대분류/20
정보통신

중분류/01
정보기술

소분류/02
정보기술개발

세분류/02
응용SW엔지니어링

능력단위/11

NCS학습모듈

서버프로그램 구현

LM2001020211_19v5

교육부

NCS 학습모듈은 교육훈련기관에서 출처를 명시하고 교육적 목적으로 활용할 수 있습니다. 다만 NCS 학습모듈에는 국가(교육부)가 저작재삭권 일체를 보유하지 않은 저작물들(출처가 표기되어 있는 도표, 사진, 삽화, 도면 등)이 포함되어 있으므로 이러한 저작물들의 변형, 복제, 공연, 배포, 공중 송신 등과 이러한 저작물들을 활용한 2차 저작물의 생성을 위해서는 반드시 원작자의 동의를 받아야 합니다.

NCS학습모듈의 이해

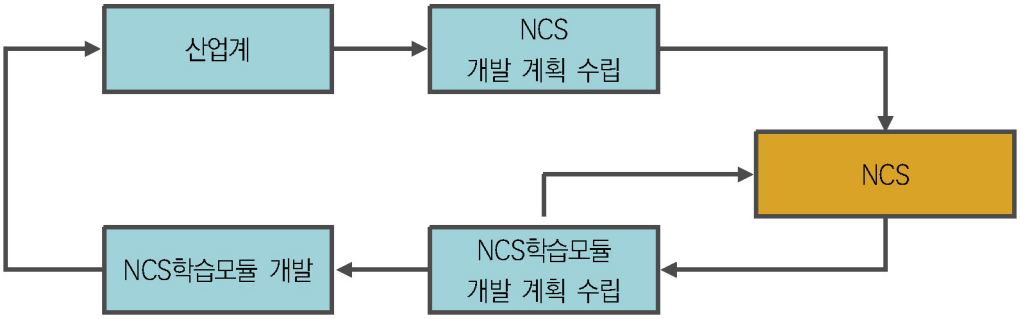
※ 본 NCS학습모듈은 「NCS 국가직무능력표준」사이트(http://www.ncs.go.kr) 에서 확인 및 다운로드할 수 있습니다.

Ⅰ NCS학습모듈이란?

❍ 국가직무능력표준(NCS: National Competency Standards)이란 산업현장에서 직무를 수행하기 위해 요구되는 지식·기술·소양 등의 내용을 국가가 산업부문별·수준별로 체계화한 것으로 산업현장의 직무를 성공적으로 수행하기 위해 필요한 능력(지식, 기술, 태도)을 국가적 차원에서 표준화한 것을 의미합니다.

❍ 국가직무능력표준(이하 NCS)이 현장의 '직무 요구서'라고 한다면, **NCS학습모듈은 NCS의 능력단위를 교육훈련에서 학습할 수 있도록 구성한 '교수·학습 자료'입니다.** NCS학습모듈은 구체적 직무를 학습할 수 있도록 이론 및 실습과 관련된 내용을 상세하게 제시하고 있습니다.

NCS 및 NCS학습모듈 개발 절차

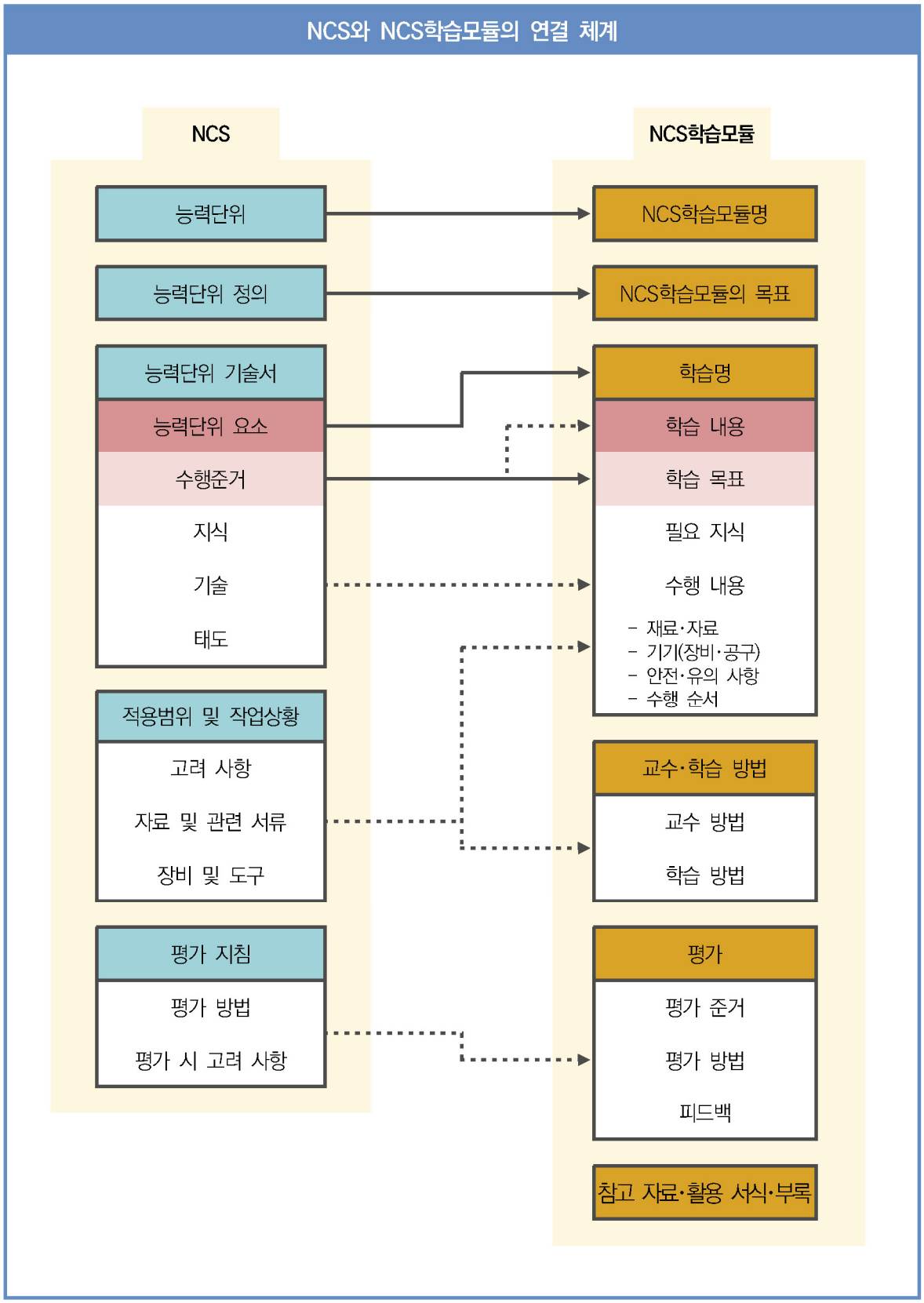


❍ **NCS학습모듈은 다음과 같은 특징을 가지고 있습니다.**

첫째, NCS학습모듈은 산업계에서 요구하는 직무능력을 교육훈련 현장에 활용할 수 있도록 성취목표와 학습의 방향을 명확히 제시하는 가이드라인의 역할을 합니다.

둘째, NCS학습모듈은 특성화고, 마이스터고, 전문대학, 4년제 대학교의 교육기관 및 훈련기관, 직장교육기관 등에서 표준교재로 활용할 수 있으며 교육과정 개편 시에도 유용하게 참고할 수 있습니다.

❍ NCS와 NCS학습모듈 간의 연결 체계를 살펴보면 아래 그림과 같습니다.



II NCS학습모듈의 체계

❍ **NCS학습모듈은** 1. NCS학습모듈의 위치, 2. NCS학습모듈의 개요, 3. NCS학습모듈의 내용 체계, 4. 참고 자료, 5. 활용서식/부록 **으로 구성되어 있습니다.**

1. NCS학습모듈의 위치

❍ **NCS학습모듈의 위치는 NCS 분류 체계에서 해당 학습모듈이 어디에 위치하는지를 한 눈에 볼 수 있도록 그림으로 제시한 것입니다.**

[NCS-학습모듈의 위치]

대분류	문화·예술·디자인·방송
중분류	문화콘텐츠
소분류	문화콘텐츠제작

세분류	능력단위	학습모듈명
방송콘텐츠제작		
영화콘텐츠제작	프로그램 기획	프로그램 기획
음악콘텐츠제작	아이템 선정	아이템 선정
광고콘텐츠제작	자료 조사	자료 조사
게임콘텐츠제작	프로그램 구성	프로그램 구성
애니메이션 콘텐츠제작	캐스팅	캐스팅
만화콘텐츠제작	제작계획	제작계획
캐릭터제작	방송 미술 준비	방송 미술 준비
스마트문화앱 콘텐츠제작	방송 리허설	방송 리허설
영사	야외촬영	야외촬영
	스튜디오 제작	스튜디오 제작
	…	…

학습모듈은

NCS 능력단위 1개당 1개의 학습모듈 개발을 원칙으로 합니다. 그러나 필요에 따라 고용단위 및 교과단위를 고려하여 능력단위 몇 개를 묶어 1개 학습모듈로 개발할 수 있으며, NCS 능력단위 1개를 여러 개의 학습모듈로 나누어 개발할 수도 있습니다.

2. NCS학습모듈의 개요

❍ **NCS학습모듈의 개요는 학습모듈이 포함하고 있는 내용을 개략적으로 설명한 것으로 학습모듈의 목표, 선수학습, 학습모듈의 내용 체계, 핵심 용어로 구성되어 있습니다.**

학습모듈의 목표	해당 NCS 능력단위의 정의를 토대로 학습 목표를 작성한 것입니다.
선수학습	해당 학습모듈에 대한 효과적인 교수·학습을 위하여 사전에 이수해야 하는 학습모듈, 학습 내용, 관련 교과목 등을 기술한 것입니다.
학습모듈의 내용 체계	해당 NCS 능력단위요소가 학습모듈에서 구조화된 체계를 제시한 것입니다.
핵심 용어	해당 학습모듈의 학습 내용, 수행 내용, 설비·기자재 등 가운데 핵심적인 용어를 제시한 것입니다.

제작계획 학습모듈의 개요

학습모듈의 목표

본격적인 촬영을 준비하는 단계로서, 촬영 대본을 확정하고 제작 스태프를 조직하며 촬영 장비와 촬영 소품을 준비할 수 있다.

선수학습

제작 준비(LM0803020105_13v1), 섭외 및 제작스태프 구성(LM0803020104_13v1), 촬영 제작(LM0803020106_13v1), 촬영 장비 준비(LM0803040204_13v1.4), 미술 디자인 협의하기(LM0803040203_13v1.4)

학습모듈의 내용체계

학습	학습 내용	NCS 능력단위 요소 코드번호	NCS 능력단위 요소 요소 명칭
1. 촬영 대본 확정하기	1-1. 촬영 구성안 검토와 수정	0803020114_16v3.1	촬영 대본 확정하기
2. 제작 스태프 조직하기	2-1. 기술 스태프 조직 2-2. 미술 스태프 조직 2-3. 전문 스태프 조직	0803020114_16v3.2	제작 스태프 조직하기
3. 촬영 장비 계획하기	3-1. 촬영 장비 점검과 준비	0803020114_16v3.3	촬영 장비 계획하기
4. 촬영 소품 계획하기	4-1. 촬영 소품 목록 작성 4-2. 촬영 소품 제작 의뢰	0803020114_16v3.4	촬영 소품 계획하기

핵심 용어

촬영 구성안, 제작 스태프, 촬영 장비, 촬영 소품

학습모듈의 목표는

학습자가 해당 학습모듈을 통해 성취해야 할 목표를 제시한 것으로, 교수자는 학습자가 학습모듈의 전체적인 내용흐름을 파악하도록 지도할 수 있습니다.

선수학습은

교수자 또는 학습자가 해당 학습모듈을 교수·학습하기 이전에 이수해야 하는 교과목 또는 학습모듈(NCS 능력단위) 등을 표기한 것입니다. 따라서 교수자는 학습자가 개별 학습, 자기 주도 학습, 방과 후 활동 등 다양한 방법을 통해 이수할 수 있도록 지도하는 것을 권장합니다.

핵심 용어는

해당 학습모듈을 대표하는 주요 용어입니다. 학습자가 해당 학습모듈을 통해 학습하고 평가받게될 주요 내용을 알 수 있습니다. 「NCS 국가직무능력표준」 사이트(www.ncs.go.kr)의 색인 (찾아보기) 중 하나로 이용할 수 있습니다.

3. NCS학습모듈의 내용 체계

❍ **NCS학습모듈의 내용은 크게 학습, 학습 내용, 교수·학습 방법, 평가로 구성되어 있습니다.**

학습	해당 NCS 능력단위요소 명칭을 사용하여 제시한 것입니다. 학습은 크게 학습 내용, 교수·학습 방법, 평가로 구성되며 해당 NCS 능력단위의 능력단위 요소별 지식, 기술, 태도 등을 토대로 내용을 제시한 것입니다.
학습 내용	학습 내용은 학습 목표, 필요 지식, 수행 내용으로 구성되며, 수행 내용은 재료·자료, 기기(장비·공구), 안전·유의 사항, 수행 순서, 수행 tip으로 구성한 것입니다. 학습모듈의 학습 내용은 실제 산업현장에서 이루어지는 업무활동을 표준화된 프로세스에 기반하여 다양한 방식으로 반영한 것입니다.
교수·학습 방법	학습 목표를 성취하기 위한 교수자와 학습자 간, 학습자와 학습자 간 상호 작용이 활발하게 일어날 수 있도록 교수자의 활동 및 교수 전략, 학습자의 활동을 제시한 것입니다.
평가	평가는 해당 학습모듈의 학습 정도를 확인할 수 있는 평가 준거 및 평가 방법, 평가 결과의 피드백 방법을 제시한 것입니다.

학습 1	촬영 대본 확정하기
학습 2	**제작 스태프 조직하기**
학습 3	촬영 장비 계획하기
학습 4	촬영 소품 계획하기

2-1. 기술 스태프 조직

학습 목표	• 프로그램 제작에 적합한 기술 스태프를 조직할 수 있다.

필요 지식 /

1 기술 스태프의 구성

프로그램의 장르에 따라 구성하는 기술 스태프는 많은 차이가 있다. 같은 장르의 프로그램이라도 그 형식이나 내용, 규모에 따라서 구성되는 기술 스태프의 종류와 인원수는 천차만별이다.

1. 스튜디오 프로그램

토크쇼, 종합 구성, 예능과 같은 스튜디오 프로그램은 부조정실과 스튜디오를 사용하여 제작하기 때문에 많은 기술 스태프가 필요하다.

학습은

해당 NCS 능력단위요소 명칭을 사용하여 제시하였습니다. 하나의 학습은 일반교과의 '대단원'에 해당되며, 학습모듈을 구성하는 가장 큰 단위가 됩니다. 또한 하나의 직무를 수행하기 위한 가장 기본적인 단위로 사용할 수 있습니다

학습 내용은

NCS 능력단위요소별 수행준거를 기준으로 제시하였습니다. 일반교과의 '중단원'에 해당합니다.

학습 목표는

학습 내용을 이수할 때 학습자가 갖춰야할 행동 수준을 의미합니다. 따라서 수업시간의 과목 목표로 활용할 수 있습니다.

필요 지식은

해당 NCS의 지식을 토대로 학습에 대한 이해와 성과를 제고하기 위해 반드시 알아야 할 주요 지식을 제시하였습니다. 필요 지식은 수행에 꼭 필요한 핵심 내용을 위주로 제시하여 교수자의 역할이 매우 중요하며, 이후 수행 순서와 연계하여 교수·학습으로 진행할 수 있습니다.

수행 내용 / 기술 스태프 구성표 작성하기

재료·자료

- 방송프로그램 제작 기획서 및 방송 대본, 콘티(continuity), 제작 일정, 운용표
- 장비 및 시설, 제작 시설 배정 의뢰서 및 배정표, 방송 기술 스태프 데이터베이스(DB) 자료

기기(장비 · 공구)

- 컴퓨터 등

안전 · 유의 사항

- 프로그램의 내용과 제작 방법을 분석하고, 각 스태프들의 역할을 신중하게 검토한다.

수행 순서

1 방송 대본이나 콘티(continuity), 큐 시트를 분석하고, 프로그램의 내용적 특성, 제작 과정에 대한 자료를 수집한다.

2 프로그램 제작 방법을 결정한다.

1. 스튜디오 녹화를 할 것인가, 야외 촬영을 할 것인가 검토한다.

수행 tip

- 스태프의 결정은 스태프 간의 호흡을 중요시하여 선정해야 프로그램의 질을 향상시킬 수 있다.

수행 내용은

해당 학습모듈에서 제시한 내용 중 기술(skill)을 습득하기 위한 실습과제로 활용할 수 있습니다.

재료·자료는

수행 내용을 수행하는데 필요한 재료 및 준비물로 실습 시 활용할 수 있습니다.

기기(장비·공구)는

수행 내용에 필요한 기본적인 장비 및 도구를 제시하였습니다. 제시된 기기 외에도 수행에 필요한 다양한 도구나 장비를 활용할 수 있습니다.

안전·유의사항은

수행 내용을 수행하는 데 있어 안전상 주의해야할 점 및 유의사항을 제시하였습니다. 실습 시 유념해야 하며, NCS의 고려사항도 추가적으로 활용할 수 있습니다.

수행 순서는

실습 과제의 진행 순서로 활용할 수 있습니다.

수행 tip은

수행 내용에서 실습을 용이하게 할 수 있는 아이디어를 제시하였습니다. 수행 tip은 지도상의 안전 및 유의사항 외에 전반적으로 적용되는 주안점 및 수행 과제 목적에 대한 보충설명, 추가사항 등으로 활용할 수 있습니다.

학습2 교수·학습 방법

교수 방법

- 방송 프로그램의 기술적 요소, 미술 구성 요소, 특수 촬영에 대해 설명한다.
- 방송 프로그램 제작에서 각 기술 스태프의 역할에 대해 설명한다.
- 방송 프로그램을 분석하고 필요한 기술 스태프를 구성할 수 있도록 지시한다.

학습 방법

- 방송 프로그램의 기술적 요소, 미술 구성 요소, 특수 촬영에 대해서 알아본다.
- 프로그램 제작에 필요한 기술 스태프의 역할을 이해하고, 기술 스태프 구성표를 작성한다.

교수·학습 방법은

학습 목표를 성취하는 데 필요한 교수 방법과 학습 방법을 제시하였습니다.

교수 방법은

해당 학습 활동에 필요한 학습 내용, 학습 내용과 관련된 자료명, 자료 형태, 수행 내용의 진행 방식 등에 대하여 제시하였습니다. 또한 학습자의 수업참여도 제고 방법 및 수업 진행상 유의사항 등도 제시하였습니다. 선수학습이 필요한 학습을 학습자가 숙지하였는지 교수자가 확인하는 과정으로 활용할 수도 있습니다.

학습 방법은

해당 학습 활동에 필요한 학습자의 자기주도 학습 방법을 제시하였습니다. 또한 학습자가 숙달해야 할 실기 능력과 학습 과정에서 주의해야 할 사항 등도 제시하였습니다. 학습자가 학습을 이수하기 전 반드시 숙지해야 할 기본 지식을 학습하였는지 스스로 확인하는 과정에 활용할 수 있습니다.

학습2 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준 상	중	하
기술 스태프 조직	- 프로그램 제작에 적합한 기술 스태프를 조직할 수 있다.			
미술 스태프 조직	- 프로그램 제작에 적합한 미술 스태프를 조직할 수 있다.			
전문 스태프 조직	- 프로그램 특수 촬영을 위한 전문 스태프를 조직할 수 있다.			

평가 방법

- 사례 연구

학습 내용	평가 항목	성취수준 상	중	하
기술 스태프 조직	- 프로그램에서 기술적 요소의 파악 여부			
	- 기술 스태프의 역할 파악 여부			
	- 프로그램에 필요한 기술 스태프 구성표 작성 능력			

피드백

1. 사례 연구
   - 프로그램을 선택하여 기술 스태프, 미술 스태프, 전문 스태프 구성표를 예시와 같이 작성하였는지 개인별 능력을 평가한 후, 그 결과를 모든 학습자에게 공유하도록 한다.

평가는

NCS 능력단위의 평가 방법과 평가 시 고려사항을 준용하여 작성합니다. 교수자와 학습자가 평가 항목별 성취수준 확인 시 활용할 수 있습니다.

평가 준거는

학습자가 학습을 어느 정도 성취하였는지 평가하기 위한 기준을 제시하고 있습니다. 학습 목표와 연계하여 단위수업 시간에 평가 항목 별 성취수준을 평가하는 데 활용할 수 있습니다.

평가 방법은

NCS 능력단위의 평가 방법을 참고하였으며, 평가 준거에 따른 평가 방법을 2개 이상 제시합니다. 평가 방법의 종류는 포트폴리오, 문제해결 시나리오, 서술형 시험, 논술형 시험, 사례 연구, 평가자 체크리스트, 작업장 평가 등이 있으며, NCS 능력단위 요소 별 수행 수준을 평가하는 데 가장 적절한 방법을 선정하여 활용할 수 있습니다.

피드백은

평가 후에 학습자들에게 평가 결과를 피드백하여 학습 목표를 달성하는 데 활용할 수 있습니다.

4. 참고 자료

참고자료

- 교육부(2013). 섭외 및 제작스태프 구성(LM0803020104_13v1). 한국직업능력개발원.

참고 자료는

해당 학습모듈에 제시된 인용 자료의 출처를 제시하였습니다. 교수·학습의 과정에서 참고로 활용할 수 있습니다.

5. 활용 서식/부록

활용서식

스튜디오 기술 스태프 구성표

직종	이름	연락처	소속	특이사항	비고
기술감독					
조명감독					

활용 서식은

평가 서식, 실습 시트 등 교수·학습 시 활용할 수 있는 다양한 서식들로 구성하였습니다. 수행에서 평가에 이르기까지 필요한 서식을 해당 모듈의 특성에 맞춰 개발하거나 기존의 양식을 활용하여 제시하였습니다.

부록

[디지털 텔레비전 방송프로그램 음량 등에 관한 기준]

제정 2014. 11. 29. 미래창조과학부 고시 제2014-87호

제1장 총칙

제1조(목적) 이 고시는 방송법 제70조의2제1항에 따라 방송사업자가 디지털 텔레비전 방송프로그램 및 방송광고의 음량을 일정하게 유지하기 위해 필요한 사항을 규정함을 목적으로 한다.

부록은

활용 서식 이외에 교수·학습 과정에서 참고할 수 있는 자료가 있는 경우 제시하였습니다.

[NCS-학습모듈의 위치]

대분류	정보 통신
중분류	정보 기술
소분류	정보기술개발

세분류
SW아키텍처
응용SW엔지니어링
임베디드SW 엔지니어링
DB엔지니어링
NW엔지니어링
보안엔지니어링
UI/UX엔지니어링
시스템SW엔지니어링
빅데이터플랫폼 구축
핀테크엔지니어링
데이터아키텍처
IoT시스템연동
인프라스트럭쳐 아키텍처 구축

능력단위	학습모듈명
요구 사항 확인	요구 사항 확인
데이터 입출력 구현	데이터 입출력 구현
통합 구현	통합 구현
정보 시스템 이행	정보 시스템 이행
제품 소프트웨어 패키징	제품 소프트웨어 패키징
서버프로그램 구현	**서버프로그램 구현**
인터페이스 구현	인터페이스 구현
애플리케이션 배포	애플리케이션 배포
애플리케이션 리팩토링	애플리케이션 리팩토링
인터페이스 설계	인터페이스 설계
애플리케이션 요구사항 분석	애플리케이션 요구사항 분석
기능 모델링	기능 모델링
애플리케이션 설계	애플리케이션 설계

정적모델 설계	정적모델 설계
동적모델 설계	동적모델 설계
화면 설계	화면 설계
화면 구현	화면 구현
애플리케이션 테스트 관리	애플리케이션 테스트 관리
애플리케이션 테스트 수행	애플리케이션 테스트 수행
소프트웨어공학 활용	소프트웨어공학 활용
소프트웨어개발 방법론 활용	소프트웨어개발 방법론 활용
프로그래밍 언어 응용	프로그래밍 언어 응용
프로그래밍 언어 활용	프로그래밍 언어 활용
응용SW 기초 기술 활용	응용SW 기초 기술 활용
개발자 환경 구축	개발자 환경 구축
개발 환경 운영 지원	개발 환경 운영 지원

차 례

학습모듈의 개요 ---------- 1

학습 1. 개발환경 구축하기

1-1. 개발환경 준비 ---------- 3

1-2. 개발환경 구축 ---------- 11

- 교수 · 학습 방법 ---------- 24
- 평가 ---------- 25

학습 2. 공통 모듈 구현하기

2-1. 공통 모듈 구현 ---------- 27

2-2. 공통 모듈 테스트 ---------- 38

- 교수 · 학습 방법 ---------- 46
- 평가 ---------- 47

학습 3. 서버 프로그램 구현하기

3-1. 업무 프로세스 확인 ---------- 49

3-2. 서버프로그램 구현 ---------- 58

3-3. 서버프로그램 테스트 ---------- 67

- 교수 · 학습 방법 ---------- 74
- 평가 ---------- 75

학습 4. 배치 프로그램 구현하기

4-1. 배치 프로그램 구현 ---------- 77

4-2. 배치 프로그램 테스트 ---------- 85

- 교수 · 학습 방법 ---------- 91

• 평가 ---------- 92

참고 자료 ---------- 94

활용 서식 ---------- 95

서버 프로그램 구현 학습모듈의 개요

학습모듈의 목표

애플리케이션 설계를 기반으로 개발에 필요한 환경을 구성하고, 프로그래밍 언어와 도구를 활용하여 공통 모듈, 업무 프로그램과 배치 프로그램을 구현할 수 있다.

선수학습

애플리케이션 설계(2001020221_19v5), 프로그램 언어 활용(2001020231_19v4), SQL 활용(2001020413_19v4), 교육부 고시 NCS 기초과목 '자료 구조'

학습모듈의 내용 체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 개발환경 구축하기	1-1. 개발환경 준비 1-2. 개발환경 구축	2001020211_19v5.1	개발환경 준비하기
2. 공통 모듈 구현하기	2-1. 공통 모듈 구현 2-2. 공통 모듈 테스트	2001020211_19v5.2	공통 모듈 구현하기
3. 서버 프로그램 구현하기	3-1. 업무 프로세스 확인 3-2. 서버 프로그램 구현 3-3. 서버 프로그램 테스트	2001020211_19v5.3	서버 프로그램 구현하기
4. 배치 프로그램 구현하기	4-1. 배치 프로그램 구현 4-2. 배치 프로그램 테스트	2001020211_19v5.4	배치 프로그램 구현하기

핵심 용어

요구 사항 분석, 시스템 아키텍처, 애플리케이션 아키텍처, 공통 모듈, 서버, 형상관리, 단위 테스트

학습 1	개발환경 구축하기
학습 2	공통 모듈 구현하기
학습 3	서버 프로그램 구현하기
학습 4	배치 프로그램 구현하기

1-1. 개발환경 준비

학습 목표	• 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어의 필요 사항을 검토하고, 이에 따라 개발환경에 필요한 준비를 수행할 수 있다.

필요 지식 /

1 개발환경 구축의 이해

개발환경을 구축하기 위해서는, 해당 프로젝트의 목적과 구축 설계에 대한 명확한 이해가 필요하며, 이에 맞는 하드웨어, 소프트웨어의 선정이 이루어져야 한다. 또 개발에 사용되는 제품들의 성능과 라이선스 그리고 사용 편의성 등에 대한 내용도 파악해야 한다.

2 개발환경 구축에 필요한 하드웨어와 소프트웨어에 대한 이해

개발환경 구성을 위해서는 어떤 종류의 개발 소프트웨어들이 존재하며 각 종류마다 어떤 특성을 갖는지에 대해 판단할 수 있어야 한다. 그리고 이러한 소프트웨어가 설치되는 하드웨어에 대한 이해도 필요하다.

1. 개발을 위해 사용되는 소프트웨어의 종류와 특성

(1) 구현 도구

프로그램을 개발할 때 가장 많이 사용되는 도구로서 코드의 작성 및 편집, 디버깅 등과 같은 다양한 작업이 가능하며 Eclipse, Visual Studio Code, IntelliJ, NetBeans 등 다양한 소프트웨어 도구들이 사용되고 있다. 구현해야 할 소프트웨어가 어떤 프로그래밍 언어로 개발되는지에 따라 선택하여 사용한다.

(2) 테스트 도구

소프트웨어의 품질을 높이기 위해 테스트에 사용되는 소프트웨어 도구들로 코드의 테스트, 테스트에 대한 리포팅 및 분석 등의 작업이 가능하다. 사용되는 테스

트 도구들에는 xUnit, Spring Test 등이 있다.

(3) 형상관리 도구

대다수의 프로젝트들은 다수의 개발자들로 구성된 팀 단위 프로젝트로 진행되며, 개발자들이 작성한 소스 및 리소스 등 산출물에 대한 버전 관리를 위해 형상관리 도구가 사용된다. 대표적인 형상관리 도구로는 CVS, Subversion, Git 등이 있다.

(4) 빌드 도구

개발자가 작성한 소스에 대한 빌드 및 배포를 지원하며, 프로젝트에서 사용되는 구성 요소들과 라이브러리들에 대한 의존성 관리를 지원하는 도구이다. 사용되는 빌드 도구에는 Ant, Maven, Gradle 등이 있다.

2. 하드웨어를 고려한 개발 소프트웨어의 선정

프로그램 개발에 사용되는 소프트웨어들은 OS에 따라 적합한 것을 선정하여야 한다. 개발에 사용되는 시스템의 하드웨어 사양을 파악하여 해당 시스템 성능에 적합한 소프트웨어를 선정하여야 한다.

수행 내용 / 개발환경 준비하기

재료·자료

- 요구 사항 정의서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구
- 개발 프로그램

안전 · 유의 사항

- 실습 후에는 형상관리 서버에 결과물을 반영한다.
- 실습 후에는 컴퓨터를 정상적으로 종료시킨다.

수행 순서

1 개발환경을 준비하기 위해서 목표 시스템의 환경을 분석한다.

제안 요청서, 제안서, 사업 수행 계획서, 요구 사항 정의서, 시스템 아키텍처, 애플리케이션 아키텍처 등 분석 및 설계 시의 산출물을 분석하여 개발환경을 준비한다.

1. 목표 시스템의 환경을 분석한다.

다음 예시와 같이 간단한 인사 관리 시스템 구축을 통해 개발환경 구축에 필요한 도구에 대해 파악한다.

〈표 1-1〉 목표 시스템의 환경 예시

항목	내용
목표 시스템	인사 관리 웹 애플리케이션 시스템
개발언어	Java, HTML5, CSS3, Spring, jQuery, Javascript, Ajax, SQL
개발 인원 및 기간	개발 인원: 5명, 개발 기간: 3개월
개발 HW 사양	프로세서: Intel(R) Core(TM) i5-7300HQ CPU @ 2.50GHz 2.50GHz 메모리(RAM): 8.00GB

2. 요구 사항을 분석한다.

〈표 1-2〉 요구 사항 목록 예시

요구 사항 분류	요구 사항 고유 번호	요구 사항 명칭
분석/설계 요구 사항 (Analysis Plan Requirement)	APR-001	요구 사항 수집 및 확정
	APR-002	프로세스 분석
	APR-003	화면 및 DB 설계
기능 요구 사항 (System Function Requirement)	SFR-001	인사 정보 등록 기능
	SFR-002	인사 정보 목록 기능
	SFR-003	출퇴근 현황 조회 기능
	SFR-004	외출 신청 기능
	SFR-005	특근 신청 관리 기능
	SFR-006	최종 점검 관리 기능
	SFR-007	초과 근무 관리 기능
	SFR-008	권한 관리 기능

요구 사항 분류	요구 사항 고유 번호	요구 사항 명칭
성능 요구 사항 (Performance Requirement)	PER-001	기존 시스템 성능 유지
인터페이스 요구 사항 (System Interface Requirement)	SIR-001	시스템 연계
	SIR-002	직관적 UI
테스트 요구 사항 (Test Requirement)	TER-001	단위 테스트
	TER-002	통합 테스트
보안 요구 사항 (Security Requirement)	SER-001	보안 기준 준수
	SER-002	응용 및 DB 보안
품질 요구 사항 (Quality Requirement)	QUR-001	기능 구현의 정확성
	QUR-002	상호 운영성

2 개발언어 및 하드웨어 사양을 고려한 구현 도구를 선정한다.

1. 개발언어의 선정

(1) 개발언어의 선정 기준

〈표 1-3〉 개발언어 선정 기준

고려 항목	설명
적정성	대상 업무의 성격, 즉 개발하고자 하는 시스템이나 응용 프로그램의 목적에 적합해야 한다.
효율성	프로그래밍의 효율성이 고려되어야 한다.
이식성	일반적인 PC 및 OS에 개발환경이 설치 가능해야 한다.
친밀성	프로그래머가 그 언어를 이해하고 사용할 수 있어야 한다.
범용성	다양한 과거 개발 실적이나 사례가 존재하고, 광범위한 분야에 사용되고 있어야 한다.

※ 위 고려 항목은 참고 사항으로 활용되며, 위 항목들 이외에도 알고리즘과 계산상의 난이도, 소프트웨어의 수행 환경, 자료 구조의 난이도, 개발 담당자와의 경험과 지식 등이 고려되어야 한다.

(2) 개발언어의 선정

인사 관리 웹 애플리케이션 시스템 프로젝트에서 범용성, 이식성이 뛰어나고 대상 업무의 성격에 적합한 Java 및 JavaScript/HTML5를 Server와 Client 개발언어로 선정한다.

(가) Server Side: 웹 애플리케이션 개발에 적합한 Java 선정

(나) Client Side: 차세대 웹 표준인 HTML5와 jQuery를 선정

2. 개발 도구의 선정

(1) 통합 개발환경의 종류 및 특징

다양한 프로그래밍 언어를 지원하는 통합 개발환경의 예로 이클립스(Eclipse), 비주얼 스튜디오(Visual Studio) 등을 들 수 있다. 이클립스는 자바를 기본적으로 지원하지만 파이썬(Phython), PHP, JSP, C, C++ 등과 같은 언어들도 추가적으로 설치할 수 있다. 설치된 언어들은 각각의 디버거를 비롯한 다양한 도구들을 지원하고 있다.

〈표 1-4〉 통합 개발환경 종류 및 주요 특징

통합 개발환경	개발사	운영체제	언어	라이선스
Eclipse	IBM 이클립스 재단	MS-Windows Linux OSX Solaris AIX	Java C C++ PHP JSP 외 다수	Eclipse Public License
Lazarus	Lazarus Team	MS-Windows Linux OSX FreeBSD	Pascal	GPL, GNU LGPL, 기타
Anjuta	GNOME 프로젝트	Linux	C C++	GPS
Wide Studio	와이드 스튜디오 프로젝트	Linux	C C++	MIT
Code::Blocks	Code::Blocks Team	MS-Windows Linux OSX	C C++	GPL v3.0
Visual Studio	Microsoft	MS-Windows	VisualBasic .Net Visual C++ Visual C# 등	상용
Visual Studio Code	Microsoft	MS-Windows MacOS Linux	C++ C# Dockerfile Go HTML Typescript PHP 외 다수	MIT

통합 개발환경	개발사	운영체제	언어	라이선스
Delphi	엠바카데로 테크놀러지	MS-Windows	Object Pascal	상용
C++ builder	엠바카데로 테크놀러지	MS-Windows	C C++	상용
J builder	엠바카데로 테크놀러지	MS-Windows	Java	상용
JDeveloper	오라클	MS-Windows Linux Unix	Java PHP SQL 외 다수	GPL, 기타
X Code	애플	MacOS	C C++ Object-C Java 외 다수	혼합된 사유 재산

(2) 통합 개발환경의 선정

목표 시스템 구축을 위한 통합 개발환경의 선정은 풍부한 기능과 Plug-in을 보유하고 있고, 다양한 적용 사례가 존재하는 통합 개발 도구(IDE: Integrated Development Environment)를 개발 도구로 선정한다.

3 프로그램의 배포 및 라이브러리 관리를 위한 빌드(Build) 도구를 선정한다.

1. 빌드 도구 Gradle, Maven, Ant의 특징

〈표 1-5〉 빌드 도구 Gradle, Maven, Ant 비교

Gradle	Maven	Ant
기구현된 Goal 수행	기구현된 Goal 수행	프로젝트 특화된 Target 수행
프로젝트 전체 정보를 정의	프로젝트 전체 정보를 정의	빌드 프로세스만 정의
Multi 프로젝트 빌드 지원	빌드 생명 주기, 표준화된 디렉터리 레이아웃	매우 복잡한 빌드 스크립트
스크립트 규모가 작고 읽기 쉬움	재사용 가능한 플러그인 및 저장소	스크립트의 재사용 불가

2. 빌드 도구의 선정

(1) 프로젝트 팀원의 빌드 도구의 친밀도와 숙련도에 따라 결정한다.

(2) 형상관리 도구를 고려하여 빌드 도구를 선정한다.

(3) 통합 개발 도구와 호환이 가능한 빌드 도구를 선정한다.

4 개발 인원을 고려한 형상관리 도구를 선정한다.

1. 형상관리 도구의 종류

〈표 1-6〉 형상관리 도구 CVS, Subversion, Git 비교

항목	CVS	Subversion	Git
라이선스	GNU GPL v2.0	Apache License v2.0	GNU GPL v2.0
적용 언어	무관	무관	무관
OS	Windows/Linux Mac은 Third-Party 도구	Windows/Linux/Max	Windows/Linux/Max
실행환경	Command Line Interface	Command Line Interface	Command Line Interface
GUI	TortoiseCVS 등	TortoiseSVN, WinSVN 등	번들로 제공 Source Tree, Git Eye 등

2. 형상관리 도구의 선정

(1) 목표 시스템 환경 등을 고려하여 선정한다.

(2) 통합 개발 도구와 호환이 가능한 빌드 도구를 선정한다.

5 프로젝트 검증에 적합한 테스트 도구를 선정한다.

1. 테스트 도구의 종류

〈표 1-7〉 테스트 활동에 따른 도구 분류

테스트 활동	테스트 도구	내용
테스트 계획	요구 사항 관리	고객 요구 사항 정의 및 변경 사항 관리
테스트 분석/설계	테스트 케이스 생성	테스트 기법에 따른 테스트 데이터 및 케이스 작성
	커버리지 분석	대상 시스템에 대한 테스트 완료 범위 척도
테스트 수행	테스트 자동화	기능 테스트 등 테스트 도구를 활용하여 자동화를 통한 테스트의 효율성을 높일 수 있음. (xUnit, STAF, NTAF 등)
	정적 분석	코딩 표준, 런타임 오류 등을 검증
	동적 분석	대상 시스템 시뮬레이션을 통한 오류 검출 (Avalanche, Valgrind 등)
	성능 테스트	가상 사용자를 인위적으로 생성하여 시스템 처리 능력 측정 (JMeter, AB, OpenSTA 등)
	모니터링	시스템 자원(CPU, Memory 등) 상태 확인 및 분석 지원 도구 (Nagios, Zenoss 등)
	형상관리	테스트 수행에 필요한 다양한 도구 및 데이터 관리
	테스트 관리	전반적인 테스트 계획 및 활동에 대한 관리
	결함 추적/관리	테스트에서 발생한 결함 관리 및 협업 지원

2. 테스트 도구의 선정

통합 개발 도구와 호환이 가능하고 널리 사용되는 테스트 자동화 도구를 선정한다.

수행 tip

- 각 종류별 개발 도구들에 대해 이해하고 장점과 단점들에 대해 파악하여 프로젝트 요구 사항에 적합한도구들을 선정하는 것이 중요하다.

1-2. 개발환경 구축

학습 목표	• 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어를 설치하고 설정하여 개발환경을 구축할 수 있다. • 사전에 수립된 형상관리 방침에 따라 운영 정책에 부합하는 형상관리 환경을 구축할 수 있다.

필요 지식 /

1 개발환경 구성

일반적으로 사용되는 시스템 환경은 프로그램 개발을 위한 개발환경, 테스트를 위한 테스트 환경, 실제 시스템이 운영되는 운영환경과 백업환경 등으로 분류할 수 있다. 이 중 개발환경 구축에 필요한 하드웨어와 소프트웨어 명세를 살펴보기로 한다.

1. 개발 하드웨어 환경

개발 하드웨어 환경은 운영환경과 유사한 구조로 구성하는 것이 원칙이며, 개발용 하드웨어 환경을 구축하기 위해서는 다음과 같은 하드웨어 구성을 고려하여야 한다.

(1) 클라이언트(Client) 환경 구성

서버 시스템에서 제공하는 서비스를 활용하기 위해 사용자와의 인터페이스(Interface)를 제공하는 하드웨어로서, 일반적으로 PC(Client/Server 화면), 웹 브라우저 화면, 핸드폰(모바일 앱)이 클라이언트로 활용된다.

(2) 서버(Server) 환경 구성

서버 활용 목적에 따라 애플리케이션 서버, 데이터베이스 서버, 파일 서버 등으로 나눌 수 있다.

웹 서비스를 제공하기 위해서 애플리케이션 서버를 웹 서버와 웹 애플리케이션 서버로 분리하여 구성하기도 한다.

(가) 웹(Web) 서버

클라이언트(웹 브라우저 화면)에서 요청하는 서비스의 속도를 향상시키기 위해 정적 파일(HTML, CSS, 이미지 등)들을 제공하는 웹 서버 애플리케이션이 설치되는 하드웨어이다.

(나) 웹 애플리케이션(Application) 서버

동적 웹 서비스를 제공하기 위해 Tomcat, Undertow, IIS 등 미들웨어인 WAS(Web Application Server)와 서비스에 관련된 애플리케이션이 설치되는 하드웨어이다.

(다) 데이터베이스(Database) 서버

MySql, Oracle, MS-SQL 등 데이터베이스가 설치되는 하드웨어이다.

(라) 파일(File) 서버

서비스 제공을 위해 파일을 저장하고, 공유하기 위한 파일 저장 하드웨어이다.

2. 개발 소프트웨어 환경

개발 소프트웨어 환경도 개발 하드웨어 환경과 마찬가지로 운영환경과 동일한 구조로 구성하는 것이 원칙이며, 개발용 소프트웨어 환경을 구축하기 위해서는 다음과 같은 유형의 소프트웨어 구성을 고려하여야 한다.

(1) 시스템 소프트웨어

(가) 운영체제(OS: Operation System)

하드웨어 운영을 위한 운영체제로서, Windows/Linux/UNIX 등의 환경으로 구성되는데, 일반적으로 상세 소프트웨어 명세는 하드웨어를 제공하는 벤더(Vendor)에서 제공한다.

(예) Windows, Linux, UNIX(HPUX, Solaris, AIX) 등

(나) JVM(Java Virtual Machine)

Java 관련 응용 프로그램을 기동하기 위한 인터프리터 환경으로, 적용 버전을 개발표준에서 명시하여 모든 개발자가 동일한 버전을 적용하는 것이 좋다.

(다) Web Server

정적 웹 서비스를 수행하는 미들웨어로서, 웹 브라우저 화면에서 요청하는 정적 파일을 제공한다.

(예) Apache, Nginx, IIS(Internet Information Server), GWS(Google Web Server) 등

(라) WAS(Web Application Server)

웹 애플리케이션을 수행하는 미들웨어로서, 웹 서버와 JSP/Servlet 애플리케이션 수행을 위한 엔진으로 구성된다.

(예) Tomcat, Undertow, JEUS, Weblogic, Websphere 등

(마) DBMS(Database Management System)

데이터 저장과 관리를 위한 데이터베이스 소프트웨어이다.

(예) Oracle, DB2, Sybase, SQL Server, MySQL 등

(2) 개발 소프트웨어

(가) 요구 사항 관리도구

목표 시스템의 기능과 제약 조건 등 고객의 요구 사항을 수집, 분석, 추적을 쉽게 할 수 있게 지원한다.

(예) JFeature, JRequisite, OSRMT, Trello 등

(나) 설계/모델링 도구

기능을 논리적으로 결정하기 위해 통합 모델링 언어(UML: Unified Modeling Language) 지원, Database 설계 지원 등 설계 및 모델링을 지원하는 도구이다.

(예) ArgoUML, DB Designer, StarUML 등

(다) 구현 도구

문제 해결 방법을 소프트웨어 언어를 통해 구현 및 개발을 지원하는 도구이다.

(예) Eclipse, IntelliJ, Visual Studio 등

(라) 테스트 도구

구현 및 개발된 모듈들에 대하여 요구 사항에 적합하게 구현되어 있는지 테스트를 지원하는 도구이다.

(예) JUnit, CppUnit, JMeter SpringTest 등

(마) 형상관리 도구

산출물의 변경 사항을 버전별로 관리하여 목표 시스템의 품질 향상을 지원하는 도구이다.

(예) Git, SVN 등

2 형상관리 구성

1. 형상관리의 정의

형상관리(SCM: Software Configuration Management)란 소프트웨어의 개발 과정에서 발생하는 산출물의 변경 사항을 버전 관리하기 위한 일련의 활동으로, 다음과 같은 특성을 가지고 있다.

(1) 소프트웨어 변경사항을 파악하고 제어하며, 적절히 변경되고 있는지에 대해 확인하여 해당 담당자에게 통보하는 작업이다.

(2) 형상관리는 프로젝트 생명주기의 전 단계에서 수행하는 활동이며, 유지 보수 단계에서도 수행되는 활동이다.

(3) 형상관리를 함으로써 소프트웨어 개발의 전체 비용을 줄이고, 개발 과정에서 발생하는 여러 가지 문제점 발생 요인이 최소화되도록 보증하는 것을 목적으로 한다.

2. 형상관리 절차

형상관리는 품질 보증을 위한 중요한 요소로서 다음과 같은 절차를 통해 수행된다.

(1) 형상 식별

(가) 형상관리 대상을 식별하여 이름과 관리 번호를 부여하고, 계층(Tree) 구조로 구

분하여 수정 및 추적이 용이하도록 하는 작업으로 베이스라인의 기준을 정하는 활동이다.

(나) 형상관리 대상이 되는 형상인 형상항목은 다음과 같다.

1) 소프트웨어 공학 기반 표준과 절차: 방법론, WBS, 개발 표준
2) 소프트웨어 프로젝트 계획서
3) 소프트웨어 요구 사항 명세서
4) 소프트웨어 아키텍처, 실행 가능한 프로토타입
5) 소프트웨어 화면, 프로그램 설계서
6) 데이터베이스 기술서: 스키마, 파일 구조, 초기 내용
7) 소스코드 목록 및 소스코드
8) 실행 프로그램
9) 테스트 계획, 절차, 결과
10) 시스템 사용 및 운영과 설치에 필요한 매뉴얼
11) 유지 보수 문서: 변경 요청서, 변경 처리 보고서 등

(2) 변경 제어

식별된 형상항목의 변경 요구를 검토, 승인하여 적절히 통제함으로써 현재의 베이스라인에 잘 반영될 수 있도록 조정하는 작업으로, 적절한 형상 통제가 이루어지기 위해서는 형상통제위원회 승인을 통한 통제가 이루어질 수 있어야 한다.

(3) 형상 상태 보고

베이스라인의 현재 상태 및 변경 항목들이 제대로 반영되는지 여부를 보고하는 절차이며 형상의 식별, 통제, 감사 작업의 결과를 기록 및 관리하고 보고서를 작성하는 작업이다.

(4) 형상 감사

베이스라인의 무결성을 평가하기 위해 확인, 검증 과정을 통해 공식적으로 승인하는작업을 말한다.

수행 내용 / 개발환경 구축하기

재료·자료

- 요구 사항 정의서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구
- 개발 프로그램

안전 · 유의 사항

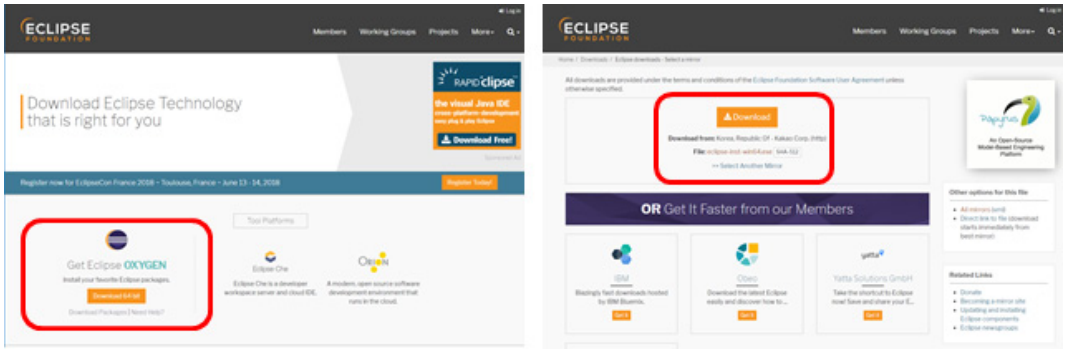
- 실습 후에는 형상관리 서버에 결과물을 반영한다.
- 실습 후에는 컴퓨터를 정상적으로 종료시킨다.

수행 순서

1 통합 개발환경 도구를 설치한다.

통합 개발환경 도구는 Eclipse, IntelliJ, Visual Studio 등 다양한 도구들이 있으나, 본 실습에서는 준비 단계에서 선정한 통합 개발환경 도구를 설치한다.

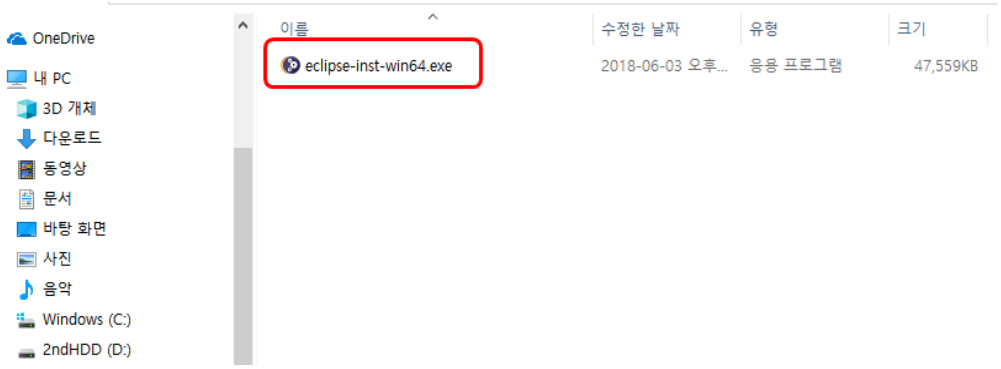
1. 요구 사항에 적합한 통합 개발환경 도구를 다운받는다.



[그림 1-1] 통합 개발 도구를 다운로드하는 화면 예시(Eclipse 예시)

2. 다운받은 통합 개발환경 도구를 설치한다.

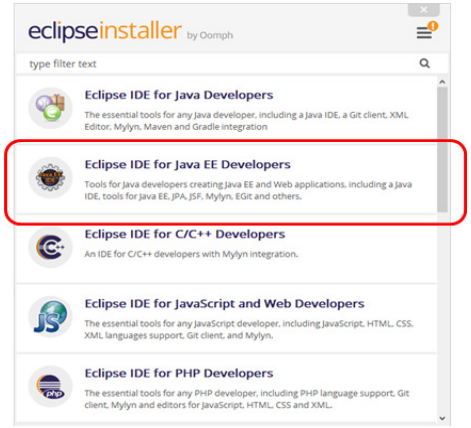
(1) 통합 개발환경 도구를 실행한다.



[그림 1-2] 통합 개발환경 도구를 설치하는 화면 예시

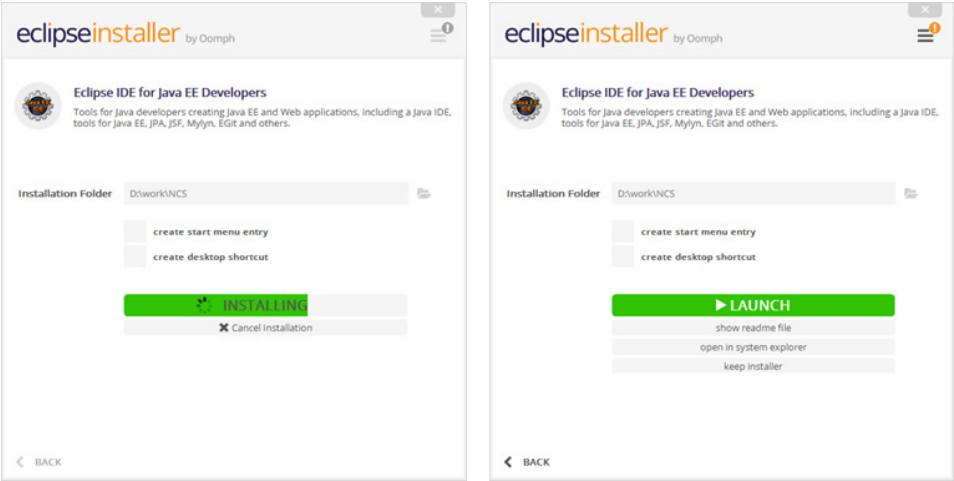
(2) 웹 애플리케이션 Installer를 선택한다.

Installer에는 다양한 패키지(package)들을 선택할 수 있지만, Java 웹 애플리케이션 프로젝트를 구성하기 위해 Eclipse IDE For Java EE Developers를 선택한다.



[그림 1-3] 웹 애플리케이션 도구 선택하기

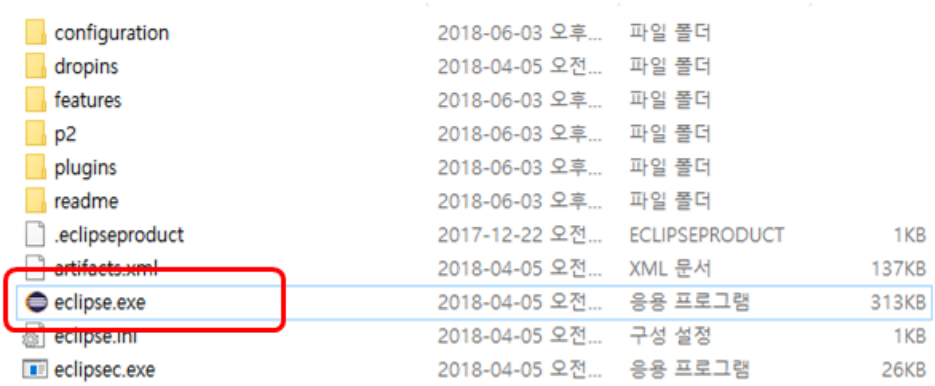
(3) 설치할 위치를 선택한 후 설치한다.



[그림 1-4] 설치 위치 선택 후 설치하는 화면 예시

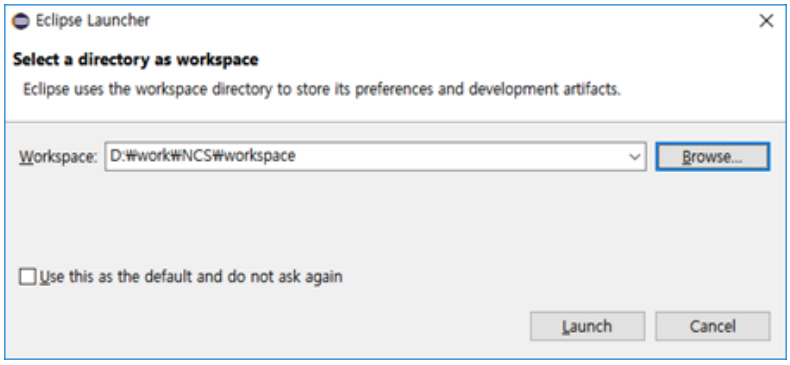
3. 통합 개발환경 도구를 실행한다.

(1) 설치된 통합 개발환경 도구를 실행한다.



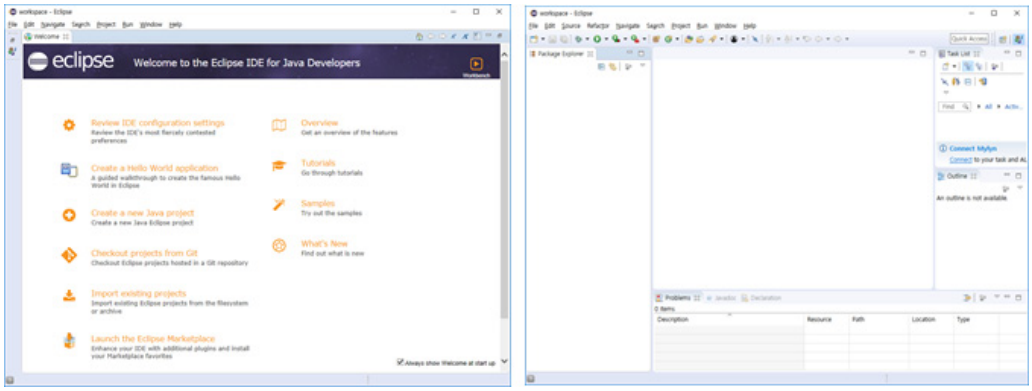
[그림 1-5] 통합 개발환경 도구를 실행하는 화면 예시

(2) 작업 공간(workspace) 위치를 선택한다.



[그림 1-6] 작업 공산 선택 화면 예시

(3) 실행 화면을 확인한다.



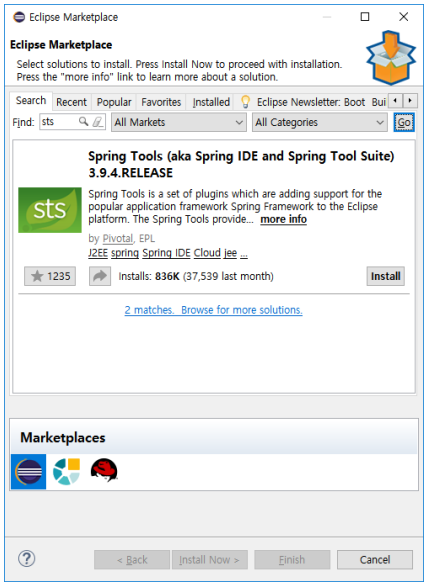
[그림 1-7] 통합 개발환경 도구의 실행 확인 화면 예시

2 Java 웹 애플리케이션 환경을 구성한다.

1. Spring 프레임워크 환경을 구성한다.

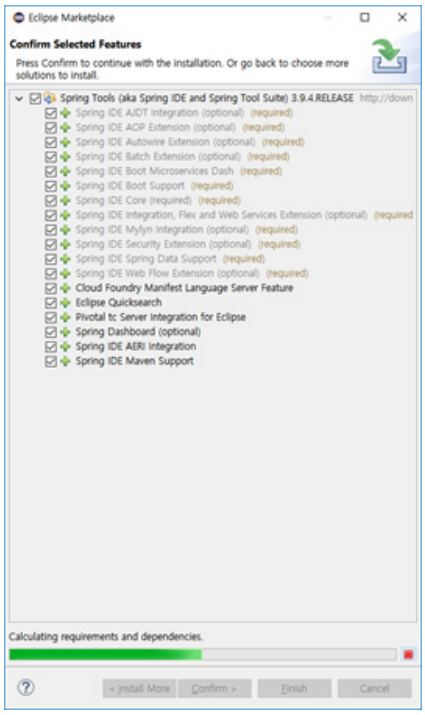
Spring 환경을 구성을 위해서는 STS(Spring Tool Suite)를 설치해야 하므로 통합 개발환경 도구에서 STS를 Plug-in 한다.

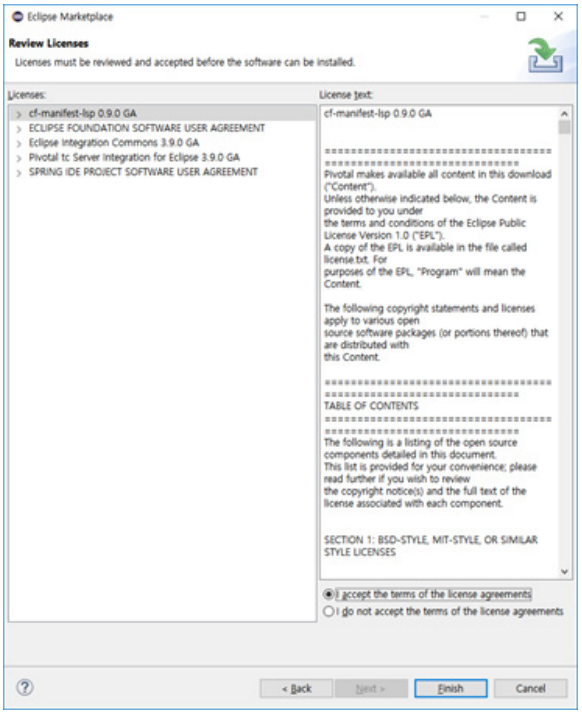
(1) STS를 Plug-in 한다.



[그림 1-8] STS 검색 화면 예시

(2) STS를 설치(Install)한다.



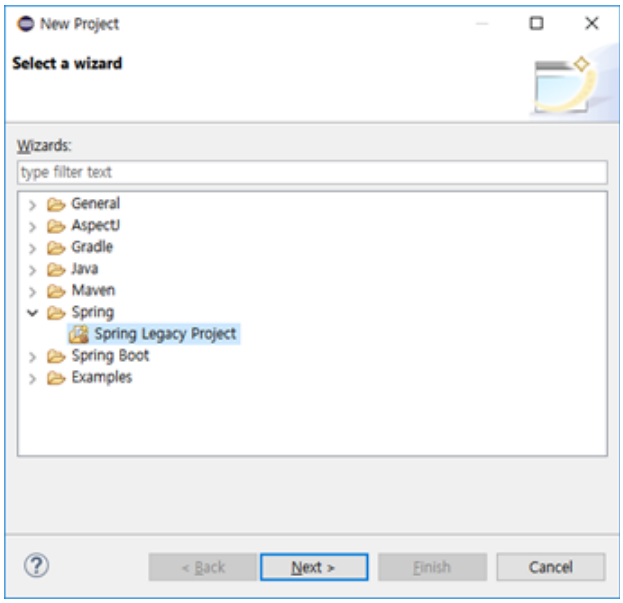


[그림 1-9] STS 설치 화면 예시 [그림 1-10] STS 설치 동의 화면 예시

2. Spring 프레임워크 기반의 웹 애플리케이션 프로젝트를 생성한다.

(1) 새로운 Spring 웹 프로젝트를 생성한다.

File > New > Project 메뉴를 선택한 후 Spring Legacy Project를 선택한다.

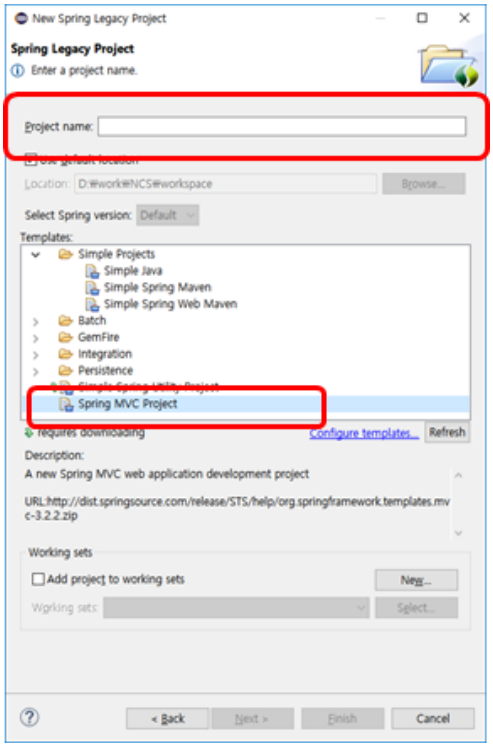


[그림 1-11] 프로젝트 생성 화면

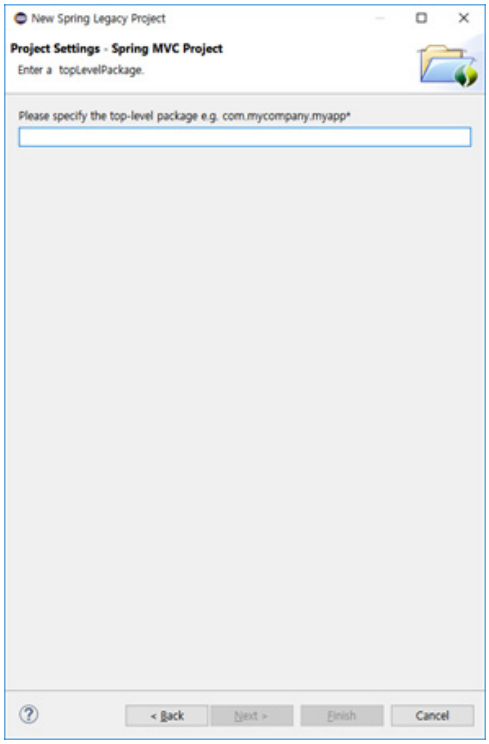
(2) Maven 프로젝트로 선택한다.

프로젝트명을 입력하고 Spring MVC Project 타입을 선택한다.

Next 버튼을 클릭하여 패키지(package)명을 입력하면 Maven 프로젝트가 구성된다.

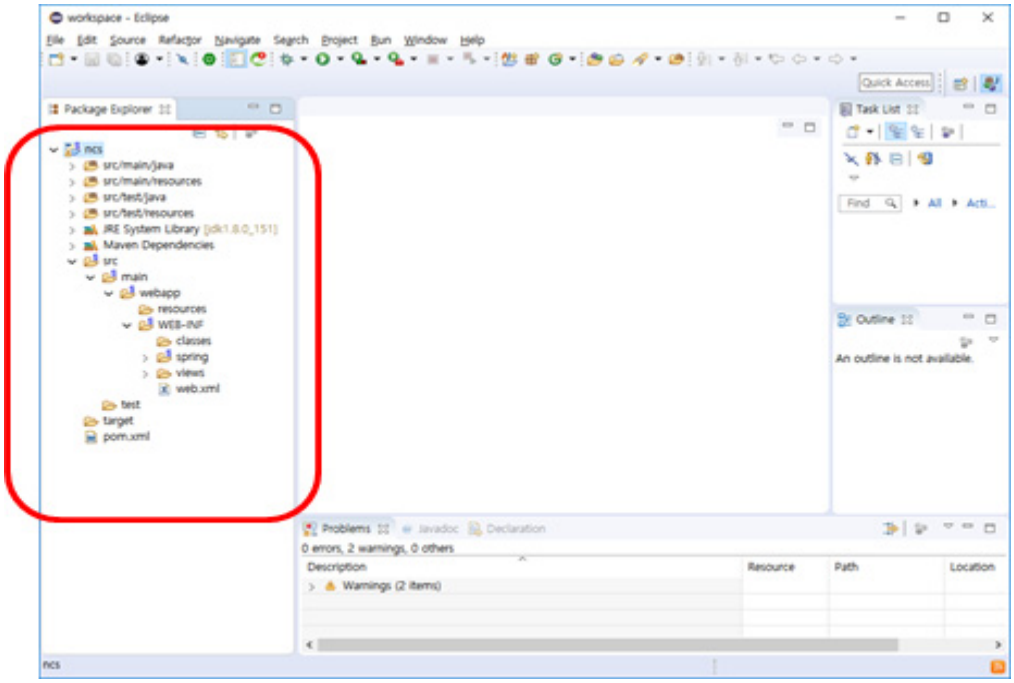


[그림 1-12] 프로젝트명 입력



[그림 1-13] 패키지 입력

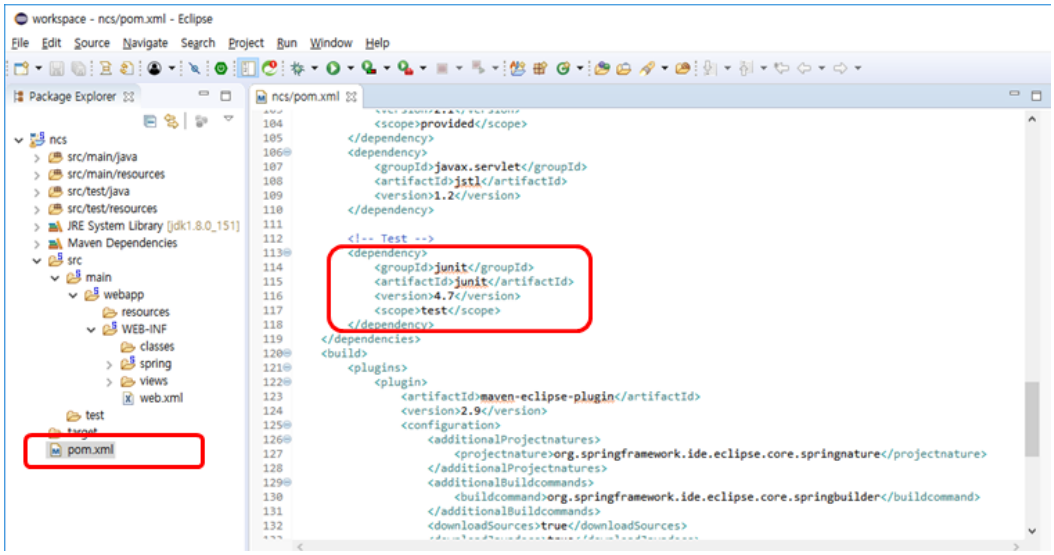
(3) 구성된 프로젝트를 확인한다.



[그림 1-14] Maven 기반의 Spring 웹 프로젝트 구성

3. 단위 테스트 자동화 도구의 설치 여부를 확인한다.

pom.xml 파일 안에 단위 테스트 도구가 포함되어 있는지 확인한다.

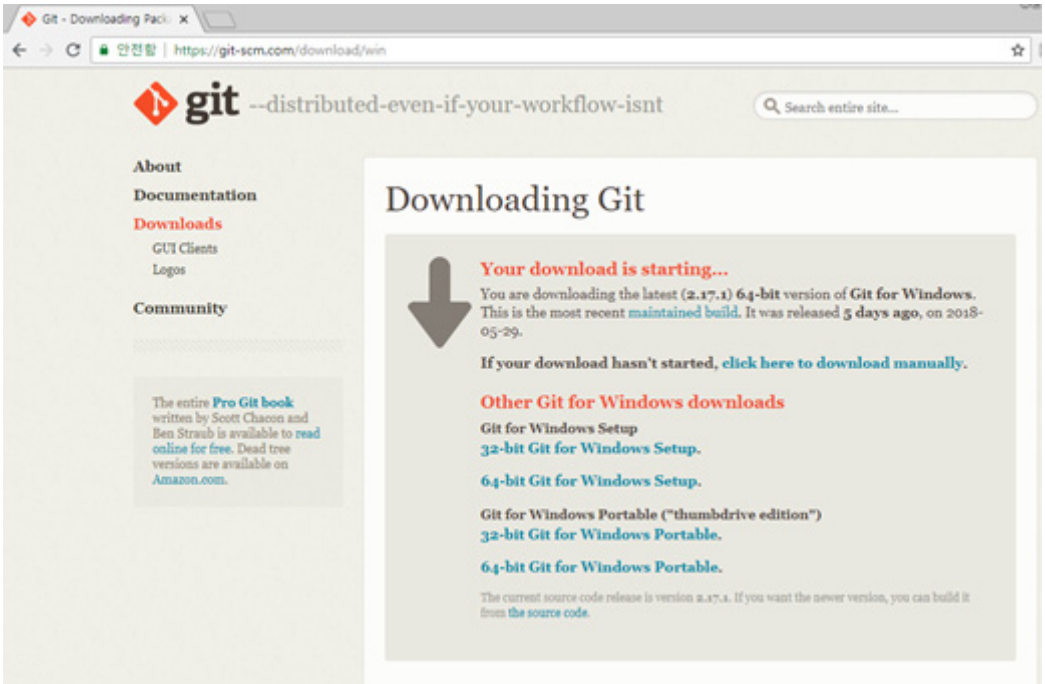


[그림 1-15] 단위 테스트 도구 설치 확인 화면 예시

3 형상관리 도구를 설치한다.

형상관리 도구는 CVS, SVN, Git 등 다양한 형상관리 도구가 사용되나, 본 예제에서는 Git를 형상관리 도구로 설치한다.

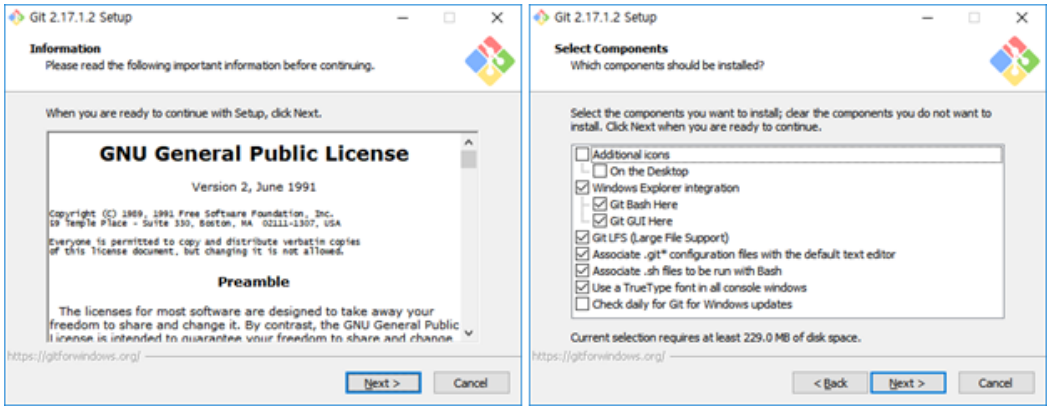
1. 형상관리 도구를 다운로드한다.



[그림 1-16] 형상관리 도구 다운로드 화면 예시(Git 예시)

2. 형상관리 도구를 설치한다.

다운로드한 형상관리 도구를 실행하여 안내에 따라 설치한다.



[그림 1-17] 형상관리 도구 설치 화면

3. 설치한 형상관리 도구의 환경을 설정한다.

(1) 설치를 완료한 이후에는 사용자 정보를 설정한다.

(2) 사용자 정보는 사용자의 이름과 e-mail을 최우선으로 설정한다.

(3) 사용자 이름 설정: git config --global user.name "사용자 이름"

(4) 사용자 e-mail 설정: git config --global user.email 이메일 주소

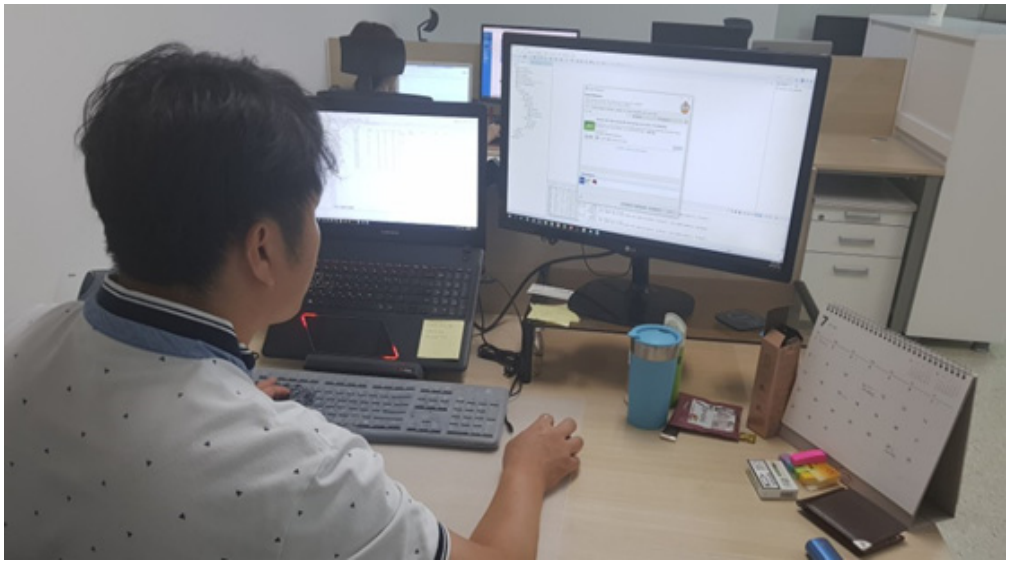
```
root@test-server:/# git config --global user.name "John"
root@test-server:/# git config --global user.email John@gmail.com
```

[그림 1-18] 사용자 이름 및 이메일 설정

(5) 설정 확인: git config --list

```
root@test-server:/# git config --list
user.name=John
user.email=John@gmail.com
```

[그림 1-19] 사용자 이름 및 이메일 설정 확인



[그림 1-20] 개발환경을 구축하는 모습

수행 tip

- 개발에 필요한 하드웨어와 소프트웨어를 파악하여 해당 명세대로 개발환경을 구축하고, 형상관리 지침에 따라 형상관리 환경을 구축할 수 있도록 한다.

학습 1 교수·학습 방법

교수 방법

- 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어 제품 유형에 대해 설명하고, 각 제품의 용도에 대해 제시한 후 수업을 진행한다.
- 여러 개발 도구들에 대한 이해 및 이전 사용 경험 여부에 대해 파악하여 개발 도구를 선정하여 수업을 진행한다.
- 여러 개발 도구의 매뉴얼 및 홈페이지 등을 방문하여 다양한 자료를 제시한 후 설명한다.
- 다양한 개발 도구의 설치 과정에 대해 단계적 실습이 이루어질 수 있도록 지도한다.
- 형상관리 방침에 포함되는 내용에 대해 설명하고, 형상관리 환경 구축 방안을 정리하여 설명한다.

학습 방법

- 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어 제품 유형에 대해 이해하고, 각 제품의 용도에 대해 학습한다.
- 여러 개발 도구들에 대한 이해 및 이전 사용 경험을 바탕으로 선정된 개발 도구로 실습한다.
- 여러 개발 도구의 매뉴얼 및 홈페이지에서 다양한 자료를 다운받아 학습한다.
- 다양한 개발 도구들의 설치 과정에 대해 단계적으로 실습하여 과정에 대해 학습한다.
- 형상관리 환경 구축을 위한 실습을 해 봄으로써 구축 필요성에 대해 학습한다.

학습 1 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
개발환경 준비	- 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어의 필요 사항을 검토하고, 이에 따라 개발환경에 필요한 준비를 수행할 수 있다.			
개발환경 구축	- 응용소프트웨어 개발에 필요한 하드웨어 및 소프트웨어를 설치하고 설정하여 개발환경을 구축할 수 있다.			
	- 사전에 수립된 형상관리 방침에 따라 운영 정책에 부합하는 형상관리 환경을 구축할 수 있다.			

평가 방법

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
개발환경 준비	- 목표 시스템의 환경 요구 사항에 대한 분석 능력			
	- 개발 도구들의 역할에 대한 파악도			
개발환경 구축	- 개발 도구들의 설치 및 활용에 대한 파악도			
	- 형상관리 지침의 이해와 환경 구축 능력			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
개발환경 준비	- 목표 시스템의 환경 요구 사항에 대한 분석 능력			
	- 개발 도구들의 역할에 대한 파악도			
개발환경 구축	- 개발 도구들의 설치 및 활용에 대한 파악도			
	- 형상관리 지침의 이해와 환경 구축 능력			

피드백

1. 포트폴리오
   - 제출한 보고서를 평가한 후에 주요 사항에 대하여 의견을 제시한다.
   - 개발환경 구축 절차에 대한 이해도를 확인하여 미비 사항에 대한 내용을 설명한다.
   - 형상관리 환경 구축 절차에 대한 정확성을 파악하여 미비 사항에 대한 내용을 설명한다.
   - 개발환경 구축에 대한 학습자의 평가 결과가 '상'인 경우 모범 사례로 공유한다.
   - 개발환경 구축에 대한 학습자의 평가 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
   - 개발환경 구축에 대한 학습자의 경가 결과가 '하'인 경우 모범 사례를 참조하여 다시 구축할 수 있도록 유도한다.

2. 평가자 체크리스트
   - 실습 과정에서 평가한 후에 개선해야 할 사항 등을 정리하여 설명한다.
   - 개발환경에 필요한 하드웨어와 소프트웨어의 누락 여부를 점검하여 보완할 수 있도록 유도한다.
   - 개발환경에 필요한 개발 도구를 학습 방향대로 구축하였는지를 점검한 후 성취수준이 낮은 학습자에게 실무 자료를 제시하여 이해를 높일 수 있게 한다.
   - 형상관리 환경 구축 후 정상적인 형상관리가 되는지를 확인하여 비정상 동작 시 그 원인을 파악한 후 학습자에게 설명한다.

2-1. 공통 모듈 구현

학습 목표	• 공통 모듈의 상세 설계를 기반으로 프로그래밍 언어와 도구를 활용하여 업무 프로세스 및 서비스의 구현에 필요한 공통 모듈을 작성할 수 있다. • 소프트웨어 측정 지표 중 모듈 간의 결합도는 줄이고 개별 모듈들의 내부 응집도를 높인 공통 모듈을 구현할 수 있다.

필요 지식

1 공통 모듈에 대한 이해

공통 모듈은 정보 시스템 구축 시 자주 사용하는 기능들로서 재사용이 가능하게 패키지로 제공하는 독립된 모듈을 의미한다.

1. 재사용(Reuse)

목표 시스템의 개발 시간 및 비용 절감을 위하여 검증된 기능을 파악하고 재구성하여 시스템에 응용하기 위한 최적화 작업이다.

〈표 2-1〉 재사용 범위에 따른 분류

분류	설명
함수와 객체 재사용	클래스(Class)나 함수(Function) 단위로 구현한 소스코드를 재사용한다.
컴포넌트 재사용	컴포넌트(Component) 단위로 재사용하며, 컴포넌트의 인터페이스를 통해 통신한다.
애플리케이션 재사용	공통된 기능을 제공하도록 구현된 애플리케이션과의 통신으로 기능을 공유하여 재사용한다.

2. 공통 모듈의 종류

목표 시스템을 구축하기 위하여 구현하는 공통 모듈들은 공통 모듈의 설계를 기반으로 프로그래밍 언어와 도구를 활용하여 작성한다.

〈표 2-2〉 공통 모듈의 분류

구분	분류	기능
공통 기술	사용자 디렉터리 /통합 인증	사용자 통합 인증, 로그인
	보안	실명 확인, 역할/권한 관리, 암호화/복호화
	통계/리포팅	통계
	협업	게시판, 동호회, 일정 관리, 전자 우편, 주소록/명함록, 문자 메시지, 전자 결재
	사용자 지원	사용자 관리, 약관 관리, 개인화, 온라인 참여, 온라인 헬프, 정보 제공/알림
	시스템 관리	공통 코드 관리, 메뉴 관리, 프로그램 관리, 로그 관리, 배치 관리, 시스템 관리, 장애 관리
	시스템/서비스 연계	시스템 연계
	디지털 자산 관리	지식 관리
모바일 공통 기술	협업	모바일 실시간 공지 서비스, 오프라인 웹 서비스, 모바일 위치 정보 연계 서비스, 모바일 기기 식별, 모바일 메뉴
	사용자 지원	모바일 메뉴 관리
	시스템/서비스 연계	시스템 연계
	디지털 자산 관리	모바일 차트/그래프, 모바일 사진 앨범, 모바일 멀티미디어제어
요소 기술		달력, 웹 에디터, 인쇄/출력, 쿠키/세션, 포맷/계산/변환, 메시지 처리, 인터페이스/화면, 시스템, 파일 업로드/다운로드, 이중 등록 방지, 지역화 처리, 웹 소켓 등

출처: 한국정보화진흥원 표준프레임워크센터(2018. 03.), 정보 시스템 구축 발주자를 위한 표준 프레임워크 적용 가이드 Ver. 3.7

2 소프트웨어 모듈 응집도

응집도는 모듈 내부에서 구성 요소 간에 밀접한 관계를 맺고 있는 정도로 평가되며, 응집도가 높을수록 필요한 요소들로 구성되어 있고 낮을수록 관련이 적은 요소들로 구성되어있다.

1. 응집도의 유형

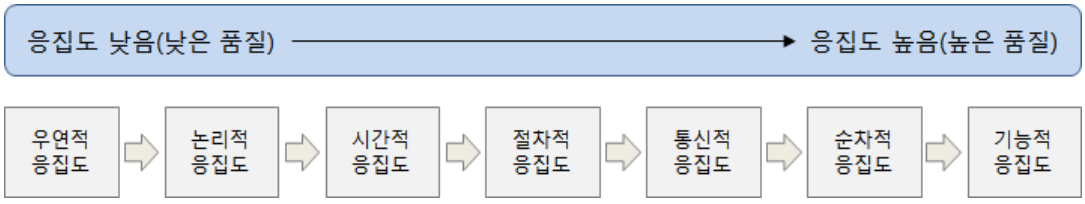
한 모듈 안의 기능들의 연관성 정도를 나타낸다.

〈표 2-3〉 응집도 유형

응집도 유형	설명
기능적 응집도 (Functional Cohesion)	모듈 내부의 모든 기능이 단일한 목적을 위해 수행되는 경우이다. 구조도 최하위 모듈에서 많이 발견된다.
순차적 응집도 (Sequential Cohesion)	모듈 내에서 한 활동으로부터 나온 출력값을 다른 활동의 입력값으로 사용하는 경우이다. 예) 행렬 입력 후 그 행렬의 역행렬을 구해서 이를 출력
통신적 응집도 (Communication Cohesion)	동일한 입력과 출력을 사용하여 다른 기능을 수행하는 활동들이 모여 있을 경우이다. 예) 같은 입력 자료를 사용하여 A를 계산한 후 B를 계산
절차적 응집도 (Procedural Cohesion)	모듈이 다수의 관련 기능을 가질 때 모듈 안의 구성 요소들이 그 기능을 순차적으로 수행할 경우이다. 예) Restart루틴: 총계 출력하고 화면을 지우고 메뉴를 표시
시간적 응집도 (Temporal Cohesion)	연관된 기능이라기보다는 특정 시간에 처리되어야 하는 활동들을 한 모듈에서 처리하는 경우이다. 예) 초기치 설정, 종료 처리 등
논리적 응집도 (Logical Cohesion)	유사한 성격을 갖거나 특정 형태로 분류되는 처리 요소들이 한 모듈에서 처리되는 경우이다. 예) 오류 처리: 자판기의 잔액 부족, 음료수 부족 출력 처리: 직원 인사 정보 출력, 회계 정보 출력
우연적 응집도 (Coincidental Cohesion)	모듈 내부의 각 구성 요소들이 연관이 없을 경우이다. 모듈화 장점이 없다. 유지 보수 작업이 어렵다.

2. 응집도와 품질

목표 시스템의 기준에 따라 모듈을 구성할 수 있으나, 품질 측면에서는 응집도의 유형 중 기능적 응집도가 가장 높고 우연적 응집도가 가장 낮다.



[그림 2-1] 응집도와 품질

3 소프트웨어 모듈 결합도

결합도는 모듈과 모듈 간의 관련성 정도를 나타내며, 관련이 적을수록 모듈의 독립성이 높아져 모듈 간 영향이 적어지게 된다.

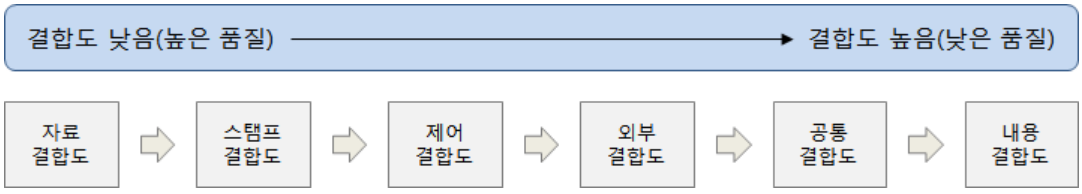
1. 결합도의 유형

〈표 2-4〉 결합도 유형

응집도 유형	설명
자료 결합도 (Data Coupling)	모듈 간의 인터페이스로 전달되는 파라미터를 통해서만 모듈 간의 상호 작용이 일어나는 경우이다. 예) 제곱근을 계산하는 함수로 하나의 정수를 전달
스탬프 결합도 (Stamp Coupling)	모듈 간의 인터페이스로 배열이나 오브젝트(Object), 스트럭처(Structure) 등이 전달되는 경우이다.
제어 결합도 (Control Coupling)	단순 처리할 대상인 값만 전달되는 게 아니라 어떻게 처리를 해야 한다는 제어 요소가 전달되는 경우이다.
외부 결합도 (External Coupling)	다수의 모듈이 모듈 밖에서 도입된 데이터, 프로토콜, 인터페이스 등을 공유할 때 발생하는 경우이다.
공통 결합도 (Common Coupling)	파라미터가 아닌 모듈 밖에 선언되어 있는 전역 변수를 참조하고 전역 변수를 갱신하는 식으로 상호 작용하는 경우이다. 예) 전역 변수
내용 결합도 (Content Coupling)	다른 모듈 내부에 있는 변수나 기능을 다른 모듈에서 사용하는 경우이다.

2. 결합도와 품질

목표 시스템의 환경에 따라 모듈 구성의 결합 정도를 결정할 수 있으나, 품질 측면에서 결합도의 유형 중 자료 결합도의 품질이 가장 높고, 내용 결합도의 품질이 가장 낮다.



[그림 2-2] 결합도와 품질

수행 내용 / 공통 모듈 구현하기

재료·자료

- 공통 모듈 관리 대장, 공통 모듈 프로그램 설계서
- 요구 사항 정의서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

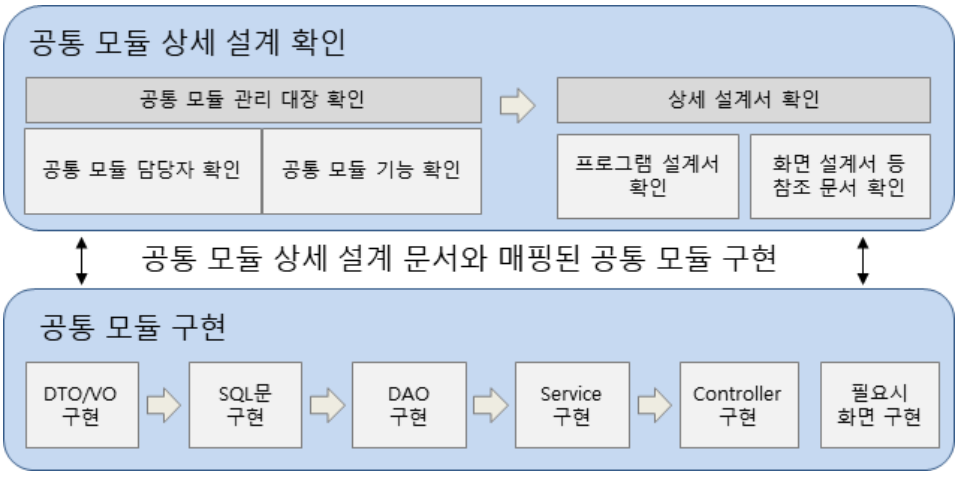
안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 공통 모듈 설계 산출물에 대한 이해가 선행되어야 한다.
- 프로그램 개발 경험과 SQL 작성 경험이 필요하다.

수행 순서

1 공통 모듈의 상세 설계를 기반으로 작성해야 할 공통 모듈을 확인한다.

공통 모듈을 작성하기 위해서는 애플리케이션 설계 단계에서 작성한 공통 모듈 관리 대장을 통해 작성해야 할 공통 모듈들을 확인하고 공통 모듈별 프로그램 설계서를 확인한다.



[그림 2-3] 공통 모듈 상세 설계를 기반으로 공통 모듈 구현 순서

1. 공통 모듈 관리 대장을 확인한다.

공통 모듈 관리 대장은 설계 단계에서 작성된 구현해야 할 전체 시스템의 공통 모듈 목록 산출물이다.

(1) 모듈 담당자 항목에서 담당해야 할 공통 모듈을 확인한다.

아래 예시는 김길동 담당자가 작성해야 할 공통 모듈을 확인한 예시이다.

공통 모듈 관리 대장

순번	ID	시스템	기능	설명	엔티티	입력	담당자	출력 (정상)	출력 (에러)	입력항목 설명	출력항목 설명
1	CMM_CODE_LIST_READ	인사	공통 코드 조회	조회 코드에 해당하는 코드 정보 제공	코드 정보	코드	김길동	[200] 코드, 코드 명	[204] 코드 없음 [500] 서버 에러	코드	1. 코드 정보를 배열로 제공 2. Code 204는 코드 없음
2	CMM_MENU_LIST_READ	인사	메뉴권한	직원별 메뉴 접근제어	메뉴 정보	사번	홍길동	[200] 메뉴 코드, 메뉴 명	[401] 권한 없음 [500] 서버 에러	사번 필수 (15자리)	1. 사번으로 조회된 메뉴 정보 2. Code 401은 권한 없음
3	CMM_MAIL_INFO_SEND	인사	메일발송	메일발송 기능 제공	메일 정보	보내는 메일, 받는 메일, 제목, 내용	홍길동	[200] 우편번호, 도로명 주소	[204] 주소 없음 [500] 서버 에러	시군구, 도로명 필수	1. 도로명으로 조회된 주소 정보 2. Code 204는 주소 없음

[그림 2-4] 공통 모듈 관리 대장에서의 담당자 확인 예시

(2) 공통 모듈의 기능을 확인한다.

작성해야 할 공통 모듈의 기능을 확인하고 상세 설계 산출물을 확인하기 위해 ID를 확인한다.

공통 모듈 관리 대장

순번	ID	시스템	기능	설명	엔티티	입력	담당자	출력 (정상)	출력 (에러)	입력항목 설명	출력항목 설명
1	CMM_CODE_LIST_READ	인사	공통 코드 조회	조회 코드에 해당하는 코드 정보 제공	코드 정보	코드	김길동	[200] 코드, 코드 명	[204] 코드 없음 [500] 서버 에러	코드	1. 코드 정보를 배열로 제공 2. Code 204는 코드 없음
2	CMM_MENU_LIST_READ	인사	메뉴권한	직원별 메뉴 접근제어	메뉴 정보	사번	홍길동	[200] 메뉴 코드, 메뉴 명	[401] 권한 없음 [500] 서버 에러	사번 필수 (15자리)	1. 사번으로 조회된 메뉴 정보 2. Code 401은 권한 없음
3	CMM_MAIL_INFO_SEND	인사	메일발송	메일발송 기능 제공	메일 정보	보내는 메일, 받는 메일, 제목, 내용	홍길동	[200] 우편번호, 도로명 주소	[204] 주소 없음 [500] 서버 에러	시군구, 도로명 필수	1. 도로명으로 조회된 주소 정보 2. Code 204는 주소 없음

[그림 2-5] 공통 모듈 관리 대장에서의 기능 확인 예시

2. 공통 모듈 프로그램 설계서를 확인한다.

공통 모듈 관리 대장 산출물에서의 ID와 일치하는 프로그램 설계서 산출물에서의 프로그램 ID를 확인한다.

프로그램 설계서

CodeController

프로그램 ID	CMM_CODE_LIST_READ	PACKAGE	com.ncs.controller
EXTENDS	N/A		
IMPLEMENTS	N/A		
IMPORT	com.ncs.service.selectCodeByGubun com.ncs.service.selectCodeByPCode		
프로그램 설명	코드의 구분값과 부모코드를 입력 시 관련 코드들을 반환하는 Class		

[속성]

Name	visibility	Type	Default Value	설명
success	private	String		코드 200 출력
none	private	String		코드 204 출력
error	private	String		코드 500 출력

[Methods]

Name	selectCodeByGubun
Visibility	public
Parameter	CodeVo
Return	json
설명	구분 값으로 해당 코드들을 조회하기 위해 서비스 메소드를 호출한다. [com.ncs..service.selectCodeByGubun]

Name	selectCodeByPCode
Visibility	public
Parameter	CodeVo
Return	json
설명	코드로 하위 코드들을 조회하기 위해 서비스 메소드를 호출한다. [com.ncs.service.selectCodeByGubun]

[그림 2-6] 공통 모듈 관리 대장에서의 ID와 일치하는 프로그램 설계서 예시

2 공통 모듈의 상세 설계 산출물을 기반으로 공통 모듈을 구현한다.

1. 공통 모듈을 구현하기 위한 I/O 오브젝트(DTO/VO)를 정의한다.

공통 코드 조회를 위해 필요한 변수들을 정의하여 계층 간 데이터 교환을 위해 사용한다.

DTO(Data Transfer Object) 또는 VO(Value Object)라고 명칭한다.

```java
package com.ncs.vo;

public class CodeVo {

    private String gubun;

    private String code;

    private String name;

    private String p_code;

    private String etc;

    public String getGubun() {
        return gubun;
    }

    public void setGubun(String gubun) {
        this.gubun = gubun;
    }

    public String getCode() {
        return code;
    }

    public void setCode(String code) {
        this.code = code;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getP_code() {
        return p_code;
    }

    public void setP_code(String p_code) {
        this.p_code = p_code;
    }
    ...
}
```

[그림 2-7] DTO/VO 구문 예시

2. 공통 모듈을 구현하기 위한 SQL을 작성한다.

공통 코드 조회를 위한 SQL 쿼리문을 마이바티스(Mybatis) XML 파일로 구현한 예제이다.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.ncs.sql">

     <select id="selectCodeByGubun" parameterType="com.ncs.vo.CodeVo" resultType="com.ncs.vo.CodeVo">
        SELECT
            CODE
            , NAME
        FROM TBL_CODE
        WHERE GUBUN = #{gubun}
    </select>
```

[그림 2-8] 구분으로 공통 코드를 조회하는 SQL 구문 예시

```xml
    <select id="selectCodeByPCode" parameterType="com.ncs.vo.CodeVo" resultType="com.ncs.vo.CodeVo">
        SELECT
            CODE
            , NAME
        FROM TBL_CODE
        WHERE GUBUN = #{gubun}
        AND P_CODE = #{p_code}
    </select>

</mapper>
```

[그림 2-9] 부모 코드로 공통 코드를 조회하는 SQL 구문 예시

3. 데이터 접근 객체(DAO: Data Access Object)를 구현한다.

DAO에서는 SQL문을 구현한 XML파일에서의 해당 ID값을 호출하여 데이터를 조회하거나 조작한다.

```java
package com.ncs.dao;

import java.util.List;

import org.apache.ibatis.session.SqlSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.ncs.vo.CodeVo;

@Repository("codeDao")
public class CodeDao {

    @Autowired
    private SqlSession sqlSession;

    //DB에서 공통 코드 조회(구분)
    public List<CodeVo> selectCodeByGubun(CodeVo codeVo) throws Exception {
        return sqlSession.selectList("com.ncs.sql.selectCodeByGubun", codeVo);
    }

    //DB에서 공통 코드 조회(부모코드)
    public List<CodeVo> selectCodeByPCode(CodeVo codeVo) throws Exception {
        return sqlSession.selectList("com.ncs.sql.selectCodeByPCode", codeVo);
    }

}
```

[그림 2-10] 공통 코드를 조회하는 DAO 구문 예시

4. 서비스(Service) 클래스를 구현한다.

동일한 시스템 내에서 사용되는 결합도가 낮은 공통 모듈을 구현하기 위해 스탬프

결합도수준의 서비스를 구현한다.

```java
package com.ncs.service;

import java.util.List;
import javax.annotation.Resource;
import org.springframework.stereotype.Service;

import com.ncs.dao.CodeDao;
import com.ncs.vo.CodeVo;

@Service("codeService")
public class CodeService {

    @Resource(name = "codeDao")
    private CodeDao codeDao;

    //공통 코드 조회(구분)
    public List<CodeVo> selectCodeByGubun(CodeVo codeVo) throws Exception {
        return codeDao.selectCodeByGubun(codeVo);
    }

    //공통 코드 조회(부모코드)
    public List<CodeVo> selectCodeByPCode(CodeVo codeVo) throws Exception {
        return codeDao.selectCodeByPCode(codeVo);
    }
}
```

[그림 2-11] 공통 코드를 조회하는 Service 구문 예시

5. 컨트롤러(Controller) 클래스를 구현한다.

시스템 내부 및 외부에서 사용할 수 있도록 REST API를 통한 공통 모듈 구현 사례이다.

```java
    /**
     * 구분으로 코드 조회
     * @param codeVo
     * @return
     */
    @PostMapping(value = "/code/gubun")
    public ResponseEntity<List<CodeVo>> selectPostByDong(@RequestBody CodeVo codeVo) {

        try {

            //공통 코드 조회 Service 호출
            List<CodeVo> codeList = codeService.selectCodeByGubun(codeVo);

            //코드가 존재 한 경우(성공 코드 : 200)
            if(codeList != null) {
                return new ResponseEntity<List<CodeVo>>(codeList, HttpStatus.OK);
            } else { //코드가 존재 하지 않는 경우(실패 코드 : 204)
                return new ResponseEntity<List<CodeVo>>(codeList, HttpStatus.NO_CONTENT);
            }


        } catch(Exception e) {
            logger.error(e.getMessage());

            //에러가 발생 한 경우(실패 코드 : 417)
            return new ResponseEntity<List<CodeVo>>(HttpStatus.INTERNAL_SERVER_ERROR);
        }
    }
```

[그림 2-12] 구분으로 공통 코드를 조회하는 REST API Controller 구문 예시

```java
58     /**
59      * 부모코드로 하위 코드 조회
60      * @param codeVo
61      * @return
62      */
63     @PostMapping(value = "/code/p_code")
64     public ResponseEntity<List<CodeVo>> selectCodeByPCode(@RequestBody CodeVo codeVo) {
65
66         try {
67
68             //공통 코드 조회 Service 호출
69             List<CodeVo> codeList = codeService.selectCodeByPCode(codeVo);
70
71             //코드가 존재 한 경우(성공 코드 : 200)
72             if(codeList != null) {
73                 return new ResponseEntity<List<CodeVo>>(codeList, HttpStatus.OK);
74             } else { //코드가 존재 하지 않는 경우(실패 코드 : 204)
75                 return new ResponseEntity<List<CodeVo>>(codeList, HttpStatus.NO_CONTENT);
76             }
77
78
79         } catch(Exception e) {
80             logger.error(e.getMessage());
81
82             //에러가 발생 한 경우(실패 코드 : 417)
83             return new ResponseEntity<List<CodeVo>>(HttpStatus.EXPECTATION_FAILED);
84         }
85     }
86 }
```

[그림 2-13] 부모 코드로 공통 코드를 조회하는 REST API Controller 구문 예시

수행 tip

- 공통 모듈 상세 설계 산출물을 통해 구현해야 할 공통 모듈을 파악해야 한다.
- 공통 모듈의 특성과 명세를 파악하여 개발언어와 도구를 활용하여 해당 모듈이 정상적으로 작동할 수 있도록 구현할 수 있어야 한다.

2-2. 공통 모듈 테스트

학습 목표	• 개발된 공통 모듈의 내부 기능과 제공하는 인터페이스에 대해 테스트할 수 있는 테스트 케이스를 작성하고, 단위 테스트를 수행하기 위한 테스트 조건을 명세화할 수 있다.

필요 지식 /

1 테스트 케이스의 이해

1. 테스트 케이스(Test Case)의 개념

   테스트 케이스란 요구 사항을 준수하는지 검증하기 위하여 테스트 조건, 입력값, 예상출력값 및 수행한 결과 등 테스트 조건을 명세한 것이다.

2. 테스트 케이스의 구성 요소

   〈표 2-5〉 테스트 케이스 구성 요소(IEEE 829)

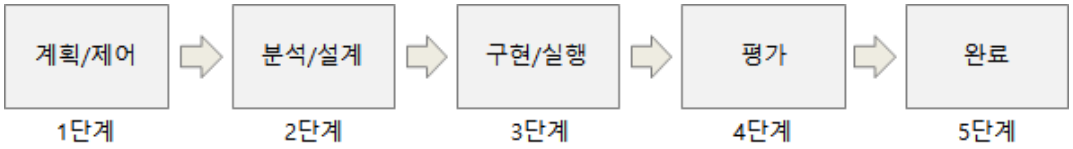
항목	설명
식별자(Identifier)	항목 식별자, 일련번호
테스트 항목(Test Item)	테스트할 모듈 또는 기능
입력 명세(Input Specification)	입력값 또는 테스트 조건
출력 명세(Output Specification)	테스트 케이스 실행 시 기대되는 출력값 결과
환경 설정(Environmental Needs)	테스트 수행 시 필요한 하드웨어나 소프트웨어 환경
특수 절차 요구 (Special Procedure Requirement)	테스트 케이스 수행 시 특별히 요구되는 절차
의존성 기술 (Inter-case Dependencies)	테스트 케이스 간의 의존성

2 테스트 프로세스

테스트 프로세스는 테스트 수행과 관련된 활동들이 의도된 테스트 목적과 조건을 달성할 수 있도록 도와주는 역할을 한다.

1. 테스트 프로세스의 구성

   테스트 프로세스는 계획 및 제어, 분석 및 설계, 구현 및 실행, 평가, 완료 단계로 구성된다.



[그림 2-14] 테스트 프로세스 5단계

2. 테스트 프로세스의 단계

(1) 계획 및 제어

테스트의 목표와 목적을 달성하기 위해 필요한 활동을 계획하고, 계획에 준수하여 진행되고 있는지 지속적인 제어 활동을 하는 단계이다.

(2) 분석 및 설계

일반적이고 추상적인 테스트의 목적을 구체화하여 테스트 시나리오와 테스트 케이스로 변환하는 활동 단계이다.

(3) 구현 및 실행

테스트를 효과적이고 효율적으로 수행하기 위해 테스트 케이스들을 조합하고 테스트 수행 시 필요한 정보들을 포함하는 테스트 프러시저를 명세하는 활동 단계이다.

(4) 평가

계획 단계에서 정의하였던 테스트 목표에 맞게 테스트가 수행되었는지를 평가하고 진행 상황을 기록하는 활동 단계이다.

(5) 완료

테스트 수행 시 명세했던 조건들을 수집하고 테스트 수행 시 발생했던 사항 및 경험들을 축적하는 활동 단계이다.

수행 내용 / 공통 모듈 테스트하기

재료·자료

- 공통 모듈 관리 대장, 공통 모듈 프로그램 설계서
- 요구 사항 정의서
- 개발된 공통 모듈 산출물

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

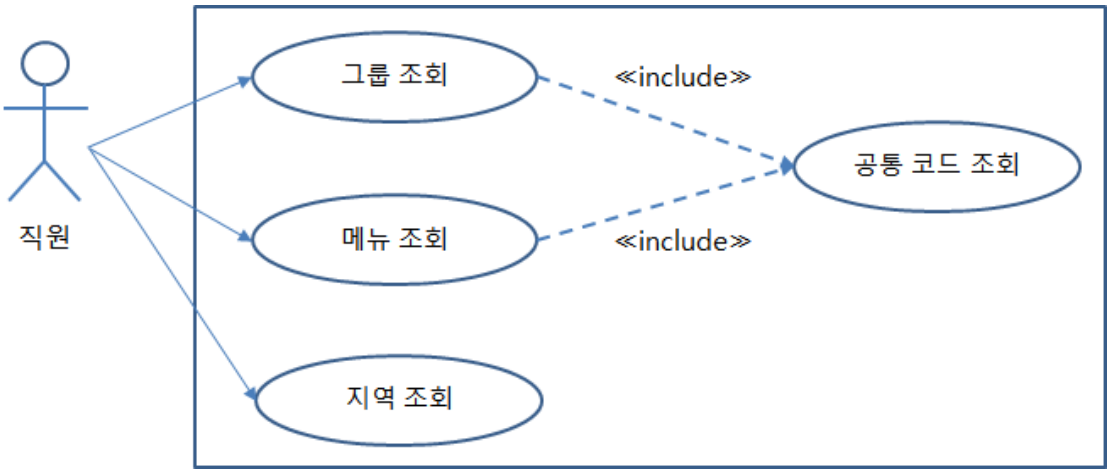
안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 공통 모듈 설계 산출물에 대한 이해가 선행되어야 한다.
- 목표 시스템의 요구 사항 범위에 대한 이해가 필요하다.

수행 순서

1 단위 테스트 케이스 작성을 위한 참조 문서를 수집한다.

공통 모듈 상세 설계 산출물과 요구 사항 정의서, 요구 사항 명세서 등 분석 및 설계 문서를 수집한다.



[그림 2-15] 공통 모듈로 식별된 유스 케이스 참고 예시

2 단위 테스트 케이스를 작성한다.

1. 단위 테스트 방식을 결정한다.
   개발된 공통 모듈을 실행하여 테스트하는 동적 테스트와 수동 또는 자동화 도구를 사용하여 검토하는 정적 테스트 중 적절한 테스트 방식을 결정한다.

2. 단위 테스트의 범위를 결정한다.
   (1) 공통 모듈의 범위 및 상위 요구 사항의 범위에 따라 테스트 범위를 판단한다.
   (2) 테스트 수행 횟수를 결정한다.
   (3) 테스트의 결과를 검증하기 위한 결괏값 산출 방식을 결정한다.

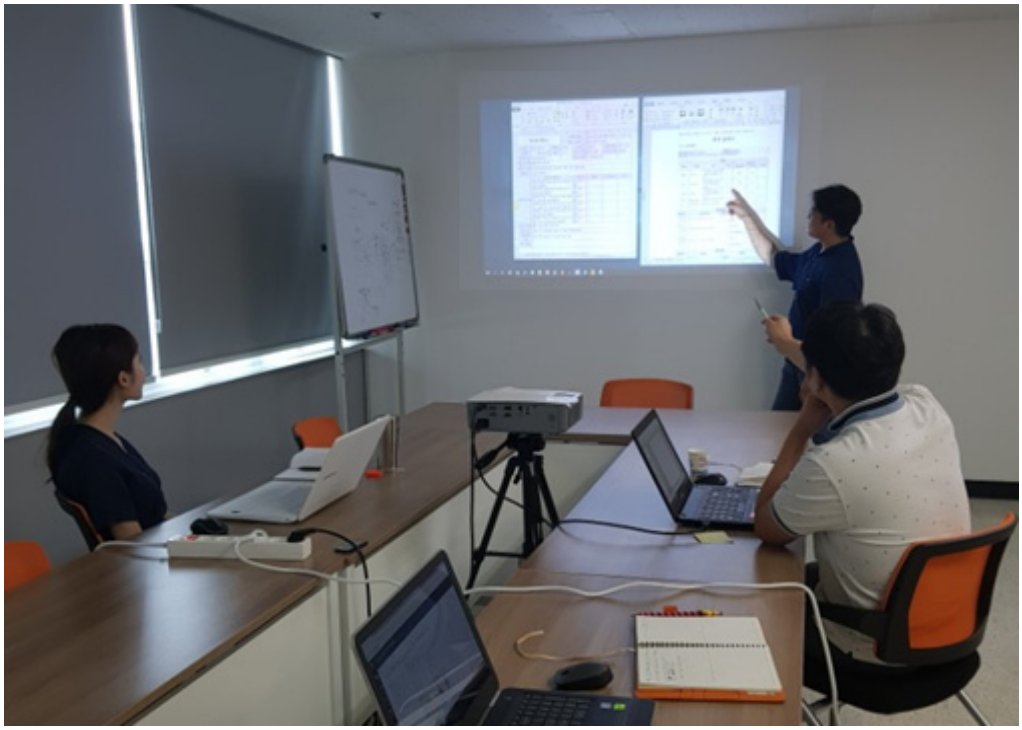
3. 단위 테스트의 방식과 범위에 따라 테스트 케이스를 작성한다.
   (1) 테스트 케이스 작성에 필요한 항목을 작성한다.
      (가) 테스트 단계명, 작성자, 작성일, 승인자, 문서 버전 등의 항목을 작성한다.
         단위 테스트, 통합 테스트, 시스템 테스트 등 테스트의 단계와 작성자, 승인자, 문서 버전, 테스트 범위, 테스트 조직 등 공통 항목을 작성한다.
      (나) 테스트 ID를 작성한다.
         테스트 케이스를 식별하기 위한 고유한 테스트 ID를 작성한다.
      (다) 테스트의 목적 및 기능을 요약하여 작성한다.
         테스트 수행의 목적과 기능을 간략하게 요약하여 작성한다.
      (라) 입력값을 작성한다.
         테스트 수행 시 입력할 데이터를 작성한다.
      (마) 예상 결괏값을 작성한다.
         테스트 수행 후 기대되는 결과 데이터(값, 메시지, 화면 등)를 작성한다.
      (바) 테스트를 수행할 환경을 작성한다.
         테스트를 수행하는 물리적, 논리적 환경을 작성한다.
      (사) 테스트를 수행하기 위한 전제 조건을 작성한다.
         테스트 케이스 간의 의존성 및 선행 조건 등 고려 사항을 작성한다.
      (아) 성공/실패 기준을 정의한다.
         테스트의 성공과 실패를 결정하는 조건을 명확하게 작성한다.
      (자) 테스트 수행 시 필요한 특수 절차를 작성한다.
         테스트를 수행 시 사용자의 요구 사항이나 필요한 특수한 절차를 작성한다.
   (2) 정의된 항목 요소를 바탕으로 테스트 케이스를 작성한다.

테스트 케이스			프로젝트명		인사 관리 웹 애플리케이션 시스템 구축		
			대상 시스템		인사 관리 시스템		
단계	단위 테스트	작성자	김길동	승인자	김승인	문서 상태	최조 작성
작성일	2018-01-01	버전	V 1.0	테스트 범위	공통 모듈	테스트 조직	개발팀
테스트 ID	TC-CMM-001			테스트 일자	2018-07-27		
테스트 목적	공통 코드 테스트						
테스트 기능	구분 및 부모코드를 기준으로 공통 코드 조회 검증						
입력 값	구분, 부모코드						
테스트 단계	테스트 케이스			예상 값	중요도	성공/실패	비고
	구분 값 없이 구분 공통 코드 조회 함수를 호출한다.			204 코드 출력			
	부모 코드 값 없이 부모 공통 코드 조회 함수를 호출한다.			204 코드 출력			
	존재하지 않는 구분 값으로 구분 공통 코드 조회 함수를 호출한다.			204 코드 출력			
	존재하지 않는 부모 코드 값으로 부모 코드 공통 코드 조회 함수를 호출한다.			204 코드 출력			
	존재하는 구분 값으로 구분 공통 코드 조회 함수를 호출한다.			200 코드 출력			
	존재하는 부모 코드 값으로 부모 코드 공통 코드 조회 함수를 호출한다.			200 코드 출력			
테스트 환경	개발 서버, 형상 관리 서버						
전제 조건	공통 코드 테이블에 공통 코드들을 입력해야 함						
성공/실패 기준	예상 값이 정상적으로 출력 되면 성공						
특수 절차							

[그림 2-16] 공통 코드 단위 테스트 케이스 예시

3 작성된 단위 테스트 케이스를 내부 검토한다.

1. 단위 테스트 케이스의 적정성을 검토한다.
   PM(Project Manager), 아키텍처, 기획자, 개발자, 테스트 담당자 등 내부 이해관계자들이 작성된 테스트 케이스의 적정성을 검토한다.
2. 요구 사항에 충족할 수 있는 테스트 범위인지 검토한다.
   단위 테스트 케이스가 요구 사항에 충족할 수 있도록 작성되었는가에 대한 테스트 범위를 검토한다.
3. 공통 모듈은 다른 모듈에서 활용되므로 활용성이나 영향도 등을 고려하여 검토한다.



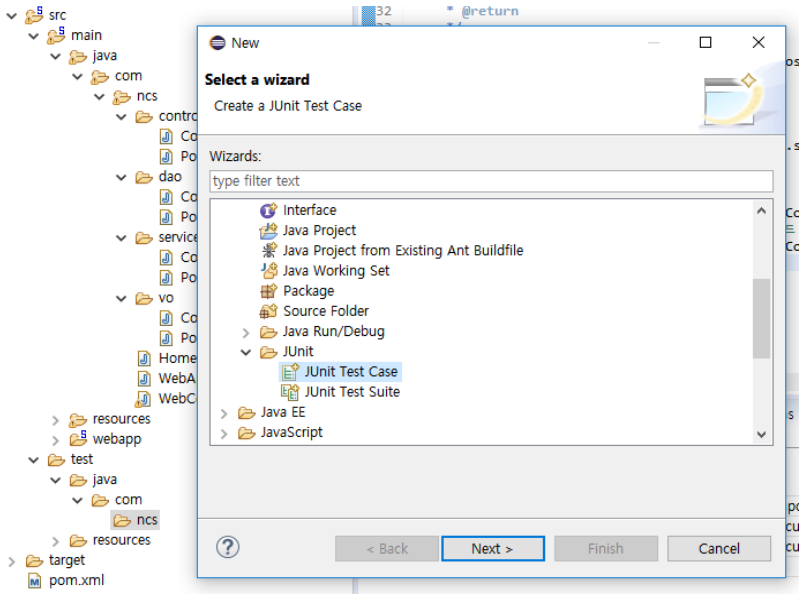
[그림 2-17] 단위 테스트 케이스의 적정성을 검토하는 모습 예시

4 작성된 단위 테스트 케이스를 고객에게 승인을 획득한다.

작성된 테스트 케이스를 고객 및 PM 등 이해관계자들에게 승인을 획득한다.

5 테스트를 명세하기 위한 테스트 도구를 설정한다.

1. 단위 테스트 도구를 사용하기 위한 Wizard를 선택한다.

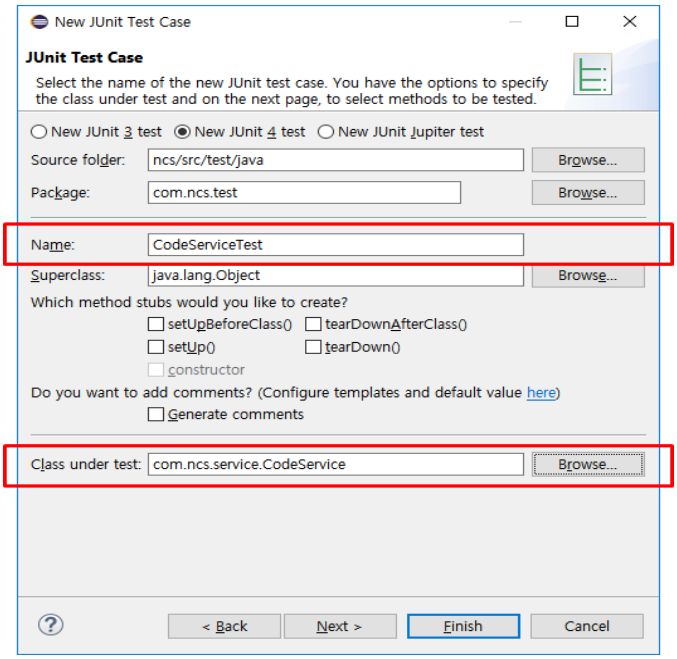


[그림 2-18] 단위 테스트 케이스 선택 예시(JUnit Test Case 예시)

2. 테스트 케이스(Test Case)의 이름을 입력하고 Next를 클릭한다.

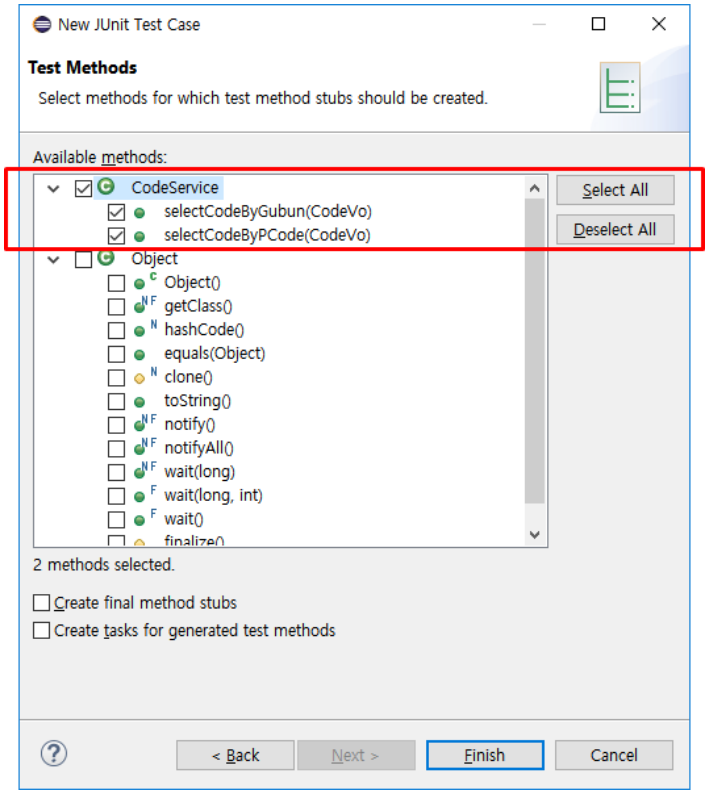
(1) 테스트 케이스 Name을 입력한다.

(2) 테스를 수행할 클래스를 선택한다.



[그림 2-19] 테스트 케이스 생성 예시

3. 테스트를 수행할 함수를 선택하고 Finish를 클릭한다.



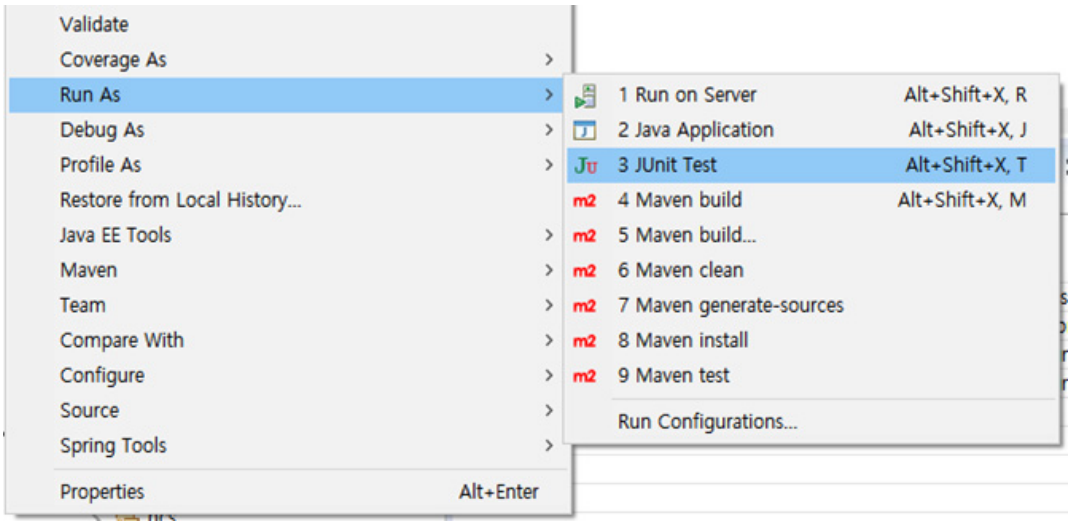
[그림 2-20] 테스트 케이스 생성 시 함수 선택 예시

4. 테스트 구문을 작성한다.

```
@Test
public void testSelectCodeByGubun() throws Exception {

    CodeVo codeVo = new CodeVo();
    codeVo.setGubun("MENU");;
    List<CodeVo> codeVoList = codeService.selectCodeByGubun(codeVo);

    assertNotNull(codeVoList);
}

@Test
public void testSelectCodeByPCode() throws Exception {

    CodeVo codeVo = new CodeVo();
    codeVo.setGubun("MENU");;
    codeVo.setP_code("0000");
    List<CodeVo> codeVoList = codeService.selectCodeByPCode(codeVo);
```

[그림 2-21] 테스트 구문 예시

5. 테스트 도구를 실행한다.



[그림 2-22] 테스트 도구 실행 화면 예시

6 테스트 결과를 명세화 한다.

테스트의 결과를 확인하여 테스트 케이스에 성공/실패 결과 및 필요 항목을 기록한다.

수행 tip

- 개발된 공통 모듈의 기능과 인터페이스 테스트를 위한 테스트 케이스를 작성하고, 그에 따른 단위 테스트를 수행할 수 있도록 한다.

학습 2 교수·학습 방법

교수 방법

- 공통 모듈의 상세 설계 내용을 이해하여 이를 개발언어를 활용하여 작성할 수 있는 절차를 구체적으로 제시한 후 내용에 대해 설명한다.
- 공통 모듈은 여러 애플리케이션에서 활용됨에 따라 최적화된 코드로 작성되어야 함을 인지시키며, 효율적 코드 작성의 필요성을 제시하고 해당 내용을 설명한다.
- 공통 모듈 식별을 위한 결합도와 응집도의 개념에 대해 구체적으로 설명하고, 이런 원리를 어떻게 공통 모듈에 적용하는지에 대한 내용을 중심으로 설명한다.
- 개발된 공통 모듈을 테스트할 수 있는 테스트 케이스 작성 기법에 대해 특성 및 차이점을 비교·분석하여 제시하고 설명한다.
- 공통 모듈 테스트를 위해 작성된 테스트 케이스를 효율적으로 테스트할 수 있는 도구 선정 시 고려 사항을 제시하고 설명한다.

학습 방법

- 공통 모듈 상세 설계서를 파악하여 개발언어로 작성할 수 있는 절차에 대해 학습한다.
- 공통 모듈의 중요성을 인지하여 효율적 코드로 작성될 수 있도록 고급 기법 습득의 필요성에 대해 인식하고 해당 내용을 학습한다.
- 공통 모듈 식별 원리인 결합도와 응집도에 대해 이해함으로써 모듈 작성 시 각 원칙에 충실한 코드를 작성할 수 있도록 학습한다.
- 개발된 공통 모듈 테스트를 위한 테스트 케이스 작성 기법에 대해 이해하고, 모듈 특성을 고려한 작성 기법 선정 및 코드 작성법에 대해 학습한다.
- 공통 모듈 테스트를 위해 작성된 테스트 케이스를 효율적으로 테스트할 수 있는 도구를 선정하고 활용하는 방법에 대해 학습한다.

학습 2 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
공통 모듈 구현	- 공통 모듈의 상세 설계를 기반으로 프로그래밍 언어와 도구를 활용하여 업무 프로세스 및 서비스의 구현에 필요한 공통 모듈을 작성할 수 있다.			
	- 소프트웨어 측정 지표 중 모듈 간의 결합도는 줄이고 개별 모듈들의 내부 응집도를 높인 공통 모듈을 구현할 수 있다.			
공통 모듈 테스트	- 개발된 공통 모듈의 내부 기능과 제공하는 인터페이스에 대해 테스트할 수 있는 테스트 케이스를 작성하고, 단위 테스트를 수행하기 위한 테스트 조건을 명세화할 수 있다.			

평가 방법

- 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
공통 모듈 구현	- 공통 모듈 명세에 따른 기능을 개발언어와 도구를 활용하여 개발된 모듈의 완전성			
	- 결합도를 줄이고 응집도를 높일 수 있는 공통 모듈 개발의 기본 원칙에 대한 파악도			
공통 모듈 테스트	- 공통 모듈의 기능과 인터페이스 테스트를 위한 테스트 케이스의 적정성			

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
공통 모듈 구현	- 공통 모듈 명세에 따른 기능을 개발언어와 도구를 활용하여 개발된 모듈의 완전성			
	- 결합도를 줄이고 응집도를 높일 수 있는 공통 모듈 개발의 기본 원칙에 대한 파악도			
공통 모듈 테스트	- 공통 모듈의 기능과 인터페이스 테스트를 위한 테스트 케이스의 적정성			

피드백

1. 평가자 체크리스트
   - 실습 과정을 체크리스트에 따라 평가한 후에 개선해야 할 사항 등을 정리한 후 학습자에게 설명한다.
   - 공통 모듈 명세를 파악하여 작성된 개발 모듈이 설계서대로 개발되었는지 확인하여 비정상 동작 시 그 문제점을 정리하여 개선 사항을 제시한다.
   - 결합도와 응집도의 개념에 대한 이해도를 파악하여 부족한 이해도에 대해 보충할 수 있도록 실무 자료를 제시한다.
   - 공통 모듈 테스트를 위한 테스트 케이스 작성 기법과 테스트 결과를 보고 미비 사항을 보완할 수 있도록 실무 자료를 제시한다.
2. 포트폴리오
   - 제출한 보고서를 평가한 후에 주요 사항에 대하여 의견을 제시한다.
   - 개발언어와 도구의 활용에 따른 공통 모듈 명세서에 정의된 기능이 정상적으로 동작하는지를 보고 비정상 동작의 원인과 조치 방안에 대해 설명한다.
   - 개발된 공통 모듈의 결합도와 응집도의 이해도를 확인하여 미비 사항을 설명한다.
   - 공통 모듈 명세를 파악하여 작성된 공통 모듈 작성 결과가 '상'인 경우 모범 사례로 공유한다.
   - 공통 모듈 명세를 파악하여 작성된 공통 모듈 작성 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
   - 공통 모듈 명세를 파악하여 작성된 공통 모듈 작성 결과가 '하'인 경우 모범 사례를 참조하여 다시 구축할 수 있도록 유도한다.

3-1. 업무 프로세스 확인

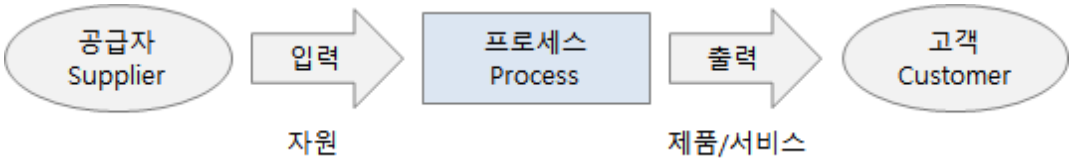
학습 목표 • 업무 프로세스 맵과 세부 업무 프로세스를 확인할 수 있다.

필요 지식 /

1 프로세스의 이해

1. 프로세스의 개념

프로세스란 개인이나 조직이 한 개 이상의 정보 자원의 입력을 통해 가치 있는 산출물을 제공하는 모든 관련 활동들의 집합이다.



[그림 3-1] 프로세스 모델

〈표 3-1〉 프로세스 모델의 구성 항목

항목	설명
고객	제품/서비스 또는 출력의 대상이 되는 개인이나 조직
프로세스	입력을 가치 있는 산출물로 변환시켜 출력하는 활동들
공급자	입력을 제공하는 개인이나 조직
입력	공급자에 의해 제공되는 정보 자원
출력	프로세스를 통해 고객에게 제공되는 가치 있는 제품/서비스

2. 프로세스의 구성 요소

프로세스의 필수 구성 요소에는 프로세스 책임자(Owner), 프로세스 맵(Map), 프로세스 Task, 프로세스 성과 지표, 프로세스 조직, 경영자의 리더십(Leadership) 등이 있다.

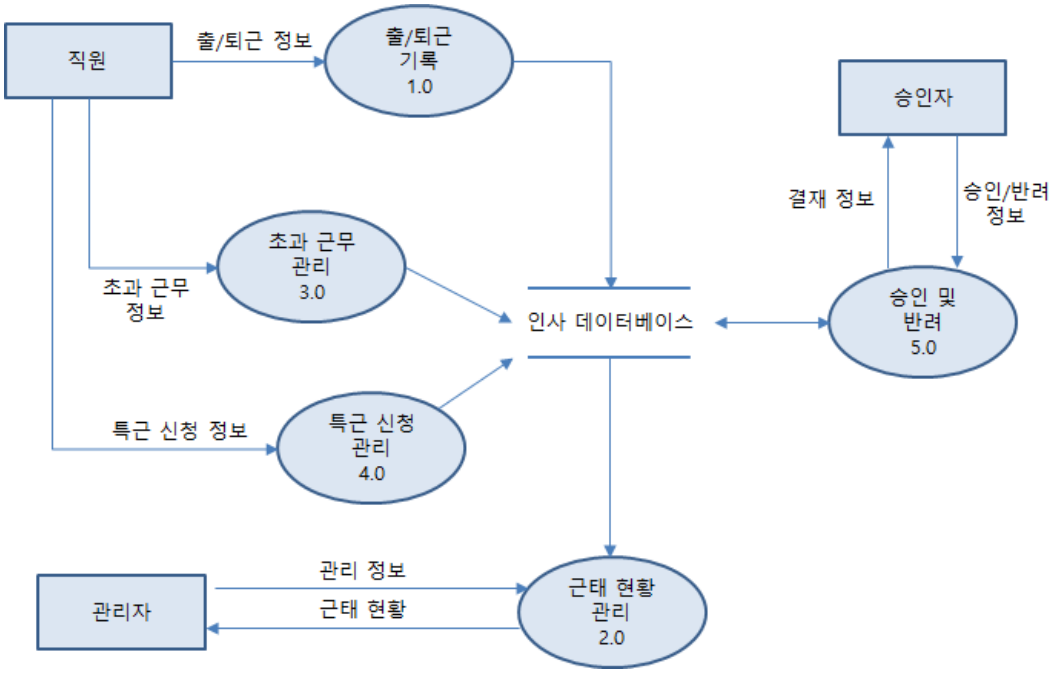
〈표 3-2〉 프로세스 구성 요소

항목	설명
프로세스 책임자 (Owner)	프로세스의 성과와 운영을 책임지는 구성원으로, 프로세스를 설계하고 지속적으로 유지하는 사람이다.
프로세스 맵(Map)	상위 프로세스와 하위 프로세스의 체계를 도식화하여 전체 업무의 청사진을 표현한다.
프로세스 Task 정의서	기대하는 결과물을 산출하기 위해 Task들이 어떻게 운영되어야 하는지에 대한 문서이다.
프로세스 성과 지표	프로세스의 과정과 결과를 고객 입장에서 정량적으로 표현한 성과 측정 지표이다.
프로세스 조직	프로세스를 성공적으로 수행하기 위해 개인들의 업무를 유기적으로 수행하는 구성원이다.
경영자의 리더십 (Leadership)	경영자는 프로세스의 중요성을 인식하고 기업의 경영 방침을 확고하게 해야 한다.

2 프로세스의 맵(Map)

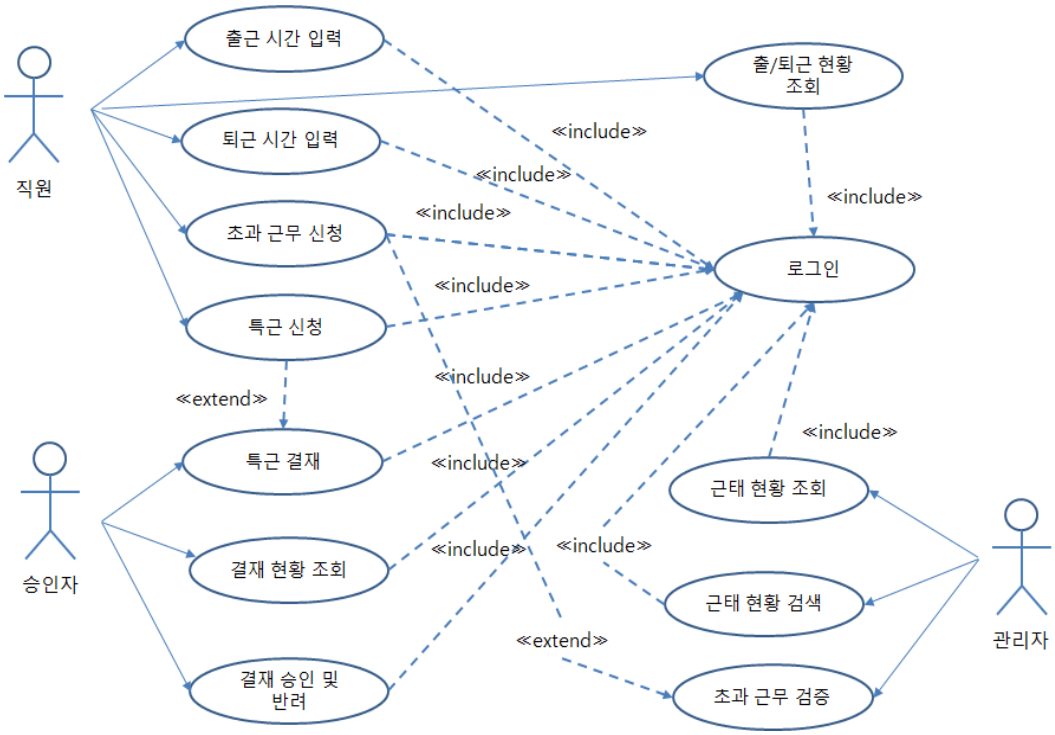
프로세스의 Map은 상위 프로세스에서 하위 프로세스로의 흐름 체계를 도식화한 청사진으로 다양한 형태로 표현할 수 있다.

1. 구조적 분석 기법에서의 자료 흐름도



[그림 3-2] 레벨(Level) 0의 자료 흐름도 예시

2. 객체 지향 분석 기법에서의 사용 사례 다이어그램



[그림 3-3] 사용 사례 다이어그램 예시

수행 내용 / 업무 프로세스 확인하기

재료·자료

- 프로그램 관리 대장, 프로그램 설계서, 화면 설계서
- 요구 사항 정의서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 설계 산출물에 대한 이해가 선행되어야 한다.
- 완료된 산출물은 형상관리 서버에 반영하여야 한다.

수행 순서

1 상세 설계를 기반으로 담당 업무를 확인한다.

업무 프로세스를 확인하기 위해서는 애플리케이션 설계 단계에서 작성한 프로그램 관리 대장을 통해 작성해야 할 업무 프로그램들을 확인하고 프로그램 설계서와 업무 프로세스 맵을 확인한다.

1. 프로그램 관리 대장을 확인한다.

   프로그램 관리 대장은 설계 단계에서 작성된 구현해야 할 전체 시스템의 프로그램 목록 산출물이다.

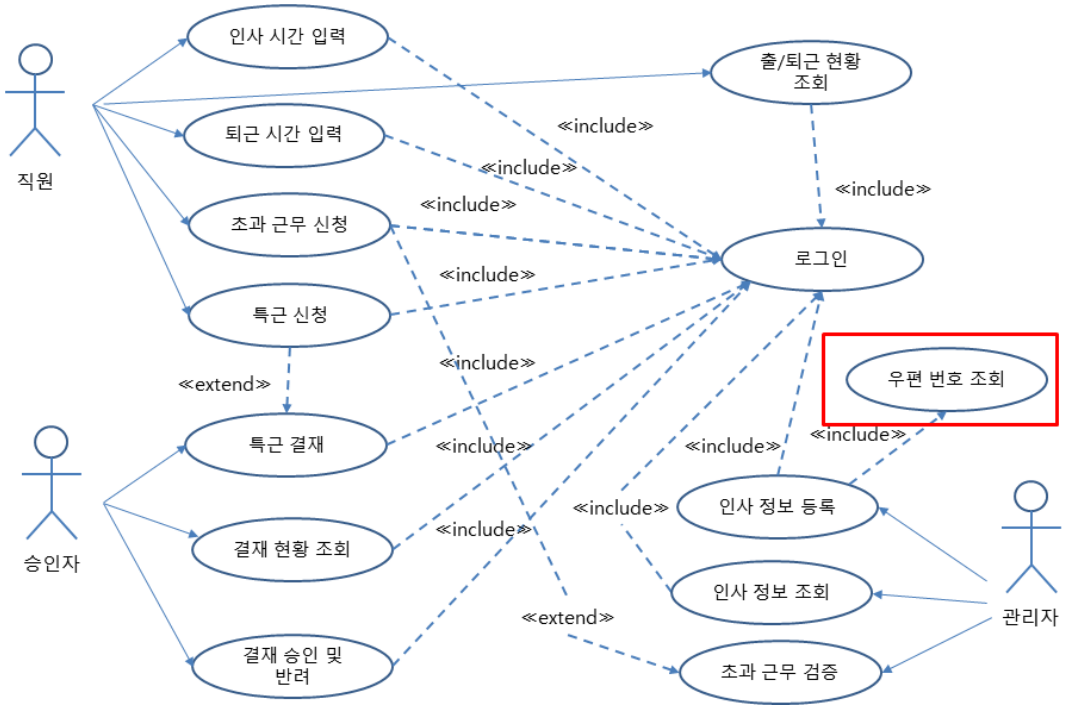
   아래 예시는 이순신 담당자가 구현해야 할 우편번호 조회 기능을 확인한 예시이다.

프로그램 관리 대장												
순번	ID	구분	시스템	기능	설명	엔티티	입력	담당자	출력 (정상)	출력 (에러)	입력항목 설명	출력항목 설명
1	CMM_CODE_LIST_READ	공통	인사	공통 코드 조회	조회 코드에 해당하는 코드 정보 제공	코드 정보	코드	김길동	[200] 코드, 코드 명	[204] 코드 없음 [500] 서버 에러	코드	1. 코드 정보를 배열로 제공 2. Code 204는 코드 없음
2	CMM_MENU_LIST_READ	공통	인사	메뉴권한	직원별 메뉴 접근제어	메뉴 정보	사번	홍길동	[200] 메뉴 코드, 메뉴 명	[401] 권한 없음 [500] 서버 에러	사번 필수 (15자리)	1. 사번으로 조회된 메뉴 정보 2. Code 401은 권한 없음
• • •												
5	EMP_INFO_DETAIL_READ	화면	인사	직원 상세 정보 조회	직원의 상세 정보 제공	직원 정보	사번	강감찬	[200] 사번, 직원 명, 부서, 직책, 이메일, 전화번호 입사일자	[204] 정보 없음 [500] 서버 에러	사번 필수 (15자리)	1. 사번으로 조회된 직원 정보 2. Code 204은 직원 아님
6	EMP_POST_READ	화면	인사	우편번호 조회	주소 조회	주소 정보	시군구, 도로명	이순신	[200] 우편번호, 도로명 주소	[204] 주소 없음 [500] 서버 에러	시군구, 도로명 필수	1. 도로명으로 조회된 주소 정보 2. Code 204는 주소 없음

[그림 3-4] 프로그램 관리 대장에서의 담당자 확인 예시

2. 업무 프로세스 맵을 확인한다.

프로그램 관리 대장에서 확인한 프로세스에 대한 업무 프로세스 체계를 확인한다.



[그림 3-5] 업무 프로세스 맵 확인 예시

2 세부 업무 프로세스를 확인한다.

1. 프로그램 설계서를 확인한다.

프로그램 관리 대장 산출물에서의 ID와 일치하는 프로그램 설계서 산출물에서의 프로그램 ID를 확인한다.

프로그램 설계서				
PostController				
프로그램 ID	EMP_POST_READ	PACKAGE	com.ncs.controller	
EXTENDS	N/A			
IMPLEMENTS	N/A			
IMPORT	com.ncs.service.selectPostByDong com.ncs.service.selectPostByDoro			
프로그램 설명	도로명 및 동 명칭 입력 시 관련 검색 결과를 반환하는 Class			
[속성]				
Name	visibility	Type	Default Value	설명
success	private	String		코드 200 출력
none	private	String		코드 204 출력
error	private	String		코드 500 출력
[Methods]				
Name	selectPostByDong			
Visibility	public			
Parameter	PostVo			
Return	post.jsp			
설명	동명칭으로 주소를 조회하기 위해 서비스 메소드를 호출한다. [com.ncs.service.selectPostByDong]			
Name	selectPostByDoro			
Visibility	public			
Parameter	PostVo			
Return	post.jsp			
설명	도로명으로 주소를 조회하기 위해 서비스 메소드를 호출한다. [com.ncs.service.selectPostByDoro]			

[그림 3-6] 프로그램 관리 대장에서의 ID와 일치하는 프로그램 설계서 예시

2. 화면 설계서를 확인한다.

프로그램 관리 대장 산출물에서의 ID와 일치하는 화면 설계서 산출물에서의 프로그램 ID를 확인한다.

화면 설계서는 화면에서 동작하는 이벤트 및 변수들을 설계한 산출물이다.

화면 설계서

주소 검색화면

프로그램 ID	EMP_POST_READ	화면 명	post.jsp
화면 설명	지번 및 도로명 주소를 입력 할 수 있는 공통 화면이다.		

항목						
항목명	영문명	설명	속성	Validation		
				필수여부	자리수	기타
동명칭	txtDong	지번주소 검색 시 동명칭을 입력하는 변수	E	필수	20	
시군구	txtSigungu	도로명주소 검색 시 시군구를 입력하는 변수	E	선택	20	
도로명	txtDoro	도로명주소 검색 시 도로명을 입력하는 변수	E	필수	20	
선택	selAddrType	입력한 주소를 선택	S			
주소	resultAddress	검색된 주소 목록	R			

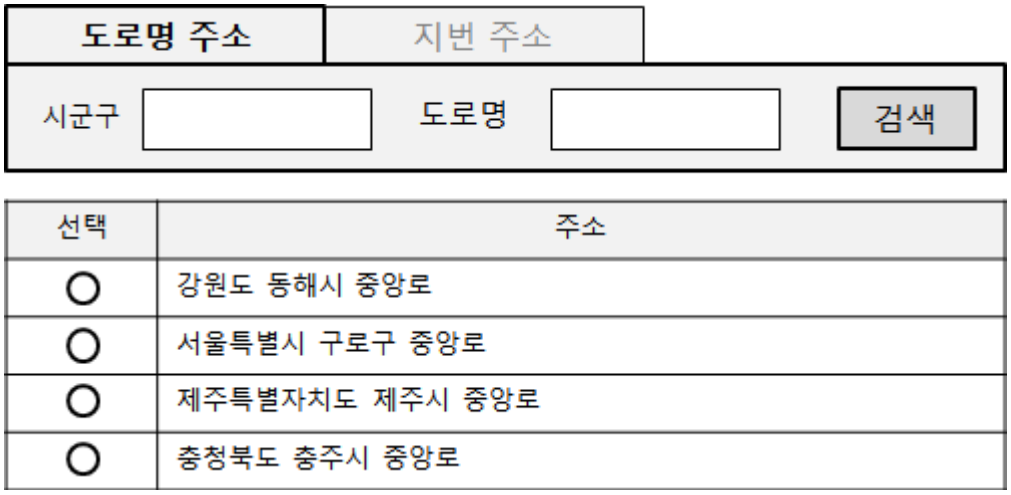
화면 이벤트				
이벤트명	이벤트 설명	관련 데이터		관련 Object
화면로딩 (onLoad)	주소 검색 화면으로 표시한다.	S	N/A	
		R	N/A	
지번주소 검색버튼 (onClick)	입력한 동명칭 정보를 조회한다.	S	txtDong	PostController
		R	resultAddress	
도로명주소 검색버튼 (onClick)	입력한 도로명 주소를 조회한다.	S	txtDoro	PostController
		R	resultAddress	

관련 파일		
파일유형	경로명	파일명
css	/common/css	common.css
jquery	/common/js	common.js

[그림 3-7] 공통 모듈 관리 대장에서의 ID와 일치하는 화면 설계서 예시

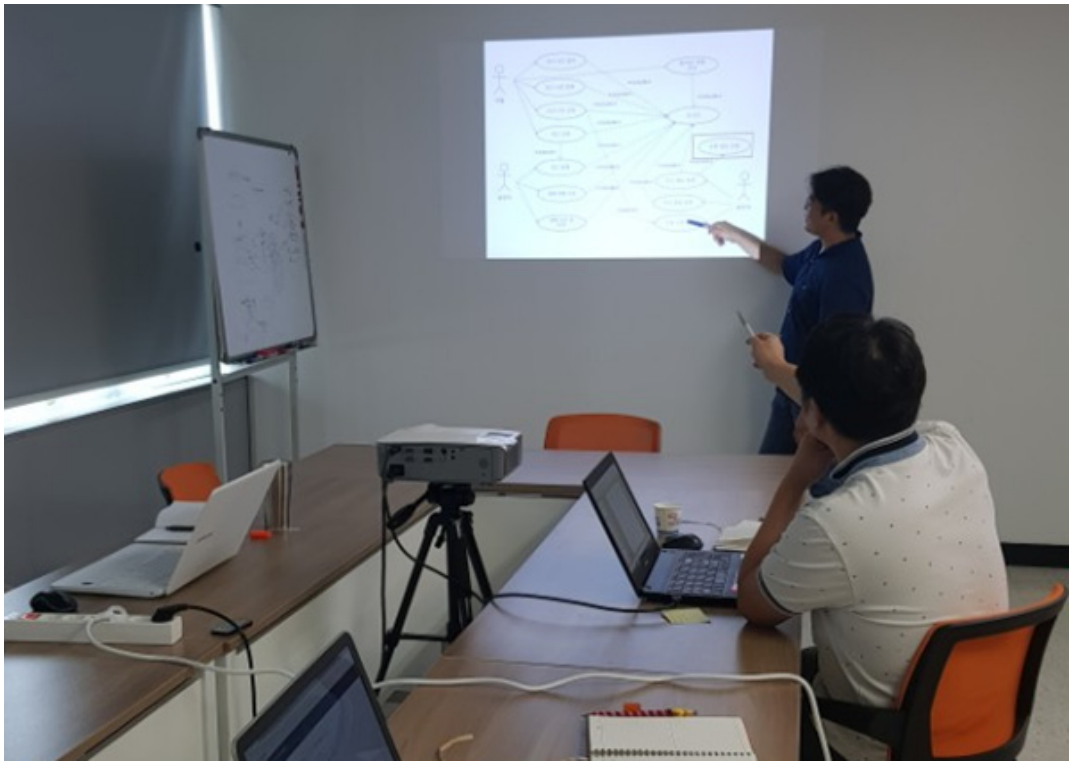
3. 화면 레이아웃을 확인한다.

애플리케이션 설계 단계에서 작성된 화면 레이아웃을 확인하여 공통 모듈의 화면을 구현한다.



[그림 3-8] 도로명 주소 화면 레이아웃 예시

[그림 3-9] 지번 주소 화면 레이아웃 예시



[그림 3-10] 업무 프로세스 맵을 확인하는 모습

수행 tip

- 목표 시스템의 업무 프로세스 맵을 확인하여 구현해야 할 프로그램의 프로세스를 이해하는 능력이 중요하다.

3-2. 서버 프로그램 구현

학습 목표	• 세부 업무 프로세스를 기반으로 프로그래밍 언어와 도구를 활용하여 서비스의 구현에 필요한 업무 프로그램을 구현할 수 있다. • 개발하고자 하는 목표 시스템의 잠재적 보안 취약성이 제거될 수 있도록 서버 프로그램을 구현할 수 있다.

필요 지식 /

1 프레임워크(Framework)에 대한 이해

1. 소프트웨어 프레임워크의 정의

(1) 효율적인 정보 시스템 개발을 위한 코드 라이브러리, 애플리케이션 인터페이스(Application Interface), 설정 정보 등의 집합으로서 재사용이 가능하도록 소프트웨어 구성에 필요한 기본 뼈대를 제공한다.

(2) 광의적으로 정보 시스템의 개발 및 운영을 지원하는 도구 및 가이드 등을 포함한다.

2. 프레임워크의 특징

〈표 3-3〉 프레임워크 특징

항목	설명
모듈화 (modularity)	프레임워크는 인터페이스에 의한 캡슐화를 통해서 모듈화를 강화하고 설계와 구현의 변경에 따르는 영향을 극소화하여 소프트웨어의 품질을 향상시킨다.
재사용성 (reusability)	프레임워크가 제공하는 인터페이스는 반복적으로 사용할 수 있는 컴포넌트를 정의할 수 있게 하여 재사용성을 높여 준다. 프레임워크 컴포넌트를 재사용하는 것은 소프트웨어의 품질을 향상시킬 뿐만 아니라 개발자의 생산성도 높여 준다.
확장성 (extensibility)	프레임워크는 다형성(polymorphism)을 통해 애플리케이션이 프레임워크의 인터페이스를 확장할 수 있게 한다. 프레임워크 확장성은 애플리케이션 서비스와 특성을 변경하고 프레임워크를 애플리케이션의 가변성으로부터 분리함으로써 재사용성의 이점을 얻게 한다.
제어의 역흐름 (inversion of control)	프레임워크 코드가 전체 애플리케이션의 처리 흐름을 제어하여 특정한 이벤트가 발생할 때 다형성(Polymorphism)을 통해 애플리케이션이 확장한 메소드를 호출함으로써 제어가 프레임워크로부터 애플리케이션으로 거꾸로 흐르게 한다.

② 데이터 저장 계층 또는 영속 계층(Persistence Layer)에 대한 이해

DAO/DTO/VO란 영속 계층(Persistence Layer)에서 사용되는 특정 패턴을 통해 구현되는 Java Bean이다.

〈표 3-4〉 영속 계층(Persistence Layer)의 객체 종류

객체	설명
DAO (Data Access Object)	특정 타입의 데이터베이스나 다른 지속적인 메커니즘(Persistence Mechanism)에 추상 인터페이스를 제공하는 객체이다. 애플리케이션 호출을 데이터 저장 부분(Persistence Layer)에 매핑함으로써 DAO는 데이터베이스의 세부 내용을 노출하지 않고 특정 데이터 조작 기능을 제공한다.
DTO (Data Transfer Object)	DTO는 프로세스 사이에서 데이터를 전송하는 객체를 의미한다. 많은 프로세스 간의 커뮤니케이션이 원격 인터페이스(예: 웹 서비스)에 의해 이루어지기 때문에 전송될 데이터를 모으는 DTO를 이용해서 한 번만 호출하게 하는 것이다. DTO는 스스로의 데이터를 저장 및 회수하는 기능을 제외하고 아무 기능도 가지고 있지 않다는 것이 DAO와의 차이이다.
VO (Value Object)	VO는 간단한 독립체(Entity)를 의미하는 작은 객체를 의미한다. 가변 클래스인 DTO와 다르게 getter기능만 제공하는 불변 클래스를 만들어서 사용한다.

③ 소프트웨어(SW) 개발 보안

1. 소프트웨어 개발 보안의 개념

SW 개발 과정에서 개발자의 실수, 논리적 오류 등으로 인해 SW에 내포될 수 있는 보안취약점(vulnerability)의 원인, 즉 보안 취약점(weakness)을 최소화하고, 사이버 보안 위협에 대응할 수 있는 안전한 SW를 개발하기 위한 일련의 보안 활동이다.

2. 소프트웨어 개발 보안 가이드의 구성

〈표 3-5〉 Java 시큐어 코딩 가이드

유형	설명
입력 데이터 검증 및 표현	프로그램 입력값에 대한 검증 누락 또는 부적절한 검증, 데이터의 잘못된 형식 지정으로 인해 발생할 수 있는 보안 약점 ※ SQL 삽입, 자원 삽입, 크로스 사이트 스크립트 등 26개
보안 기능	보안 기능(인증, 접근 제어, 기밀성, 암호화, 권한 관리 등)을 적절하지 않게 구현 시 발생할 수 있는 보안 약점 ※ 부적절한 인가, 중요 정보 평문 저장(또는 전송) 등 24개
시간 및 상태	동시 또는 거의 동시 수행을 지원하는 병렬 시스템, 하나 이상의 프로세스가 동작하는 환경에서 시간 및 상태를 부적절하게 관리하여 발생할 수 있는 보안 약점 ※ 경쟁 조건, 제어문을 사용하지 않는 재귀 함수 등 7개

유형	설명
에러 처리	에러를 처리하지 않거나, 불충분하게 처리하여 에러 정보에 중요 정보(시스템 등)가 포함될 때 발생할 수 있는 보안 약점 ※ 취약한 패스워드 요구 조건, 오류 메시지를 통한 정보 노출 등 4대
코드 오류	타입 변환 오류, 자원(메모리 등)의 부적절한 반환 등과 같이 개발자가 범할 수 있는 코딩 오류로 인해 유발되는 보안 약점 ※ 널 포인터 역참조, 부적절한 자원 해제 등 7개
캡슐화	중요한 데이터 또는 기능성을 불충분하게 캡슐화하였을 때 인가되지 않는 사용자에게 데이터 누출이 가능해지는 보안 약점 ※ 제거되지 않고 남은 디버그 코드, 시스템 데이터 정보 노출 등 8개
API 오용	의도된 사용에 반하는 방법으로 API를 사용하거나, 보안에 취약한 API를 사용하여 발생할 수 있는 보안 약점 ※ DNS Lookup에 의존한 보안 결정, 널 매개 변수 미조사 등 7개

출처: 행정자치부(2012. 09.), 전자정부 SW 개발·운영자를 위한 Java 시큐어 코딩 가이드

수행 내용 / 서버 프로그램 구현하기

재료·자료

- 프로그램 설계서, 화면 설계서, 화면 레이아웃
- 요구 사항 정의서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

안전 · 유의 사항

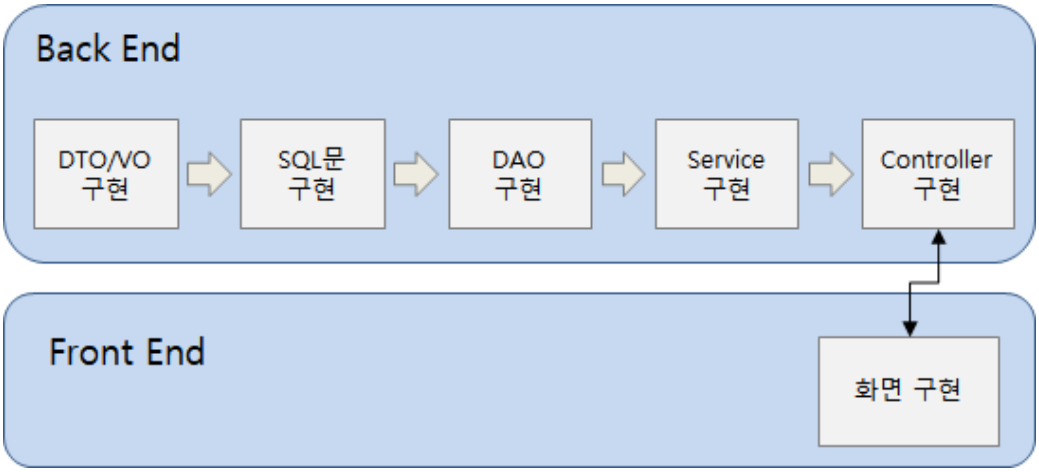
- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 공통 모듈 설계 산출물에 대한 이해가 선행되어야 한다.

- 프로그램 개발 경험과 SQL 작성 경험이 필요하다.

수행 순서

1 세부 업무 프로세스를 기반으로 업무 프로그램을 구현한다.

프로그램을 구현하기 위해 서버 영역(Back End)과 화면 영역(Front End)을 구현한다.



[그림 3-11] 모듈 구현 순서 예시

순서에 상관없이 구현을 해도 무방하다.

1. 업무 프로그램을 구현하기 위한 I/O 오브젝트(DTO/VO)를 정의한다.

   주소 검색을 위해 필요한 변수들을 정의하여 계층 간 데이터 교환을 위해 사용한다. DTO(Data Transfer object) 또는 VO(Value Object)라고 명칭한다.

```java
package com.ncs.vo;

public class PostVo {

    private String doro_code;

    private String zip_code;

    private String doro_kor;

    private String doro_rome;

    private String dong_serial;

    private String sido_kor;

    private String sido_rome;

    private String sigungu_kor;

    private String sigungu_rome;

    private String dong_kor;

    private String dong_rome;

    private String dong_gubun;

    private String dong_code;

    private String use_yn;

    private String modify_reason;

    private String modify_history;

    private String notify_date;

    private String expire_date;

    public String getDoro_code() {
        return doro_code;
    }

    ...
}
```

[그림 3-12] DTO/VO 구문 예시

2. 업무 프로그램을 구현하기 위한 Data를 준비한다.

   주소 검색을 위해 주소 테이블(Table)을 정의하고 테이블을 생성한다.

이름	데이터 유형	길이/설정	부호 ...	NULL 허용
DORO_CODE	**VARCHAR**	**12**	☐	☐
ZIP_CODE	VARCHAR	5	☐	☐
DORO_KOR	VARCHAR	80	☐	☑
DORO_ROME	VARCHAR	80	☐	☑
DONG_SERIAL	**VARCHAR**	**2**	☐	☐
SIDO_KOR	VARCHAR	20	☐	☑
SIDO_ROME	VARCHAR	40	☐	☑
SIGUNGU_KOR	VARCHAR	20	☐	☑
SIGUNGU_ROME	VARCHAR	40	☐	☑
DONG_KOR	VARCHAR	20	☐	☑
DONG_ROME	VARCHAR	40	☐	☑
DONG_GUBUN	VARCHAR	1	☐	☑
DONG_CODE	VARCHAR	3	☐	☑
USE_YN	VARCHAR	1	☐	☑
MODIFY_REASON	VARCHAR	1	☐	☑
MODIFY_HISTORY	VARCHAR	14	☐	☑
NOTIFY_DATE	VARCHAR	8	☐	☑
EXPIRE_DATE	VARCHAR	8	☐	☑

[그림 3-13] 주소 테이블(Table) 예시

3. 업무 프로그램을 구현하기 위한 SQL을 작성한다.

주소 검색을 위한 SQL 쿼리문을 마이바티스(Mybatis) XML 파일로 구현한 예제이다.

```
<mapper namespace="com.ncs.sql">

     <select id="selectPostByDong" parameterType="com.ncs.vo.PostVo" resultType="com.ncs.vo.PostVo">
        SELECT DISTINCT
            ZIP_CODE
            , SIDO_KOR
            , SIGUNGU_KOR
            , DONG_KOR
        FROM
            TBL_ADDRESS
        WHERE DORO_KOR LIKE CONCAT(CONCAT('%', #{dong_kor}), '%');
    </select>
```

[그림 3-14] 동으로 주소를 조회하는 SQL 구문 예시

```
    <select id="selectPostByDoro" parameterType="com.ncs.vo.PostVo" resultType="com.ncs.vo.PostVo">
        SELECT DISTINCT
            ZIP_CODE
            , SIDO_KOR
            , SIGUNGU_KOR
            , DORO_KOR
        FROM
            TBL_ADDRESS
        WHERE DORO_KOR LIKE CONCAT(CONCAT('%', #{doro_kor}), '%');
    </select>

</mapper>
```

[그림 3-15] 도로명으로 주소를 조회하는 SQL 구문 예시

4. 데이터 접근 객체(DAO: Data Access Object)를 구현한다.

DAO에서는 SQL문을 구현한 XML의 id를 호출하여 데이터를 조회하거나 조작한다.

```java
package com.ncs.dao;

import java.util.List;

import org.apache.ibatis.session.SqlSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.ncs.vo.PostVo;

@Repository("postDao")
public class PostDao {

    @Autowired
    private SqlSession sqlSession;

    //DB에서 주소 검색(동)
    public List<PostVo> selectPostByDong(PostVo postVo) throws Exception {
        return sqlSession.selectList("com.ncs.sql.selectPostByDong", postVo);
    }

    //DB에서 주소 검색(도로)
    public List<PostVo> selectPostByDoro(PostVo postVo) throws Exception {
        return sqlSession.selectList("com.ncs.sql.selectPostByDoro", postVo);
    }

}
```

[그림 3-16] 주소를 조회하는 DAO 구문 예시

5. 서비스(Service) 클래스를 구현한다.

```java
package com.ncs.service;

import java.util.List;
import javax.annotation.Resource;
import org.springframework.stereotype.Service;

import com.ncs.dao.PostDao;
import com.ncs.vo.PostVo;

@Service("postService")
public class PostService {

    @Resource(name = "postDao")
    private PostDao postDao;

    //주소 검색(동)
    public List<PostVo> selectPostByDong(PostVo postVo) throws Exception {
        return postDao.selectPostByDong(postVo);
    }

    //주소 검색(도로)
    public List<PostVo> selectPostByDoro(PostVo postVo) throws Exception {
        return postDao.selectPostByDoro(postVo);
    }
}
```

[그림 3-17] 주소를 조회하는 Service 구문 예시

6. Java 시큐어 코딩 가이드에 의한 보안 취약성을 제거하는 코드를 구현한다.

서버 프로그램은 다양한 보안 취약점에 존재하기 때문에 소프트웨어 개발 보안 가이

드에 의해 알려진 보안 취약점을 대응해야 한다.

아래 예시는 크로스 사이트 스크립트에 대응하는 안전한 코드 예시이다.

```
/**
 * 크로스 사이트 스크립트 보안
 * @param param
 * @return
 */
private String relaceParameter(String param) {

    String result = param;

    if(param != null) {
        result = result.replaceAll("<","&lt;");
        result = result.replaceAll(">","&gt;");
        result = result.replaceAll("&","&amp;");
        result = result.replaceAll("\"","&quot;");
    }

    return result;
}
```

[그림 3-18] 안전한 코드(크로스 사이트 스크립트 보안 약점) 예시

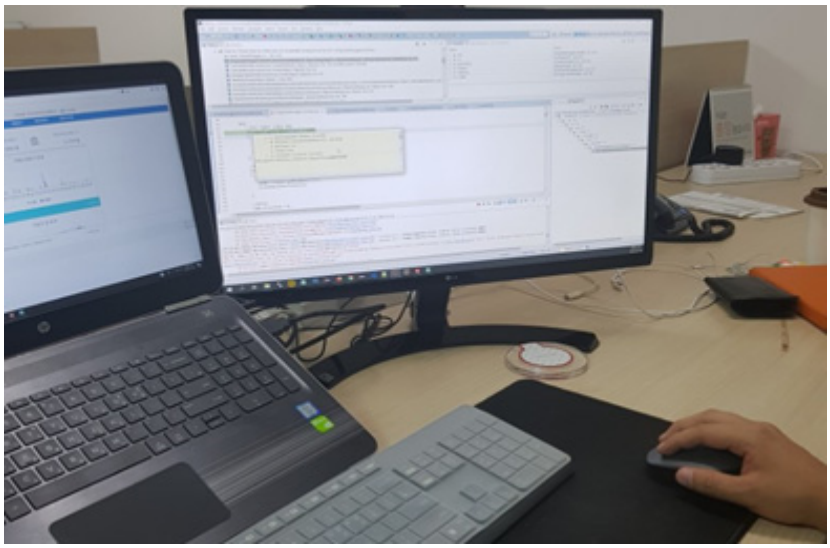
7. 컨트롤러(Controller) 클래스를 구현한다.

```
/**
 * 동으로 주소 검색
 * @param postVo
 * @param request
 * @param response
 * @param model
 * @return
 */
@RequestMapping(value = "/zip/dong", method = RequestMethod.POST)
public String selectPostByDong(PostVo postVo
        , HttpServletRequest request, HttpServletResponse response
        , Model model) {

    try {

        //문자열 필터(크로스 사이트 스크립트 보안)
        String dong_kor = this.relaceParameter(postVo.getDong_kor());
        postVo.setDong_kor(dong_kor);

        //주소 검색 Service 호출
        model.addAttribute("resultAddress", postService.selectPostByDong(postVo));

    } catch(Exception e) {
        logger.error(e.getMessage());
        e.printStackTrace();
    }

    //주소검색 화면인 post.jsp 호출
    return "post";
}
```

[그림 3-19] 동으로 주소를 조회하는 Controller 구문 예시

```java
/**
 * 도로명으로 주소 검색
 * @param postVo
 * @param request
 * @param response
 * @param model
 * @return
 */
@RequestMapping(value = "/zip/doro", method = RequestMethod.POST)
public String selectPostByDoro(PostVo postVo
        , HttpServletRequest request, HttpServletResponse response
        , Model model) {

    try {

        //문자열 필터(크로스 사이트 스크립트 보안)
        String doro_kor = this.relaceParameter(postVo.getDoro_kor());
        postVo.setDoro_kor(doro_kor);

        //주소 검색 Service 호출
        model.addAttribute("resultAddress", postService.selectPostByDoro(postVo));

    } catch(Exception e) {
        logger.error(e.getMessage());
        e.printStackTrace();
    }

    //주소검색 화면인 post.jsp 호출
    return "post";
}
```

[그림 3-20] 도로명으로 주소를 조회하는 Controller 구문 예시



[그림 3-21] 서버 프로그램을 구현하는 모습

수행 tip

- 클라이언트 프로그램에 독립적으로 활용 가능하며, 개발환경으로 정의된 개발언어와 도구의 특성을 충분히 반영하여 생산성과 품질이 확보된 서버프로그램을 개발하는 것이 중요하다.

3-3. 서버 프로그램 테스트

학습 목표	• 개발된 업무 프로그램의 내부 기능과 제공하는 인터페이스에 대해 테스트를 수행할 수 있다.

필요 지식 /

1 소프트웨어 테스트의 이해

1. 소프트웨어 테스트의 개념

소프트웨어 테스트란 구현된 애플리케이션이나 시스템을 사용자의 요구 사항이 만족되었는지 확인하기 위하여 기능 및 비기능 요소의 결함을 찾아내는 활동이다.

2. 소프트웨어 테스트의 원칙

(1) 개발자가 자신이 개발한 프로그램 및 소스코드를 테스팅하지 않는다.

일반적으로 개발자가 자신이 개발한 소스코드에 대해서는 자신이 테스팅을 할 경우 결함을 발견하는 것이 쉬운 일이 아니다.

(2) 효율적인 결함 제거 법칙 사용(낚시의 법칙, 파레토의 법칙)

효율적으로 결함을 발견, 가시화, 제거, 예방의 순서로 하여 정량적으로 관리할 수 있어야 한다.

(가) 낚시의 법칙

낚시를 즐겨하는 사람들은 특정 자리에서 물고기가 잘 잡힌다는 사실을 경험적으로 알고 있다. 소프트웨어 제품의 결함도 특정 기능, 모듈, 라이브러리에서 결함이 많이 발견된다는 것이 소프트웨어 테스트에서의 낚시의 법칙이다.

(나) 파레토의 법칙

소프트웨어 제품에서 발견되는 전체 결함의 80%는 소프트웨어 제품의 전체 기능 중 20%에 집중되어 있다.

(3) 완벽한 소프트웨어 테스팅은 불가능하다.

단순한 애플리케이션이라도 테스트 케이스의 수는 무한대로 발생되기 때문에 완벽한 테스트는 불가능하다.

(4) 테스트는 계획 단계부터 해야 한다.

소프트웨어 테스트는 결함의 발견이 목적이긴 하지만 개발 초기 이전인 계획 단계에서부터 할 수 있다면 결함을 예방할 수 있다.

(5) 살충제 패러독스(Pesticide Paradox)

동일한 테스트 케이스로 반복 실행하면 더 이상 새로운 결함을 발견할 수 없으므로 주기적으로 테스트 케이스를 점검하고 개선해야 한다.

(6) 오류-부재의 궤변(Absence of Errors Fallacy)

사용자의 요구 사항을 만족하지 못한다면 오류를 발견하고 제거해도 품질이 높다고 말할 수 없다.

2 소프트웨어 테스트의 명세

1. 테스트 결과 정리

테스트가 완료되면 테스트 계획과 테스트 케이스 설계부터 단계별 테스트 시나리오, 테스트 결과까지 모두 포함된 문서를 일관성 있게 작성한다.

2. 테스트 요약 문서

테스트 계획, 소요 비용, 테스트 결과에 의해 판단 가능한 대상 소프트웨어의 품질 상태를

포함한 요약 문서를 작성한다.

3. 품질 상태

품질 상태는 품질 지표인 테스트 성공률, 발생한 결함의 수와 결함의 중요도, 테스트 커버리지 등이 포함된다.

4. 테스트 결과서

테스트 결과서는 결함에 관련된 내용을 중점적으로 기록하며, 결함의 내용, 결함의 재현 순서를 상세하게 기록한다.

5. 테스트 실행 절차 및 평가

단계별 테스트 종료 시 테스트 실행 절차를 리뷰하고 결과에 대한 평가를 수행하며, 그 결과에 따라 실행 절차를 최적화하여 다음 테스트에 적용한다.

수행 내용 / 서버 프로그램 테스트하기

재료·자료

- 프로그램 설계서, 화면 설계서, 화면 레이아웃
- 요구 사항 정의서
- 개발된 프로그램 산출물

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 프로그램 설계 산출물에 대한 이해가 선행되어야 한다.
- 테스트와 디버그의 이해가 선행되어야 한다.

수행 순서

1 단위 테스트 케이스를 작성한다.

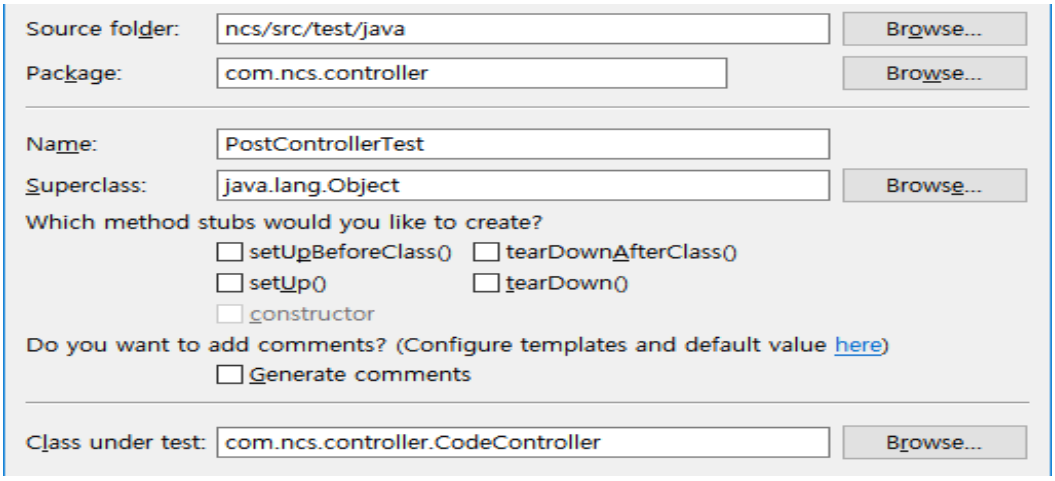
1. 단위 테스트 방식을 결정한다.
2. 단위 테스트의 범위를 결정한다.
   (1) 공통 모듈의 범위 및 상위 요구 사항의 범위에 따라 테스트 범위를 판단한다.
   (2) 테스트 수행 횟수를 결정한다.
   (3) 테스트의 결과를 검증하기 위한 결괏값 산출 방식을 결정한다.
3. 단위 테스트의 방식과 범위에 따라 테스트 케이스를 작성한다.
4. 작성된 단위 테스트 케이스를 내부 검토한다.
5. 작성된 단위 테스트 케이스를 고객에게 승인을 획득한다.

테스트 케이스			프로젝트명		인사 관리 웹 애플리케이션 시스템 구축		
			대상 시스템		인사 관리 시스템		
단계	단위 테스트	작성자	이순신	승인자	김승인	문서 상태	최초 작성
작성일	2018-01-01	버전	V 1.0	테스트 범위	우편 번호 조회	테스트 조직	개발팀
테스트 ID	TC-CMM-006			테스트 일자	2018-07-27		
테스트 목적	우편번호 조회 테스트						
테스트 기능	동 및 도로명으로 우편번호 조회 기능 점검						
입력 값	동 명, 시도, 도로명						
테스트 단계	테스트 케이스			예상 값	중요도	성공/실패	비고
	동 명 없이 동 주소 조회 함수를 호출한다.			204 코드 출력			
	도로명 없이 도로명 주소 조회 함수를 호출한다.			204 코드 출력			
	존재하지 않는 동 명으로 동 주소 조회 함수를 호출한다.			204 코드 출력			
	존재하지 않는 도로명으로 도로명 주소 조회 함수를 호출한다.			204 코드 출력			
	존재하는 동 명으로 동 주소 조회 함수를 호출한다.			200 코드 출력			
	존재하는 도로명 값으로 도로명 주소 조회 함수를 호출한다.			200 코드 출력			
테스트 환경	개발 서버, 형상 관리 서버						
전제 조건	주소 테이블에 주소들을 입력해야 함						
성공/실패 기준	예상 값이 정상적으로 출력 되면 성공						
특수 절차							

[그림 3-22] 우편번호 조회 단위 테스트 케이스 예시

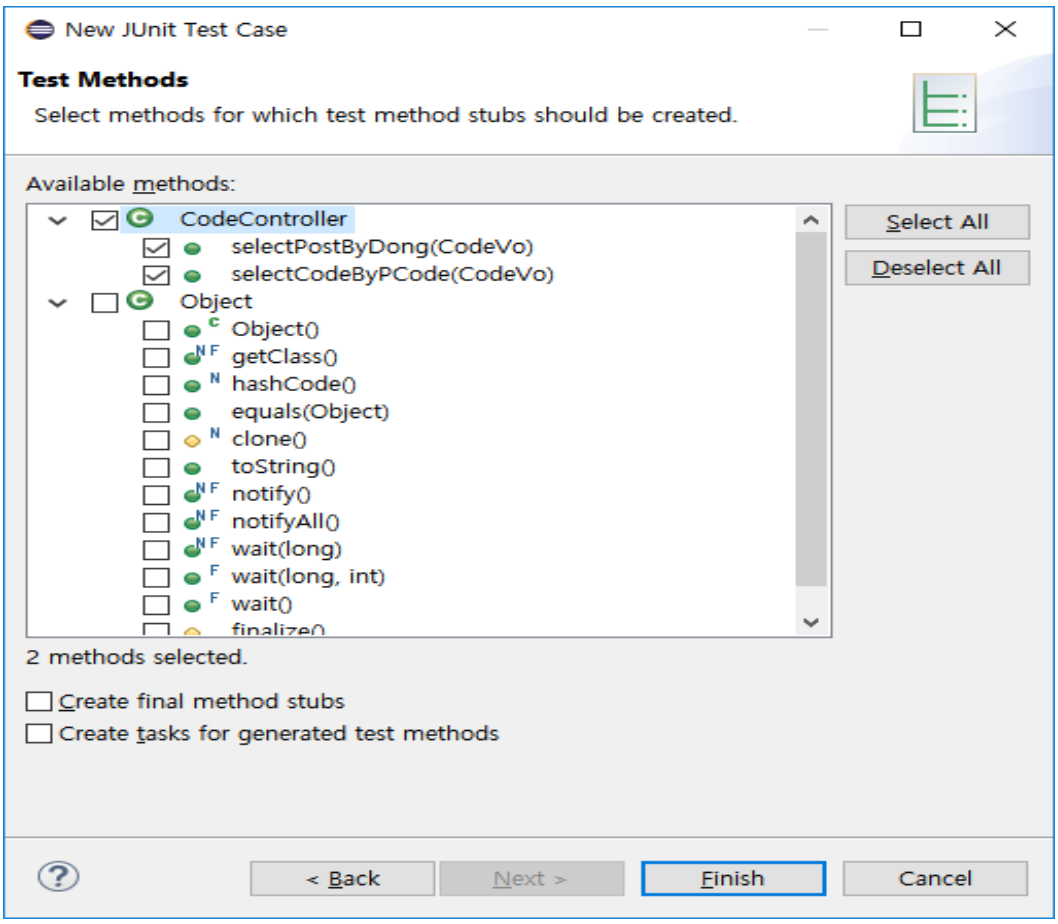
2 테스트를 명세하기 위한 테스트 도구를 설정한다.

1. 단위 테스트 도구를 실행한 후 테스트 케이스(Test Case)를 생성한다.



[그림 3-23] 우편번호 조회 단위 테스트 케이스 생성 예시(JUnit 예시)

2. 테스트를 수행할 함수를 선택하고 Finish를 클릭한다.



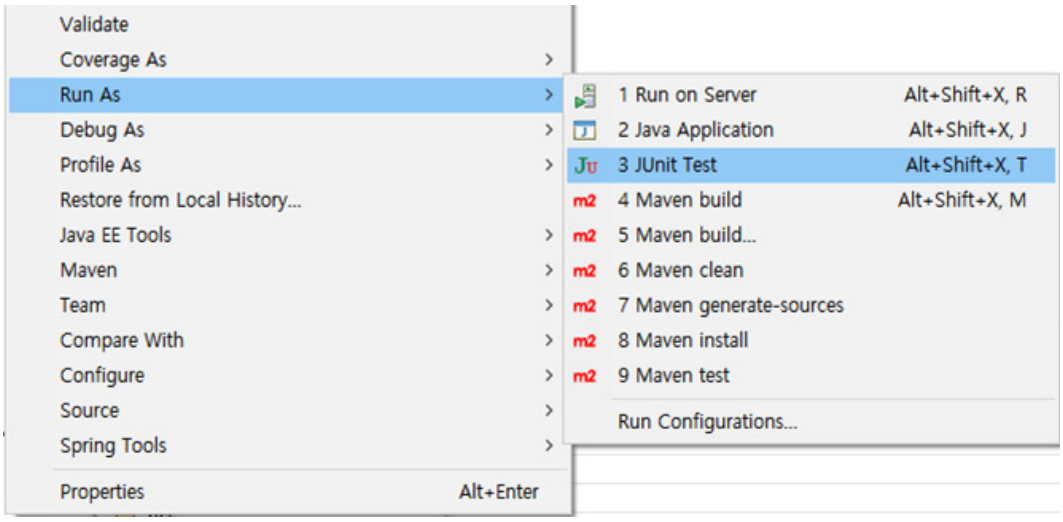
[그림 3-24] 우편번호 조회 단위 테스트 케이스 함수 선택 화면 예시

3. 테스트 구문을 작성한다.

```
@Test
public void testSelectPostByDong() throws Exception {

    mockMvc.perform
    (
        post("/zip/dong")
            .param("dong_kor", "고봉동")
    )
    .andDo(print())
    .andExpect(status().isOk());
}

@Test
public void testSelectPostByDoro() throws Exception {

    mockMvc.perform
    (
        post("/zip/doro")
            .param("doro_kor", "고봉로")
    )
    .andDo(print())
```

[그림 3-25] 우편번호 조회 단위 테스트 구문 예시

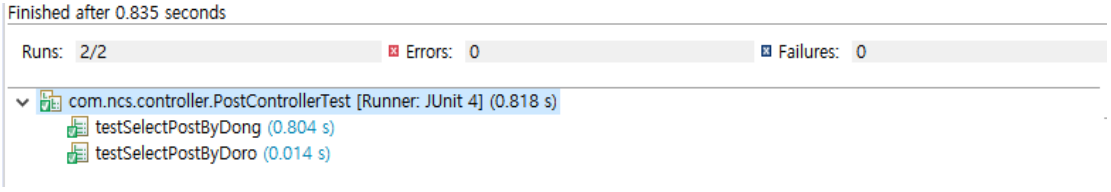
4. 테스트 도구를 실행한다.



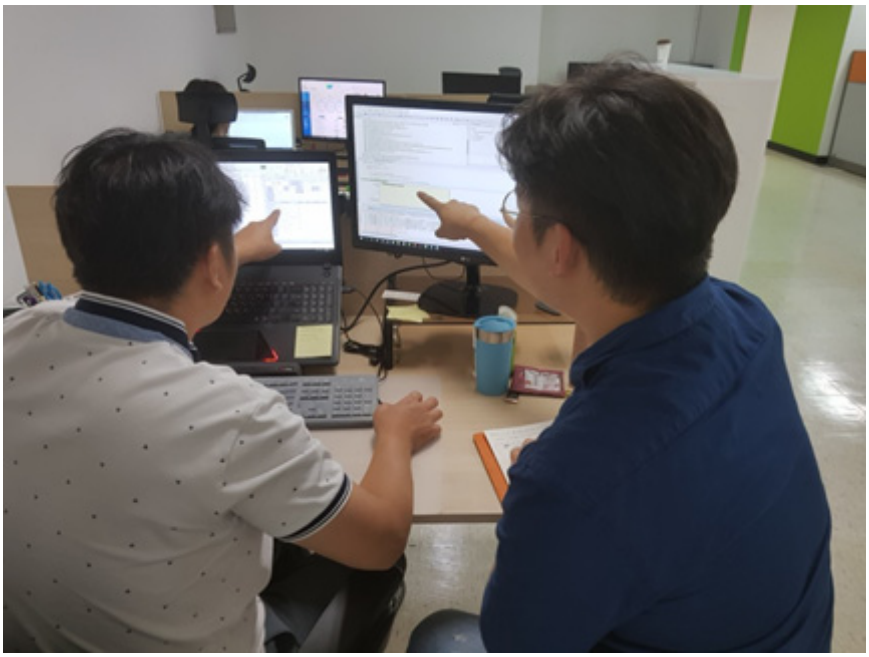
[그림 3-26] 단위 테스트 도구 실행(JUnit Test 예시)

3 테스트 결과를 명세화한다.

테스트의 결과를 확인하여 테스트 케이스에 성공/실패 결과 및 필요 항목을 기록한다.



[그림 3-27] 테스트 결과 화면 예시



[그림 3-28] 단위 테스트를 수행하는 모습

수행 tip

- 개발된 업무 프로그램의 내부 기능과 인터페이스 테스트를 위한 테스트 케이스를 작성하고, 그에 따른 단위 테스트를 수행할 수 있도록 한다.

학습 3 교수·학습 방법

교수 방법

- 애플리케이션 설계 내용을 개발언어와 도구를 활용하여 개발하는 절차에 대해 구체적으로 설명한다.
- 애플리케이션 개발 시 개발 생산성을 높이기 위한 도구 유형과 각 도구의 특성에 대해 제시하여 효율적인 개발을 할 수 있도록 도구 활용 방안에 대해 설명한다.
- 서버 프로그램이 클라이언트 프로그램에 종속되지 않도록 유연성을 확보하는 방안에 대해 정리하여 설명한다.
- 개발된 서버 프로그램에서 제공하는 기능에 대한 테스트 케이스 작성 기법에 대해 정리하여 설명한다.
- 개발 모듈의 단위 테스트 수행을 위한 테스트 조건 명세화 절차와 내용에 대해 정리한 후 상세히 설명한다.

학습 방법

- 애플리케이션 설계 내용을 개발언어와 도구를 활용하여 개발하는 절차에 대해 학습하고 숙지할 수 있도록 학습한다.
- 애플리케이션 개발 시 개발 생산성을 높이기 위한 도구 유형과 각 도구의 특성에 대해 인지하여 효율적인 개발을 할 수 있도록 실습한다.
- 서버 프로그램이 클라이언트 프로그램에 종속되지 않도록 유연성을 확보하는 방안에 대해 학습한다.
- 개발된 서버 프로그램에서 제공하는 기능에 대한 테스트 케이스 작성 기법에 대해 학습하여, 이를 실무에 활용할 수 있을 정도로 반복적으로 실습한다.
- 개발 모듈의 단위 테스트 수행을 위한 테스트 조건 명세화 절차와 내용에 대해 학습한다.

학습 3 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
업무 프로세스 확인	- 업무 프로세스 맵과 세부 업무 프로세스를 확인할 수 있다.			
서버 프로그램 구현	- 세부 업무 프로세스를 기반으로 프로그래밍 언어와 도구를 활용하여 서비스의 구현에 필요한 업무 프로그램을 구현할 수 있다.			
	- 개발하고자 하는 목표 시스템의 잠재적 보안 취약성이 제거될 수 있도록 서버 프로그램을 구현할 수 있다.			
서버 프로그램 테스트	- 개발된 업무 프로그램의 내부 기능과 제공하는 인터페이스에 대해 테스트를 수행할 수 있다.			

평가 방법

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
업무 프로세스 확인	- 업무 프로세스 맵에서의 세부 업무 프로세스의 관계를 확인하는 능력			
서버 프로그램 구현	- 애플리케이션 설계서와 서버 프로그램 간 정합성 정도			
	- 잠재적 보안 취약성이 제거될 수 있도록 SW 개발 보안이 준수된 서버 프로그램을 구현하는 능력			
서버 프로그램 테스트	- 서버 프로그램 테스트 케이스와 테스트 조건의 적정성			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
업무 프로세스 확인	- 업무 프로세스 맵에서의 세부 업무 프로세스의 관계 파악도			
서버 프로그램 구현	- 애플리케이션 설계서와 서버 프로그램 간 정합성 정도			
	- 잠재적 보안 취약성이 제거될 수 있는지에 대한 SW 개발 보안의 파악 능력			
서버 프로그램 테스트	- 서버 프로그램 테스트 케이스와 테스트 조건의 적정성			

피드백

1. 포트폴리오
   - 업무 프로세스 맵을 확인한 후 세부 업무 프로세스의 관계에 대해 이해했는지 평가한 후에 미비 사항을 보완할 수 있도록 실무 자료를 제시한다.
   - 서버 프로그램 테스트를 위한 테스트 케이스 적용 기법에 대해 파악한 후, 프로그램 특성에 맞는 기법 선정 방안을 확인하여 부족한 경우 보완할 수 있도록 설명한다.
   - 보안 취약점이 제거될 수 있도록 서버 프로그램이 구현되었는지를 확인하고 미비한 부분은 SW 개발 보안 가이드를 참고 자료로 제시한다.
   - 서버 프로그램 테스트 케이스 생성 방법을 확인하고 미비 사항에 대해 설명한다.
   - 서버 프로그램 구현 결과가 '상'인 경우 모범 사례로 공유한다.
   - 서버 프로그램 구현 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
   - 서버 프로그램 구현 결과가 '하'인 경우 모범 사례를 참조하여 다시 구축할 수 있도록 유도한다.
2. 평가자 체크리스트
   - 실습 과정에서 평가한 후에 개선해야 할 사항 등을 정리한 후 학습자에게 설명한다.
   - 애플리케이션 설계서에 따라 서버 프로그램이 작성되었는지를 확인하여 미비 사항을 보완할 수 있도록 보완 사항을 제시한다.
   - 보안 취약점이 제거될 수 있도록 서버 프로그램을 구현할 수 있는지에 대한 이해도를 확인한 후 SW 개발 보안 가이드에서의 참고 위치를 제시한다.
   - 서버 프로그램 테스트 케이스 작성 기법을 확인하고, 미비 사항을 정리하여 설명한다.

학습 4 배치 프로그램 구현하기

4-1. 배치 프로그램 구현

학습 목표	• 애플리케이션 설계를 기반으로 프로그래밍 언어와 도구를 활용하여 배치 프로그램 구현 기술에 부합하는 배치 프로그램을 구현할 수 있다. • 목표 시스템을 구성하는 하위 시스템 간의 연동 시 안정적이고 안전하게 동작할 수 있는 배치 프로그램을 구현할 수 있다.

필요 지식 /

1 배치 프로그램의 이해

1. 배치 프로그램의 개념

배치 프로그램이란 사용자와의 상호 작용 없이 일련의 작업들을 작업 단위로 묶어 정기적으로 반복 수행하거나 정해진 규칙에 따라 일괄 처리하는 것이다.

2. 배치 프로그램의 필수 요소

〈표 4-1〉 배치 프로그램의 필수 요소

요소	설명
대용량 데이터	대용량의 데이터를 처리할 수 있어야 한다.
자동화	심각한 오류 상황 외에는 사용자의 개입 없이 동작해야 한다.
견고함	유효하지 않은 데이터의 경우도 처리해서 비정상적인 동작 중단이 발생하지 않아야 한다.
안정성	어떤 문제가 생겼는지, 언제 발생했는지 등을 추적할 수 있어야 한다.
성능	주어진 시간 내에 처리를 완료할 수 있어야 하고, 동시에 동작하고 있는 다른 애플리케이션을 방해하지 말아야 한다.

2 배치 스케줄러(Scheduler)

1. 배치 스케줄러의 개념

배치 스케줄러는 일괄 처리(Batch Processing)를 위해 주기적으로 발생하거나 반복적으로 발생하는 작업을 지원하는 도구이다.

2. 배치 스케줄러의 종류

(1) 스프링 배치(Spring Batch)

스프링 배치는 Spring Source사와 Accenture사의 공동 작업으로 2007년에 탄생한 배치 기반 오픈소스 프레임워크이다.

(가) 스프링 배치(Spring Batch)의 핵심 컴포넌트

〈표 4-2〉 스프링 배치의 컴포넌트

컴포넌트	설명
Job Repository	Job Execution 관련 메타데이터를 저장하는 기반 컴포넌트이다.
Job Launcher	Job Execution을 실행하는 기반 컴포넌트이다.
JPA(Java Persistence API)	페이징 기능을 제공한다.
Job	배치 처리를 의미하는 애플리케이션 컴포넌트이다.
Step	Job의 각 단계를 의미한다. Job은 일련의 연속된 Step으로 구성된다.
Item	Data Source로부터 읽거나 Data Source로 저장하는 각 레코드를 의미한다.
Chunk	특정 크기를 갖는 아이템 목록을 의미한다.
Item Reader	데이터 소스로부터 아이템을 읽어 들이는 컴포넌트이다.
Item Processor	Item Reader로 읽어 들인 아이템을 Item Writer를 사용해 저장하기 전에 처리하는 컴포넌트이다.
Item Writer	Item Chunk를 데이터 소스에 저장하는 컴포넌트이다.

(나) 스프링 배치(Spring Batch)의 핵심 기능

〈표 4-3〉 스프링 배치의 핵심 기능

기능	설명
스프링 프레임워크 기반	스프링의 DI, AOP 및 다양한 엔터프라이즈 지원 기능을 사용한다.
자체 제공 컴포넌트	배치 처리(데이터베이스나 파일로부터 데이터를 읽거나 쓰는 등) 시 공통적으로 필요한 컴포넌트를 제공한다.
견고함과 안정성	선언적 생략과 처리 실패 후 재시도 설정을 제공한다.

(2) Quartz 스케줄러(Scheduler)

Quartz 스케줄러란 Spring Framework에 플러그인(Plug-in)되어 수행하는 Job과 실행 스케줄을 정의하는 Trigger를 분리하여 유연성을 제공하는 오픈소스 스케줄러이다.

(가) Quartz 스케줄러의 구성 요소

〈표 4-4〉 Quartz 스케줄러의 구성 요소

구성 요소	설명
Scheduler	Quartz 실행환경을 관리하는 핵심 개체이다.
Job	사용자가 수행할 작업을 정의하는 인터페이스로서 Trigger 개체를 이용하여 스케줄할 수 있다.
JobDetail	작업명과 작업 그룹과 같은 수행할 Job에 대한 상세 정보를 정의하는 개체이다.
Trigger	정의한 Job 개체의 실행 스케줄을 정의하는 개체로서 Scheduler 개체에게 Job 수행 시점을 알려 주는 개체이다.

(나) Quartz 스케줄러의 사용 예제

```
public class SampleJob extends QuartzJobBean {

        private MyService myService;

        private String name;

        // Inject "MyService" bean
        public void setMyService(MyService myService) { ... }

        // Inject the "name" job data property
        public void setName(String name) { ... }

        @Override
        protected void executeInternal(JobExecutionContext context)
                        throws JobExecutionException {
                ...
        }
}
```

수행 내용 / 배치 프로그램 구현하기

재료·자료

- 배치 설계서
- 시스템 아키텍처

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 배치 설계 산출물에 대한 이해가 선행되어야 한다.
- 프로그램 개발 경험과 SQL 작성 경험이 필요하다.

수행 순서

1 애플리케이션 설계를 기반으로 배치 프로그램을 확인한다.

1. 프로그램 관리 대장을 확인한다.
   아래 예시는 이순신 담당자가 구현해야 할 배치 기능을 확인한 예시이다.

프로그램 관리 대장

순번	ID	구분	시스템	기능	설명	엔티티	입력	담당자	출력 (정상)	출력 (에러)	입력항목 설명	출력항목 설명
1	CMM_CODE_LIST_READ	공통	인사	공통 코드 조회	조회 코드에 해당하는 코드 정보 제공	코드 정보	코드	김길동	[200] 코드, 코드 명	[204] 코드 없음 [500] 서버 에러	코드	1. 코드 정보를 배열로 제공 2. Code 204는 코드 없음
● ● ●												
7	BAT_EMP_001	배치	인사	업무종료 삭제	업무종료 데이터 삭제	인사 정보	업무종료 된 데이터	이순신	[200] 사용자 정보, 시간	[500] 시간	N/A	1. 업무 종료 된 사용자 정보 2. Code 500은 에러 발생

[그림 4-1] 프로그램 관리 대장에서 담당 기능 확인 예시

2. 배치 설계서를 확인한다.

프로그램 관리 대장의 ID와 일치하는 배치 설계를 확인한다.

배치 설계서					
배치기능ID	BAT_EMP_001	배치파일명	RetireProcBatchRunner	배치기능명	업무종료 데이터 삭제
기능 설명	업무종료가 된 사용자 정보를 대상으로 삭제 작업을 진행함				
선행 조건	없음	작업주기	매일	소요시간	30분
입력 값	업무종료 된 데이터		출력 값	작업로그(삭제된 목록 리스트)	
작업 내용					
1. 인사DB에서 업무 종료가 된 사용자 정보 목록정보를 조회한다. 2. 업무 종료가 된 사용자 정보가 존재하지 않은 경우 4번을 진행하고 존재하는 경우 3번 작업을 진행한다. 3. 해당 사용자 정보를 삭제한다. - 해당 사용자의 업무 정보를 삭제한다. - 해당 사용자의 출퇴근 정보를삭제한다. - 해당 사용자의 권한 정보를 삭제한다. - 해당 사용자의 정보를 삭제한다. 4. 삭제 대상 정보에 대한 로그 정보를 생성한다. - 삭제한 사용자 정보 - 삭제 시간 및 종료시간, 총 소요시간 - 파일명 : /log/batch/YYYYMMDD_result.log					

[그림 4-2] 배치 설계서 예시

2 애플리케이션 설계를 기반으로 배치 프로그램을 구현한다.

1. 배치 프로그램을 구현하기 위한 SQL을 작성한다.

업무 종료가 된 사용자 정보를 조회 및 삭제하는 SQL 구문이다.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="com.ncs.sql">

     <select id="selectRetireUsers" resultType="com.ncs.vo.UserVo">

        SELECT SABUN
        FROM TBL_USERS
        WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') <![CDATA[<]]> DATE_FORMAT(CURDATE(), '%Y-%m-%d')

    </select>

    <delete id="deleteRetireUser" parameterType="com.ncs.vo.UserVo">

        DELETE
        FROM TBL_USERS
        WHERE SABUN = #{sabun}

    </delete>

</mapper>
```

[그림 4-3] 배치 프로그램 SQL 구문 예시

2. 배치 프로그램을 구현하기 위한 I/O 오브젝트(DTO/VO)를 정의한다.

```
1  package com.ncs.vo;
2
3  public class UserVo {
4
5      private String sabun;
6
7      private String userid;
8
9      private String userpwd;
10
11     private String name;
12
13     private String email;
14
15     private String mobile;
16
17     private String authcd;
18
19     private String orgcd;
20
21     private String regdate;
22
23     private String retiredate;
24
25     public String getSabun() {
26         return sabun;
27     }
28
29     public void setSabun(String sabun) {
30         this.sabun = sabun;
31     }
32
33     public String getUserid() {
34         return userid;
35     }
36
37     public void setUserid(String userid) {
38         this.userid = userid;
39     }
       ...
   }
```

[그림 4-4] 배치 프로그램 DTO/VO 구문 예시

3. 배치 프로그램을 구현하기 위한 데이터 접근 오브젝트(DAO)를 작성한다.

```java
package com.ncs.dao;

import java.util.List;

import org.apache.ibatis.session.SqlSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.ncs.vo.UserVo;

@Repository("userDao")
public class UserDao {

    @Autowired
    private SqlSession sqlSession;

    //DB에서 업무 종료 된 사용자 목록 조회
    public List<UserVo> selectRetireUsers() throws Exception {
        return sqlSession.selectList("com.ncs.sql.selectRetireUsers");
    }

    //DB에서 업무 종료 된 사용자 삭제
    public int deleteRetireUser(UserVo userVo) throws Exception {
        return sqlSession.delete("com.ncs.sql.deleteRetireUser", userVo);
    }

}
```

[그림 4-5] 배치 프로그램 DAO 구문 예시

4. 배치 프로그램을 구현하기 위한 스케줄러 클래스를 작성한다.

```java
@Scheduled(cron="0 0 1 * * ?")
public void scheduleRun() {

    String batchResult = "성공";
    try {

        // 1. 업무 종료된 사용자 정보를 조회한다.
        List<UserVo> userList = userDao.selectRetireUsers();

        // 2. 업무 종료가 된 사용자 정보가 존재하지 않은 경우 4번을 진행하고 존재하는 경우 3번 작업을 진행한다.
        if(userList != null) {

            for(int i = 0; i < userList.size(); i++) {
                UserVo userVo = userList.get(i);

                // 3. 해당 사용자 정보를 삭제한다.
                userDao.deleteRetireUser(userVo);
            }
        }

    } catch(Exception e) {
        batchResult = "실패";
    }


    // 4. 삭제 대상 정보에 대한 로그 정보를 생성한다.
    Calendar calendar = Calendar.getInstance();
    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");
    logger.info("스케줄 실행 : [" + batchResult + "] " + dateFormat.format(calendar.getTime()));
}
```

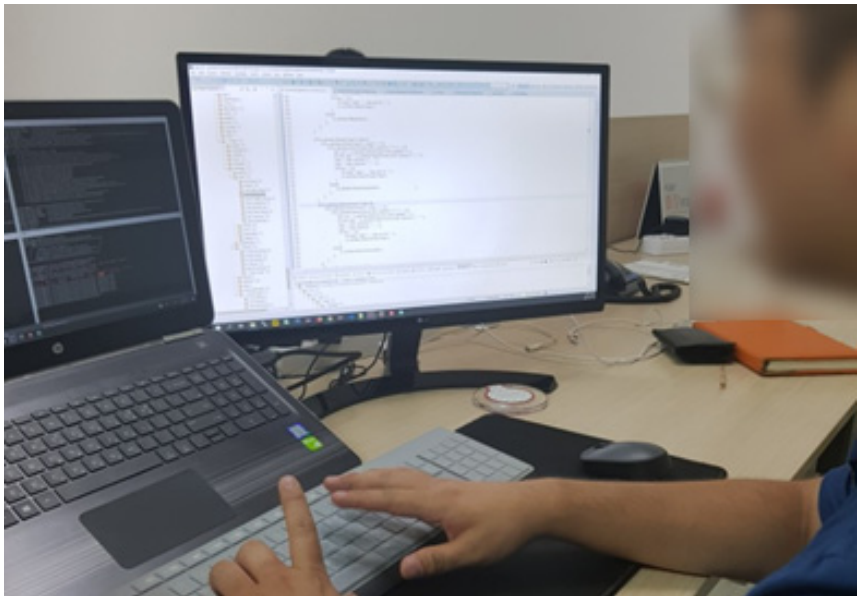
[그림 4-6] 배치 프로그램 스케줄러 클래스 구문 예시

위 예시는 @Scheduled 어노테이션(Annotation) 구문 지정으로 매일 1시에 배치

스케줄이 실행되게 구현한 예시이다.

〈표 4-5〉 Cron 표현식(Expression)

필드명	허용값(범위)	허용된 특수 문자
Seconds	0 ~ 59	, - * /
Minutes	0 ~ 59	, - * /
Hours	0 ~ 23	, - * /
Day-of-month	1 ~ 31	, - * ? / L W
Month	1 ~ 12 or JAN ~ DEC	, - * /
Day-Of-Week	1 ~ 7 or SUN ~ SAT	, - * ? / L #
Year(optional)	empty, 1970 ~ 2099	, - * /



[그림 4-7] 배치 프로그램을 구현하는 모습

수행 tip

- 애플리케이션 설계인 배치 설계서를 이해하고 적합한 배치 프로그램 방식을 선정하여 구현하는 능력이 중요하다.
- 목표 시스템을 구성하는 하위 시스템 간의 연동 시 안정적이고 안전하게 동작할 수 있는 배치 프로그램을 구현하는 것이 중요하다.

4-2. 배치 프로그램 테스트

학습 목표	• 개발된 배치 프로그램의 테스트를 수행할 수 있다.

필요 지식 /

1 디버그(Debug)와 디버거(Debugger)

1. 디버그(Debug)의 개념

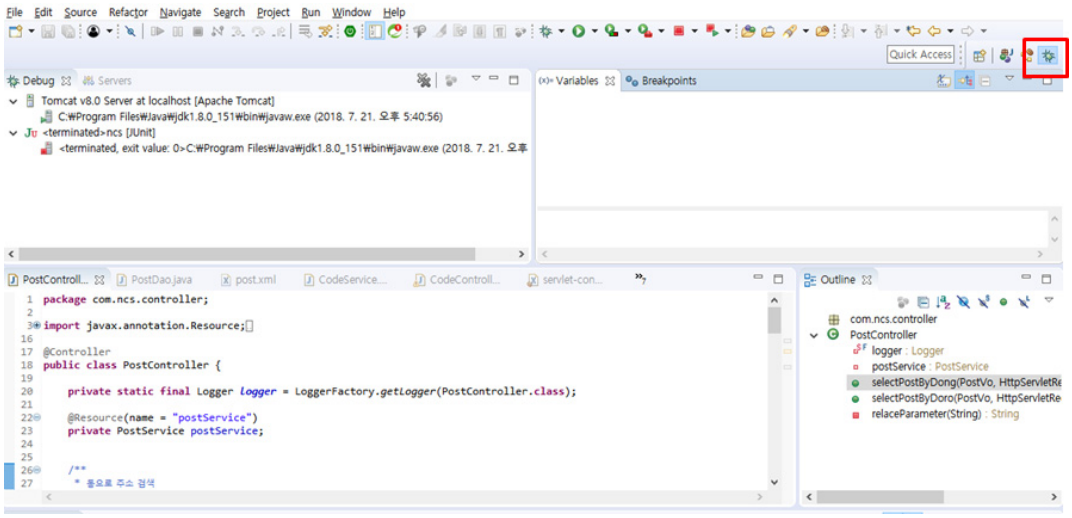
   디버그(Debug) 또는 디버깅(Debugging)은 컴퓨터 프로그램의 논리적인 오류(Bug)를 찾아내는 과정을 말한다.

2. 디버거(Debugger)의 개념

   디버거는 디버그를 돕는 도구이다. 디버거는 디버깅을 하려는 코드에 중단점을 지정하여 프로그램 실행을 중단하고, 코드를 단계적으로 실행하여 저장된 값을 확인 할 수 있도록 지원한다.

2 디버거(Debugger) 사용법

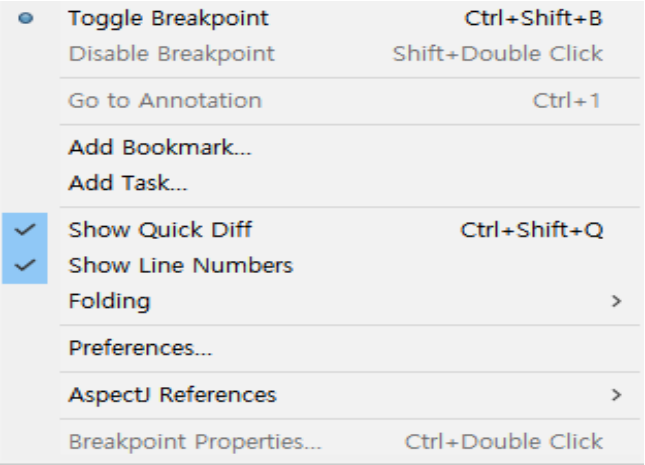
1. 디버그 뷰(Debug View)



[그림 4-8] 디버그 뷰(Debug View) 화면 예시(Eclipse 화면 예시)

2. 중단점(Break Pointer) 설정

편집기 왼쪽에 파란 부분(마커 바)을 더블 클릭하거나 해당 위치에서 마우스 오른쪽을 클릭하여 `Toggle Breakpoint'를 선택한다.



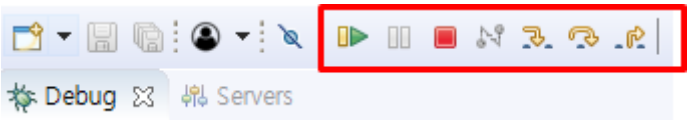
[그림 4-9] 브레이크 포인트 설정 화면 예시

3. 디버그 모드(Debug Mode)에서의 스텝(Step) 단위 진행

(1) 디버그 모드로 실행 중 지정된 중단점에 오면 디버거에 의해 중단되고 라인 단위로 실행이 가능해진다.

[그림 4-10] 중단점에서 중단된 화면 예시

(2) 디버그 뷰 버튼 사용법은 함수(Method) 안으로 진입하는 Step into, 함수 안으로 진입하지 않는 Step Over 등의 기능으로 스텝 단위로 디버깅을 한다.



[그림 4-11] 디버그 뷰(Debug View) 버튼 화면 예시

수행 내용 / 배치 프로그램 테스트하기

재료·자료

- 배치 설계서
- 개발된 프로그램 산출물

기기(장비 · 공구)

- 컴퓨터 또는 PC, 네트워크 장비, 프린터, 빔 프로젝트
- CASE 도구(SW 개발 지원 도구, SW 테스트 지원 도구 등)
- 형상관리 도구

안전 · 유의 사항

- 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 설계 산출물에 대한 이해가 선행되어야 한다.
- 테스트와 디버그의 이해가 선행되어야 한다.

수행 순서

1 배치 프로그램 주기를 변경한다.

배치 스케줄(Schedule) 주기를 테스트가 가능한 주기로 변경하여 로그에 정상적으로 출력되는지 확인한다.

```
@Scheduled(cron="0/10 * * * * *")
public void scheduleRun() {

    String batchResult = "성공";
    try {

        // 1. 업무 종료된 사용자 정보를 조회한다.
        List<UserVo> userList = userDao.selectRetireUsers();

        // 2. 업무 종료가 된 사용자 정보가 존재하지 않은 경우 4번을 진행하고 존재하는 경우 3번 작업을 진행한다.
        if(userList != null) {

            for(int i = 0; i < userList.size(); i++) {
                UserVo userVo = userList.get(i);

                // 3. 해당 사용자 정보를 삭제한다.
                userDao.deleteRetireUser(userVo);
```

[그림 4-12] 배치 스케줄 주기가 변경된 스케줄러 클래스 구문 예시

위 예시는 테스트를 위해 @Scheduled 어노테이션(Annotation) 구문 지정으로 10초

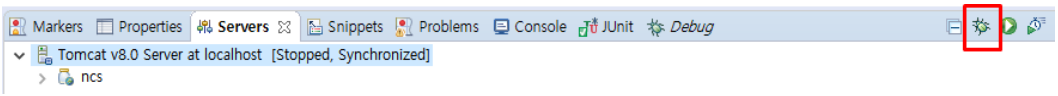
주기로 배치 스케줄이 실행되게 구현한 예시이다.

2 배치 프로그램을 디버깅하여 정상적으로 동작하는지 검증한다.

배치 프로그램은 목표 시스템의 품질에 영향도가 높은 작업이므로 디버깅(Debugging)을 통해 구문 확인이 필요하다.

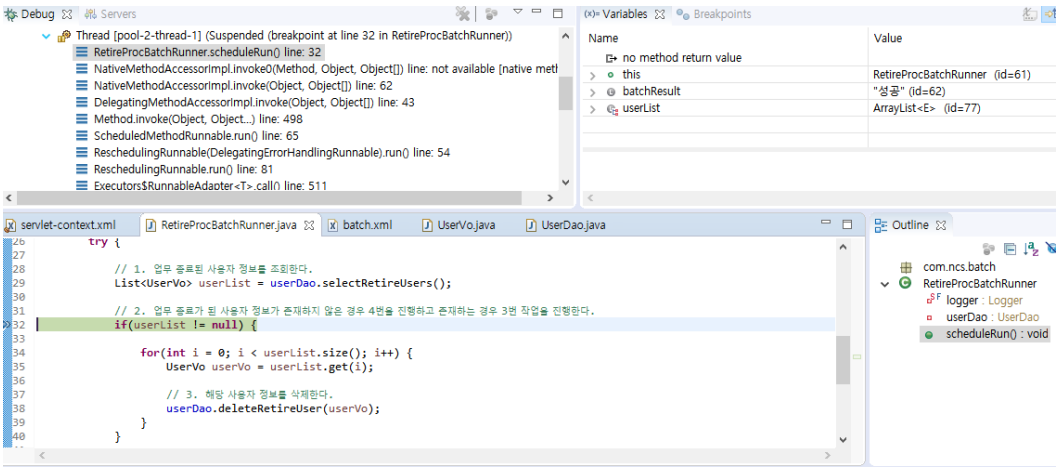
1. 디버그 모드(Debug Mode)로 실행한다.

   아래 예시는 디버그 모드로 실행하기 위해 붉은색 디버그 아이콘을 실행하는 예시이다.

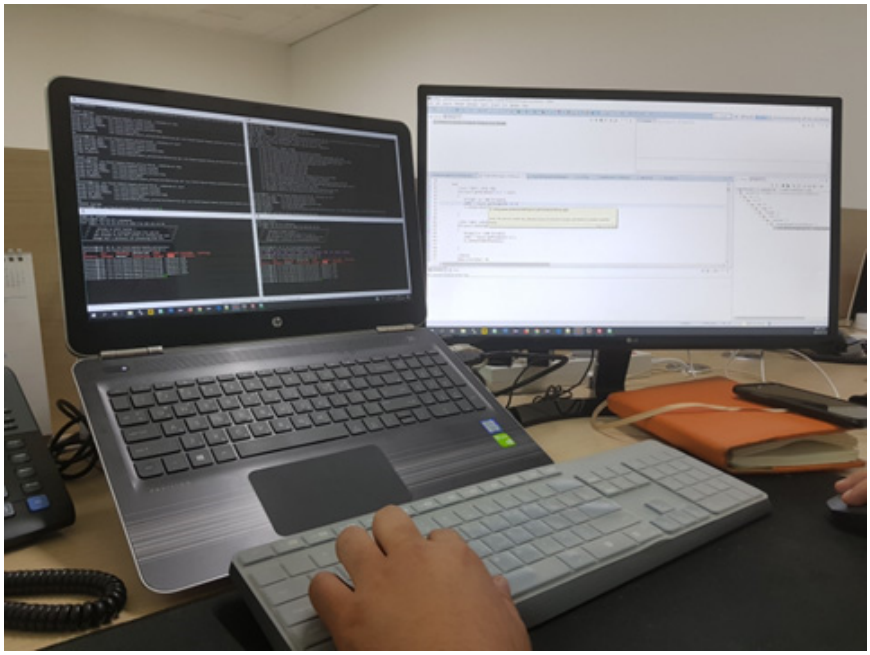


[그림 4-13] 디버그 모드 실행 예시

2. 검증할 코드 라인에 중단점(Break Point)를 지정하여 코드를 검증한다.



[그림 4-14] 중단점이 지정된 코드에서 실행이 중단된 화면 예시



[그림 4-15] 배치 프로그램을 디버깅하는 모습

3 배치 프로그램의 로그를 확인한다.

```
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:26:10
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:26:20
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:26:30
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:26:40
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:26:50
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:27:00
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:27:10
DEBUG: com.ncs.sql.selectRetireUsers - ==>  Preparing: SELECT SABUN FROM TBL_USERS WHERE DATE_FORMAT(RETIREDATE, '%Y-%m-%d') < DATE_FORMAT(CURDATE(), '%Y-%m-%d')
DEBUG: com.ncs.sql.selectRetireUsers - ==> Parameters:
DEBUG: com.ncs.sql.selectRetireUsers - <==      Total: 0
INFO : com.ncs.batch.RetireProcBatchRunner - 스케줄 실행 : [성공] 2018-07-24 15:27:20
```

[그림 4-16] 배치 프로그램 로그 예시

4 목적 테이블에서 정상적으로 배치 프로그램이 실행되었는지 확인한다.

1. 배치 프로그램을 실행하기 전 데이터를 확인한다.

SELECT SABUN, RETIREDATE
FROM TBL_USERS;

실행 결과

SABUN	RETIREDATE
0001	2018-07-23 16:39:37
0002	2018-07-24 16:40:33
0003	2018-07-21 16:41:30
0004	2018-07-20 16:41:54
0005	2018-07-24 16:42:10

[그림 4-17] 배치 프로그램 실행 전 데이터 확인 예시

2. 배치 프로그램을 실행하였을 경우의 대상 데이터를 확인한다.
   아래 예시는 업무 종료된 사용자 정보를 삭제하는 경우이다.

```
SELECT SABUN, RETIREDATE
FROM TBL_USERS
WHERE
DATE_FORMAT(RETIREDATE,
'%Y-%m-%d')
< DATE_FORMAT(CURDATE
(), '%Y-%m-%d')
```

실행 결과

SABUN	RETIREDATE
0001	2018-07-23 16:39:37
0003	2018-07-21 16:41:30
0004	2018-07-20 16:41:54

[그림 4-18] 배치 프로그램의 대상 데이터 확인 예시

3. 배치 프로그램을 실행한 후의 데이터를 확인한다.

```
SELECT SABUN, RETIREDATE
FROM TBL_USERS;
```

실행 결과

SABUN	RETIREDATE
0002	2018-07-24 16:40:33
0005	2018-07-24 16:42:10

[그림 4-19] 배치 프로그램 실행 후 데이터 확인 예시

5 배치 프로그램이 정상적으로 실행되지 않았을 경우 오류를 확인한다.

1. 데이터가 목적에 맞지 않게 처리된 경우는 참조 테이블의 데이터를 확인하여 오류를 확인한다.
2. 로그에 실패가 출력된 경우 디버깅을 통해 오류를 확인한다.

수행 tip

- 개발된 배치 프로그램이 정상적으로 실행될 수 있도록 디버깅을 수행하는 방식과 문제점을 찾아내는 능력을 향상시킬 수 있도록 테스트를 수행하는 것이 중요하다.

학습 4 교수·학습 방법

교수 방법

- 애플리케이션 설계 명세를 기반으로 선정된 개발언어와 도구를 활용하여 개발하는 절차에 대해 정리하여 설명한다.
- 배치 프로그램 작성 시 연동되는 하위 시스템과 연동하는 방안에 대해 정리하여 설명한다.
- 배치 프로그램을 안정적이고 안전하게 동작하기 위한 절차와 기법에 대해 정리하여 설명한다.
- 배치 프로그램 테스트를 위한 디버깅 절차와 기법에 대해 자료로 정리하여 제시하고, 이에 대해 상세히 설명한다.
- 배치 프로그램 테스트를 수행하기 위한 테스트 조건을 명세화하는 방안에 대해 제시하고, 상세히 설명한다.

학습 방법

- 애플리케이션 설계 명세를 기반으로 선정된 개발언어와 도구를 활용하여 개발하는 절차에 대해 학습한다.
- 배치 프로그램 작성 시 연동되는 하위 시스템과 연동하는 방안을 학습하고 실습한다.
- 배치 프로그램을 안정적이고 안전하게 동작하기 위한 절차와 기법에 대해 학습한다.
- 배치 프로그램 테스트를 위한 디버깅 절차와 기법에 대해 학습하고 해당 절차와 기법에 따라 실습함으로써 실무에서 활용할 수 있도록 한다.
- 배치 프로그램 테스트를 수행하기 위한 테스트 조건을 명세화하는 방안에 대해 학습한다.

학습 4 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
배치 프로그램 구현	- 애플리케이션 설계를 기반으로 프로그래밍 언어와 도구를 활용하여 배치 프로그램 구현 기술에 부합하는 배치 프로그램을 구현할 수 있다.			
	- 목표 시스템을 구성하는 하위 시스템 간의 연동 시 안정적이고 안전하게 동작할 수 있는 배치 프로그램을 구현할 수 있다.			
배치 프로그램 테스트	- 개발된 배치 프로그램의 테스트를 수행할 수 있다.			

평가 방법

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
배치 프로그램 구현	- 애플리케이션 설계서와 배치 프로그램 간의 정합성			
	- 배치 프로그램 구현 시 안정적이고 안전하게 동작할 수 있는 정도			
배치 프로그램 테스트	- 배치 프로그램 테스트 시 오류 발견과 문제점을 해결해 가는가에 대한 적정성			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
배치 프로그램 구현	- 애플리케이션 설계서와 배치 프로그램 간의 정합성			
	- 배치 프로그램 구현 시 안정적이고 안전하게 동작할 수 있는 정도			
배치 프로그램 테스트	- 배치 프로그램 테스트 시 오류 발견과 문제점을 해결해 가는가에 대한 적정성			

피드백

1. 포트폴리오
   - 배치 프로그램 테스트를 위한 테스트 케이스 적용 기법에 대해 파악한 후, 프로그램 특성에 맞는 기법 선정 방안과 테스트 케이스 생성 방법에 대해 설명한다.
   - 배치 프로그램 수행 시 목표 시스템을 구성하는 하위 시스템 간의 연동 시 안정적이고 안전적으로 구현되는지를 점검하고 미비 사항을 정리하여 보완 사항을 제시한다.
   - 배치 프로그램 기능 검증을 위해 디버깅 방식과 로그의 완전성을 검토하고 미비 사항을 보완할 수 있도록 설명한다.
   - 배치 프로그램 구현 결과가 '상'인 경우 모범 사례로 공유한다.
   - 배치 프로그램 구현 결과가 '중'인 경우 부족한 부분을 보완할 수 있도록 설명한다.
   - 배치 프로그램 구현 결과가 '하'인 경우 모범 사례를 참조하여 다시 구축할 수 있도록 유도한다.
2. 평가자 체크리스트
   - 실습 과정에서 평가한 후에 개선해야 할 사항 등을 정리하여 설명한다.
   - 애플리케이션 설계서에 따라 배치 프로그램이 작성되었는지를 확인하여 미비 사항을 보완할 수 있도록 보완 사항을 제시한다.
   - 배치 프로그램 수행 시 목표 시스템을 구성하는 하위 시스템 간의 연동 시 안정적이고 안전적으로 구현할 수 있는지를 점검하고 미비 사항을 정리하여 설명한다.
   - 배치 프로그램 테스트를 위해 디버깅 방식과 도구 활용도를 파악하고 미비 사항을 정리하여 실무 자료를 참고 자료로 제시한다.

참고자료

- 교육부(2014). 애플리케이션 구현(LM2001020203_14v2). 세종: 한국직업능력개발원.
- 교육부(2015). 프로그램 언어 활용(LM2001020215_15v3). 세종: 한국직업능력개발원.
- 교육부(2016). SQL 활용(LM2001020413_16v3). 세종: 한국직업능력개발원.
- 교육부(2016). 기능 모델링(LM2001020220_16v1). 세종: 한국직업능력개발원.
- 교육부(2016). 애플리케이션 설계(LM2001020221_19v5). 세종: 한국직업능력개발원.
- 소프트웨어공학포털(2019). SW 개발도구. http://www.sw-eng.kr에서 2018. 06. 05. 검색.
- 정보통신산업진흥원(2018). 공개SW 테스트 도구. https://www.oss.kr에서 2018. 06. 04. 검색.
- 한국정보화진흥원 표준프레임워크센터(2018. 03.). 『정보 시스템 구축 발주자를 위한 표준 프레임워크 적용 가이드 Ver. 3.7』.
- 한국정보화진흥원(2011. 9.). 『제안 요청서의 요구 사항 작성 가이드』.
- 행정자치부(2012. 09.). 『전자정부 SW개발·운영자를 위한 Java 시큐어 코딩 가이드』.

활용서식

단위 시스템 공통 모듈 관리 대장

순번	ID	시스템	기능	설명	엔티티	입력	담당자	출력 (정상)	출력 (에러)	입력 항목 설명	출력 항목 설명

프로그램 관리 대장

순번	ID	구분	시스템	기능	설명	엔티티	입력	담당자	출력(정상)	출력(에러)	입력 항목 설명	출력 항목 설명

단위 테스트 케이스

테스트 케이스	프로젝트명:		
	대상 시스템:		
단계: 단위 테스트	작성자:	승인자:	문서 상태:
작성일:	버전:	테스트 범위:	테스트 조직:

테스트 ID		테스트 일자	
테스트 목적			
테스트 기능			
입력값			

테스트 단계	테스트 케이스	예상값	중요도	성공 /실패	비고

테스트 환경	
전제 조건	
성공/실패 기준	
특수 절차	

NCS학습모듈 개발이력

발행일	2015년 12월 31일		
세분류명	응용SW엔지니어링(20010202)		
개발기관	한국소프트웨어기술진흥협회, 한국직업능력개발원		
집필진	강석진(이비스톰)* 김보운(이화여자대학교) 김홍진(LG CNS) 유은희 장현섭((주)커리텍) 주선태(T3Q) 진권기(이비스톰) 최재준	검토진	김승현(경희대학교) 엄기영(우리에프아이에스) 장온순(한국IT컨설팅) 조상욱(세종대학교) 조성호(삼성카드) *표시는 대표집필자임
발행일	2017년 12월 31일		
세분류명	응용SW엔지니어링(20010202)		
개발기관	(사)한국정보통신기술사협회, 한국직업능력개발원		
집필진	박미화(동국대학교)* 김승환((주)캐롯아이) 김원기(LG CNS) 김종명(SM신용정보) 박현기(프리랜서) 유현주((사)한국정보통신기술사협회) 이구성(한국아이씨티(주)) 이숙희(서초문화예술정보학교) 최창선(한빛디엔에스(주)) 홍민표(한화S&C)	검토진	권순명(㈜씨에이에스) 김태형((사)한국정보통신기술사협회) 양승화(라이나생명보험) 이성화(시스원) 황극인(㈜코스콤) *표시는 대표집필자임
발행일	2018년 12월 31일		
학습모듈명	서버프로그램 구현(LM2001020211_16v4)		
개발기관	(사)한국정보통신기술사협회, 한국직업능력개발원		
집필진	김광국(㈜코스콤)* 권기학(세종대학교) 김승환(㈜캐롯아이) 김유석(한국정보통신기술사협회) 문정기(교보정보통신) 박미화(동국대학교 융합SW교육원) 이주희(㈜유알피시스템) 홍민표(한화S&C)	검토진	류갑상(동신대학교) 엄기영(우리에프아이에스) 장성수(세명컴퓨터고등학교) 장희수(㈜유알피시스템) *표시는 대표집필자임
발행일	2021년 12월 31일		
학습모듈명	서버프로그램 구현(LM2001020211_19v5)		
개발기관	(사)한국정보통신기술사협회(개발책임자: 온기현), 한국직업능력연구원		

서버프로그램 구현(LM2001020211_19v5)

저작권자	교육부
연구기관	한국직업능력연구원
발행일	2021. 12. 31.

※ 이 학습모듈은 자격기본법 시행령(제8조 국가직무능력표준의 활용)에 의거하여 개발하였으며, NCS통합포털사이트(http://www.ncs.go.kr)에서 다운로드 할 수 있습니다.

National Competency Standards Learning Module

NCS
학습모듈

www.ncs.go.kr