1. **객체지향 프로그래밍(OOP) 관련**

* **객체 지향 프로그래밍(OOP)에 대해 설명 하시오**

Object Oriented Programing

컴퓨터 프로그래밍 패러다임중 하나 프로그램의 순서와 자료의 흐름을 우선으로 두고 데이터와 함수를 설계하는 방식을 절차 지향 프로그래밍

장점 : 코드 재사용 용이, 유지보수가 쉬움, 대형프로젝트에 적합

단점 : 처리속도가 상대적으로 느림, 객체가 많으면 용량이 커짐, 설계시 많은 시간과 노력이 필요

데이터 함수를 우선 정의하고 자료의 흐름을 만드는 것이 객체지향프로그래밍

장점 : Data(변수)가 객체별로 독립되어 관리 되기 때문에 프로그래머에 의한 실수나 실행중 예기치 못한 상황으로 인한 오류를 줄일수 있고 프로세스의 변화에도 유연하게 대처 할 수 있다. 또한 절차 지향 프로그래밍 방식에 비해 생산성과 코드의 재사용성 증대 그리고 유지보수와 확장의 유연성이 증대된다.

* **데이터 추상화(캡슐화, 정보은닉)에 대해 설명하시오**

불필요한 정보는 숨기고 중요한 정보만을 표현함으로써 공통의 속성이나 기능을 묶어 이름을 붙이는것

* **상속에 대해 설명 하시오**

부모클래스의 속성과 기능을 그대로 이어받아 사용할 수 있게하고 기능의 일부분을 변경해야 할 경우 상속받은 자식클래스에서 해당 기능만 다시 정의 하여 사용할 수 있게 하는것

* **다형성에 대해 설명 하시오**

하나의 변수명, 함수명 등이 상황에 따라 다른 의미로 해석될 수 있는 것

* **클래스와 인스턴스에 대해 설명하시오**

클래스는 속성과 행위를 변수와 메서드로 정의하는것

인스턴스는 클래스에서 정의한 것을 토대로 실제 메모리 상에 할당 된 것

1. **JAVA의 구조**

* **JAVA Virtual Machine 에 대해 설명 하시오**

JAVA 가상 머신으로 자바 바이트 코드를 실행할 수 있는 주체

CPU나 운영체제의 종류와 무관하게 실행가능 즉, JAVA 코드를 컴파일해서 얻은 바이트 코드를 해당 운영체제가 이해할 수 있는 기계어로 바꿔 실행시켜주는 역할

* **JAVA Memory 구조는 어떻게 구분 되는가**

Method area -클래스 멤버 변수의 이름, 데이터 타입, 접근 제어자 정보같은 필드 정보와 메소드의 이름, 리턴 타입, 파라미터, 접근 제어자 정보같은 메소드 정보, Type정보(Interface인지 class인지), Constant Pool(상수 풀 : 문자 상수, 타입, 필드, 객체 참조가 저장됨), static 변수, final class 변수등이 생성되는 영역이다.

Heap area - new 키워드로 생성된 객체와 배열이 생성되는 영역이다.

메소드 영역에 로드된 클래스만 생성이 가능하고 Garbage Collector가 참조되지 않는 메모리를 확인하고 제거하는 영역이다.

Stack area - 지역 변수, 파라미터, 리턴 값, 연산에 사용되는 임시 값등이 생성되는 영역이다.

PC Register - Thread(쓰레드)가 생성될 때마다 생성되는 영역으로 Program

Counter 즉, 현재 쓰레드가 실행되는 부분의 주소와 명령을 저장하고 있는 영역이다.

(\*CPU의 레지스터와 다름)

이것을 이용해서 쓰레드를 돌아가면서 수행할 수 있게 한다.

Native method stack - 자바 외 언어로 작성된 네이티브 코드를 위한 메모리 영역이다.보통 C/C++등의 코드를 수행하기 위한 스택이다. (JNI)

* **Garbage Collector는 언제 작동되는가**

Garbage Collector(GC)는 Heap 메모리 영역에 생성(적재)된 객체들 중에 참조되지 않는 객체들을 탐색 후 제거하는 역할을 한다.GC가 역할을 하는 시간은 정확히 언제인지를 알 수 없다.

1. **JAVA의 문법**

* **기본 자료형과 참조 자료형에 대해 설명 하시오**

기본 자료형 Primitive data type = 정수형(byte, short, int, long), 실수형(float, double), 문자형(char=character), 논리형(boolean) 참조 자료형 Reference data type = 배열, 클래스 객체, 인터페이스

* **각 접근 지정자의 접근 제한 범위에 대해 설명 하시오**

**public** 공개 정도가 가장 높고 어디에서든 자유롭게 접근할 수 있다.

**protected** 같은 패키지 내에서 접근이 가능하고 다른 패키지에서도 상속을 받은 클래스 내부에서는 사용 가능하다. public과 다른 점은 다른 패키지의 자식 클래스 외부에서는 접근할 수 없다는 것이다.

**default (package private)** 같은 패키지 내에서만 접근이 가능하다. 아무런 접근 지정자도 없을 경우 이 옵션이 자동으로 적용된다.

**private** 동일 클래스 내에서만 접근이 가능한 가장 낮은 단계의 자유도를 갖는다.

* **static과 final에 대해 설명 하시오**

클래스를 설계할 때, 멤버변수 중 모든 인스턴스에 공통적으로 사용해야하는 것에 static을 붙인다. 인스턴스들이 공통적으로 같은 값이 유지되어야 하는 경우 static을 붙인다. static이 붙은 멤버변수는 인스턴스를 생성하지 않아도 사용할 수 있다.

static이 붙은 멤버변수(클래스변수)는 클래스가 메모리에 올라갈때 이미 자동적으로 생성되기 때문이다. static이 붙은 메서드(함수)에서는 인스턴스 변수를 사용할 수 없다. static이 붙은 메서드(클래스메서드)를 호출할 때 인스턴스가 생성되어있을수도 그렇지 않을 수도 있어서 static이 붙은 메서드에서 인스턴스변수의 사용을 허용하지 않는다.

클래스 설계시 static의 사용지침

먼저 클래스의 멤버변수중 모든 인스턴스에 공통된 값을 유지해야하는 것이 있는지

살펴보고 있으면, static을 붙여준다. 작성한 메서드 중에서 인스턴스 변수를 사용하지 않는 메서드에 대해서 static을 붙일 것을 고려한다.

일반적으로 인스턴스변수와 관련된 작업을 하는 메서드는 인스턴스메서드(static이 안붙은 메서드)이고 static변수(클래스변수)와 관련된 작업을 하는 메서드는 클래스메서드static이 붙은 메서드)라고 보면 된다.

요약

마법같은 단어 HeapMemry에 생성하지않고 사용할 수 있도록 해줌

계속 사용하게 되면 Memory를 차지하게됨

공유의 개념 = static 으로 설정하면 같은 곳의 메모리 주소만을 바라보기 때문에 static 변수의 값을 공유하게 되는 것

final

final이 클래스 이름 앞에 사용되면 클래스를 상속받을 수 없음 을 지정한다.

* **List, Set, Map에 대해 설명 하시오**

list

순서가 있는 데이터의 집합, 데이터의 중복을 허용한다.

--> 데이터를 add하면 앞에서 부터 순차적(순서대로)으로 데이터가 들어간다.

       그래서 각각의 저장되어 있는 공간들은 고유한 index를 갖는다.

set

순서를 유지하지 않는 데이터의 집합, 데이터의 중복을 허용하지 않는다.

--> 집합이다. 데이터가 순서와는 상관없이 add된다. 중복되지 않는다.

map

키와 값의 쌍으로 이루어진 데이터의 집합. 순서는 유지되지 않으며, 키는 중복을 허용하지 않고, 값을 중복을 허용한다.

* **Abstract Class(추상 클래스)와 Interface에 대해 설명 하시오**

Class 내에 일부 method가 구현되지 않고 interface만을 가진 형태의 method로 포함 되어 있을때 그 Class를 AbstractClass라고 부른다.

특징

abstract 한 method가 하나 이상 포함되어 있으면 해당 Class는 Abstract Class로 선언되어야 한다.

abstract한 method가 없어도 Abstract Class로 선언할 수 없다.

new keyword를 이용해서 object를 생성할 수 없다.

Abstract Class 상속받는 Calss는 반드시 Abstract Class의 모든 Abstract Method를 overriding해야한다.

장점

반드시 overriding해야 한다는 강제성을 부여한다.

Class의 일관성을 줄 수 있다.

단점

수정단계에서 번거러움이 많아지면서 생산성이 떨어진다.

Interface

100% Abstract Method로 이루어져 있음

여러번 상속이 가능함

Class가 아니라 Interface로 존재함

implements로 다른 Class에 상속 시킬 수 있음

장점

서로의 기능 구현 방법에 상관없이 작업할 수 있다. => 규격을 제공

Method만 구현 된다면 각 다른 Class에서는 호출만 하면 작성이 가능 동시 작업이 가능

Abstract Class와 차이점

다중 상속이 가능

다중 상속에 대한 것을 구현

Children Class 간 공통점이 많다면 Abstract Class로 만드는것이 좋음

Interface로 만들면 getter, setter를 계속해서 만들어 줘야함

Interface는 Childre Class간 공통점이 적을때 사용하는 것이

* **Overload와 Override에 대해 설명 하시오**

**오버로딩 (Overloading)**

오버로딩(Overloading)은 메소드 오버로딩과 생성자 오버로딩이 있다. 하지만 둘다 같은 개념이다. 같은 이름의 함수를 여러 개 정의하고, 매개변수의 유형과 개수를 다르게 하여 다양한 유형의 호출에 응답하게 한다.

**오버라이딩 (Overriding)**

상위 클래스가 가지고 있는 멤버변수가 하위 클래스로 상속되는 것처럼 상위 클래스가 가지고 있는 메소드도 하위 클래스로 상속되어 하위 클래스에서 사용할 수 있다. 하지만, 하위 클래스에서 메소드를 재정의해서 사용할 수 있다.상속 관계에 있는 클래스 간에 같은 이름의 메소드를 정의하는 기술을 오버라이딩(Overriding) 이라고 한다.

* **Generic에 대해 설명 하시오**

클래스 내부에서 사용될 자료형을 지정하는 것이다. 대표적인 적용 예는 ArrayList가 있다.

* **Wrapper Class에 대해 설명 하시오**

모든 숫자와 관련된 wrapper 클래스(Byte. Short, Integer, Long, Float, Double, BigInteger, BigDecimal)들은 모두 Number 클래스의 자손 이다.

기본형(primitive type) 변수도 때로는 객체로 다루어져야 하는 경우가 있다.

**1. 매개변수로 객체가 요구 될때.**

**2. 기본형 값이 아닌 객체로 저장해야 할 때.**

**3. 객체간의 비교가 필요할 때. 등등**

이 때 사용되는 것이 wrapper class 이다.

1. **Spring/Web 개발 관련**

* **관점 지향 프로그래밍(AOP)에 대해 설명 하시오**

Spring은 핵심적인 비즈니스 로직과 관련이 없으나 여러 곳에서 공통적으로 쓰이는 기능들을 분리(공통 관심사를 분리)하여 개발하고 실행 시에 서로 조합 할 수 있는 AOP를 지원 -> 코드를 단순하고 깔끔하게 작성 (반복 적인 코드의 제거)

횡단 관심을 수행하는 코드(Logging, Security, Transaction 등)는 aspect라는 특별한 객체로 모듈화 weaving이라는 작업을 통해 모듈화한 코드를 핵심 기능에 끼워넣을 수 있음

* **의존성 주입(DI)에 대해 설명 하시오**

의존성 주입 -> "미리 찜해 놓음" 으로 해석

Dependency : 하나의 객체가 다른 객체없이 제대로 된 역할을 할 수 없음을 의미

Injection : 말 그대로 외부에서 '밀어 넣는 것'

두가지 방식은 필요한 객체를 얻기 위해서 주체가 능동적인지 수동적인지에 대한 문제

'어떤 객체가 필요한 객체를 외부에서 밀어 넣는다.' -> '왜 외부에서 객체를 주입하는 방식' 을 사용하는가

'의존성 주입' 방식을 사용하려면 추가적인 하나의 존재가 필요하고 이 존재는 의존성이 필요한 객체를 찾아서 '주입'하는 역할 을 담당

따라서 Spring개발자들은 기존의 프로그래밍과 달리 객체와 개체를 분리해서 생성하고, 이러한 객체들을 엮는(wiring) 작업을 하는형태

이렇게 개발된 Spring에서 ApplicationContext가 관리하는 객체들을 'Bean'이라는 용어로 부름

Bean과 Bean 사이의 의존관계를 처리하는 방식으로 XML 설정, 어노테이션 설정, Java 설정 방식을 이용할 수 있음

* **Servlet과 JSP에 대해 설명 하시오**

Servlet

웹프로그래밍에서 클라이언트의 요청을 처리하고 그 결과를 다시 클라이언트에게

전송하는 Servlet 클래스의 구현 규칙을 지킨 자바 프로그래밍 기술

Servlet이란 Java를 사용하여 웹을 만들기 위해 필요한 기술

클라이언트가 어떠한 요청을 하면 그에 대한 결과를 다시 전송필요 -> 해당 역할을 하는 Java 프로그램

Servlet은 Java로 구현 된 CGI

Servlet 특징

클라이언트의 요청에 대해 동적으로 작동하는 웹 어플리케이션 컴포넌트

html을 사용하여 요청에 응답, Java Thread를 이용하여 동작, MVC 패턴에서 Controller로 이용,

HTTP 프로토콜 서비스를 지원하는 javax.servlet.http.HttpServlet 클래스를 상속 UDP보다 속도가 느림,

HTML 변경 시 Servlet을 재컴파일해야 하는 단점, 일반적으로 웹서버는 정적인 페이지만을 제공

동적인 페이지를 제공하기 위해서 웹서버는 다른 곳에 도움을 요청하여 동적인 페이지를 작성

-> 동적인 페이지로는 임의의 이미지만을 보여주는 페이지와 같이 사용자가 요청한 시점에 페이지를 생성해서 전달해 주는 것

-> 여기서 웹서버가 동적인 페이지를 제공할 수 있도록 도와주는 어플리케이션이 Servlet, 동적인 페이지를 생성하는 어플리케이션이 CGI

JSP

Java Code가 들어가 있는 HTML Code

Servlet은 Saurce Code드 속에 HTML Code가 들어가는 형태

JSP는 이와 반대로 HTML Saurce Code 속에 Java Saurce Code가 들어가는 구조를 갖는 웹어플리케이션 프로그래밍 기술

* **GET방식과 POST방식에 대해 설명 하시오**

GET은 Idempotent, POST는 Non-idempotent하게 설계

 GET으로 서버에게 동일한 요청을 여러 번 전송하더라도 동일한 응답이 돌아와야 한다는 것을 의미

POST는 Non-idempotent하기 때문에 서버에게 동일한 요청을 여러 번 전송해도 응답은 항상 다를 수 있습니다

* **Session과 Cookie에 대해 설명 하시오**

쿠키(Cookie)

Set-Cookie, Cookie Header

쿠키는 서버가 사용자의 웹 브라우저에 저장하는 데이터라고 했습니다.

쿠키의 데이터 형태는 Key 와 Value로 구성되고 String으로만 이루어져 있습니다. 4KB 이상 저장할 수는 없습니다.

Session 쿠키 & Permanent 쿠키

위에서 살펴본 쿠키는 Session 쿠키입니다. Session 쿠키란 웹 브라우저가 종료될 때 제거되는 쿠키를 말합니다.

브라우저가 종료되더라도 쿠키를 유지하고 싶으면 Permanent를 이용하면 됩니다.

쿠키를 생성할 때 Expires 혹은 Max-Age 옵션을 추가하면됩니다.

Session은 비밀번호와 같은 인증 정보를 쿠키에 저장하지 않고 대신에 사용자의 식별자인 JSESSIONID(session id)를 저장합니다.

서버에는 인증 정보와 더불어 이 ID에 해당하는 로그인 상태, 마지막 로그인 시간, 닉네임, 만료기한 등의 정보를 저장합니다.

보안상 서버는 사용자의 개인 컴퓨터보다는 훨씬 안전하기 때문에 인증에 Session을 이용

1. **컴퓨터 과학/개발 방법론**

* **Thread와 Process에 대해 설명 하시오**

프로세스(process)란?

실행 중인 프로그램

프로그램 자체 메모리 상에서 실행되고 있는 작업 단위

스레드(thread)란?

프로세스 안에서 실행되는 흐름의 단위

프로세스의 업무를 여러개로 나누면 스레드

* **API와 Framework에 대해 설명 하시오**

Framework

'프레임워크는 라이브러리와 달리 애플리케이션의 틀과 구조를 결정할 뿐 아니라, 그 위에 개발된 개발자의 코드를 제어한다.

프레임워크는 구체적이며 확장 가능한 기반 코드를 가지고 있으며, 설계자가 의도하는 여러 디자인 패턴의 집합으로 구성되어 있다.'

API

미리 만든 함수들을 어딘가에 저장해 놓고, 개발자들은 필요시 가져와 편리하게 사용할 수 있는 것이다.

라이브러리가 함수들의 묶음 그 자체라면,

API는 그것들에 대한 명세까지 포함한다고 볼 수 있다.

* **Singleton pattern에 대해 설명 하시오**

해당 클래스의 인스턴스가 하나만 만들어지고, 어디서든지 그 인스턴스에 접근할 수 있도록 하기 위한 패턴.

클래스에서 자신의 단 하나뿐인 인스턴스를 관리하도록 만든다.

다른 어떤클래스에서도 자신의 인스턴스를 추가로 만들지 못하도록 해야한다.

* **MVC Pattern에 대해 설명 하시오**

디자인 패턴이란 프로그램이나 어떤 특정한 것을 개발하는 중에 발생했던 문제점들을 정리해서 상황에 따라 간편하게 적용해서 쓸 수 있는 것을 정리하여 특정한 "규약"을 통해 쉽게 쓸 수 있는 형태로 만든 것을 말합니다.

1. **프로젝트 관련**

* **본인이 참여한 프로젝트에 대해 소개 하시오**
* **프로젝트에서 본인이 맡은 부분에 대해 설명하시오**
* **프로젝트를 진행하면서 어떤점이 어려웠는지?(해결 방법까지)**

1. **기타**

* **어떤 프로그래밍 언어를 사용할 수 있는가**
* **어떤 버전 관리 프로그램을 사용할 수 있는가**
* **가장 자신있는 기술은 무엇인가**
* **어디서 IT관련 최신 정보를 얻는가**
* **만약 스스로 해결이 불가능한 문제가 발생하면 어떻게 할것인가**
* **단기간 내에 언어와 Framework를 학습해야 한다면 어떻게 할 것인가**

DDL, DML, DCL의 종류

DDL : Data Definition Language (Create, Alter, Drop, Rename, Truncate)

테이블과 같은 데이터 구조를 정의 하는데 사용되는 명령어들로 데이터구조와 관려된 명령어들을 말함

DML : Data MAnipulaton Language (Select, Insert, Update, Delete)

데이터베이스에 들어있는 데이터를 조회하거나 검색하기 위한 명령어를 말하고

또한 데이터에 변형을 가하는 종류의 명령어를 가리킴

DCL : Data Control Language (Grant, Revoke)

데이터베이스에 접근하고 객체들을 사용하도록 권한을 주고 회수하는 명령어들을 말함

AJAX란?

javascript 를 이용한 비동기 통신 방식, JavaScript와 XML (Asynchronous JavaScript And XML)

서버와의 통신이 이루어지는 동안, 다른 작업이 가능 / 화면 전환 불필요 통신하기 위해 XMLHttpRequest 객체를 사용

xml, text, json 등의 데이터를 이용하여 통신 가능

collection framework의 종류

배열의 단점을 보완한 데이터 군을 저장하는 클래스들을 표준화한 설계이다.

다수의 데이터를 쉽게 처리할 수 있는 방법을 제공하는 클래스들로 구성된다.

컬렉션(collection) : 다수의 데이터, 데이터 그룹을 의미한다.

프레임워크(framework) : 표준화, 정형화된 체게적인 프로그래밍 방식이다.

컬렉션 클래스(collection class) : 다수의 데이터를 저장할 수 있는 클래스이다.

ArrayList와 List의 차이점

데이터가 많아지면 그룹 관리의 필요성이 생긴다. 이럴 때 프로그래밍에서 사용하는 것이 배열

여러 데이터를 하나의 이름으로 그룹핑해서 관리 하기 위한 자료구조

배열을 이용하면 하나의 변수에 여러 정보를 담을 수 있고, 반복문과 결합 하면 많은 정보도 효율적으로 처리할 수 있다.

배열 인덱스는 값에 대한 유일무이한 식별자 (참고로 리스트에서 인덱스는 몇 번째 데이터인가 정도의 의미를 가짐)

리스트는 배열이 가지고 있는 인덱스라는 장점을 버리고 대신 빈틈없는 데이터의 적재 라는 장점을 취한 데이터 스트럭쳐

리스트 자료구조의 핵심은 엘리먼트들 간의 순서. 따라서 리스트를 다른 말로는 시퀀스(sequence) 라고도 부른다.

즉 순서가 있는 데이터의 모임이 리스트이다.

리스트에서 인덱스는 몇 번째 데이터인가 정도의 의미를 가진다.

빈 엘리먼트는 허용하지 않는다.

순차성을 보장하지 못하기 때문에 spacial locality 보장이 되지 않아서 cash hit가 어렵다.

데이터 갯수가 확실하게 정해져 있고, 자주 사용된다면 array가 더 효율적이다.

리스트에 대한 효용은 어떤 언어를 사용하느냐에 따라서 달라진다.

특히 많은 언어가 리스트를 기본적으로 지원하기 때문에 리스트를 직접 구현하는 경우는 줄고 있다.

spring framework의 구동순서

웹 어플리케이션이 실행되면 Tomcat(WAS)에 의해 web.xml이 Loading

web.xml에 등록되어 있는 ContextLoaderListener (Java Class) 생성

생성된 ContextLoaderListener는 root-context.xml을 Loading

root-context.xml에 등록되어 있는 Spring Container가 구동

클라이언트로부터 웹 어플리케이션 요청이 옴

DispatcherServlet이 생성됨

DispatcherServlet은 servlet-context.xml을 Loading

클라이언트가 해당 어플리케이션에 접근하면 접근한 URL 요청을 DispatcherServlet이 가로챔

RequestMappingHandlerMapping이 해당 요청을 처리할 컨트롤러를 찾음

DefaultListableBeanFactory 가 mainContoller 를 쓰면 된다고 알려줌

RequestResponseBodyMethodProcessor가 servlet-context.xml 에 선언해놓은 MessageConverter을 이용하여 요청 바디(파라미터)를 읽음

DataSourceTransactionManager 로 DB 접속

응답결과 받아다 파라미터에 써줌

컨트롤러에서 view를 리턴하면 ViewResolver가 먼저 받아 해당 view가 존재하는지 검색

DispatcherServlet은 ViewResolver를 통해 접두사와 접미사가 붙은 JSP 파일의 이름과 경로를 리턴받아 최종적으로 JSP를 실행.

string의 메소드의 종류를 아는대로 말하라

**startWith:** 문자열이 지정한 문자로 시작하는지 판단 같으면 true반환 아니면 false를 반환한다.(대소문자구별)

**endWith:**문자열 마지막에 지정한 문자가 있는지를 판단후 있으면 true, 없으면 false를 반환한다.(대소문자구별)

**equals:**두개의 String에 값만을 비교해서 같으면 true, 다르면 false를 반환한다.(대소비교)

**indexOf:**지정한 문자가 문자열에 몇번째에 있는지를 반환한다.

**lastindexOf:**문자열에 지정한 문자가 마지막몇번째에 있는 int를 반환한다.

**length:**문자열의 길이를 반환한다.

**replace:**문자열에 지정한 문자" "가 있으면 새로 지정한 문자" "로 바꿔서 출력한다.

**replaceAll:**정규표현식을 지정한 문자로 바꿔서 출력한다.

**split:**지정한 문자로 문자열을 나눌수 있다.(배열로 반환)

**substring:**문자열에 지정한 범위에 속하는 문자열을 반환한다.(시작범위에 값은 포함하고, 끝나는 범위에 값은 포함하지않는다.)

**toLowerCase:** 문자열에 대문자를 소문자로 변환한다.

**toUpperCase:**문자열에 소문자를 대문자로 변환한다.

**toString:**문자열을 그대로 반환해준다.

**trim:**문자열에 공백을 없에준다.

**valueOf:**지정한 개체의 원시 값을 반환

**compareTo:**두개의 String를 앞에서부터 순사적으로 비교하다가 틀린부분이 있으면 비교하는 String에 캐릭터값을 반환한다.(대소문자를 구별)

**contains:**두개의 String을 비교해서 비교대상 String을 포함하고 있으면true, 다르면 false를 반환한다.

**charAt:**지정한 index번째에 문자를 반환한다.

**concat:**문자와 문자를 결합해준다.

**format:**서식문자열을 이용해서 서식화된 문자열을 반환한다.

**matches:**지정한 정규 표현과 일치 할때 true를 반환한다.

**replaceFirst:**문자열에 지정한 문자" "가 있으면 첫번째만 새로지정한 문자" "로 바꿔서 출력한다.

반복문에 대해서 아는대로 이야기해라

for, while 두가지 문장을 주로 사용 for문은 조건을 주고 조건식이 true일때 반복되어 실행함

while문은 조건을 만족할 동안 반복시킴 반복문 안에 증감식 또는 while문을 빠져나가는 식을 적어 두어야 한다.

조건문에 대해서 아는대로 이야기해라

조건문이랑 조건에 따라 코드의 실행 흐름을 다르게 동작하도록 제어 하는것

조건문은 if, switch문으로 구분 할 수 있고 if문은 if, if else, else if 로 구분 함

if문은 조건식이 참일때 내부 문장이 실행되며 거짓이면 문장을 벗어남

if else문은 if문의 조건식이 참인 경우에만 실행문장이 실행되면 거짓인경우 문장을 벗어남

switch 하나의 식이 값에 따라 여러 방향으로 분기하고자 하는 경우 사용

case문을 사용하여 원하는 값으로 분기할 수 있음

JDBC 구동순서

JDBC 드라이버 로드

Connection 객체 생성 (데이터베이스와 연결)

Statement 객체 또는 PrparedStatement 객체 생성 (SQL문 실행 객체 생성)

SQL 문장 실행

ResultSet에 저장된 데이터 추출 (SELECT 쿼리를 사용할 때만 사용)

DB 연결 해제 (생성된 객체의 자원 반환)

Stream이란?

입출력을 뜻한다.

데이터가 있던 장소에서 다른곳으로 흘러가는것을 stream으로 표현

단반향으로 진행된다.

In&Out put Stream

Input Stream 대상의 입장에서 data를 받으면 Input Stream

Output Stream 대상의 입장에서 data를 보내면 Output Stream

System.out.print : 표준 출력, Output Stream

특징

스트림은 데이터 소스를 변경하지 않는다.

스트림은 데이터 소스로 부터 데이터를 읽기만할 뿐, 데이터 소스를 변경하지 않는다.

필요하다면, 정렬된 결과를 컬렉션이나 배열에 담아서 반환

스트림은 일회용이다.

스트림은 Iterator처럼 일회용이다. Iterator로 컬렉션의 요소를 모두 읽고 나면 다시 사용할 수 없는 것처럼,

스트림도 한번 사용하면 닫혀서 다시 사용할 수 없다. 필요하다면 스트림을 다시 생성해야함

스트림은 작업을 내부 반복으로 처리한다.

스트림을 이용한 작업이 간결할 수 있는 비결중의 하나가 바로 '내부 반복'이다.

내부 반복이라는 것은 반복문을 메서드의 내부에 숨길 수 있다는 것을 의미

LNUX와 UNIX에 대해 알고 있나

|  |  |  |
| --- | --- | --- |
|  | **Linux(리눅스)** | **Unix(유닉스)** |
| **설명** | 리눅스는 오픈소스이며 무료OS이다. | 유닉스는 대학, 회사, 큰  기업에서 주로 선호하는 OS이다. |
| **비용** | 리눅스는 무료이다. 따라서 잡지나 책을 동해서도 배포될 수 있다.  유료버전의 리눅스도 있지만 대부분 윈도우보다는 저렴하다. | 유닉스는 판매 회사별로 가격이 다르다. |
| **사용자** | 모든사람. 집에서 사용하는 사람에서부터 개발자 그리고 컴퓨터 열혈팬 같은 사람들 모두. | 유닉스 운영체제는 주로 메인프레임, 서버 그리고  모든사람을 위해 설계된 OSX를 제외한 워크스테이션을 위해 개발되었다. 유닉스 환경과 클라이언트-서버 프로그램 모델은 인터넷의 개발의 필수 요소이다. |
| **제조사** | 리눅스 커널은 커뮤니티에 의해 개발되었다. 리누스 토발즈(Linus Benedict Torvalds, 리눅스개발자)가 이를 감독하고 있다. | Solaris (Oracle), AIX (IBM) 그리고 HP-UX (Hewlett Packard)가 3개의 큰 제조사이다. Apple의 OSX도 있다. |
| **가격** | 무료. 그러나 유료로 지원되는 것이 있다. | 몇몇은 개발용도로 무료이다(Solaris)그러나 유료로 지원되는 것이 있다. |
| **사용량** | 리눅스는 모바일폰, 태블랫 컴퓨터 그리고 비디오게임콘솔에서부터 메인프레임이나 슈퍼컴퓨터까지 다양한 컴퓨터 하드웨어에 설치가 가능하다. | 유닉스는 인터넷서버, 워크스테이션과 PC들에 사용된다. 다수의 금융인프라 그리고 많은 24x365 고 가용솔루션의 뼈대를 이루는 인프라에 사용된다. (백본). |
| **프로세서** | 다양한 종류가 있다. | x86/x64, Sparc, Power, Itanium, PA-RISC, PowerPC 등 |
| **개발과 배포** | 리눅스는 오픈소스로 개발되었다.  리눅스 코드의 공유와 공동작업과 각 포럼 등의 특성을 통해 개발되었다. 그리고 다양한 판매사에 의해 배포된다. | 유닉스 시스템은 다양한 제조사로 나누어진다. (대부분 AT&T 그리고 다양한 상업적 판매사와 비영리 단체에 의해 개발됨) |
| **아키텍처** | 원래 인텔의 x86 하드웨어를 위해 개발되었으며, ARM을 포함한 24개 이상의 CPU 타입들도 지원가능하다. | PA-RISC 그리고 Itanium 기계에 사용가능하다. Solaris는 x86/x64기본시스템에도 사용가능하다. OSX는 PowerPC(10.0-10.5)/x86(10.4)/x64(10.5-10.8)에 가능 |
| **GUI** | 리눅스는 기본적으로 KDE와 Gnome의 두 가지 GUI를 제공한다. 그 외에 LXDE, Xfce, Unity, Mate, twm 등도 있음. | 초기에 유닉스는 커맨드기반의 OS였다. 그러나 후에 GUI가 생성되어 공통 데스크톱 환경으로 불렸다. 대부분 배포는 현재 Gnome을 탑재하고 있다. |
| **파일 시스템 지원** | Ext2, Ext3, Ext4, Jfs, ReiserFS, Xfs, Btrfs, FAT, FAT32, NTFS | jfs, gpfs, hfs, hfs+, ufs, xfs, zfs format |
| **텍스트모드 인터페이스** | BASH (Bourne Again SHell)는 리눅스의 기본 쉘이다. BASH는 다수의 명령어를 지원한다. | 원래는 Bourne Shell이다. 현재는 BASH, Korn & C를 포함한 다른 많은많은 쉘들과 호환된다. |
| **보안** | 리눅스는 지금까지 약 60-100개의 바이러스 명단이 있다. 이 중 요즘은 아무도 활발하게 퍼지고 있지 않다. | 유닉스의 바이러스는 지금까지 85-120정도가 보고되었다. |
| **위협 탐지와 해결** | 리눅스의 경우 위협탐지와 해결이 매우 빠르다. 리눅스가 주로 커뮤니티 기반이기 때문이다. 그리고 리눅스 사용자가 위협을 커뮤니티에 올리면, 세계의 다른 지역의 개발자들이 해결을 위한 작업을 시작한다. | 유닉스의 본래 독점적인 성격으로 인해, 사용자는 적절한 버그수정 패치를 얻기 위해선 한동안 기다려야 했다. 그러나 이것은 일반적이진 않다. |
| **첫 시작** | 1992년 MINIX(유닉스와 비슷한 시스템)에 의해 영감을 받은 리누스 토발즈가. GUI, Drivers 등의 많은 특성들을 추가한 후에,지금의 리눅스가 된OS의 프레임워크를 개발하였다 리눅스 커널은 1991년9월 17일 출시되었다. | 1969년, Bell연구소의 AT&T직원들과 데니스리치에 의해 개발되었다. 유닉스는 C언어로 작성되었으며 휴대용제품, 시분할 환경의 다중작업과 다중사용자 시스템을 위해 설계되었다. |
| **예시** | Ubuntu, Fedora, Red Hat, Debian, Archlinux, Android 등 | OS X, Solaris, 모든 리눅스 |

json이란 무엇인가

JSON (JavaScript Object Notation)

경량의 DATA-교환 형식 : 사람이 읽고 쓰기에 용이, 기계가 분석하고 생성함에도 용이

언어로 부터 독립적이지만 프로그래머들에게 친숙한 관습을 사용하는 텍스트 형식

두개의 기본 구조

name/value 형태의 쌍으로 collection 타입 다양한 언어들에서, 이는 object, record, struct(구조체), dictionary, hash table, 키가 있는 list, 또는 연상배열로서 실현

값들의 순서화된 리스트. 대부분의 언어들에서, 이는 array, vector, list, 또는 sequence로서 실현

JSON 형식

object는 name/value 쌍들의 비순서화된 SET

object는 {좌 중괄호로 시작하고 }우 중괄호로 끝내어 표현

각 name 뒤에 :colon을 붙이고 ,comma로 name/value 쌍들 간을 구분

array은 값들의 순서화된 collection

array는 [left bracket로 시작해서 ]right bracket로 끝내어 표현

,comma로 array의 값들을 구분

value는 큰따옴표안에 string, number ,true ,false , null, object ,array이 올수 있다. 이러한 구조들을 포함

string은 큰따옴표안에 둘러 싸인 zero 이상 Unicode 문자들의 조합 쌍다옴표안에 감싸지며,backslash escape가 적용

하나의 문자(character)도 하나의 문자열(character string)로서 표현 string은 C 또는 Java 문자열 처럼 매우 많이 비슷

number는 8진수와 16진수 형식을 사용하지 않는것을 제외하면 C와 Java number 처럼 매우 많이 비슷

토근들의 어떤 쌍 사이에 공백을 삽입할수 있다. 드물게 encode된 세부 항목을 제외하면, 이렇게 설명된 JSON의 형식은 완벽하게 그 언어를 설명

스레드의 개념과 스레드를 동기화하는 메소드를 아는대로 말해주세요

wait(), notify(), notifyAll()

동기화된 스레드(보호구역에서 실행하고있는 스레드) 혼자서는 동기화 블록(보호구역 안)에서 다른 스레드에게 제어권을 넘기지 못합니다.

동기화된 블록에서 스레드간의 통신(제어권을 넘김)하기 위해서는 wait(), notify(), notifyAll() 메소드를 사용해야 합니다.

메소드를 사용할 때 주의해야할 점은 synchronized 블록에서만 의미가 있습니다.

Synchronized 블록이 아닌 경우에 사용할 경우 java.lang.illegalMonitorStateException이 발생합니다.

wait() 메소드는 어떤 객체에 대해 스레드를 대기하게 만듭니다.

notify() 메소드는 객체에 대해 대기중인 스레드가 있을 경우 우선순위가 높은 스레드 하나만을 깨웁니다.

notifyAll() 메소드는 대기중인 스레드 전부를 깨웁니다.