JavaSE(standard Edition)

-데스크탑에서 실행되는 모든 응용프로그램

-아래의 두 플랫폼 마저도 SE를 설치해야 한다

JavaEE(Enterprise Edition)

-웹분야

-기업용 어플리케이션은 대규모의 배포가 필요

즉 큰규모의 다수의 사용자가 이용해야 하므로

인터넷 기반으로 간다.

JavaME(Micro Edition)

-임베디드 분야(하드웨어 분야)에 사용되는

자바.

-스마트폰 출시 이전 피처폰 시대에 주로 많은

개발이 이루어졌으나 스마트폰 출시 이후 안드

로이드에게 밀렸다.

JDK(Java Development Kit)-자바 개발도구

=개발도구+실행환경(JRE)포함되어 있다.

JRE(Java Runtime Enviroment)-자바 실행환경

-96년도에 JAVA 출시

용도: 세탁기, 임베디드..

그 이후 용도: 웹개발 최적화.

-2000년대(닷넷(C#)vs자바) 자바승

program files 디렉토리에 넣지말자

JRE에는 JVM이라는 Java Virtual Machine(자바용 가상 os)

오직 JVM만이 .class 라는 바이트코드라 불리는 기계어를 해석할 수 있다.

Compile

-----------------------------------------------------------------------------------

1.자바의 개발 플랫폼(자바 언어로 개발할 수 있는 분야)

java SE:응용 프로그램(실행할 수 있는 코드가 반드시 유저의 PC에 설치되야 한다.)

java EE:기업용 어플리케이션(웹)

java ME:임베디드 어플리케이션 -->안드로이드로 대체할 것임.

2.JDK와 JRE란 무엇이고 차이점은?

-개발, 실행환경이다.JRE는 JDK에 포함되어 있다.

3.자바와 자바스크립트의 차이점

-컴파일 여부:Java는 하지만 스크립트는 X

속도: JAVA(컴파일 언어)가 훨씬 빠르다. 자바스크립트(인터프리터)

접근성: 자바스크립트가 좋다.

4.컴파일이란?--원본소스를 기계어로 변환하는 과정.

5.자바의 슬로건인 Write once run anywhere의 의미

-한번 작성된 자바 프로그램은 해당 플랫폼에 독립적으로

수행될 수 있다.(OS종류의 상관없이 돌아간다)

6.바이트코드와 자바가상머신

-바이트코드란? .class를 의미하며 오직 JVM만이 해석가능

Java Virtual Machine: 자바 코드 해석 및 메모리를 관리하는 소프트웨어.

7.환경변수란 무엇이고 사용 목적은?

-OS에 등록하여 사용할 수 있는 변수를 의미하며 특히 path라는 변수에는 os가

가동되고 있는 한 언제든 접근이 가능한 전역데이터를 등록할 수 있다.

어느 위치에서건 등록된 경로를 참조할 수 있다.

8.자료형이란 무엇이고, 자바의 자료형의 종류 및 크기는?

-데이터의 종류를 의미하여 종류를 구분하는 이유는 메모리에 차지할 데이터의 용량을 결정하기 위함이다

문자:char(2)

숫자:byte(1)<short(2)<int(4)<long(5)

논리값:boolean(1)

--------------------------------------------------------------------------------------------------------

자료형(데이터의 종류)

-문자,숫자,논리값-->1,0

-일거리 제공시 그 용량을 결정할수 있도록 제공되는 방법

문자 :char-->숫자형(양수)아스키코드+유니코드

숫자 :정수-byte short int long

1 2 4 8

실수-float double

논리값: boolean

작은자료형--->큰자료형 promotion

int x=3;

byte b=5;

x=b;//형변환이 일어난다. 자동형변환

b=x;//에러가 생긴다. 하지만 변환하려면 b=(byte)x; 된다. 강제형변환

자바의 자료형은 총 4개이다

[기본자료형]

문자

숫자

논리값

[객체자료형]

-----------------------------------------------------------------------------------------------------

자바의 자료형은 총 몇개? 4개

기본자료형(문자, 숫자, 논리값), 객체형

기본자료형 중 형변환 서로 불가능한 자료형은 boolean

숫자=문자형(주의: 숫자간 연산에는 부적절..

왜? 문자에 매핑되는 키보드값이 목적이므로 음수는 지원하지 않는다!)

환경변수란: 환경변수(os에 등록되는 전역변수)

시스템이 가동되는 동안 참조할 데이터를 담아 놓는 변수

path:시스템이 가동되는 동안 경로를 참조하기 위함.

javac.exe, java.exe를 참조하기 위해서..

main()메서드란? 프로그램의 시작 시점을 결정하는 실행부

자바클래스 작성시 main()은 둬야 하나??

둘 경우- 실행할 수 있는 클래스이긴 하나, 클래스가 재사용성은 떨어진다.

왜? -자바는 한번 작성된 클래스가 javaSE,javaEE, javaME에 모두 사용될

수 있는데, mian을 두면 javaSE에 한정됨.., 객체를 표현하기 위한 용도와 맞지 않음

-----------------------------------------------------------------------------------------------------

1.클래스 작성 방법

변수, 메서드로 구성됨.

2.클래스를 사용하는 법

-즉 메모리에 올리는법=객체의 인스턴스를 생성하면 된다!

3.메모리 구조

static: 정적 메모리 영역

클래스 원본 코드

4.초기화 블럭

sttic 초기화 블럭 { }:클래스 로드 시

인스턴스 초기화 블럭:각 인스턴스 생성 시마다.

----------------------------------------------------------------------------------------------------

클래스 작성법

-메서드 대해서..

일반적인 함수의 특징 가지고 있었다.

(1)반환형 지원

(2)매개변수

-메서드 중, 생성자.

생성자란? 객체 생성시 초기화를 목적으로 무조건 호출되는 메서드!

특징?

클래명과 반드시 일치해야 한다

생성자의 명시는 개발시 생략이 가능하나,

만약 생략하면 에러가 나지는 않고, 컴파일러에 의해 default생성자가 관여된다. 이때 디폴트 생성자의 코드는 최소한의 관여만 하므로, 로직이 없다!(결국 목적은 에러 방지)

반환형이 없다!

만일 반환형을 두면? 안난다는게 정말 무섭다. New 뒤에 오는 생성자로 생각하지 않기 때문에 객체 생성시 호출되지 않는다.

-하나의 클래스내에 메서드명은 중복이 가능하다

(대부분의 유명한 객체지향 언어에서는 추세)

Overloading(=메서드 중첩)

왜?? 메서드명은 소중하니깐.. 명명규칙을 힘들게 고민했기 때문에.. 사소한 기능차이로 새로운 메서드명을 지정해야 한다는 것은 상당히 비효율적이다.

Call by value(메서드 호출시 값으로 호출)

변수의 값이 복사되어 전달되므로, 해당 변수는 영향을

전혀 받지 않는다.

Call by reference(메서드 호출시 그 값이 특히나 객체의 주소값일 경우)

변수의 값이 주소값이므로, 해당 객체가 영향을 받는다.

package와 import

클래스들을 모아놓은 집합을 패키지라 한다.

왜 모아놓나? 클래스명의 중복방지, 관련성있는 클래스들끼리 묶어서 관리

개발시 환경 구축

프로젝트/src- .java 패키지명/Dog.java

Bin- .class 패키지명/Dog.class

.java를 .class로 만들되, 지정한 경로에 패키지를 생성까지 해주는 명령어 : Javac –d 생성경로 컴파일대상.java

특정 클래스를 패키지에 넣었을 때, 외부의 클래스가 접근하기 위해서는 그 경로를 알아야 하는데 이때 사용하는 경로를 위한 환경변수는? classpath

1.접근제한자

- 자바에서 지원하는 보안 등급을 결정하는 키워드

Private – default-protected-public

2.클래스에서 메서드의 역할

- 멤버변수를 제어하는 목적으로 많이 사용함

3.은닉화(C#,객체지향 언어의 일반적 특징 중 하나)

- 객체가 보유한 데이터는 보호하고, 객체가 보유한 데이터에 대한 제어는 개발자가 제공하는 메서드에 의해 제어하도록 개발법

멤버변수는 private 외부의 접근을 막고, 이 막혀진 데이터에 대한 조희 getter로 제공하고, 수정은 setter로 정의한다.

get+단어(첫 철자는 대문자로)

-public String getName(){

return name;

}

public void setName(String name){

this.nam=name;

}

4.API Document의 뜻과 문서 html 생성법

- Application programming interface

어플리케이션 제작을 위해 제공되는 라이브러리.

java설치루트/bin/javadoc.exe

주의) 클래스는 public으로 선언되어 있어야 한다.

5.API 활용 방법 실습

rt.jar (.class의 집합이므로 메서드와 변수를 개발자가 알수 없다. 따라서 API Document를 참조해야 한다)

일반파일을 압축 – zip등..

클래스파일을 압축 – jar(java archive)

6.String 클래스의 특징 – 실습

7.자바에서 배열은 객체취급

boolean[] arr=new boolean[3];

8.main()메서드 호출시 매개변수에 대해서..

9.has a

- 현실을 제대로 반영하려면, 객체 단독으로는 개발이 불가

10.call by reference 실습

- 메서드 호출시 넘겨지는 데이터가 주소값일 때.

넘겨진 주소값으로 어떤 처리를 할 경우 해당 주소값의 인스턴스가 영향을 받는다!

개발시 객체와 객체는 관련성을 맺는다.

1. has a
2. is a : 상속

[로직 작성시 메서드 정의기법]

오버로딩 : 하나의 클래스. 같은 클래스내에서 기능이 비슷한 메서드가 있다면 새롭게 이름을 부여하지 말자!

오버라이딩 : 상속관계에 있는 클래스. 이미 부모의 메서드가 존재하더라도 자식클래스에서 그 메서드를 재정의하자!=업그레이드

1.(객체와 객체간)관계의 종류

-has a 관계 : 부품관계

-is a 관계 : 상속관계

2. 상속

-자식 클래스가 부모클래스의 능력을 부여받는 것.

🡪부모 클래스의 속성과 메서드를 사용할 수 있는 개발방법

🡪즉 자식이 자신이 가진 속성과 메서드 이외에 부모의 영역까지 확장하였으므로 extends라 한다.

🡪부모클래스가 인수있는 생성자만 보유할 경우

3.자바의 최상위 객체

Object

4.오버라이드(=업그레이드)

- 상속관계에서 자식클래스가 부모클래스의 메서드를 재정의하는 메서드 정의 기법

5.객체간 형변환

기본자료형간 형변환 가능하다.

객체자료형간 형변환 가능?-객체형도 자료형이니까

주의!

기본자료형간 형변환사엔 크기가 중심

Bird > Duck

객체가 레퍼런스 할 수 있는 대상이 넓다면 그게 큰 자료형이다! 결국 부모가 더 크다!

6.다형성

하나의 자료형으로 여러가지 다양한 동작을 일으키는 기법

7.Wrapper 클래스

기본자료형과 객체자료형간의 형변환을 지원하기 위한 클래스가 Wrapper 클래스이다!

자바의 기본자료형은 1:1로 객체로 변환될 수 있는 클래스가 지원된다!

byte🡪Byte

short🡪Short

int🡪Integer.parseInt(“1”)🡪1

long🡪Long

1.추상클래스란?

-불완전한 클래스

-클래스를 불완전하게 하는 법 : 메서드 몸체를 없이 정의

-사용이유 : 클래스 정의 그 내용을 확정지을 수 없거나 확정지어서도 안되는 경우. 즉 계획 단계에서 사용될 수 있는 클래스.

2.인터페이스 : 기능만을 보유한 객체

- 클래스가 아니므로 현실의 다중성을 반영하기에 적절하다.

3.extends, implements = is a

A extends B= A와 B는 같은 종류의 자료형

A implements B=A와 B는 같은 종류의 자료형

4.상수

상수=숫자(현실)

전산에서의 상수란, 프로그램이 종류될때까지 그 값이 변하지 않고, 유지되고 있다면 상수라 한다.

따라서 상수가 될 수 있느 것들은 자바의 모든 자료형

숫자, 문자,논리값, 객체

Public Final static intX=5;//🡨상수

5.GUI API 지원한다!

윈도우-Window🡪Frame

BorderLayout (많이 쓰임)

FLowLayout (많이 쓰임)

GridLayout

GridBagLayout

CardLayout

**GUI-플리케이션을 그림으로 구현하는 방식(직관적이면서 편함)**

**CLI- Command Line Interface(명령어기반이기 때문에 불편함)**

**Dos, SQlpuls**

**GUI프로그래밍- 윈도우 프로그래밍**

**결론: 자바언어도 윈도우 프로그래밍을 지원한다**

**GUI 컴포넌트의 분류**

**-남을 포함하는 컴포넌트(컨테이너)**

**JFrame, JPanel, Applet**

**-포함 당하는 컴포넌트(비쥬얼 컴포넌트)**

**컨테이너를 제외한 모든 것.**

**레이아웃 관리자**

**-모든 컨테이너는 자신의 몸체에 컴포넌트들을 어떻게 배치할지를 고민. 5가지 유형의 배치관리자가 지원된다.**

**JFrame : 디폴트-BorderLayout**

**JPanel : 디폴트-FlowLayout**

**Applet : 디폴트- FlowLayout**

**-Frame 내에서의 부분적 레이아웃 적용하기 위해선 JPanle 이용해야 한다!**

**이벤트**

**Click🡪Action**

**주의!! Action은 버튼에만 부여할 수 있는게 아니다.**

**Html도 Click을 Button에만 적용하지 않았다.**

**이벤트 처리과정**

**API Document**

**수 많은 클래스가 정의되고 있다.**

**자바의 객체의 유형**

**1단계) 객체의 종류를 파악!**

**-일반클래스**

**-추상클래스**

**-인터페이스**

**2단계) 생성하는 방벙을 결정하라**

**-일반클래스 - new한다 (따라서 생성자를 보러간다)**

**-추상클래스 - 자식에서 완성한 후 자식을 올린다!**

**-인터페이스 - 자식에서 완성한 후 자식을 올린다!**

**3단계) API상의 각종 메서드 적절하게 적용**

1. **자바의 그래픽 처리**

**-주체(화가) : 컴포넌트**

**-행위 : paint()메서드**

**-그래픽스타일 : Graphics**

**g.draw~~..**

**-손대지 말아야 할 컴포넌트 : JButton, JTextFiled, Checkbox.**

**-손대야 할 경우 : 텅 빈 컴포넌트,**

**Canvas, JFrame,JPanel**

1. **내부 익명 클래스**

**-클래스안에 선언할 수 있는 이름없는 클래스**

**-언제쓰나? .java소스로 분리시켜놓기에는 재사용성이 떨어지는경우.**

**컬렉션 프레임워크(=Collection Framework)**

**모음, 수집**

**컬렉션 프레임웍에서 수집, 모음의 대상은 객체이다!**

**Java.util**

**순서있는 객체를 처리하는데 유용한 List 계열**

**-ArrayList(거의 배열과 같다.**

**차이점 - 크기가 동적으로 변경된다. 처리할데이터가객체에 한정.)**

**ArrayList에 숫자를 넣고 싶다면? – 원칙:숫자를 객체화시켜야 한다. 3->new Integer(3);**

**-예외 : 자동으로 객체화 시켜준다(Auto Boxing)**

**순서있는 객체를 처리하는데 유용한 Set 계열**

**-Hashset 구현.**

**-모든 요소를 출력하려면 일렬로 늘어뜨려야 한다**

**Iterator, Enumeration.**

**hasNext():다음 요소의 존재여부 true/false 반복문의 범위 설정가능.**

**Next():각 요소를 접근할 수 있는 메서드. 호출할때마다 다음요소에 자동접근**

**순서없이 Key-value의 한쌍으로 모아진 데이터를 처리하는데 유용한 Map**

**Hashmap**

**JTable🡪**

**개발방법 이론이 반영되어 있다.**

**디자인과 로직은 분리시켜 개발해야 한다!**

**선배들에 의해 구전.**

**유지보수성이 좋아진다.**

**개발자에겐 업무의 효율성**

**사장님에겐 돈이 덜 나간다.**

**Stream(흐름)**

**-현실 : 대상은 물**

**-전산 : 대상은 데이터.**

**방향 : 절대적? 상대적?->상대적이다.**

**실행중인 프로그램을 기준으로**

**입력(Input): 데이터가 들어오면**

**출력(Output): 데이터가 나가면**

**IO**

**-주변의 스트림을 이용한 예**

**파일복사, 바코드 리더기, 모니터, 빔프로젝트, 프린터, 마우스**

**-스트림은 Java.io 패키지에서 지원**

**Exception(예외- 정상수행이 될 수 없는 예외적상황)**

* **실행시 에러 가능성이 다분한 코드 영역은 try문으로 감싸고, 이영역에서 에러상황이 발생하면 원칙상 프로그램의 비정상 종료가 발생하게 된다. 따라서 비정상종료를 방지하기 위한 블록블 catch문을 마련하여 실행부로 하여금 정상 수행을 유도.**
* **catch문에서 작성할 내용은?**

**정상수행을 유도해 내거나, 비정상 수행의 원인 공지**

* **프로그램에서 발생할 수 있는 예외 상황은 Sun에서 정해놓은 것만 예외다!**
* **예외가 발생하면 JVM의 메모리영역에는 해당 예외 객체가 올라오고, catch(예외객체)**

**e.printStackTrace();**

**자바 문법**

**GUI**

**컬렉션프레임웍**

**스트림**

**Json🡪외부 라이브러리(class 집합=jar)**

classpath🡪.class

JTable

–View Controller Model 방법론이 반영된 컴포넌트

View : JTable은 껍데기에 불과함

Model : 개발자가 작성하면 됨.

TableModel : jtable에게 보여질 정보를 제공함.

스트림(흐름)

데이터의 흐름

1.방향성에 따라 –

실행중인 프로그램을 기준!!

-입력

-출력

2.처리방법

- 1 byte씩 이해 : 바이트 기반(기본 스트림)

영문자 처리, 모든 스트림 처리시 사용가능

- 2byte씩 이해 : 문자 기반 스트림

한 문자씩 입력되므로 read() 빈번하게 발생!

~Reader

~Writer

* 버퍼처리된 스트림 : bufferd~~접두어

입력이 표시되기 전까지 입력을 일으키지 않고, 메모리에 버퍼라는 임시공간에 모아둔다. 따라서 입력행위가 빈번하지 않다!

쓰레드 : 하나의 프로세스내에서 독립적으로 실행될 수 있는 세부실행단위!

멀티쓰레드!

추상클래스- 불완전한 클래스

메서드의 완성 여부!!(브레이스)

실수가 아닌 의도에 의해 정의

GameObject

Tick(); 물리량 변화

Render(Graphics g);//g는 JPanel의 것 넘겨받음