Power Java

제1장 자바 소개

자바(Java)란?

(Quiz) 우리가 학습하려는 자바는 다음 중 무엇일까?

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!");
   }
}
```

프로그램이란?

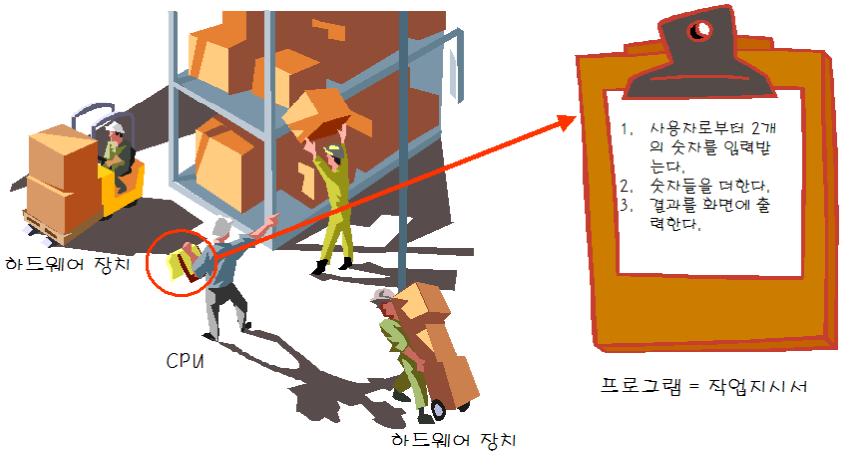


그림 1.1 프로그램은 작업 지시서와 같다.

컴퓨터가 이해하는 언어

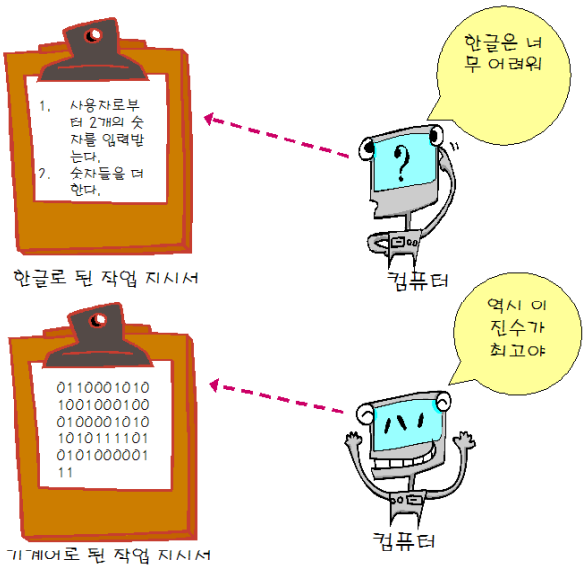


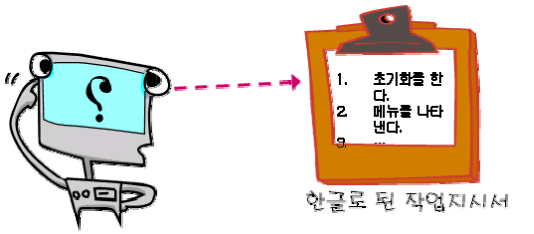
그림 1.2 컴퓨터는 한글로 된 작업 지시서는 이해하지 못하는 반면 기계어로 된 작업 지시서는 이해할 수 있다.

기계어

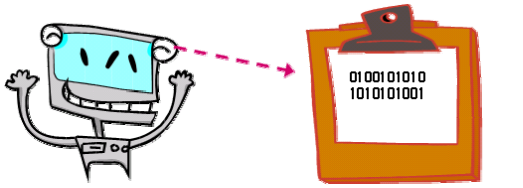
Q) 컴퓨터가 이해할 수 있는 언어는 어떤 것인가?

A) 컴퓨터가 알아듣는 언어는 한가지이다. 즉 0과 1로 구성되어 있는 "0011011110001010..."과 같은 기계어이다.

A) 컴퓨터는 모든 것을 0과 1로 표현하고 0과 1에 의하여 내부 스위치 회로들이 ON/OFF 상태로 변경되면서 작업을 한다.



한글로 된 작업지시서

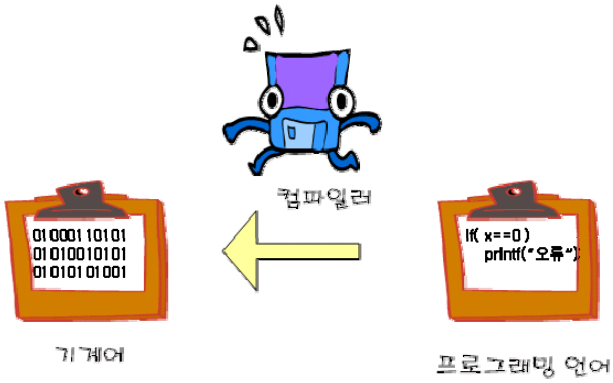


기계어로 된 작업지시서

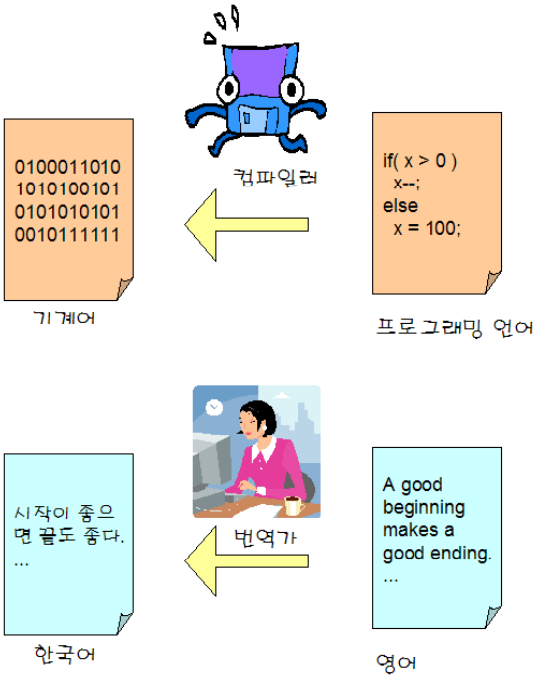
프로그래밍 언어의 필요성

Q) 그렇다면 인간이 기계어를 사용하면 어떤가?

- 기계어를 사용할 수는 있으나 이진수로 프로그램을 작성하여야 하기 때문에 아주 불편하다.
- 프로그래밍 언어는 자연어와 기계어 중간쯤에 위치
- 컴파일러가 프로그래밍 언어를 기계어로 통역

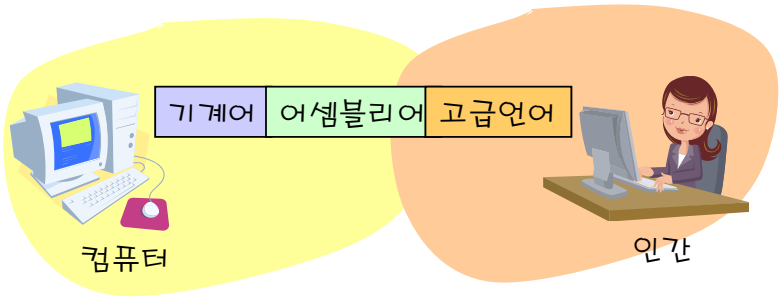


자바는 프로그래밍 언어의 일종



프로그래밍 언어의 분류

- 기계어(machine language)
- 어셈블리어(assembly language)
- 고급 언어(high-level language): 자바, C++, C언어



중간 점검

중간점검

1. 컴퓨터가 범용성을 가질 수 있는 근본 이유는 무엇인가?
2. 인터넷에서 컴퓨터의 명령어(instruction)의 구체적인 예를 찾아보자.
3. 왜 mp3 플레이어는 컴퓨터에 비하여 덜 범용적인가?

중간점검

1. 컴퓨터가 직접 이해할 수 있는 단 하나의 언어는 기계어이다.
2. 프로그래밍 언어를 기계어로 변환시켜주는 프로그램을 컴파일러한다.
3. 우리는 왜 기계어를 사용해서 프로그램하지 않는가? 힘들어서

자바란?

- 패러다임(**Paradigm**)
  - 객체 지향 프로그래밍(Object-oriented),
  - 구조적 프로그래밍(structured)
  - 절차적 프로그래밍(imperative)
- 등장
  - 1995
- 누가 설계하였나?
  - Sun Microsystems
- 가장 최근 버전
  - Java Standard Edition 6 (1.6.0_14)
- 설계 원칙
  - Static, strong, safe, nominative, manifest

www.wikipedia.com참조

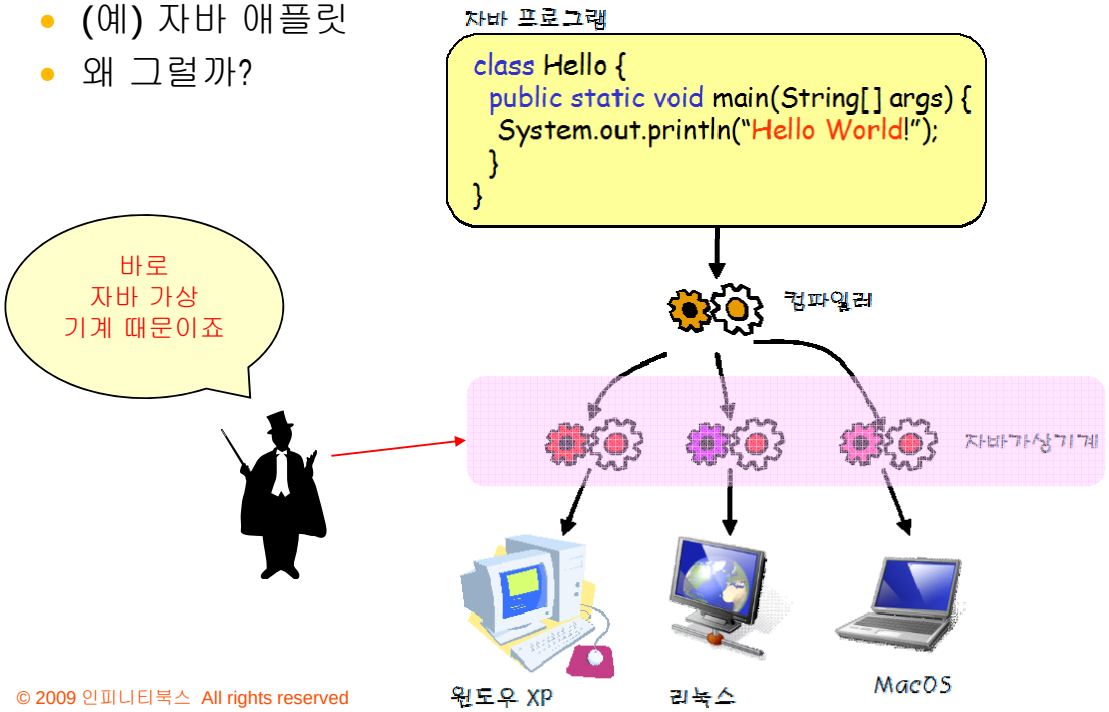
자바란?

- 비슷한 언어
  - Generic Java, Pizza
- 자바에게 영향을 준 언어
  - Objective-C, Ada 83, Pascal, C++, C#, Eiffel, Smalltalk, Mesa, Modula-3, Generic Java
- 영향을 끼친 언어
  - Ada 2005, C#, D, ECMAScript, Groovy, J#, PHP, Scala, JavaScript, Python
- 운영 체제
  - Cross-platform (multi-platform)
- 라이센스
  - GNU General Public License / Java Community Process
- 웹사이트
  - http://java.sun.com

www.wikipedia.com참조

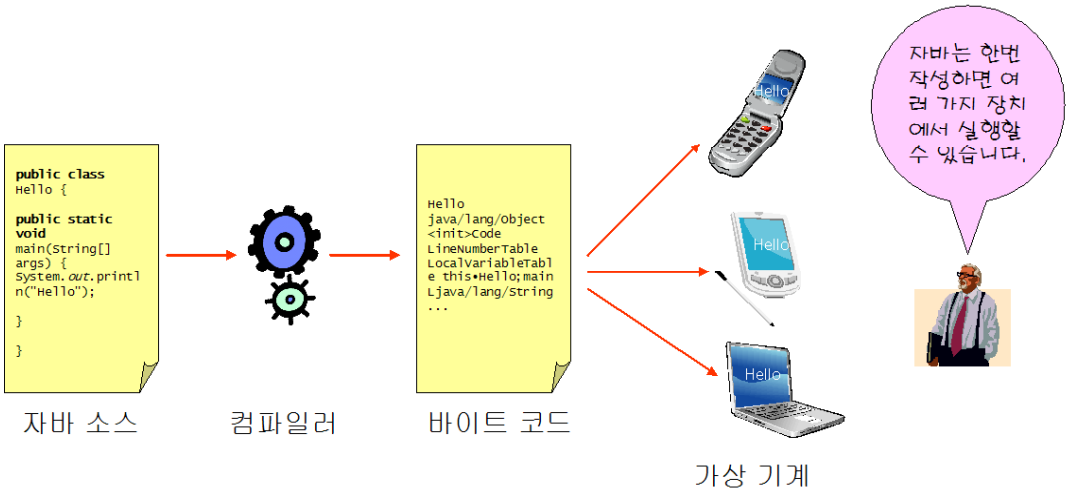
자바 가상 기계

- 자바는 다양한 컴퓨터에서 동일한 모습으로 실행이 가능하다.
  - (예) 자바 애플릿
  - 왜 그럴까?



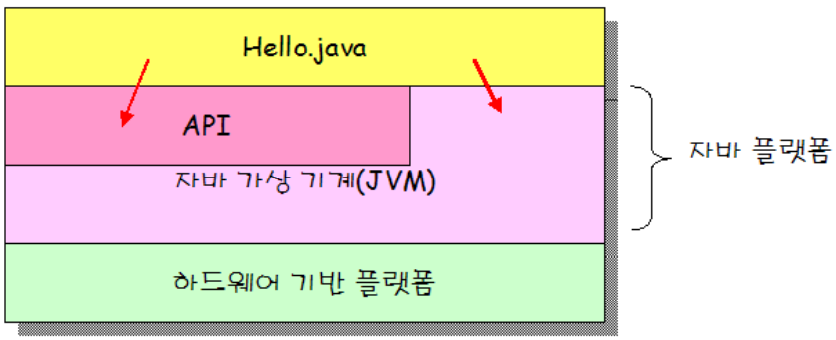
자바 가상 기계

- 자바 컴파일러는 특정한 컴퓨터가 아닌 가상적인 기계(virtual machine)를 위한 코드를 생성한다.



자바 플랫폼

- 플랫폼(platform)이란 프로그램이 실행되는 하드웨어나 소프트웨어 환경이다.
- 일반적으로 API란 많은 유용한 기능을 제공하는 라이브러리들의 모임이다.



중간 점검 문제

중간점검

1. 자바 컴파일러가 소스 코드를 컴파일하면 바이트 코드가 생성된다.
2. 바이트 코드를 해석하여 실행하는 소프트웨어는 자바 가상 기계 이다.
3. 자바가 어떤 컴퓨터에서나 실행이 가능한 이유는 무엇인가?

특정한 컴퓨터가 아닌 중간적인 코드를 생성하고 이것을 해석하여 실행하는 구조로 되어 있기 때문이다.

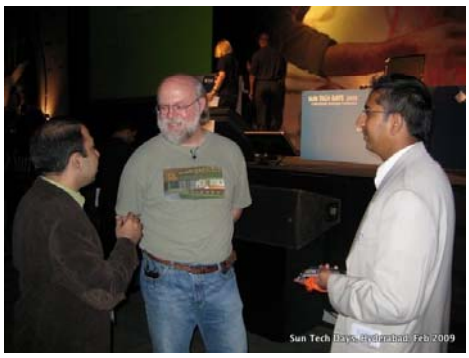
자바의 역사

- 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 그래픽스 사를 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 중립적인 언어가 이상적이라는 것을 발견

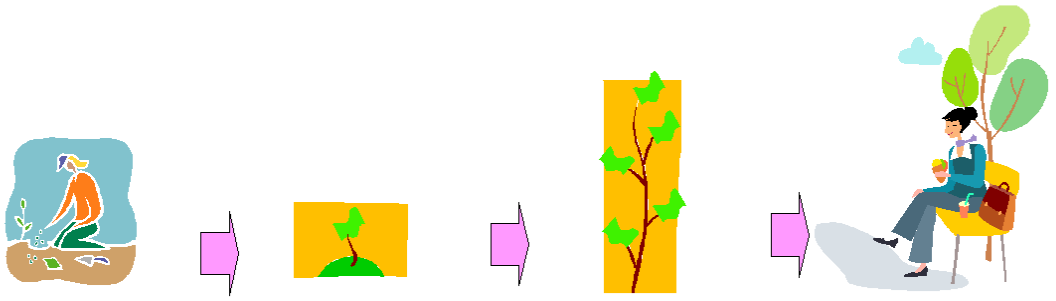
자바는 누가 만들었을까?

- James Gosling





자바 버전



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

- 단순(Simple),
- 객체 지향(Object-Orineted),
- 고성능(High-performance),
- 견고(Robust),
- 안전(Secure),
- 이식 가능(Potable),
- 인터프리트 방식(Interpreted),
- 멀티 스레드 지원(Multithreaded),
- 동적(Dynamic),
- 분산 처리 지원(Distributed),
- 컴퓨터 구조에 중립적(Architecture-neutral).

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 지난 30년간의 연구를 통하여 그 가치를 입증한, 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

자바의 특징

- 견고하다
  - 오류를 만들 수 있는 원인들을 제거
  - (예) 포인터 개념을 삭제하였으며 컴파일시에 강력하게 데이터 타입을 검사
- 안전하다.
  - 바이러스, 파일의 삭제나 수정, 데이터 파괴 작업이나 컴퓨터 오류 연산 등을 방지하면서 실행되도록 설계되었다.
- 컴퓨터 구조에 중립적이다.
  - 컴퓨터 구조에 중립적인 바이트 코드로 번역
  - 이러한 바이트 코드 특성 때문에 인터넷에 연결된 서로 다른 기종의 컴퓨터에서도 자바는 실행될 수 있다.

자바의 특징

- 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.
- 기타 장점
  - 비교적 배우기 쉽고 특히 C언어를 이미 학습하였다면 더욱 쉽다.
  - 자바 웹 스타트(Java Web Start) 소프트웨어를 사용하면 제작된 응용 프로그램을 한 번의 마우스 클릭으로 실행

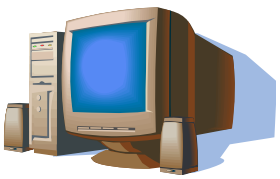
중간 점검

중간점검

1. 자바의 어떤 특징 때문에 컴퓨터 구조에 중립적인가? 바이트 코드와 자바 가상 기계
2. 자바와 C언어를 비교하여 보자. C 언어는 절차적 언어, 자바는 객체 지향 언어
3. 멀티스레딩이란 무엇인가? 여러 작업을 동시에 실행하는 것

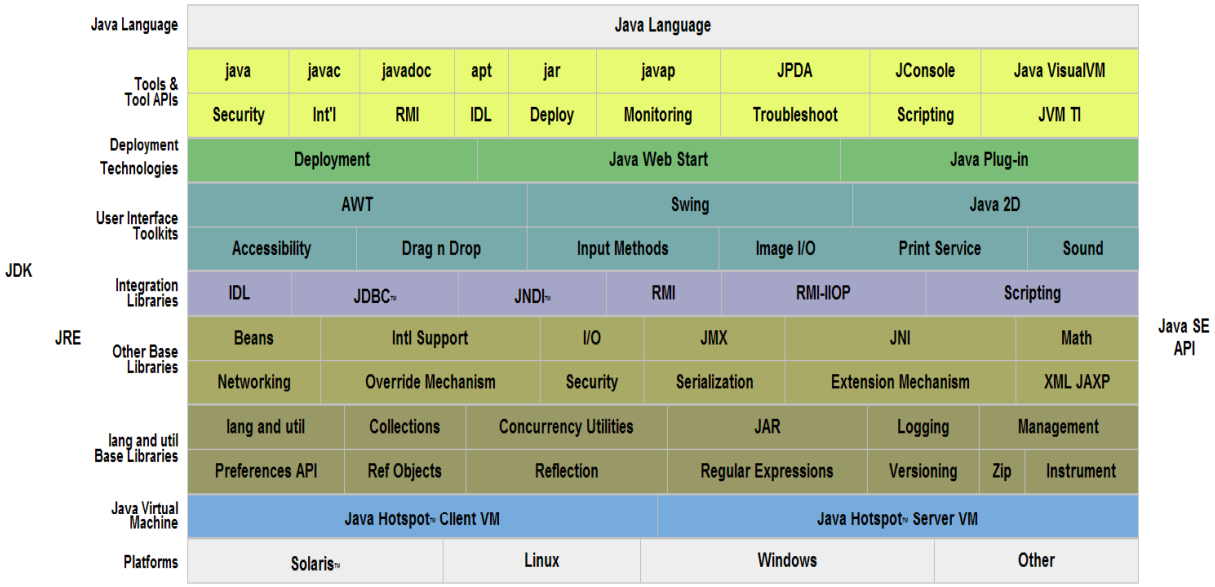
자바의 에디션

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



Java SE

- Java SE는 데스크탑과 서버에서 자바 애플리케이션을 개발하고 실행할 수 있게 해주며 임베디드 환경(embedded environment)과 실시간 환경(real-Time environments)도 지원

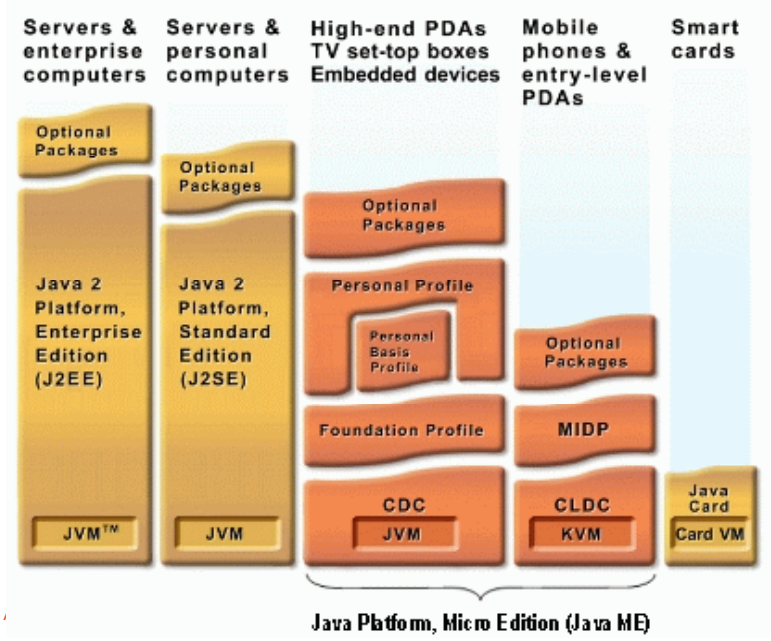


Java EE

- Java EE는 기업용 애플리케이션을 개발하는 데 필요한 여러 가지 도구 및 라이브러리들을 모아 놓은 것
- 응용 서버, 웹서버, J2EE API, 엔터프라이즈 자바 빈즈(JavaBeans) 지원, 자바 서블릿 API 와 JSP 등을 포함

Java ME

- Java ME는 핸드폰, PDA, TV 셉톱박스, 프린터와 같은 모바일 기기나 다른 엠베디드 장치들에서 실행되는 애플리케이션을 위한 강인하고 유연한 환경을 제공



자바로 만들 수 있는 것

- 자바 애플리케이션(Java application)
  - 독립적으로 실행될 수 있는 일반 응용 프로그램
- 자바 애플릿(Java applet)
  - 웹 브라우저 안에서 실행되는 작은 프로그램이다.

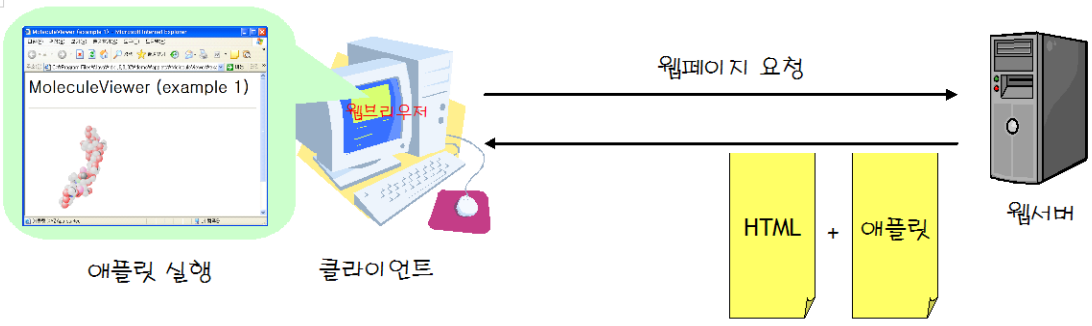


그림 1.12 자바 애플릿의 실행 과정

자바로 만들 수 있는 것

- 자바 서블릿(Java servlet)
  - 웹서버에서 동작하는 서버 모듈로서 클라이언트의 요구를 받아서 그에 대한 처리를 한 후에, 실행 결과를 HTML 문서 형태로 클라이언트 컴퓨터로 전송

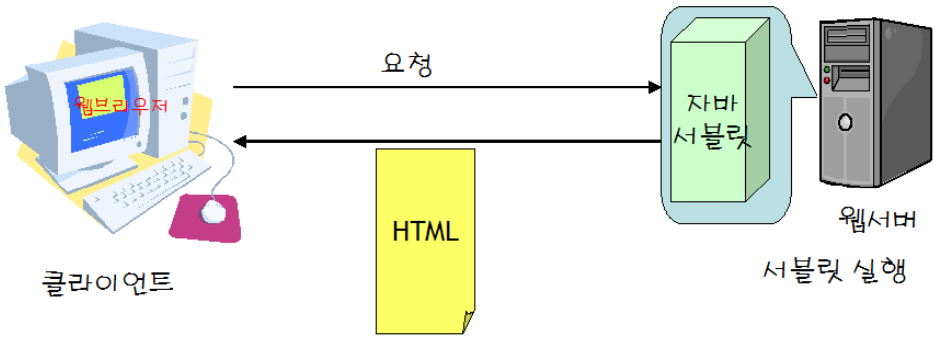


그림 1.13 자바 서블릿의 실행 과정

자바로 만들 수 있는 것

- HTML안에 자바 코드를 넣으면 웹페이지를 사용자와 상호작용하도록 만들 수 있다. JSP는 서버에서 실행되고 결과를 사용자에게 보여준다.

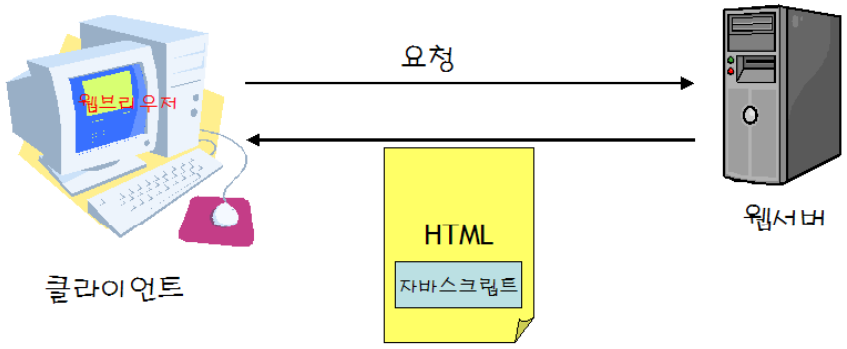


그림 1.14 JSP의 실행 과정

자바로 만들 수 있는 것

- 자바 빈즈(Java Beans)
  - 자바로 작성된 컴포넌트를 자바 빈즈(Java beans)라고 한다. 컴포넌트는 애플리케이션을 형성하기 위한 프로그램 빌딩 블록

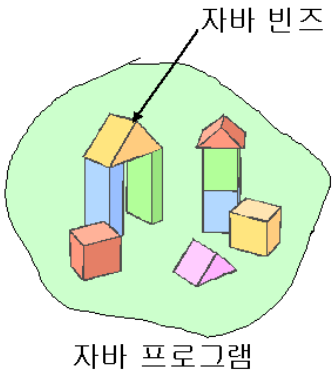


그림 1.15 자바 빈즈의 개념

중간 점검 문제

1. 애플릿과 서블릿을 비교하여 보자.

   애플릿: 서버에서 다운로드되어서 웹 페이지 안에서 실행
   서블릿: 웹 서버 안에서 사용자의 요청을 처리

2. JSP에 대하여 웹을 통하여 조사하여 보자.

   JavaServer Pages (JSP)는 서버 측의 자바 기술로서 클라이언트 컴퓨터의 요청에 따라서 동적으로, HTML, XML 등이 포함된 웹 페이지를 생성한다.
   → 자세한 것은 다음 기회에…

Q & A