**환경설정**

□ 이클립스

∙ 이클립스 세팅

- 23-03 버전 다운로드

- 압축해제 후 eclipse.exe 실행 → workspace 경로 설정 가능

- Window - Preference - General - content type - text

→ CSS, HTML, Java prop., Java Source, JSP : UTF-8로 설정(각각 Update 클릭)

- General - web: HTML, CSS, JSP → UTF-8로 변경

- General - web browser: chrome으로 설정

- installed jre : edit - browse - jdk17 설치 폴더 선택. 이름도 ‘jdk17’로 변경

- General - show heap status 켜기

- General - server - runtime - apache 9.X edit(없으면 add) 상단은 아파치 9.X으로, 하단은

C:\workspaceSpring\tomcat9(경로 다를 수 있음)

- Windows - Show View - Servers → 하단 서버 - 아파치 9.X열기

→ tomcat v9.X 더블클릭-> 좌측하단 serve, publish 체크 후 저장

- ERMaster : https://zoosso.tistory.com/627

- tern : https://code-nen.tistory.com/8

- SpringFramework : eclipse - help - eclipse marketplace - sts 검색 - Spring Tools 3 설치

(설치가 안 될 경우, 두 번째 플러그인 설치)

- Mybatis : eclipse - help - eclipse marketplace - myBatis 검색

□ Tomcat

∙ Tomcat 세팅

- apache tomcat 9.X 다운로드 및 압축해제

- 폴더 안에 있는 'apache-tomcat-9.X' 잘라내어, eclipse의 workspace 안에 붙여넣기

- 'apache-tomcat-9.X' 폴더명을 tomcat9로 변경

∙ Tomcat 배포

- 이클립스 Export - Web – WARfile

- 프로젝트명(=artifactId, 그대로 사용)

- 톰캣압축을 새로풀고, tomcat9으로 이름을 바꾸고, (단, 해당 경로폴더명들은 영문이어야 함)

- conf 폴더의 server.xml 열어, 이클립스의 서버 포트번호와 일치시키고, 그 옆에

URIEncoding="UTF-8" 추가

- conf 폴더의 tomcat-users.xml 열어, 맨 아래에 유저정보 추가

<user username="admin" password="admin12"

roles="admin-gui, admin-script, manager-script

, manager-gui, manager-jmx, manager-status"/>

- bin 폴더로 가서, 그 경로 복사, 관리자 권한으로 CMD열어서 'cd'에 하나 띄고, 경로

복사하고 엔터, startup.bat 엔터치고, 서버 실행

- 서버 종료는 bin 폴더에 shutdown.bat 파일 실행으로만 가능

- Export했던 WAR파일을 webapps폴더에 붙여넣음

- 'localhost:OOOO(설정한 포트번호)'를 브라우저 주소창에 입력

- ManagerApp 에서 로그인(tomcat-users.xml에 입력한 유저정보)

□ GitHub

∙ Git을 사용할 때 소스코드를 올려놓는 서버가 필요한데, 이 서버를 제공해 주는 서비스

- 리눅스 토발즈가 리눅스 개발에 이용하기 위해 처음 개발한 소스코드 관리 시스템

- 버전관리 시스템에서의 특징 : 빠른 속도 / 단순한 구조 / 완벽한 분산

/ 대형 프로젝트에 유용 / 거의 모든 명령을 로컬에서 실행

∙ Git의 데이터 상태 관리

- Modified : 수정한 파일을 아직 로컬 데이터에 적용하지 않은 상태

- Staged : 현재 수정한 파일을 곧 적용할 것이라고 표시한 상태

- Committed : 로컬 데이터에 안전하게 저장 완료된 상태

∙ Git의 기본 용어

- Repository : 저장소, 소스코드의 모든 내용을 저장

각 소스코드의 변경 사항을 관리

- Working Tree : 원격 저장소에 있는 모든 소스코드와 변경 정보를 복사한 내 컴퓨터의 작업

디렉토리의 현재 상태

- Directory : 원격 저장소에 저장된 소스코드와 그 변경사항을 관리하기 위해 내 컴퓨터에

변경 점에 대한 정보만을 따로 관리

- Branch : 소프트웨어 개발에서 다양한 버전이나 특정 기능들을 따로 구현해야 할 때에 기본

개발 부분을 건드리지 않고 분기

- Merge : 합치기 기능. Branch를 생성해서 작업하던 내용이 Master Branch에 통합되어

반영되어야 할 때

∙ Git과 GitHub의 차이

- Git은 개인 컴퓨터에서 돌아가는 Version Control System

- Git을 사용하기 위해서는 개인 계정을 등록할 필요도 없으며, 인터넷을 연결할 필요도 없음

- GitHub는 GitHub라 불리는 회사에서 서비스하고 있는 서버에 올라간 Git이기 때문에 사용

하기 위해 개인 계정을 등록해야 하며, 인터넷에 연결되어 있어야 한다

- GitHub는 Git 소프트웨어를 지원하는 일종의 클라우드 서비스

- Git에서 제공하지 않는 기능도 추가적으로 제공해주고 있다 (ex) UI제공)

∙ GitHub 기능

- 소스코드 관리

- 프로젝트 설명서

- 커뮤니티

∙ 소스코드 관리 서버

- Git시스템을 사용하여 발생되는 소스코드의 변경 정보와 모든 소스코드 파일을 저장하기

위한 공간

- Git을 사용하는데 있어 갱인적으로 서버를 구성하여 프로젝트 관리도 가능

- GitHub 이용 비용 ( 오픈소스 프로젝트 : 무료 / private 프로젝트 : 유료)

- 소스코드 관리서버 기능 개인적으로 서버 구현을 하는 번거로움을 GitHub에 프로젝트만

생성하면, 누구나 쉽게 Git의 중앙 서버로 사용할 수 있음

- 각 개발자들의 소스코드의 상태를 관리 & 저장해주는 저장소 역할

∙ GitHub의 커뮤니티 기능

- GitHub 프로젝트 소개 홈페이지로도 활용 가능

- GitHub 사이트에서 프로젝트 페이지에 접속 시 누구나 현재버전 소스코드를 확인 가능

- 질문과 답변 가능

- 이슈를 등록 및 상태 체크

∙ GitHub 기타 기능

- 문서작성 기능 : 프로젝트 소개 및 사용방법 등을 별도의 홈페이지 없이도 보여줄 수 있음

- 이슈관리 기능 : 개발 중에 발생하는 문제에 대한 관리 이슈를 등록하면 개발자 토론 기능

- 개발진행 관리 기능 : 개발 진행 관리와 차기 버전에서 개선해야 할 부분 정리에 큰 도움

지속적인 업데이트가 필요한 서비스의 개념 강화

- 커뮤니티 기능 : 개발자들 간의 의견 교류를 할 수 있는 커뮤니티 페이지를 제공해,

스타트업이나 오픈소스 프로젝트를 비롯해 상용 프로젝트에서도 소스코드

관리, 개발과 서비스의 연속성을 위해서 활용하는 사례가 증가

- 통계 기능 : 이슈에 대해 Graph를 통해 통계 데이터 확인 가능하고, 시각화를 통해 어떻게

진행되는지 파악이 쉬움

∙ 순서

- Pull (내용 최신 업데이트) -> 코드 작성 -> 완벽할 때 (팀원들 확인) -> 비교 -> Commit

-> Push (Branch 'main')