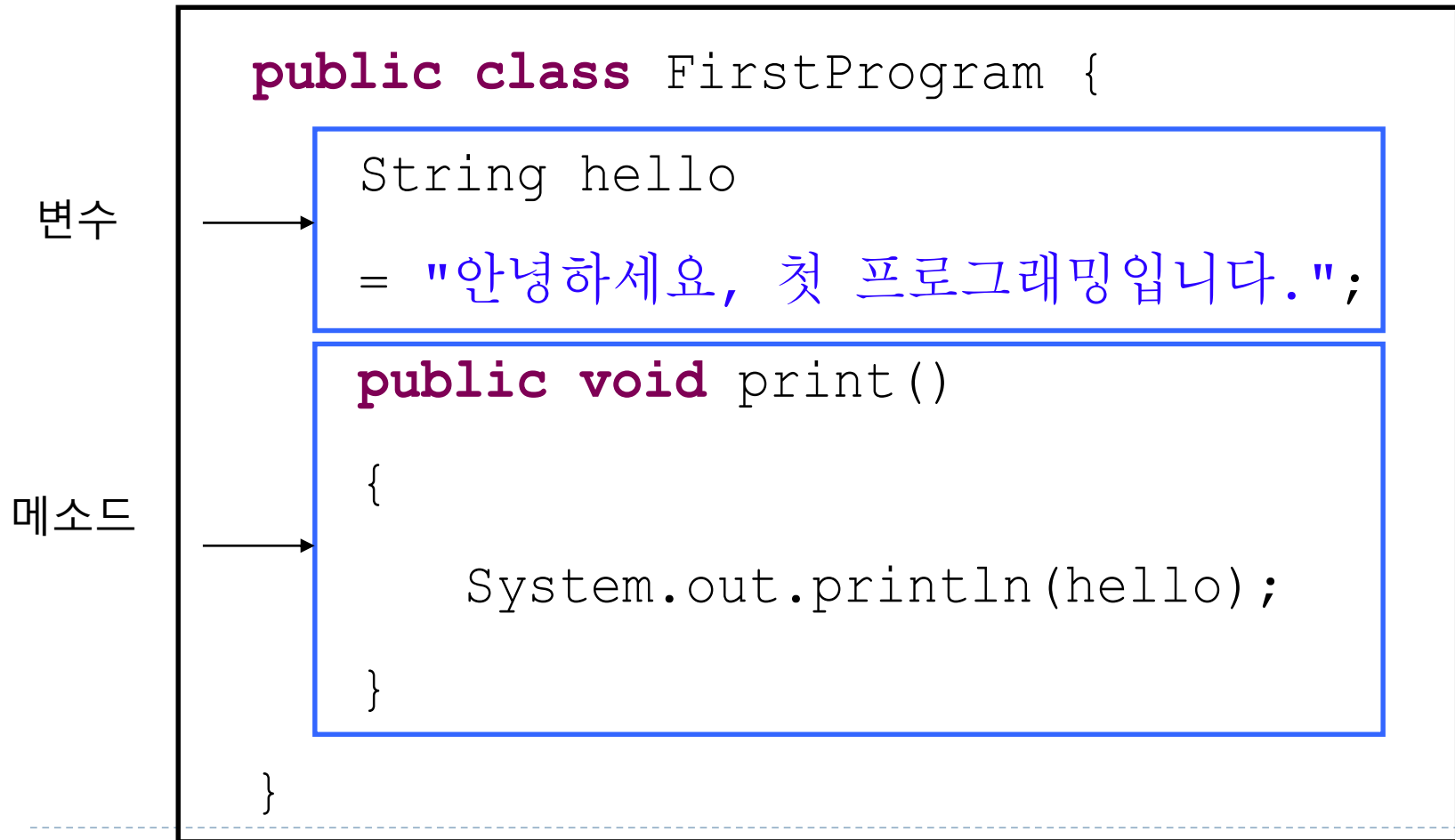
```
public class FirstProgram {  
  
}
```

- ▶ 단, 소스 파일 이름은 클래스의 이름과 동일하고 확장자는 .java임
 - ▶ FirstProgram.java



FirstProgram.java

▶ 3단계: 변수와 메소드 정의하기



FirstProgram.java

▶ 4단계: 프로그램이 시작되는 메인 함수 구현하기

```
public class FirstProgram {  
    String hello = "안녕하세요, 첫 프로그래밍입니다.";  
    public void print()  
    {  
        System.out.println(hello);  
    }  
    public static void main(String[] args) {  
        FirstProgram fp = new FirstProgram();  
        fp.print();  
    }  
}
```