

1. Spring

① Spring MVC

: 초기기 Spring, maven build 사용

② 정부전자

: 정부기업에서 사용하는 Spring, Maven build 사용

③ Spring boot

: 대부분의 설정이 생략됨, 요즘에 많이 사용, gradle build를 주로 사용.

2. build 를

① 컴파일 (실행)의 기능이 있다.

② 라이브러리를 관리하는 기능이 있다.

③ 배포를 하는 기능이 있다.

→ 내가 사용하는 build 를 **maven**.

: 자바에서 JDBC를 코드하기 위해 build path → 라이브러리 → External 하는 표정을 알아야 해요.

3. Pom.xml

: maven을 환경설정하는 곳.

구성

① 프로젝트 정보

<groupId> www.silver > 도메인

<artifactId> hom > 프로젝트 이름

<packaging> war > 배포형식

② 변수

<properties> 변수

③ 의존관계

<dependency> > 필요로하는 (의존하는) 라이브러리 기능을 build 한다.

* 필요한 라이브러리 검색 : maven repository — (많이 쓰는 것 선택)

보통 아래에 놓음. 공통적으로 사용하는 라이브러리를 위쪽에.

* framework : 개방에 대한 성격을 미리 만들어 놓은 를.

장점 ① 팀 개발이 수월하다.

② 유지보수가 수월하다. (다른 사람도 코드 읽기 쉬운)

<dependency> 의 핵심 라이브러리는 Spring - WebMVC이다.

④ 컴파일

<build>

1. Spring의 틈 - MVC (Web 개발은 모두 MVC 디자인 패턴을 사용)

클라이언트 : web browser

서버 : WAS : Tomcat 8.5 (WAS: Web Application Server)

→ WAS는 동적권리조차 정적권리조 모두 다룬다

→ NAS와 반대로 Web server은 정적 컨텐츠만 다룬다.

下

① Filter (언어설정)

② Interceptor

③ Spring - MVC

• Controller

: URI의 식별자로 클라이언트 서비스 처리

@Controller

@ RequestMapping

: 클라이언트의 URI 요청을 어떤식으로 처리할지 (value = "작별자", method =)

@RequestParam

: 피라미터 (form의 name)을 저장한다. (@ — ("Id") String id)

- Model

DB 관련 서비스 처리

① Service - DB관련

② Dao - DB CRUD 작업

◦ View

: 클라이언트에게 보여줄 화면을 만든다.

∴ JSP 파일이 html 을 서로 변환된다.

2. URI

http:// 도메인주소 : 포트번호 / contextpath / 서브자 / 파라미터.

커피 아마 1잔

3. 어려코드

① 404 : 찾을수없다 (@RequestMapping 이상한 헤더이거나)

② 405 : 식별자 찾았는데 코드가 이상하다.

핵심 키워드.

- ① Spring MVC Project
- ② build
- ③ maven
- ④ 배포
- ⑤ war
- ⑥ Pom.xml
- ⑦ dependency
- ⑧ MVC 디자인 패턴
- ⑨ model / view / controller
- ⑩ contextPath
- ⑪ 요청시 메서드 - get / Post , 요청시 파라미터 -
- ⑫ WAS

1. Maven Project 구조.

① Src / main / Java

- : Java 소스 파일 (.java)
- : 컨트롤러, dao, vo (dto와 비슷), utils (부가적 기능)

② Src / main / resources

- : 부가적으로 필요한 설정 파일
- : mapper (쿼리만 작성하면 된다.)

③ Src / test / Java

- : 툴별 코드를 만들지 않고 test 할 수 있다. → JUnit 라이브러리가 실행시 사용된다.

④ Maven Dependencies

- : maven 라이브러리 → spring에서 사용하고 싶은 라이브러리를 pom.xml에 설정한다.

⑤ Src / main / webapp / resources

- : 정적 컨텐츠 (html, css, js, 그림 등)

⑥ Spring 풀더

- : Spring 환경 설정, Bean 설정 (Servlet-context.xml)

* Bean: spring은 필요한 객체를 직접 만들지 않고 spring에 객체의 주소를 달라고 한다 (주입)

⑦ views / jsp

- : view 파일

⑧ web.xml

- : MVC 프로젝트 설정, Filter 설정 (한글필터) → 제일 먼저 실행하는 xml

* DispatcherServlet 객체 MVC 프로젝트 동작시킨다.

↳ 컨트롤러 설정, Model 작업, View 지정, 응답

⑨ pom.xml

- : maven project 환경 설정 (프로젝트 버전, dependency 설정 등)

* dependency 추가방법 : "maven repository 라이브러리" 검색 후 추가

1. JUNIT

: 단위테스트를 가능하게 하는 라이브러리이다.

: main 메서드 없이 실행 가능.

단위테스트

① ② Test ② @Before (먼저실행) ③ ④ After (마지막)

2. STATIC

: 공용변수

: 객체를 생성하지 않아도 어디서든 접근 가능. 누구나 사용해도 고려해야 한다 (보안↓)

3. FINAL

: 상수. (수정이 불가능하다)

4. 메서드 overload

: 메서드 이름은 같지만 매개변수 개수, 자료형, 리턴값이 다를 수 있다.

: 메서드 헤출할 때 이름 오류가 된다.

예시

[
Public int add(int a, int b) { return a + b; }
Public int add(int a, int b, int c) { return a + b + c; }]

5. 메서드 override

: 부모·인터페이스에 정의된 메서드를 상속·구현 받아서 재정의 한다.

6. 추상메서드

: abstract

: 추상 메서드는 상속받은 하위 클래스에서 반드시 재정의 해야 한다.

: 인터페이스 메서드 → 모두 추상메서드 → 인터페이스를 구현받은 모든 class는 재정의해야 한다.

: {} 가 없는 메서드.

: 추상 class
↳ 추상 메서드를 한 개 이상 보유하는 클래스.
↳ new 연산자로 객체를 만들 수 없다.

예시.

Public abstract void abc();

Maven Project 구조

앙기! maven 프로젝트 외에 mvc를 찾기.

Package Explor... ☰

Servers

> www [www main]

> src/main/java
 > www.silver.hom
 > HomeController.java
 > src/main/resources
 > src/test/java
 > src/test/resources
 > Maven Dependencies
 > JRE System Library [JavaSE-1.8]
 > src
 > main
 > webapp
 > resources

정적 컨텐츠 : html, css, js, 그림 파일 등
 > WEB-INF
 > classes
 > spring
 > views
 > home.jsp

 > web.xml

 > test
 > target
 > pom.xml

 > Dispat... servlet 폴더 : Spring은 웹환경 기본설정이 필요하니깐
 > MVC 폴더 : 동작시작

 > Maven project 환경설정 - 버전, 디펜던시 설정
 > pom.xml

* maven repository 라이브러리 이용 정책. 추가 + maven > update project

java 소스파일 : 컨트롤러, dao, vo, utils * VO : dao와 비슷
부가적으로 필요한 [설정] 파일 : mapper : 커리언 작성해면된다.
* Test code : 원본코드를 전등이 알고 테스트하기 위해 (단위로 테스트)
Maven 라이브러리 : pom.xml 사용설정 한다.
Spring에서 사용하고 싶은 라이브러리를 추가한다.

* Bean(부록과 동일한 설정으로bean이라는 이름의 주제로 단축문자를 쓰자!)
Spring환경설정, Bean설정 * Bean : Spring은 웹환경 기본설정이 필요하니깐
Servlet-context.xml > viewResolver
view Files :

MVC프로젝트 설정, Filter 설정-제일먼저 실행

* DispatcherServlet 폴더 : 모든 요청은 model 적용 처리. View 계획. 응답

Maven project 환경설정 - 버전, 디펜던시 설정

1. 객체지향 언어 특징

- ① 추상화 : 객체를 정의 (객체 : class, 구체적이지 않음)
- ② 캡슐화 : 접근제어자 → 은닉화 (외부에서 자원을 알 수 없게 숨김) ex) DTO - getter, setter
- ③ 상속 : Super class, Sub class, Super & this, 추상 class, 추상메서드, override
- ④ 다형성 : B b = new A(); 조건 1. 상속, 2. 인터페이스.
조건 1. 상속, 2. 인터페이스.
다양한 형태의 객체의 주소를 저장할 수 있다.

2. 할 것 (고통수단)

<시스템개발>

호텔에서 예약한 후 푹입서비스 선택 가능. 예약자는 버스 or 택시 선택할 수 있다.

- BUS 멤버변수 : 날짜, 일정수, 예약자
액세스 : 타다, 내리다.
 - TAXI 멤버변수 : 날짜, 일정수, 예약자
액세스 : 타다, 내리다, 한립인자체로
- 국적을 추가하고 싶다 ① 멤버변수, 액세스 중복, 유지보수 힘들음.
② 택시, 버스를 여러개의 객체로 생성해야 한다
- ↓ 해결

[공통된 것 : Super (부모) 으로 합대면 Super class 만 남아야 한다.
세부적인 것 : Sub (자식)

* Java에서 최상위 class는 Object다. (특례상속)

Bus, Taxi를 ArrayList 틀로 관리한다.

ArrayList<TadaSuper>t = new ArrayList<>();

TadaSuper tempT = new BUS();

t.add(tempT);

tempT = new TAXI();

t.add(tempT);

for (TadaSuper getT : t) {

getT.info();

}

ArrayList<>에 추가.

다형성.

→ 소스 → override methods.

3. 상속

: Super class의 자식(생략변수·메서드)을 Sub class에 확장하는 개념.

Sub class는 Super class의 메서드를 재정의 (override)할 수 있다.

Super class에서 추상 메서드를 정의해둔다면

Sub class에서는 반드시 재정의 (override) 해야 하는 강제성을 가진다.

4. 디자인패턴

: 슈퍼클래스의 객체로 상속받은 객체를 컨트롤하겠다.

5. 인터페이스

: 상수와 추상메서드가 있다.

① 인터페이스는 구현하기 할 수 없다.

② 인터페이스의 메서드는 절의만 할 수 있다. 기능을 구현할 수 없다.

→ 추상메서드이다.

→ 때문에 인터페이스를 구현받은 클래스의 메서드는 반드시 override 해야한다. ☆

장점: 메서드 이름을 동일할 수 있다.

인터페이스 객체 만들기 목적, 다양성을 통한 다양한 디자인 패턴 설계

③ 인터페이스의 모든 변수는 상수이다. (수정X)

* 인터페이스 사용 이유

: 원하는 기능의 이름만 정의할 뿐니 기능 구현을 알아서해라.

24-05-20

[개발 환경 설정]

1. 개발환경 이해.

① Maven Project

: 빌드합니다.

version, dependency, 배포형식 등을 담당합니다.

② MVC 디자인패턴

: model, view, controller 기능을 담당한다 → 요청부터 응답까지.

: MVC를 올리면 web.xml, root.xml, servlet.xml이 생긴다.

2. Web.xml의 환경설정 [config 파일]

① web.xml

: Spring을 run on server하면 제일 먼저 실행된다.

: 가장 중요한 DispatcherServlet 객체가 있다.

MVC 프로젝트를 동작시키는 중요한 기능을 한다. → '요청' 응답' 담당한다.

: 필터로 먼저설정

: root-context.xml과 Servlet-context.xml을 로딩한다.

② root-context.xml

: 스프링의 환경설정을 정의한다.

: 객체생성 등 백엔드 영역

: <Bean>으로 객체 생성

③ Servlet-context.xml

: DispatcherServlet의 환경설정을 한다.

예시 - value = "/WEB-INF/views" を 통해 String만 아니라 view.jsp로 인식.

3. 객체

: 스프링에게 Bean을 통해 객체 생성을 위임한다. → IOC

: 스프링의 객체의 '생성'과 '소멸'을 관리합니다

: 개발자는 객체의 주소를 넣여 객체 사용 → 주입 받는다 → 의존 DI

: 싱글톤 (객체 1개만 생성) 을 선호한다.

: new는 일회성, new로 객체생성하면 소멸도 직접.

객체 생성 방법

① 등록

◦ Servlet.xml에 <context:component-scan base-package = " " > 추가

◦ root.xml에 <bean class = " " id = " " > 생성

② 주입

ⓐ inject → 컨테이너로부터 객체의 주소 주입받음

1. Mybatis

: Persistence Framework

→ 데이터의 영속성 (데이터가 날라가지 않는다.)

: DB와 CRUD 작업을 위한 프레임워크

기능

- ① 안정된 데이터베이스 지원 사용 가능 → 일정한 connection 유지. 개별자에게 블리주고 뺄.
- ② 데이터베이스 코드와 서비스코드가 분리

2. 의존성을 낮추기

: 주입하기

주입방법 ① 생성자 : 반드시 실행되는 메서드

② Setter : 선택에 의해 실행되는 메서드.

3. Stateless

: 서버는 클라이언트의 정보를 저장하지 않는다. *

: Web 환경. http의 특징이다. 기본값이다.

: Connection → request → 서버처리 → response → 연결종료. → 클라이언트 기억X

: Session → 서버가 클라이언트의 정보 저장. → 서버가 만든다. *

: 쿠키 → 클라이언트의 신분증. 클라이언트가 보인의 정보 저장. → 서버가 만든다.

[git 명령어]

- mkdir 디렉토리명 : 디렉토리 생성
 - cd 디렉토리명 : 디렉토리로 이동
 - git init : 현재 디렉토리에서 작업하겠다. ①
 - ls -al : 슬링파일 목록까지 보기.
 - vim 파일명 : 파일을 생성 / 편집하겠다. + i(insert) + :wq ②
 - git status : 파일상태확인 (untracked: 추가x) ③
 - git add 파일명 : git에게 이 파일 관리하라는 명령어. 추적하라. ④
 - git config --global user.name 브네임) 1번만
 - git config --global user.email 이메일) 1번만
 - git commit → vim실행 → commit message 작성 (i) → :wq /git commit -m "
 - git log : 버전확인
-
- git reset : git add 취소
 - git commit : git commit 취소
 - git reset HEAD^ :
git push origin <branch> -f] git push 취소

① 차이점확인

- git log -p : 소스사이의 차이점.

②

- git diff 차이내시지 .. 차이내시지

- git branch
 - git branch 새로운브랜치명
 - git branch -d
 - git remote add origin 서버주소 : 원격저장소에 연결
 - git push (-u) origin main(master) : github에 업로드
-
- git checkout -b 새로운브랜치 : 생성과 동시에 체크아웃.

24 - 07 - 03

[DB]

① 관계형데이터

: 관계형 데이터베이스 (RDB)

관계형인 이유 : table, 투플 속성, key 등
예시

- Oracle
- MySQL
- MariaDB

② 비관계형데이터

: 빅데이터

비관계형인 이유: table 없고 key와 value 구조만 있다.

예시

- MongoDB

★ [자료구조]

: 편하게 자료 관리 할 수 있다.

예시 ◦ List

- Map

: 코딩 테스트의 목적.

[네트워크]

IP 주소 : 자신의 컴퓨터 주소, 고유하다.

게이트웨이 : 월드로 나가는 통로의 주소. LAN과 WAN의 연결 통로.

Ping : 연결 여부 확인.

공인 IP : WAN. 도내. ex) 8.8.8.8

내장 IP : LAN ex) 192.168. ~

[OS 운영체제]

① 윈도우 - GUI (마이크로)

① 디스크 → 디렉토리 / 파일 관리

② 프로그램 추가 / 삭제

③ 네트워크 설정 조회, 간단한 테스팅

④ 방화벽 설정 (활성화, 비활성화, 특정 Port에만 Permit/deny)

⑤ 계정 관리 (사용자의 권한 관리)

② 리눅스 - TUI (명령어)

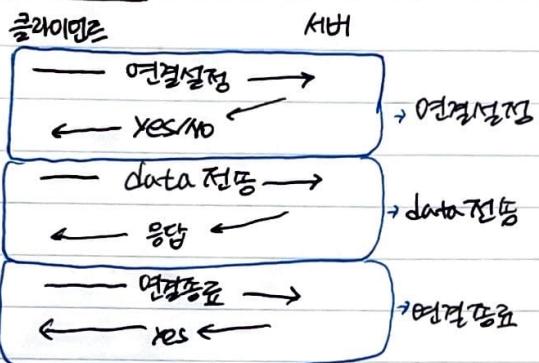
24-01-03.

1. 통신방법.

① HTTP [IP : 찾아가기 위한 주소]

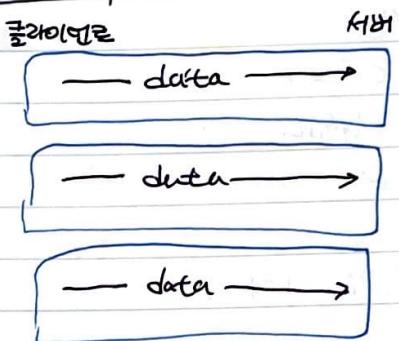
TCP / UDP : 통신방법.

TCP 방식



ex) TCP

UDP 방식



ex) 문자

속도가 더 빠름

1. 컴퓨터 구성 요소.

- ① CPU
- ② 메모리 RAM
- ③ 보조기억장치
- + 네트워크카드

2. 가상컴퓨터

: ① vm wear

② Virtual box - 리눅스 기스로 운영체제 설치하여 만들어봄.

가상 컴퓨터는 전원을 종료하면 컴퓨터도 종료됨.

클라우드는 언제 어디서나 접속 가능하다.

↳ 가상의 서버로 사용 가능.

3. 리눅스 운영체제

: 터미널 기반.

① 리눅스의 계정 이해

- 현재사용자
- root (/)

② 퍼미션

: 파일별로 접근 권한이 다르다.

: 명령어 ls -l을 통해 권한을 알 수 있다.

rwx rwx rwx → 1 1 1
User Group Other. 2^2 + 2^1 + 2^0 = 7

r: read (읽기)

x: execute (실행하기)

w: write (쓰기)

: 명령어 chmod ---

③ 명령어 종류

- su : 계정변경
- sudo : 루트권한으로 변경
- d : 디렉토리
- - : 파일
- ss -ltn : 리눅스 소켓상태 확인
- ifconfig : IP주소 확인

- apt install : 온라인 상에서 프로그램 다운
- apt remove : 설치된 프로그램 삭제
- apt update : 프로그램 최신상태로 업데이트

④ 리눅스에 톰캣 설치하기.

: 프로젝트가 설치되는 디렉토리의 구조를 이해해야 한다.

① Sudo apt install tomcat9 tomcat9-admin.

② 위치확인 : Pwd

③ 디렉토리 이동 : Cd /var/lib/tomcat9.

④ 평점 100% 정한 설정 : Cd root - sudo vi index.html

⑤ 프로젝트 파일 이동 : sudo cp . /www.war /var/lib/tomcat9/webapps.

⑥ visual box 확인.

1. 클라우드 컴퓨터.

- 가상 컴퓨터 (물리적 X)
- 유비쿼터스 (24시간, 언제 어디서나 접속 가능)

사용 이유

- 배포
- 클라우드 데이터베이스 사용을 위해.

2. GCP

: Google Cloud Platform.

: VM (Virtual Machine) 가상 컴퓨터를 만들 수 있다.

설치

① 인스턴스 만들기

② 운영체제 선택.

③ 방화벽 http, https 허용

④ 내부 IP, 외부 IP 확인.

내부 IP = 내 컴퓨터 주소 / 외부 IP = 공인 IP

⑤ SSH 인증

- tfconfig 확인
- jar 파일 업로드.

⑥ 방화벽 규칙 설정

: 포트번호 허용.