*** 형상관리 시스템**

- 서비스 제공 대상의 형상 항목을 식별하여 기준선을 설정하고, 형상 항목 변경과정에서 체계적인 통제를 통해 형상항목간의 일관성과 추적성을 확보하기 위한 시스템. 즉, 어떤 문서나 파일이 변경됐을 경우 변경 원인과 변경 사항을 확인해야 하는 경우에 대한 관리를 의미함.

*** 형상관리 도구 종류**

- CVS, SVN, Git (최근들어 가장 대중화 되고있음)

*** 버전(형상) 관리 시스템의 이점**

- 형상관리된 정보들(변경된 이력들)을 관리할 수 있어 추적성이 높으며 언제든지 과거에 접근 변경 및 수정 작업을 반복할 수 있음. 뿐만 아니라 여러 사람이 소스코드를 공유해 개발하면서 공유할때 생기는 버전 충돌 문제를 해결할 수 있다.

*** Git에서 로컬저장소(local repository)와 원격저장소(remote repository)의 차이점**

- 로컬저장소는 내 pc에 저장되는 개인용 저장소이고 원격저장소는 여러 사람이 함께 공유해서 사용하는 저장소이다. 따라서 로컬저장소에 변경사항이 생길 경우 commit and push를 이용하여 원격 저장소에 올릴 수 있다. 반대로 원격저장소에 변경사항이 생길 경우 pull을 이용하여 로컬저장소로 내려받을 수 있다.

> **commit** : 파일의 변경사항들에 대해 기록하는 것

> **push** : 원격저장소로 변경된 파일을 업로드 하는 것

> **pull** : 원격저장소로부터 변경된 파일을 다운로드 받는 것

>> 현재 변경된 파일의 내용과 그 아래 모든 파일을 add하고 commit할 때 명령어

```
git add .   /   git commit –m "<이번 확정본에 대한 설명>"
```

>> 로컬저장소에 있는 commit한 정보를 원격서버(origin/master)에 올릴 때 명령어

```
git push origin master
```

>> 원격서버에서 소스를 내려받을 때 사용하는 명령어

```
git pull
```

*** 개발이 완료된 프로젝트를 아파치 톰캣 서버와 연동하여 운영할 수 있는 파일**

> war파일 : 웹 어플리케이션 프로젝트를 압축하여 배포에 사용되는 파일 형식

> jar파일 : 여러 자바클래스들과 각종 자원 등을 모아 s/w나 라이브러리로 배포를 위한 파일형식

*** 웹 프로젝트를 톰캣 서버에 올리기 위한 배포파일(war)로 만들기 위한 순서**

1) 웹 프로젝트 선택

2) export

3) web 선택

4) war file 선택

5) folder select

6) tomcat version select

7) finish

>> 위의 과정으로 얻어낸 배포파일(war)를 톰캣 서버에 올릴 때의 폴더위치

- 해당 Tomcat 폴더 내의 webapps 폴더 안에(즉, Tomcat폴더/webapps)

*** 테스트 커버리지** : 전체 프로그램의 범위 테스트 시 해당 테스트 수행을 위해 동작된 프로그램의 범위를 의미함.