

CTIP 환경 구축 - Basic

※ Contents

1. CI Server
2. Code Configuration Management
3. Requirements Management & Bug Tracking
4. Team Communication
5. Unit Testing
6. Automatic Build
7. IDE
8. Aggregation

Team #1

201411273 박재범
201411295 이상훈
201510436 허윤아
201511244 김민우

1. CI Server : Jenkins

CI (Continuous Integration) 도구로 **Jenkins** 채택.

- 오랜 시간 널리 상용화된 도구 → 많은 커뮤니티 및 플러그인 지원, 높은 호환성 제공.
- 오픈소스 프로젝트로 무료 이용 가능.

또한, Jenkins를 클라우드 **가상 서버**에서 실행하여 서버 관리가 용이하고 언제든지 Jenkins 접근 가능

클라우드 가상 Ubuntu 서버



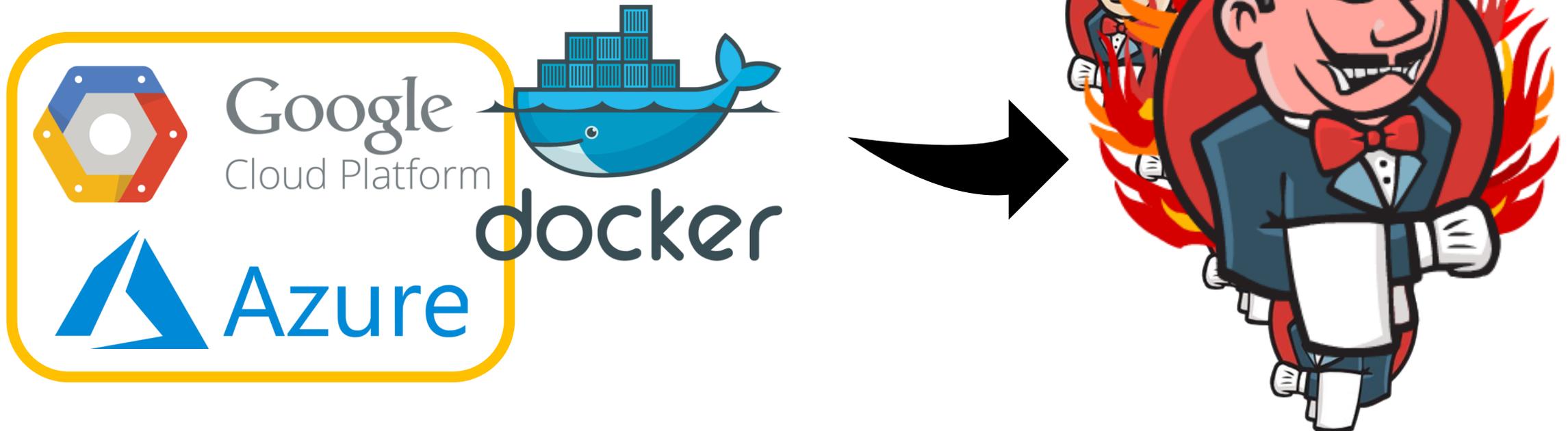
1. CI Server : Docker ?

Docker란?

- 컨테이너 기반 오픈소스 프로세스 격리 플랫폼
- 프로그램, 메시지 등 다양한 프로세스를 컨테이너로 만들어 동작
- 가상머신보다 가볍기 때문에 서버에서 여러 프로그램 실행에 용이
- 이미지 파일을 이용하여 새로운 서버 구축 혹은 유지 관리가 매우 편리

그러나..

가상 서버 하나에서 4개의 Jenkins 컨테이너를 돌리기엔 무리

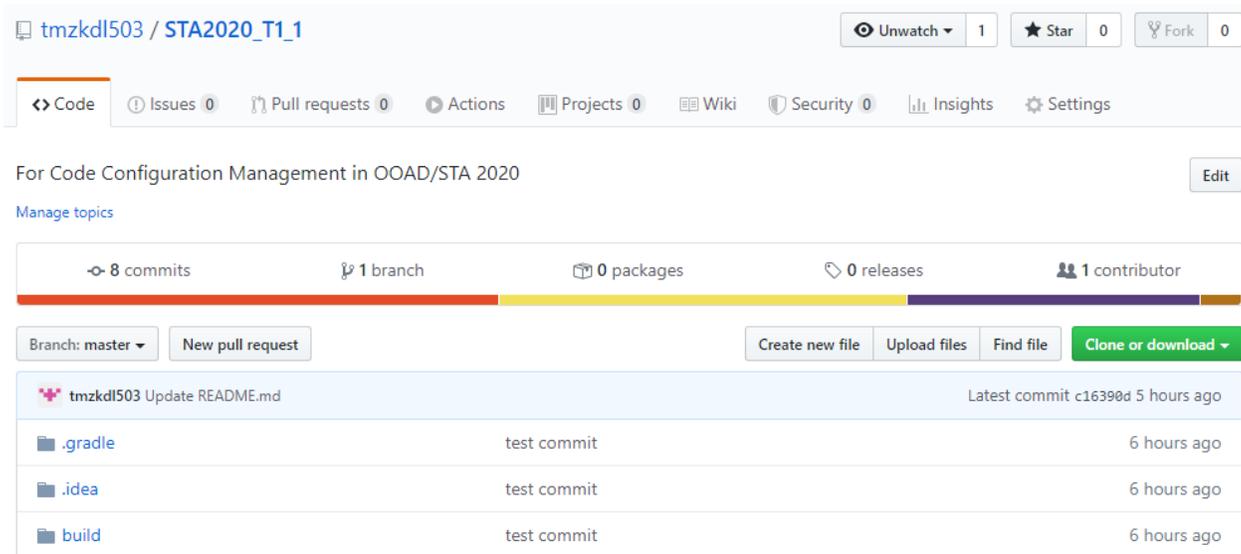


2. Code Configuration Management : Git

Code Configuration Management 도구로 **Git** 채택.

- 간단하게 병렬 협업 환경 구축이 가능.
- 모든 변경 기록 유지 및 열람 가능. → Issue Tracking에 활용할 수 있음.
- 분산 버전 관리 시스템으로 commit시 부담이 적음.
- 다른 도구들과 연동성이 좋음.

Github의 Repository를 적극 활용하여 Code configuration을 원활히 하고 다른 도구와 활발히 연동. Github는 웹에서 바로 접근이 가능하다는 장점이 있음.



The screenshot shows a GitHub repository page for 'tmzkdI503 / STA2020_T1_1'. The repository is for 'Code Configuration Management in OOAD/STA 2020'. It has 8 commits, 1 branch, 0 packages, 0 releases, and 1 contributor. The current branch is 'master'. The repository contains files: '.gradle', '.idea', and 'build', all with 'test commit' messages and committed 6 hours ago. The latest commit is 'c16390d' from 5 hours ago.



3. Requirements Management & Bug Tracking

Issue Tracking(Requirements Management & Bug Tracking) 도구로 **Jira** 또는 **Notion**.

Requirement Management와 Bug Tracking을 모두 **Issue Tracking** 도구에서 관리하는 경향.
- 여러 도구의 병행으로 발생하는 오버헤드 방지. → 안정적인 초기 학습곡선, 업무 효율성 증진.

현재 가장 유명하고 많이 쓰이고 있는 Issue Tracking 도구 2개

- **Jira**

- **Notion**

(2가지 모두 웹으로 접근이 가능.)



Jira Software

VS



Notion

3. Requirements Management & Bug Tracking : Jira

Jira는...

장점

- 애자일 SW 개발팀에 최적화된 이슈 및 버그 트래킹, 프로젝트 관리 목적으로 만들어진 서비스.
- 정형화된 사용법으로 가독성과 일관성을 유지할 수 있다.
- 통계 및 시각화 기능을 자체적으로 제공.
- 복잡한 대규모 프로젝트에도 적합.
- Git과의 연동이 가능. → 관련 이슈를 추적하는 데 용이.
- 대기업을 포함한 많은 기업에서 채택하여 사용 중.

단점

- 대부분의 유용한 기능은 유료.
- 처음 배우고 익숙해지는 것에 더 많은 시간이 소요.
- 실질적으로 불필요한 기능이라도 완전히 제거할 수 없어 그에 맞춘 사용이 강요.



3. Requirements Management & Bug Tracking : Notion

Notion은...

장점

- All-in-one Workspace를 지향하는 플래닝, 이슈 트래킹, 프로젝트 관리 등의 다목적 서비스.
- 모든 부분이 커스텀이 가능한 유연하면서도 편리한 템플릿으로 쉽게 익히고 사용 가능.
- 필요한 기능이 있으면 직접 만들어서 사용할 수 있음.
- Git Gist, Code Block을 지원하여 코드를 직접 보면서 피드백을 주고받을 수 있음.
- 누가 어떤 항목을 열람 또는 수정 중인지 태그를 통해 실시간으로 확인할 수 있음.
- 많은 실리콘 벨리, 스타트업 기업에서 채택하여 사용 중임.

단점

- 코드 관리에 특화된 어플리케이션이 아니다.
- 다른 도구들과 호환성, 이식성이 떨어짐
- 그래프 기능을 지원하지 않고 통계 시스템이 부족함.
- 높은 자유도로 인해 일관성이 무너질 가능성이 있음.



3. Requirements Management & Bug Tracking : Notion

Issue Tracking(Requirements Management & Bug Tracking) Notion 채택.

- 오픈소스 소프트웨어
- Jira에 비해 사용에 용이 → 빠르게 적응할 수 있다.

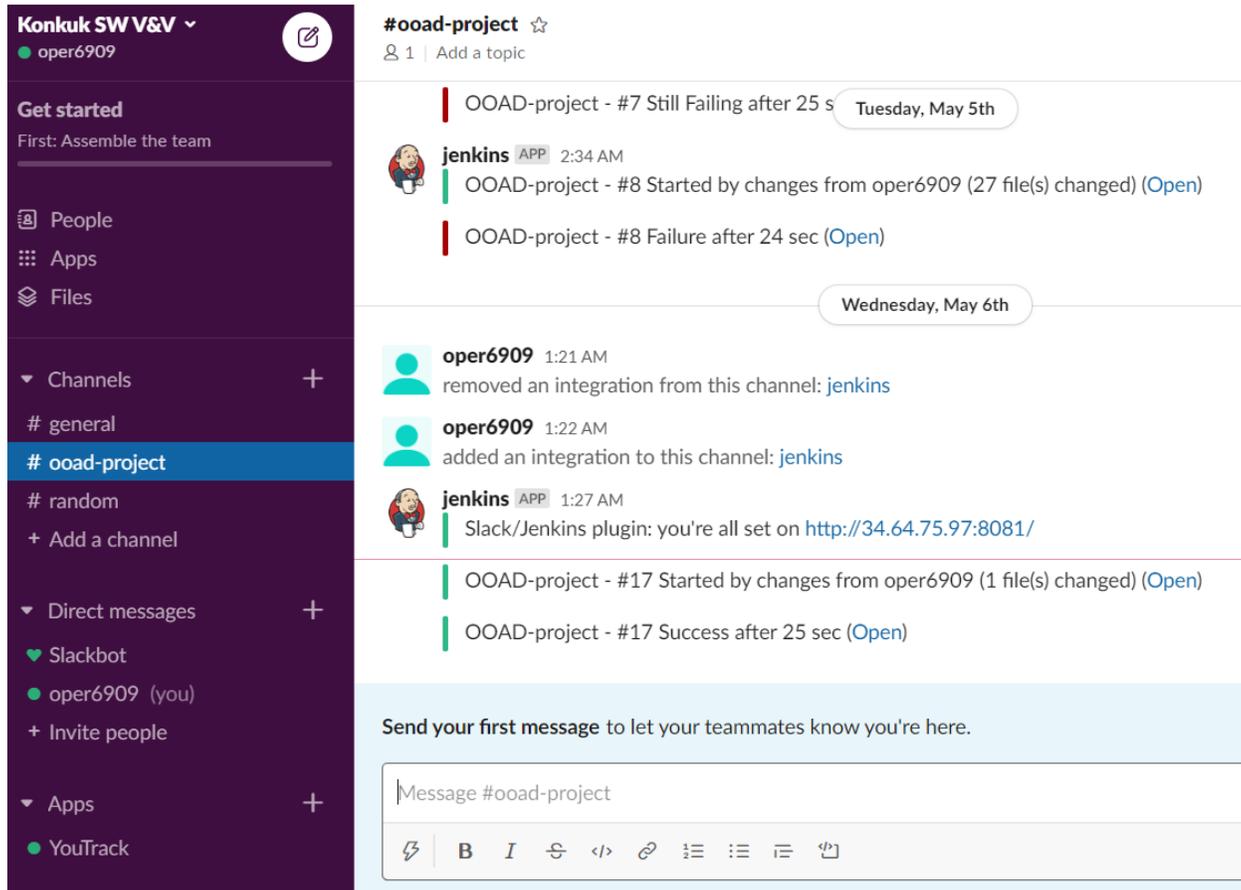
The screenshot displays a Notion workspace for 'Issue Management'. The left sidebar contains navigation options: 'Quick Find', 'All Updates', 'Settings & Members', 'Workspace' (with '엔지니어링 위키' and '이슈 관리'), and 'Private' (with 'Add a page', 'Templates', 'Import', and 'Trash'). The main content area shows a Kanban-style board with three columns: '할 일' (1 item), '진행 중' (1 item), and '완료' (1 item). Each item is a task card with a title, description, and metadata. For example, the 'In Progress' item is 'CTIP 구축' (CTIP Construction) with a due date of May 3, 2020, 3:46 AM, and assigned to '민우 김' and '달가'. The 'Completed' item is '기능안전성 표준 조사' (Functional Safety Standard Investigation) with a due date of May 3, 2020, 4:09 AM, and assigned to '민우 김' and '달가'. The 'To-do' item is 'Jenkins - JUnit 연동 오류' (Jenkins - JUnit Integration Error) with a due date of May 8, 2020, 11:59 AM.



4. Team Communication : Slack

Team Communication 도구로 **Slack** 채택.

- Git, Jenkins, Notion 등 CTIP 환경 내 다른 도구들과 연동이 가능.
- 목적 별 채널 분리로 명확한 의사소통이 가능.
- 스마트폰 App을 통한 알림 서비스 이용 가능. → 실시간 피드백에 도움.
- 일반적인 메신저와 크게 다르지 않은 익숙한 인터페이스.



The screenshot shows a Slack interface for a workspace named 'Konkuk SW V&V'. The user 'oper6909' is in a channel named '# ooad-project'. The channel history shows several messages from the 'jenkins' app and the user 'oper6909'. The messages include status updates about failed and successful builds, and announcements regarding the removal and addition of the Jenkins integration. A light blue banner at the bottom prompts the user to 'Send your first message to let your teammates know you're here.' Below the banner is a text input field with a placeholder 'Message #ood-project' and a rich text editor toolbar.



5. Unit Testing : JUnit

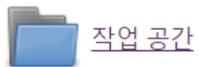
Unit Testing 도구로 **JUnit** 채택.

- 가장 유명한 Java Unit Test Platform.
- Build 자동화 도구에서 쉽게 사용 가능.
- Jenkins에서 연동하면 Unit Test 결과를 한눈에 보기 쉽게 보여준다.

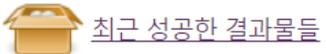
JUnit

Project OOAD-project

OOAD-project



작업 공간



최근 성공한 결과물들



최근 변경사항



가장 최근 테스트 결과 (실패가 없습니다)



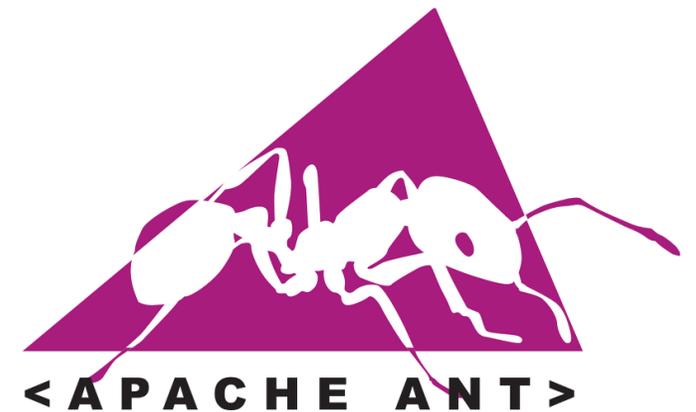
6. Automatic Build

Automatic Build 도구로 **Gradle** vs **Maven** vs **Apache Ant**

Build란?

- 컴파일, 패키징, 테스트 등의 작업을 통해 소스 코드파일을 실행가능한 소프트웨어로 변환하는 작업

가장 유명한 자동화 빌드 툴 3개.



6. Automatic Build : Apache Ant

Ant는...

장점

- xml파일에 기반하기 때문에 플랫폼에 종속성이 없다.
- xml파일에 의해 계층성 확인이 쉽다.
- 개발자가 프로젝트에 따라 자유롭게 빌드 절차 정의.

단점

- 스크립트 재사용성이 떨어진다.
- xml, 원격 저장소를 가져오기 위해 별도의 어플리케이션 필요

Maven™

 **Gradle**



6. Automatic Build : Maven

Maven은...

장점

- xml 파일을 이용한 빌드 정형화.
- 프로젝트 전반적인 관리 기능 포함.
- 다양한 플러그인.

단점

- 라이브러리 종속성이 발생하면 xml이 복잡해 진다.
- 복잡한 xml 파일 관리에 의한 급격한 러닝 커브.

The logo for Apache Maven, featuring the word "Maven" in a bold, black, sans-serif font. The letter 'v' is replaced by a red and black feather. A small "TM" trademark symbol is located to the right of the word.The logo for Gradle, consisting of a dark blue elephant silhouette on the left and the word "Gradle" in a green, sans-serif font on the right.The logo for Apache Ant, featuring a white ant silhouette on a purple triangular background. Below the triangle, the text "< APACHE ANT >" is written in a black, sans-serif font.

6. Automatic Build : Gradle

Gradle은...

장점

- Java 외에도 여러 언어의 빌드 환경 구성 가능
- 많은 레퍼런스 문서와 활발한 플러그인 허브 덕분에 플러그인 사용이 편리.
- 빌드 속도 향상을 위한 기능 제공 → 빠른 빌드 속도
- 스크립트 기반의 간편한 사용법.

단점

- 딱히 없다.



Maven™



6. Automatic Build : Gradle

Automatic Build 도구로 **Gradle** 채택.

- Java 외에도 다양한 언어 빌드 환경 구성 가능.
- 복잡하지 않은 스크립트 파일 관리.
- Ant와 Maven의 장점을 계승



```
build.gradle (testGradleProject) x Calculator.java x CalculatorTest.java x
1  plugins {
2      id 'java'
3  }
4
5  group 'org.example'
6  version '1.0-SNAPSHOT'
7
8  repositories {
9      mavenCentral()
10 }
11
12 dependencies {
13     testCompile group: 'junit', name: 'junit', version: '4.12'
14 }
15
16 test {
17     useJUnit()
18 }
19
20
21 jar {
22     manifest {
23         attributes 'Main-Class': 'Calculator'
24     }
25 }
```

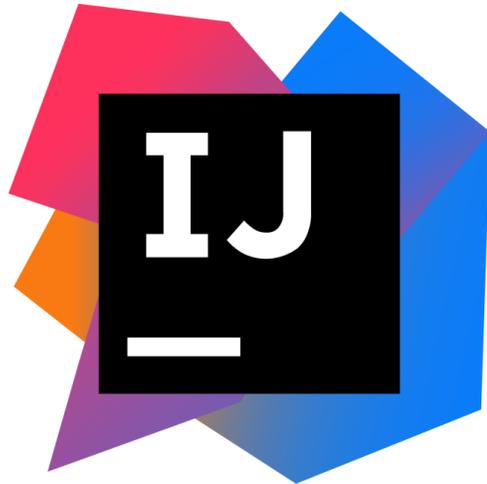
7. IDE

IDE 도구로 **IntelliJ** vs **Eclipse**

IDE란?

- 여러 개발자 툴 및 개발 작업을 하나의 프로그램에서 진행할 수 있는 개발 환경.

가장 유명한 자동화 Java IDE 2개.



7. IDE : IntelliJ

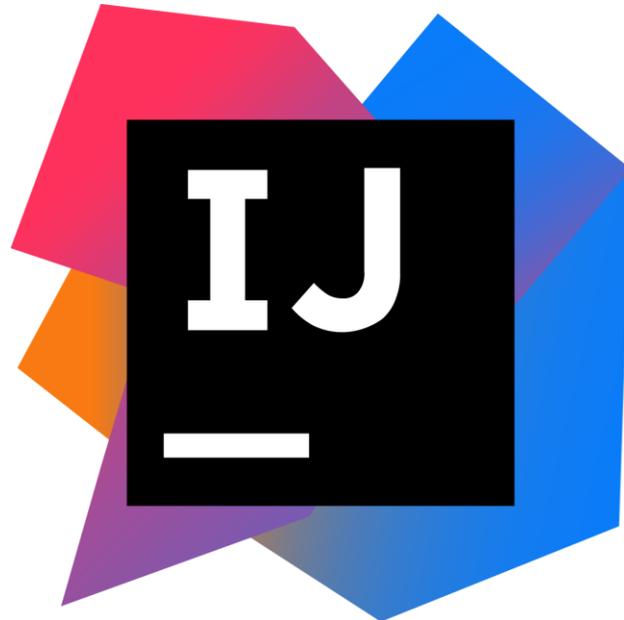
IntelliJ는...

장점

- 편리한 코드 자동 완성 기능. 유사하거나 최근 사용된 코드를 추천해준다.
- 안정적인 플러그인
- 많은 자체 빌트인 기능 제공

단점

- 많은 플러그인 때문에 무겁다.



7. IDE : Eclipse

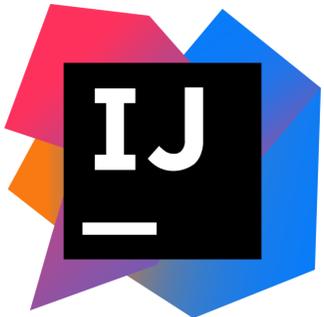
Eclipse는...

장점

- 무료
- 아주 많은 플러그인

단점

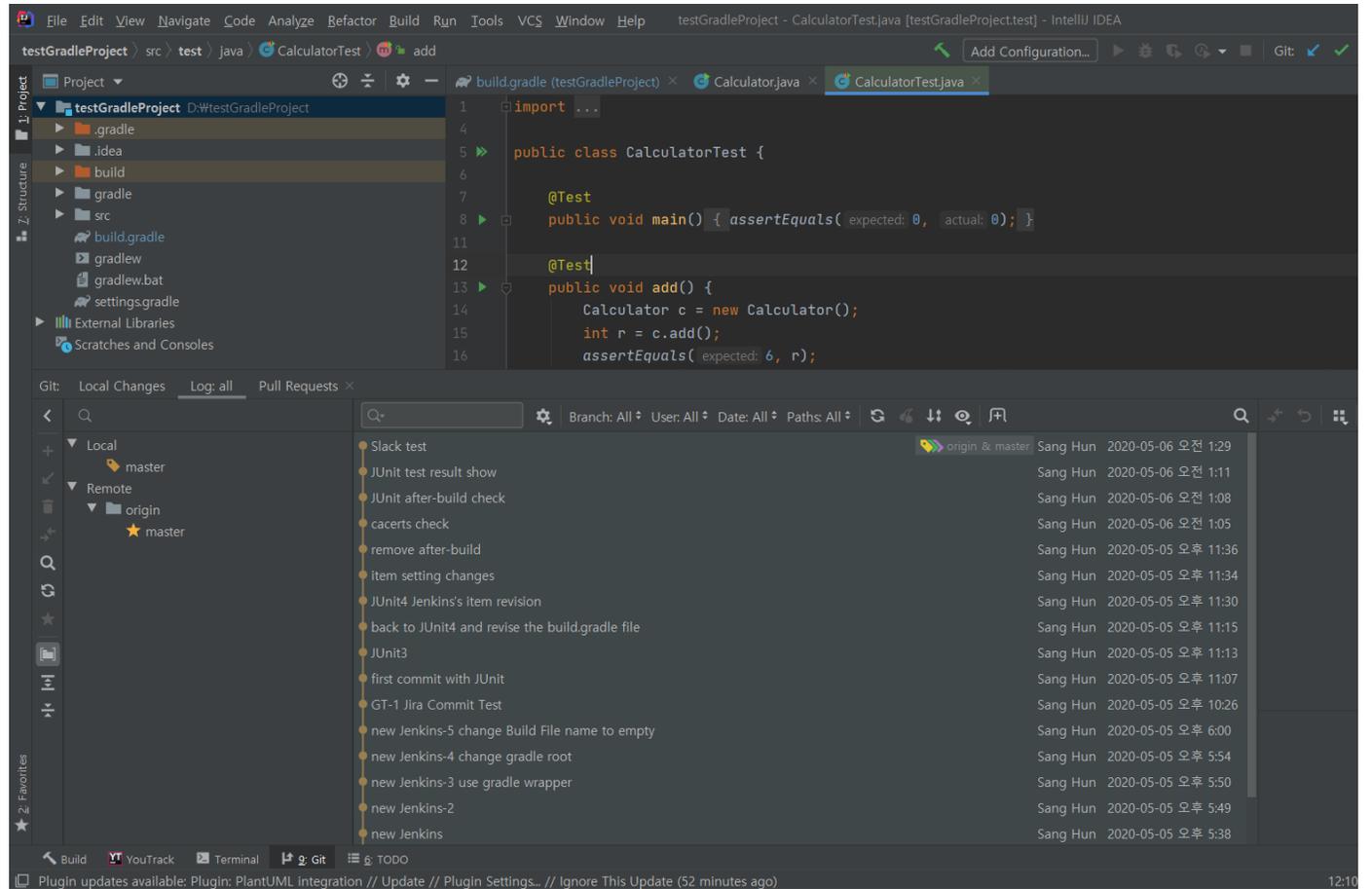
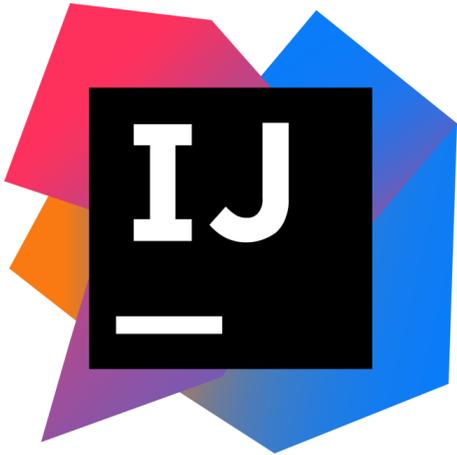
- 불안정한 플러그인이 많다.
- 그리 똑똑하지 않은 자동완성.



7. IDE : IntelliJ

IDE 도구로 **IntelliJ** 채택.

- Java IDE 중 가장 큰 점유율 차지.
- 코드 자동 완성 기능 덕분에 유사한 단어가 많은 객체지향 프로젝트에 유용
- 다양하고 안정적인 플러그인 지원
- 자체 내장된 Git



8. Aggregation

현재까지의 CTIP 환경 전체 구성도

