오픈 소스 라이브러리

오픈 소스

- Open Source
  - ✓ 오픈 소스: 소프트웨어 혹은 하드웨어 제작자의 권리를 지키면서 원시 코드를 누구나 열람할 수 있도록 한 소프트웨어 혹은 오픈 소스 라이선스에 준하는 모든 것
  - ✓ 오픈 소스의 장점
    - ❑ 편리성: 사용자가 직접 처음부터 개발하려고 하면 그 만큼의 시간과 노동력이 소요됨
    - ❑ 뛰어난 성능: 대부분 다수의 개발자들이 협업해서 만들었기 때문에 성능이 우수할 가능성이 높음
    - ❑ 신뢰성: 오픈 소스 라이브러리의 버전이 높다면 많은 프로그래머들이 지속적으로 기능과 성능을 검증해 주었으므로 신뢰성이 높다.

Java 오픈 소스

- Java Open Source Project
  - ✓ 하둡(Hadoop)
    - 관련 URL: http://hadoop.apache.org/
    - 아파치 재단에서 제공하는 오픈 소스 분산 처리 시스템으로서 획기적인 방법으로 분산 처리 플랫폼을 제공
  - ✓ 카산드라 데이터베이스(Cassandra Database)
    - 관련 URL: http://cassandra.apache.org/
    - 확장성을 위해서 분산 저장 시스템 기능을 제공하는 동시에 RDBMS보다 간단한 데이터 저장 형식을 지원
    - 기존의 RDBMS에서 사용하던 테이블 대신 Hashtable과 같이 Key/Value 형식의 데이터 저장 형식을 갖고 있음 → NoSQL 데이터베이스
  - ✓ 루씬(Lucene)
    - 관련 URL: http://lucene.apache.org/
    - 고성능의 인덱싱(색인 작업) 및 텍스트 기반의 검색 엔진 플랫폼

Java 오픈 소스

- Java Open Source Project
  - ✓ 메이븐(Maven)
    - ❑ 관련 URL : http://maven.apache.org/
    - ❑ Java Build 및 외부 라이브러리 사용을 편리하게 하는 도구
  - ✓ 아파치 웹 서버(Apache WEB Server)
    - ❑ 관련 URL : http://httpd.apache.org/
    - ❑ 윈도우 계열부터 유닉스 계열의 서버까지 다양한 플랫폼에서 동작하고 기능 확장성이 뛰어난 웹 애플리케이션 서버
  - ✓ Log4j
    - ❑ Java 클래스에서 로그를 생성 및 기록하고 관리하기 위한 오픈 소스 프레임워크

Java 오픈 소스

- Java Open Source Project
  - ✓ Commons Project
    - ❑ URL : http://commons apache.org/

컴포넌트	설명
BeanUtils	클래스가 어떤 메소드와 속성으로 구성되어 있는지 알 수 있는 기능을 제공하는 리플렉션(reflection)과 객체의 상태와 그 정보를 수집하고 변경할 수 있는 인트로스펙션(introspection)을 쉽게 구현할 수 있는 API를 제공한다.
Collections	앞서 알아본 ArrayList, Hashmap과 같은 자바 기본 컬렉션(Collections)과 관련된 기능의 확장이라고 생각하면 된다.
DBCP	자바 프로그램에서 데이터베이스로부터 데이터를 가져오기 위해서는 Connection이라는 객체가 필요하다. 이 Connection 객체를 생성하려면 시스템 리소스가 많이 필요하며, IO 관련 클래스와 같이 반환(close)을 해주어야 한다. 이렇게 Connection 객체들을 미리 만들어 놓고 저장한 다음 애플리케이션이 데이터베이스에 접속할 때마다 저장한 Connection 객체를 전달하는 기능을 풀링(pooling)이라고 한다. 풀링은 또한 Connection 객체의 개수를 제한하는 등 다양한 제어 기능을 제공하고 있다. DBCP 컴포넌트는 이런 풀링 기능을 제공해준다.

Java 오픈 소스

- Java Open Source Project
  - ✓ Commons Project
    - ❑ URL : http://commons apache.org/

Email	이메일을 발송하는데 필요한 여러 가지 기능들을 모아 놓은 컴포넌트다.
FileUpload	일반적으로 웹 환경의 클라이언트에서 서버로 파일을 전송하는데 사용하는 컴포넌트다.
HttpClient	웹에서 사용하는 HTTP 프로토콜을 사용해서 클라이언트가 할 수 있는 기능을 제공해준다. 예를 들어 HttpClient를 사용하여 특정 웹 사이트의 정보를 가져오거나 서버와 데이터를 주고 받을 수 있다.
JCS	자바 캐시 시스템(Java Cache System)으로 메모리에 데이터를 올려놓고 데이터를 전달해주는 기능을 한다. 보통 쓰기 빈도가 적고, 읽기 빈도가 많은 데이터들을 메모리와 같은 읽기 속도가 빠른 저장 매체에 올려놓기 위해서 사용한다. 읽기 속도가 빠르다는 장점이 있다.

Maven

Maven

- Build
  - ✓ Source Code를 Compile을 수행하고 최종적인 실행 파일로 만드는 작업
  - ✓ 자바에서는 .java 파일을 컴파일하여 적절한 단위로 JAR 파일로 합친 후 경우에 따라서는 실행을 위한 스크립트나 설정 파일을 부여하여 다른 시스템에서 실행 가능한 파일의 세트를 만드는 작업
  - ✓ JDK에 들어 있는 명령 등을 이용하면 가능
  - ✓ 개발자가 개발 지침서를 바탕으로 각각 실시하다 보면 시간이 오래 걸리고 작업 순서를 빼먹는 실수가 있어서 제대로 동작하지 않는 파일이 만들어질 수 있기 때문에 이러한 문제를 해결하기 위해 빌드용(자동화) 도구라고 불리는 소프트웨어가 제공되는데 빌드 도구는 빌드 스크립트 라고 불리는 빌드 순서가 적힌 파일을 로드하면서 그 대상이 되는 파일들을 순서대로 잘 정리하여 결과물로 만들어내는 도구
  - ✓ 최근의 개발 방법으로는 빌드 시에 Checkstyle이나 FindBugs에 의한 코드 체크, JUnit에 의한 테스트 등도 함께 실시하여 빌드 대상을 빌드하기에 적합한 품질인지를 확인하는 것도 일반화되어 있는데 이러한 처리를 빌드 스크립트에 기술하여 빌드 도구로부터 실행하는 경우가 많음
  - ✓ 빌드 처리 자체를 Jenkins 등의 CI(지속적인 통합) 도구에서 자동 실시하고 정기적으로 적절한 빌드를 할 수 있는지에 대해 확인하는 것도 일반화되고 있는데 Jenkins로 빌드한 결과물을 시험 환경에 투입하는(배치-Deploy라고도 부름) 부분까지 자동으로 하고 있는 곳도 많음
  - ✓ IDE만으로도 빌드를 수행할 수 있지만 문서 생성 및 테스트 실행 등을 자동으로 수행하기 위해서는 IDE에 의존하지 않는 빌드 도구를 사용할 필요가 있음

Maven

- Java Build Tool
  - ✓ Ant: XML로 기술한 빌드 설정에 따라 컴파일 및 의존 관계의 해결을 실시하는 도구로 최근에는 사용빈도가 많지 않음
  - ✓ Maven: Apache 에서 개발되고 있는 소프트웨어 프로젝트 관리 툴로 외부 라이브러리 사용을 편리하게 해주는 빌드 도구
  - ✓ Gradle: Groovy로 작성하며 Ant 또는 Maven의 특징을 도입하여 만든 빌드 도구로 Android의 IDE 인 Android Studio 의 기본 빌드 도구 – build.gradle 파일을 이용
- Maven
  - ✓ POM(Project Object Model) 이라는 것에 기초를 두어 프로젝트의 빌드, 테스트, 도큐멘테이션, 성과물의 배치 등등의 라이프 사이클 전체를 관리
  - ✓ 프로젝트 빌드 툴인 ANT 와는 달리 Maven은 프로젝트 관리 툴로써 프로젝트에 관련한 여러 정보를 POM에 집약하여 POM의 정보에 기초를 두어 프로젝트 전체를 관리
  - ✓ 기존 방법
    - ❑ Library를 다운로드해서 Project에 복사하고 Build path를 설정
    - ❑ 추가로 Javadoc, Source 설정
  - ✓ Maven 사용
    - ❑ Maven을 이용한 Library 다운로드 및 Build Path 추가
    - ❑ 마우스를 이용한 Javadoc, Source Download 메뉴 클릭

pom.xml

- Maven 프로젝트의 설정 파일 – pom.xml
  - ✓ Modelversion: 4.0.0 으로 고정
  - ✓ groupId: 프로젝트를 생성하는 조직의 고유 아이디(식별자)로 일반적으로 도메인 이름 사용
  - ✓ artifactId: 프로젝트를 식별하는 유일한 아이디로 출력물의 결과물의 명칭
  - ✓ packaging: 프로젝트를 어떤 형태로 패키징 할 지 결정 – jar, war 등
  - ✓ version: 빌드되는 프로젝트의 버전
  - ✓ name: 프로젝트 이름
  - ✓ url: 프로젝트의 사이트가 있다면 사이트 URL을 등록
  - ✓ repositories: repositories 와 하위 엘리먼트 인 repository는 프로젝트와 의존 관계에 있는 라이브러리를 중앙 저장소가 아닌 곳에서 다운로드 받고자 할 때 설정
  - ✓ dependencies: dependencies 와 하위 엘리먼트 인 dependency는 프로젝트와 의존 관계에 있는 라이브러리를 관리 – http://mvnrepository.com/에서 검색

```
- <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>net.test</groupId>
    <artifactId>myproject</artifactId>
    <packaging>jar</packaging>
    <version>1.0-SNAPSHOT</version>
    <name>myproject</name>
    <url>http://maven.apache.org</url>
  - <dependencies>
    - <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>${junit.version}</version>
        <scope>test</scope>
      </dependency>
    </dependencies>
  </project>
```

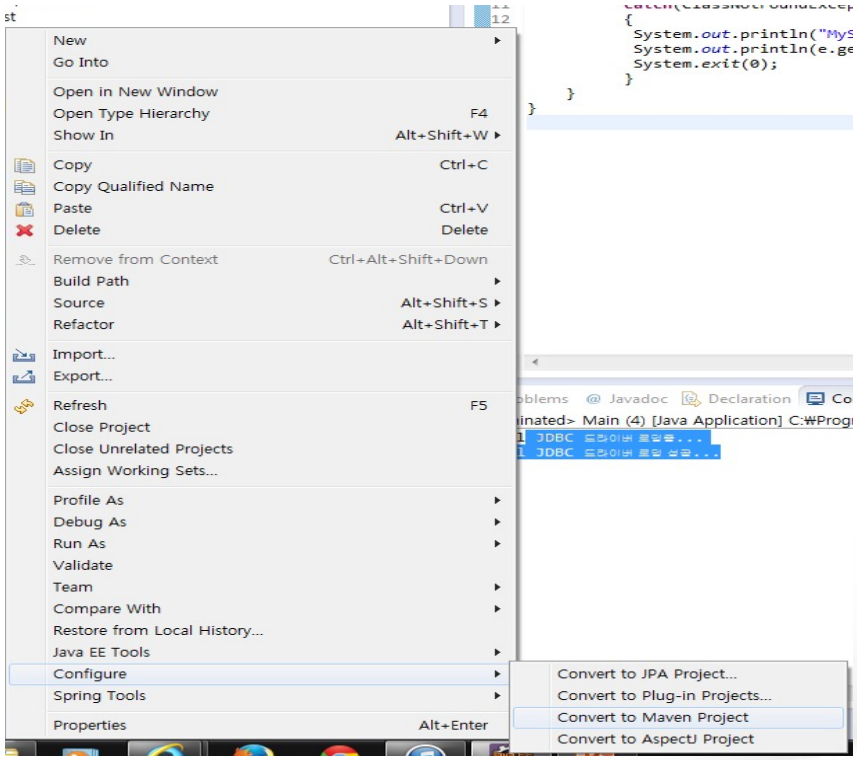
Application 변환

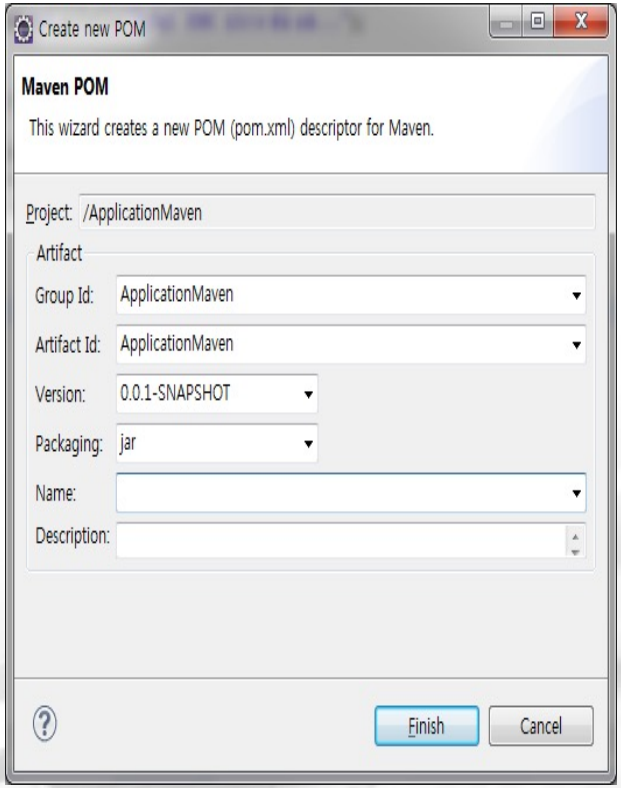
- Java Application 생성
- 프로젝트에 클래스를 추가하고 main 함수를 작성하고 실행

```
public class Main {
        public static void main(String[] args) {
                System.out.println("MySql JDBC 드라이버 로딩중...");
                try
                {
                 Class.forName("com.mysql.jdbc.Driver");
                 System.out.println("MySql JDBC 드라이버 로딩 성공...");

                }
                catch(ClassNotFoundException e)
                {
                 System.err.println("MySql JDBC 드라이버 로딩 실패...");
                 System.err.println(e.getMessage());
                 System.exit(0);
                }
        }
}
```

Application 변환

- 프로젝트를 선택하고 Maven 프로젝트로 변환
- 프로젝트를 선택하고 마우스 오른쪽을 클릭하고 [Configure] - [Convert to Maven Project]





Application 변환

- pom.xml 파일에 추가

```
<dependencies>
        <dependency>
                <groupId>mysql</groupId>
                <artifactId>mysql-connector-java</artifactId>
                <version>5.1.48</version>
        </dependency>
</dependencies>
```

- Main 클래스를 다시 실행

Maven 명령 및 플러그 인

- Maven 외부 라이브러리 사용 원리
  - ✓ pom.xml 파일의 dependencies 에 작성된 dependency 코드를 확인
  - ✓ 라이브러리를 로컬 컴퓨터의 .m2 디렉토리에서 조회
  - ✓ .m2 디렉토리에 라이브러리가 존재하지 않는다면 저장소에서 다운로드해서 .m2에 저장
  - ✓ 라이브러리를 build path에 추가
- 프로젝트를 선택하고 마우스 오른쪽을 클릭하고 [Run As]에서 선택
  - ✓ Maven Build: 결과물을 target 디렉토리 안에 작성 – 프로젝트를 실행하지 않음
  - ✓ Maven Install: 결과물을 로컬 저장소에 설치하고 로컬의 다른 프로젝트로부터 참조할 수 있도록 함
  - ✓ Maven Clean: target 디렉토리를 삭제하는데 빌드 할 때 이전에 빌드한 결과가 영향을 미칠 수 있는 가능성이 있기 때문에 빌드 전에 clean 하는 것이 좋음
- 플러그 인
  - ✓ <project><build><plugins> 아래에 빌드 시에 포함시킬 플러그 인을 열거
  - ✓ <project><reporting><plugins>에는 보고서 생성 시에 포함시킬 플러그 인을 열거

Maven 명령 및 플러그 인

- 플러그 인 오류가 나는 경우
  - ✓ dependencies에 아래 코드를 추가

```
<dependency>
    <groupId>org.apache.maven.plugins</groupId>
    <artifactId>maven-resources-plugin</artifactId>
    <version>2.4.3</version>
</dependency>
```

  - ✓ 프로젝트를 선택한 후 마우스 오른쪽 클릭 > Run As > Maven Install
  - ✓ 프로젝트를 선택한 후 F5(새로 고침)
  - ✓ 프로젝트를 선택한 후 마우스 오른쪽 클릭 > Maven > Update Project

중앙 레포지트리에 없는 경우

- 오라클은 중앙 repository에 없어서 dependency만 추가해서는 안됨
- 이런 경우에는 다운로드 받을 repository를 추가한 후 dependency를 추가
- 오라클 다운로드
  - ✓ repository 추가

```
<repositories>
        <repository>
                <id>oracle</id>
                <url>http://maven.jahia.org/maven2</url>
        </repository>
</repositories>
```

  - ✓ dependency 추가

```
<dependencies>
        <dependency>
                <groupId>com.oracle</groupId>
                <artifactId>ojdbc7</artifactId>
                <version>12.1.0.2</version>
        </dependency>
</dependencies>
```

CheckStyle

CheckStyle

- CheckStyle
  - ✓ 초보자 시절에는 코드를 작성해서 동작하도록 하는 것만으로도 능력의 한계치여서 코드의 구성이나 유지보수는 그다지 신경 쓸 여력이 없었던 경우가 많은데 구성을 신경 쓰지 않고 여기저기 마구 작성한 코드는 수정 및 기능 추가를 하려고 할 때 그 코드의 가독성이 떨어져 생산성과 품질을 저하되기 때문에 그런 일이 발생하지 않도록 평소에 읽기 쉬운 코드를 작성하는 것이 중요
  - ✓ 읽기 쉬운 코드를 작성하는 노력을 해도 사람의 눈으로 하는 체크는 소홀해지거나 결여가 발생할 것이므로 도구로 체크를 실시할 수 있도록 하여 품질을 높게 유지할 수 있도록 하면 매우 편리
  - ✓ 자바의 소스 코드의 포맷(구성)을 체크하여 규칙에 따르지 않은 기술에 대해서는 경고해 주는 도구가 Checkstyle

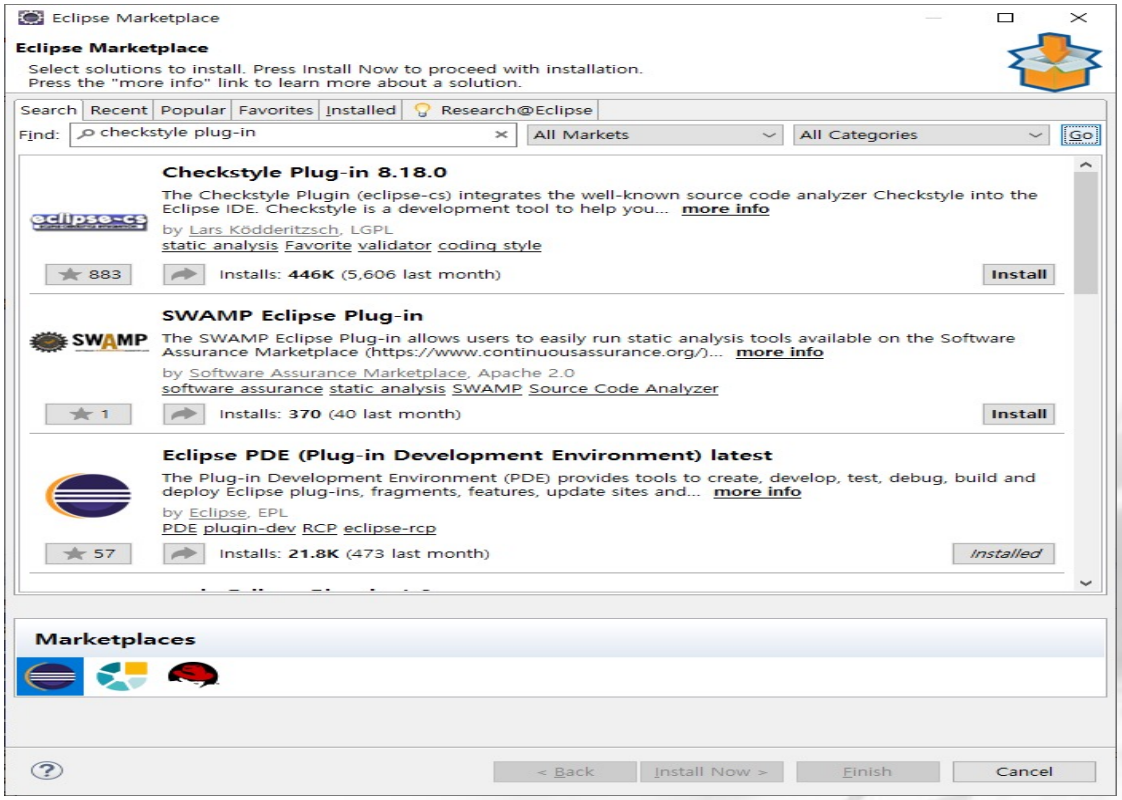
CheckStyle

❖ 아래 코드는 에러 없이 잘 수행됨

```
public class CheckStyleMain {

        public static void main(String[] args) {
                Object emp = new Object();
                System.out.println(emp);
        }
}
```

CheckStyle

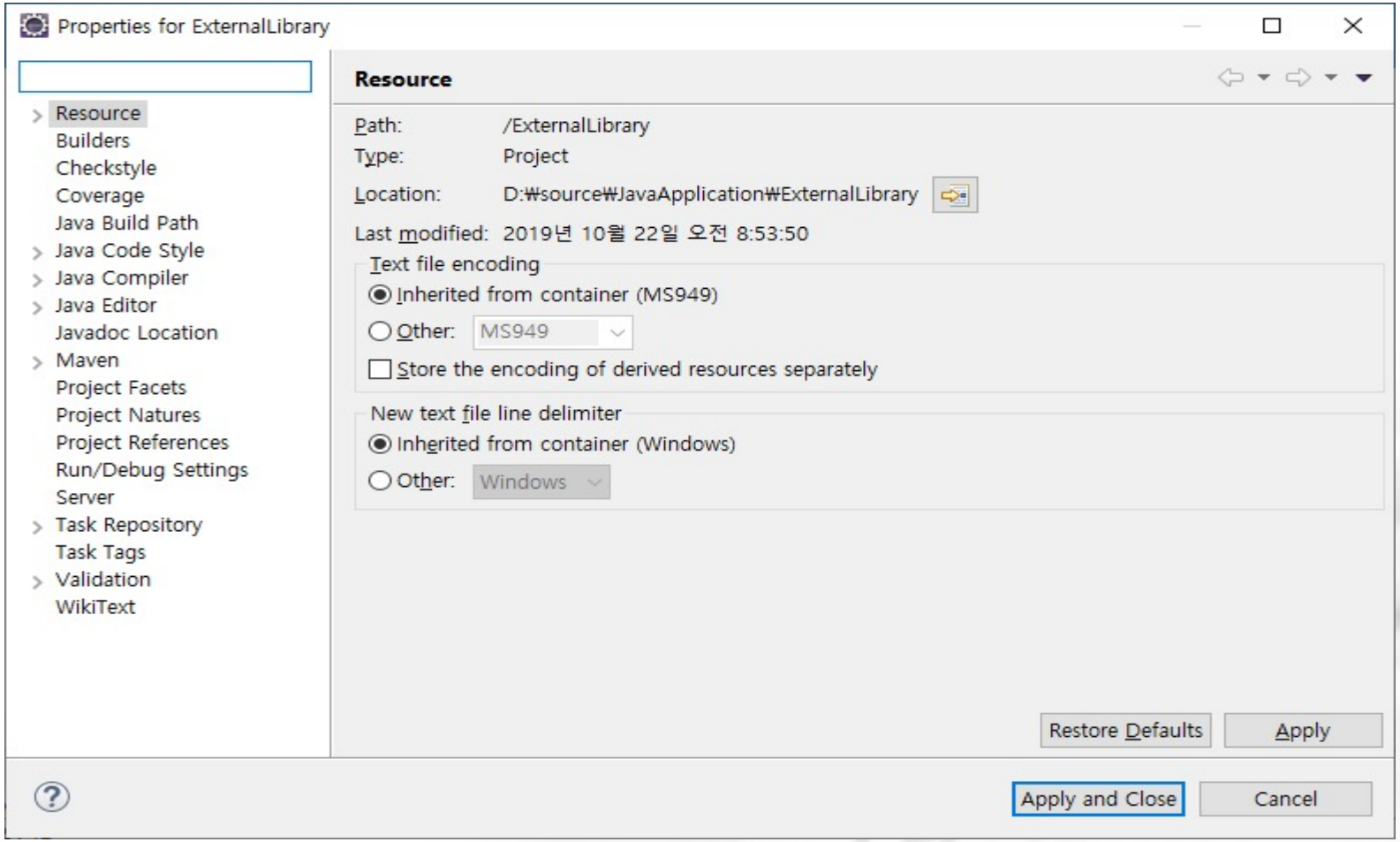
- 프로그램의 구성을 확인하고자 할 때 사용하는 라이브러리로 자바의 구성에 맞지 않게 작성된 코드에 경고를 표시해 줌
- Eclipse에서의 설치
  - ✓ Eclipse MarketPlace ([Help] – [Eclipse MarketPlace]에서 CheckStyle Plug-in 을 검색해서 설치



CheckStyle

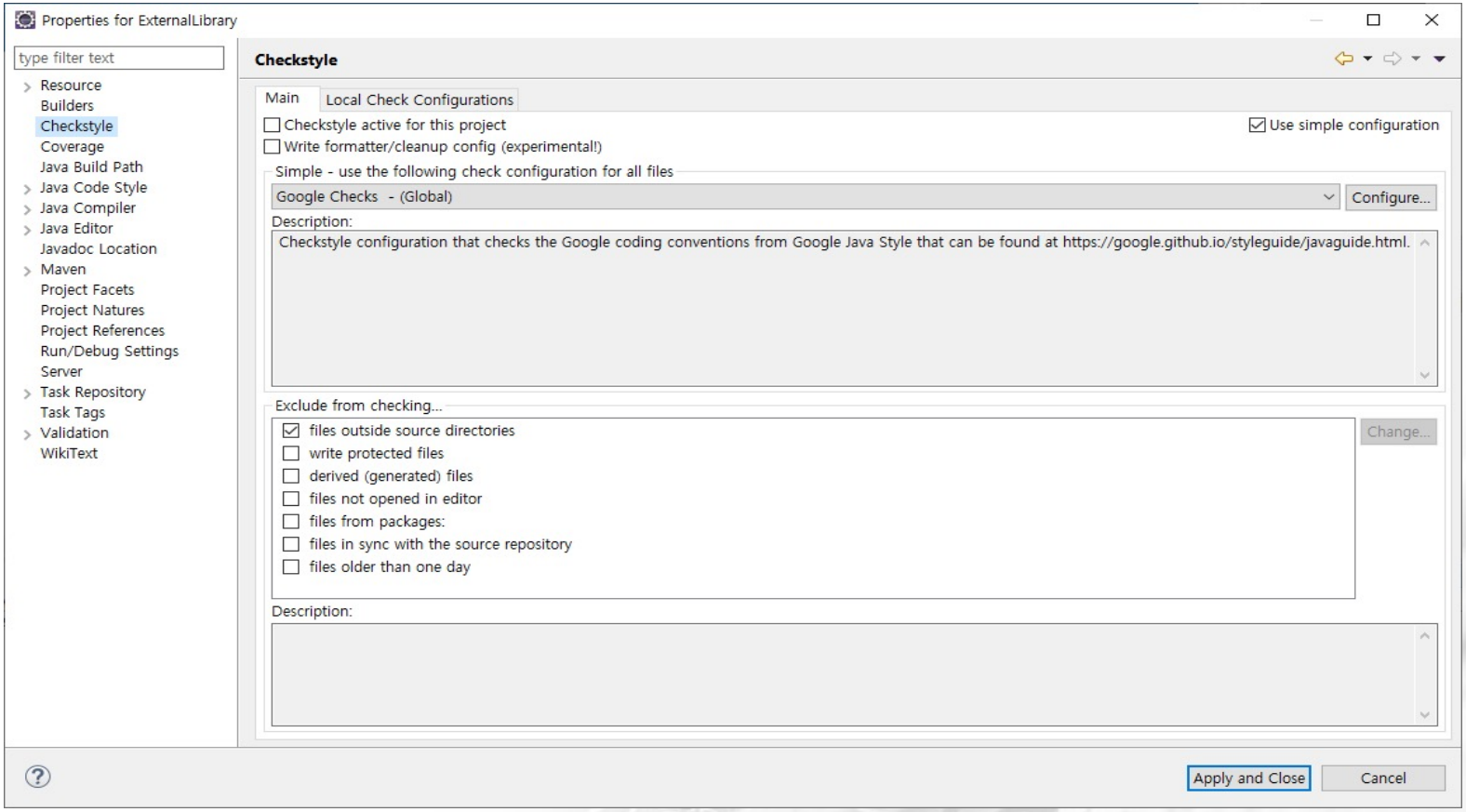
❖ Eclipse에서의 실행

✓ 프로젝트를 선택하고 마우스 오른쪽을 클릭해서 [Properties] 클릭



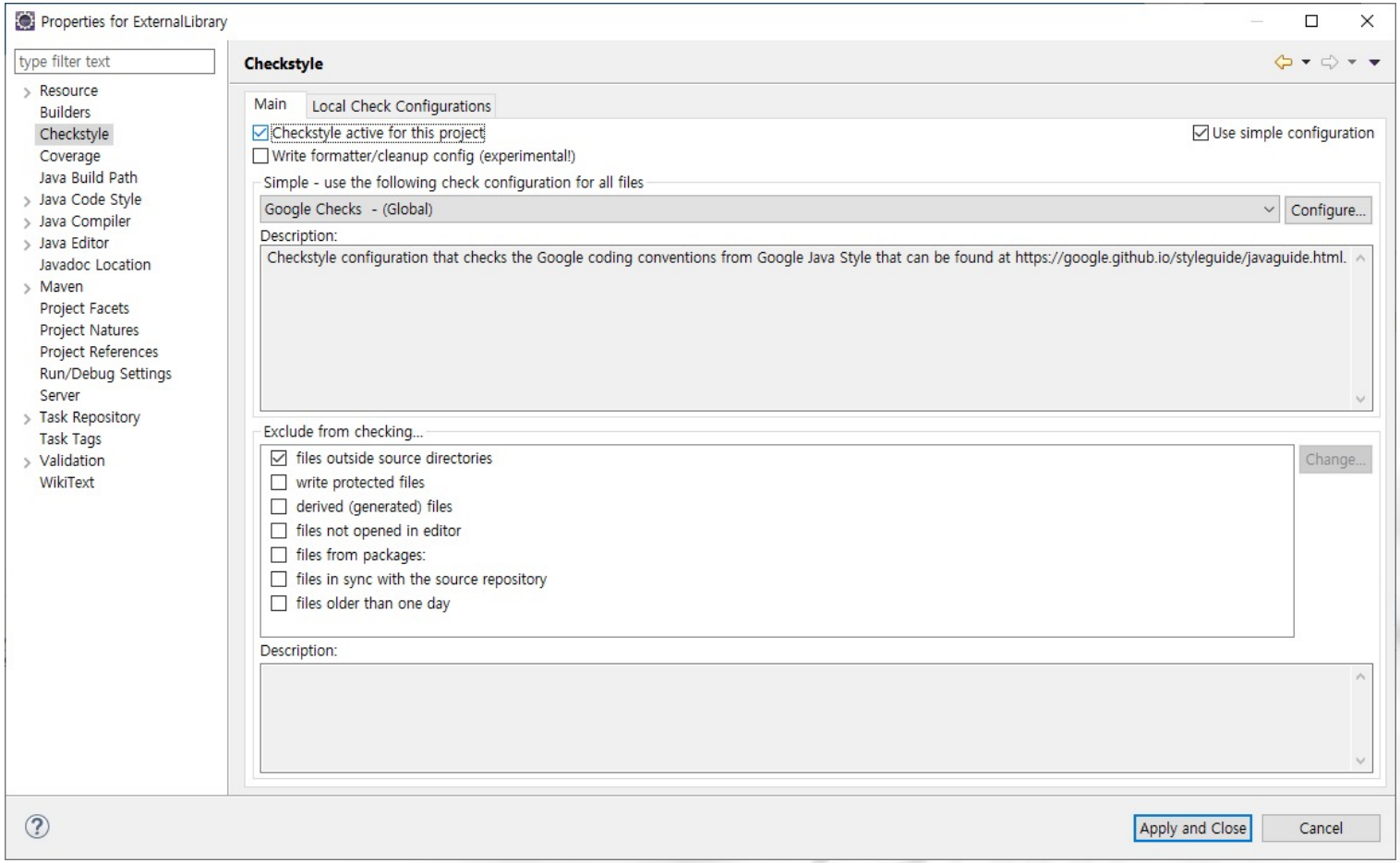
CheckStyle

- Eclipse에서의 실행
  - ✓ 왼쪽 창에서 Checkstyle 선택



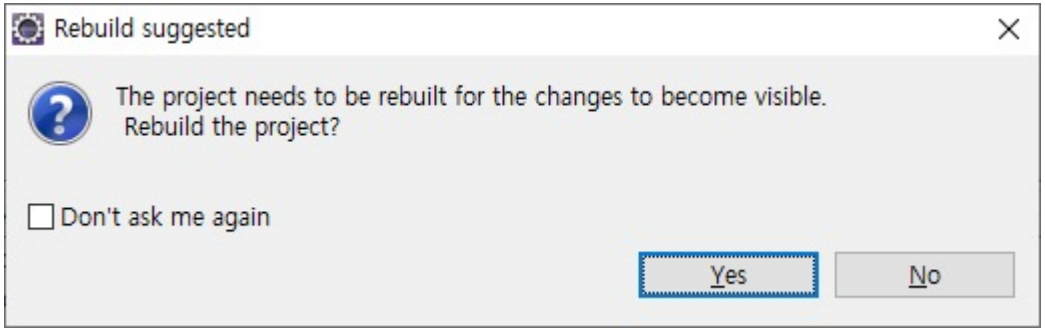
CheckStyle

- Eclipse에서의 실행
  - ✓ 오른쪽 창에서 'Checkstyle active for this project' 체크



CheckStyle

- Eclipse에서의 실행
  - ✓ Rebuild 할 것인지 물어보면 Yes



CheckStyle

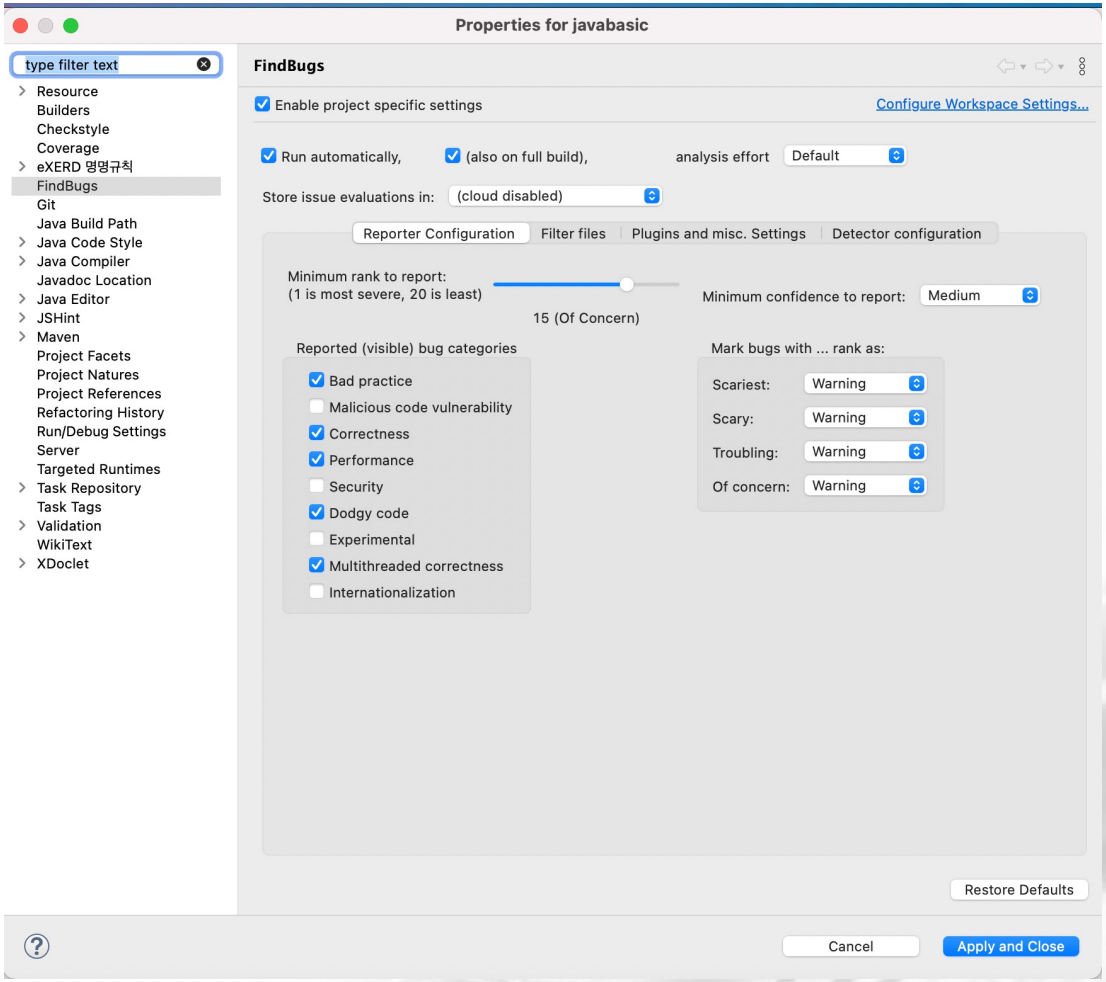
```java
/**
 * CheckStyle 적용을 위한 클래스
 */
public class CheckStyleMain {

    public static void main(String[] args) {
        Object emp = new Object();
        System.out.println(emp);
    }
}
```

FindBugs

FindBugs

- FindBugs
  - ✓ 프로그램의 버그의 종류는 천차만별인데 예를 들어, '변수에 할당된 값이 사용되지 않는다', '리소스가 닫히지 않았다' 등의 문제는 흔히 볼 수 있는 종류의 것
  - ✓ 일정한 패턴의 단순한 버그를 검출하고, 프로그램의 품질을 끌어 올리는 데 도움이 되는 도구가 FindBugs
  - ✓ 정적으로 체크를 실시하기 때문에 프로그램을 동작시킬 필요없이 간단하게 체크할 수 있다는 것이 특징
  - ✓ 테스트하지 않고도 버그를 줄일 수 있기 때문에 자바로 프로그램을 만들 때 필수 도구
  - ✓ 설치
    - ❑ Eclipse Marketplace에서 'FindBugs Eclipse Plugin'을 설치
    - ❑ 설치가 안되면 [Help] – [Install New Software] 를 실행한 후 [Work With] 에 http://findbugs.cs.umd.edu/eclipse/를 입력해서 직접 설치
  - ✓ 사용
    - ❑ 이클립스의 패키지 익스플로러에서 이클립스 프로젝트를 오른쪽 클릭 후 'Properties'를 선택
    - ❑ 열린 윈도우의 왼쪽 리스트에서 'FindBugs'를 선택
    - ❑ 오른쪽 화면에서 'Enable project specific settings'를 체크
    - ❑ 'Run automatically'와 '(also on full build)'에 체크
    - ❑ 'OK' 버튼을 클릭하면 빌드와 함께 체크가 수행

FindBugs


Properties for javabasic
type filter text
Resource
Builders
Checkstyle
Coverage
eXERD 명명규칙
FindBugs
Git
Java Build Path
Java Code Style
Java Compiler
Javadoc Location
Java Editor
JSHint
Maven
Project Facets
Project Natures
Project References
Refactoring History
Run/Debug Settings
Server
Targeted Runtimes
Task Repository
Task Tags
Validation
WikiText
XDoclet
FindBugs
Enable project specific settings
Configure Workspace Settings...
Run automatically,
(also on full build),
analysis effort
Default
Store issue evaluations in:
(cloud disabled)
Reporter Configuration
Filter files
Plugins and misc. Settings
Detector configuration
Minimum rank to report:
(1 is most severe, 20 is least)
15 (Of Concern)
Minimum confidence to report:
Medium
Reported (visible) bug categories
Bad practice
Malicious code vulnerability
Correctness
Performance
Security
Dodgy code
Experimental
Multithreaded correctness
Internationalization
Mark bugs with ... rank as:
Scariest:
Warning
Scary:
Warning
Troubling:
Warning
Of concern:
Warning
Restore Defaults
Cancel
Apply and Close

FindBugs

Description	Resource	Path	Location	Type
FindBugs Problem (Of concern) (13 items)				
Dead store to color in javagui.JColorChooserTest.actionPerformed(ActionEvent) [...	JColorChoose...	/javabasic/src/javagui	line 28	FindBugs Pro...
Dead store to frame in drawing.Paint2.main(String[]) [Of Concern(15), High confi...	Paint2.java	/javabasic/src/drawing	line 67	FindBugs Pro...
Dead store to n in oop.Calculate.inc1(int) [Of Concern(15), High confidence]	Calculate.java	/javabasic/src/oop	line 9	FindBugs Pro...
Dead store to obj1 in oop.StaticInit.main(String[]) [Of Concern(15), High confidence]	StaticInit.java	/javabasic/src/oop	line 7	FindBugs Pro...
Dead store to obj1 in oop.StaticVariable.main(String[]) [Of Concern(15), High con...	StaticVariable....	/javabasic/src/oop	line 5	FindBugs Pro...
Dead store to obj2 in oop.StaticInit.main(String[]) [Of Concern(15), High confidence]	StaticInit.java	/javabasic/src/oop	line 8	FindBugs Pro...
Dead store to obj2 in oop.StaticVariable.main(String[]) [Of Concern(15), High con...	StaticVariable....	/javabasic/src/oop	line 6	FindBugs Pro...
Dead store to Obj in javagui.FrameMain.main(String[]) [Of Concern(15), High con...	FrameMain.java	/javabasic/src/javagui	line 5	FindBugs Pro...
Dereference of the result of readLine() without nullcheck in io.BufferReaderMain....	BufferReader...	/javabasic/src/io	line 20	FindBugs Pro...
Dereference of the result of readLine() without nullcheck in jdkapi.Cal.main(String...	Cal.java	/javabasic/src/jdkapi	line 18	FindBugs Pro...
Dereference of the result of readLine() without nullcheck in network.ChatSendThr...	ChatSendThre...	/javabasic/src/network	line 19	FindBugs Pro...
Dereference of the result of readLine() without nullcheck in network.MulticastSen...	MulticastSend...	/javabasic/src/network	line 29	FindBugs Pro...
Dereference of the result of readLine() without nullcheck in network.UDPClient.m...	UDPClient.java	/javabasic/src/network	line 14	FindBugs Pro...
FindBugs Problem (Scary) (3 items)				
Comparison of String objects using == or != in javalangpackage.StringCompare.m...	StringCompar...	/javabasic/src/javala...	line 12	FindBugs Pro...
Comparison of String objects using == or != in javalangpackage.StringIntern.main...	StringIntern.java	/javabasic/src/javala...	line 10	FindBugs Pro...
Format-string method java.io.PrintStream.printf(String, Object[]) called with form...	ConnectThrea...	/javabasic/src/network	line 44	FindBugs Pro...
FindBugs Problem (Troubling) (15 items)				
javalangpackage.ObjectClass.equals(Object) does not check for null argument [Tr...	ObjectClass.java	/javabasic/src/javala...	line 17	FindBugs Pro...
javalangpackage.ObjectClass defines equals and uses Object.hashCode() [Troubli...	ObjectClass.java	/javabasic/src/javala...	line 17	FindBugs Pro...
javalangpackage.SampleAnnotationTest.main(String[]) might ignore java.lang.Exc...	SampleAnnota...	/javabasic/src/javala...	line 28	FindBugs Pro...
new javagui.GUIChatServer(int) invokes javagui.ChatHandler.start() [Troubling(14...	GUIChatServe...	/javabasic/src/javagui	line 36	FindBugs Pro...

JUnit

JUnit

- JUnit
  - ✓ 단위 테스트를 위한 프레임워크
  - ✓ JUnit은 테스트 주도 개발(TDD, Test-Driven Development, 테스트를 먼저 한 뒤 코드를 작성하는 방법)에서 많이 사용하는 프레임워크이며 자동화된 테스트가 가능
  - ✓ 단위 테스트 : 전체 프로그램을 구성하고 있는 기본 단위(unit) 프로그램이 정상적으로 동작하는지 테스트하는 것

JUnit

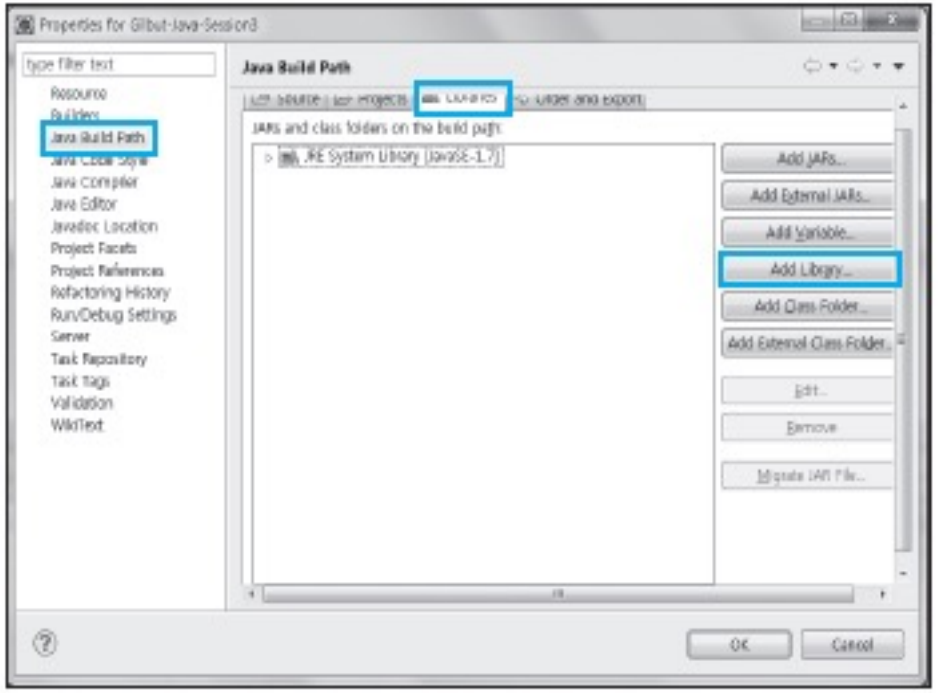
❖ 이클립스에서 JUnit 라이브러리 추가하기

✓ Java project를 선택하고 마우스 오른쪽 버튼을 클릭

✓ 메뉴의 가장 아래에 위치한 Properties를 클릭

JUnit

❖ 이클립스에서 JUnit 라이브러리 추가하기

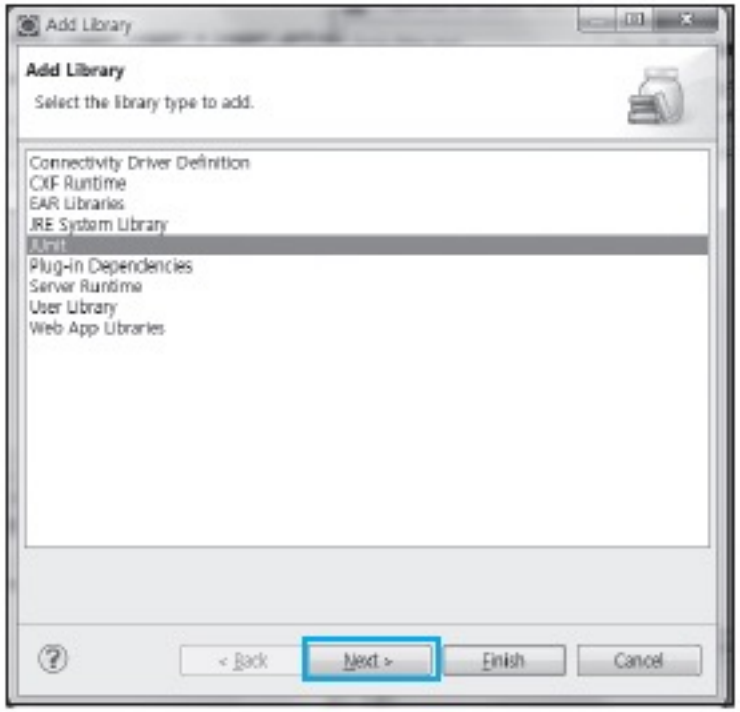


✓ Properties 윈도우가 나타나면 왼쪽 창의 속성 중에서 Java Build Path를 클릭

✓ 오른쪽 창에서 Library 탭을 선택하면 추가된 Library를 확인 가능, <Add Library> 버튼 클릭

JUnit

❖ 이클립스에서 JUnit 라이브러리 추가하기

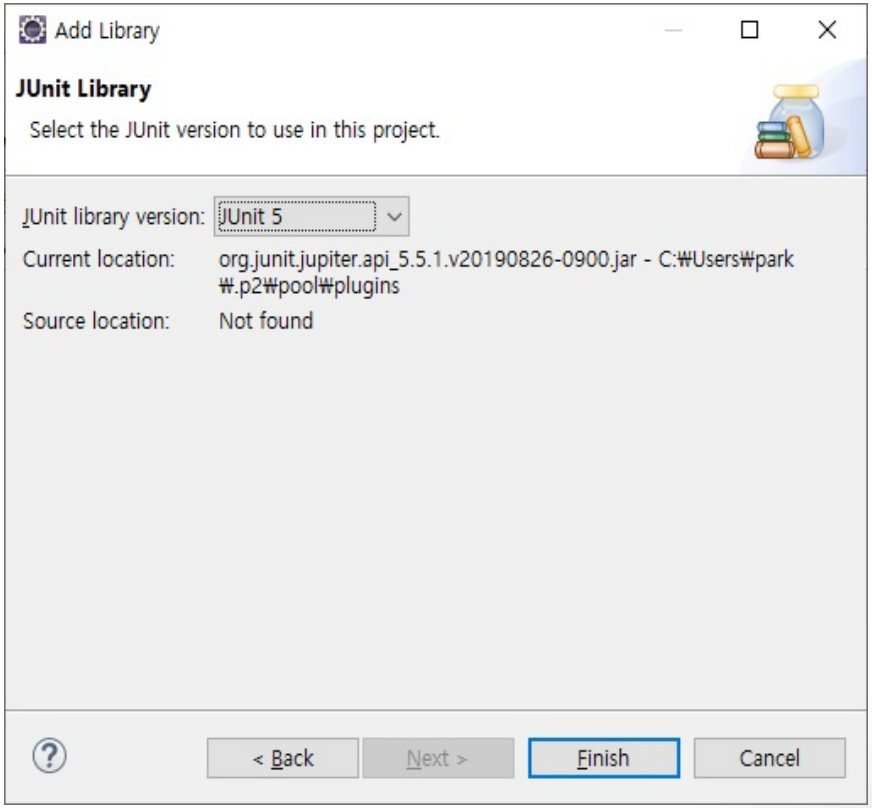


✓ Library를 선택할 수 있는 화면이 생성

✓ JUnit을 선택하고, <Next> 버튼을 클릭

JUnit

❖ 이클립스에서 JUnit 라이브러리 추가하기



✓ JUnit 버전을 설정할 수 있는 화면이 나타나면 JUnit 5를 선택하고 <Finish> 버튼을 클릭

JUnit

- JUnit을 이용해서 테스트를 진행하기 방법
  - ✓ TestCase 클래스로부터 상속받는 클래스를 생성하는 방법 – 클래스 안에 생성된 모든 메서드를 테스트
  - ✓ Object로 부터 상속받는 클래스를 작성한 후 테스트를 수행할 메서드 위에 @Test라는 annotation을 기재하는 방법
  - ✓ TestCase로 부터 상속받는 클래스나 @Test 메서드 안에 assertThat 메서드 또는 AssertEquals 메서드의 첫번째 인수로 실제 값을 두번째 인수에 기대하는 값을 기술하면 양쪽의 값이 다른 경우에는 AssertionError 예외가 발생

JUnit

- 테스트를 위한 Source 클래스를 작성하고 메서드 작성

```
public class Source {
        public int add(int a, int b){
                return a+b;
        }
}
```

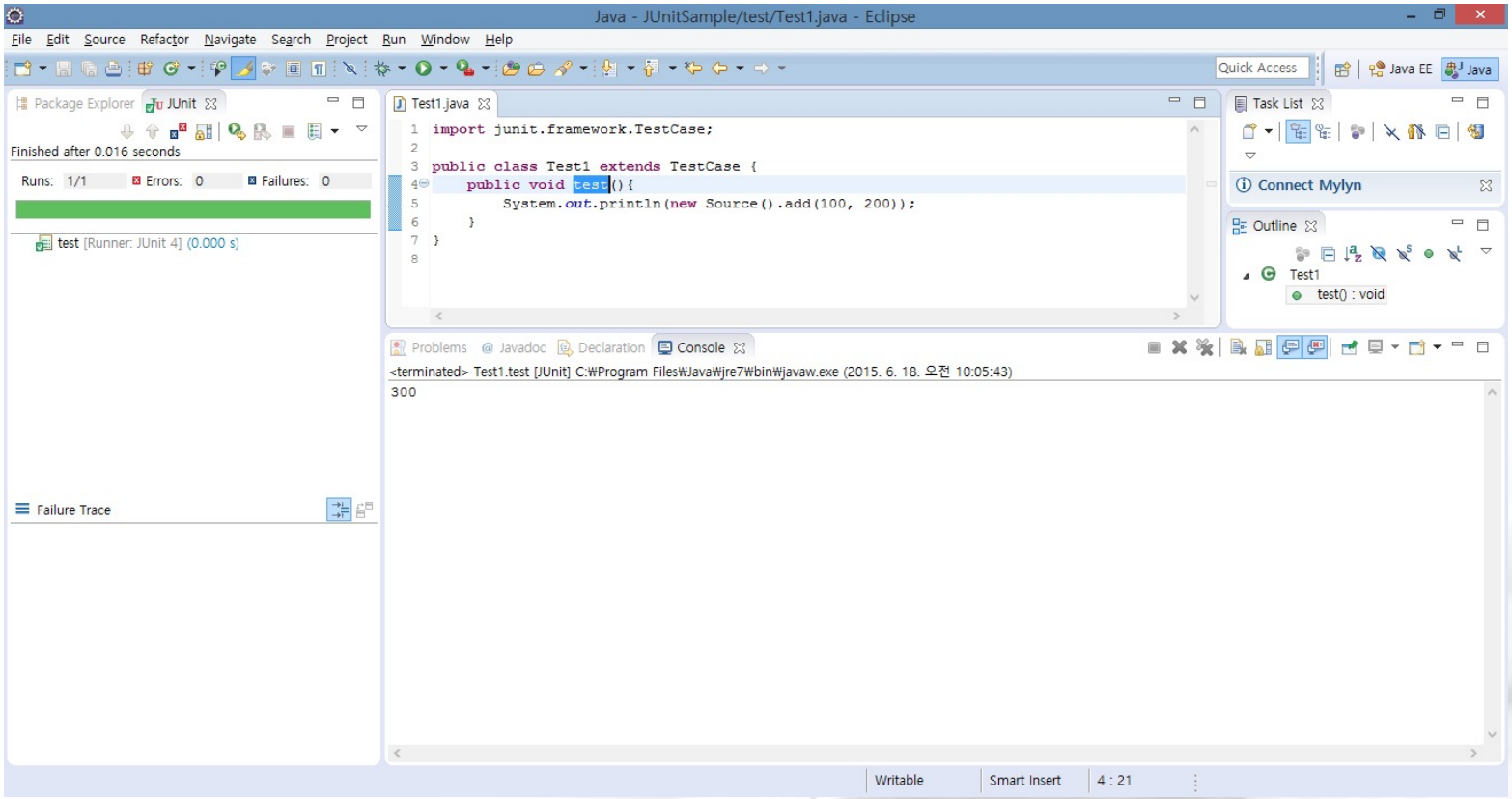
JUnit

- 프로젝트 내에 Source Folder를 생성하고 TestCase클래스로부터 상속받는 클래스를 생성하고 테스트를 위한 메서드를 작성

```
import junit.framework.TestCase;

public class Test1 extends TestCase {
	public void test(){
		System.out.println(new Source().add(100, 200));
	}
}
```

JUnit

❖ Test 메서드를 선택하고 Run As – JUnit Test를 선택



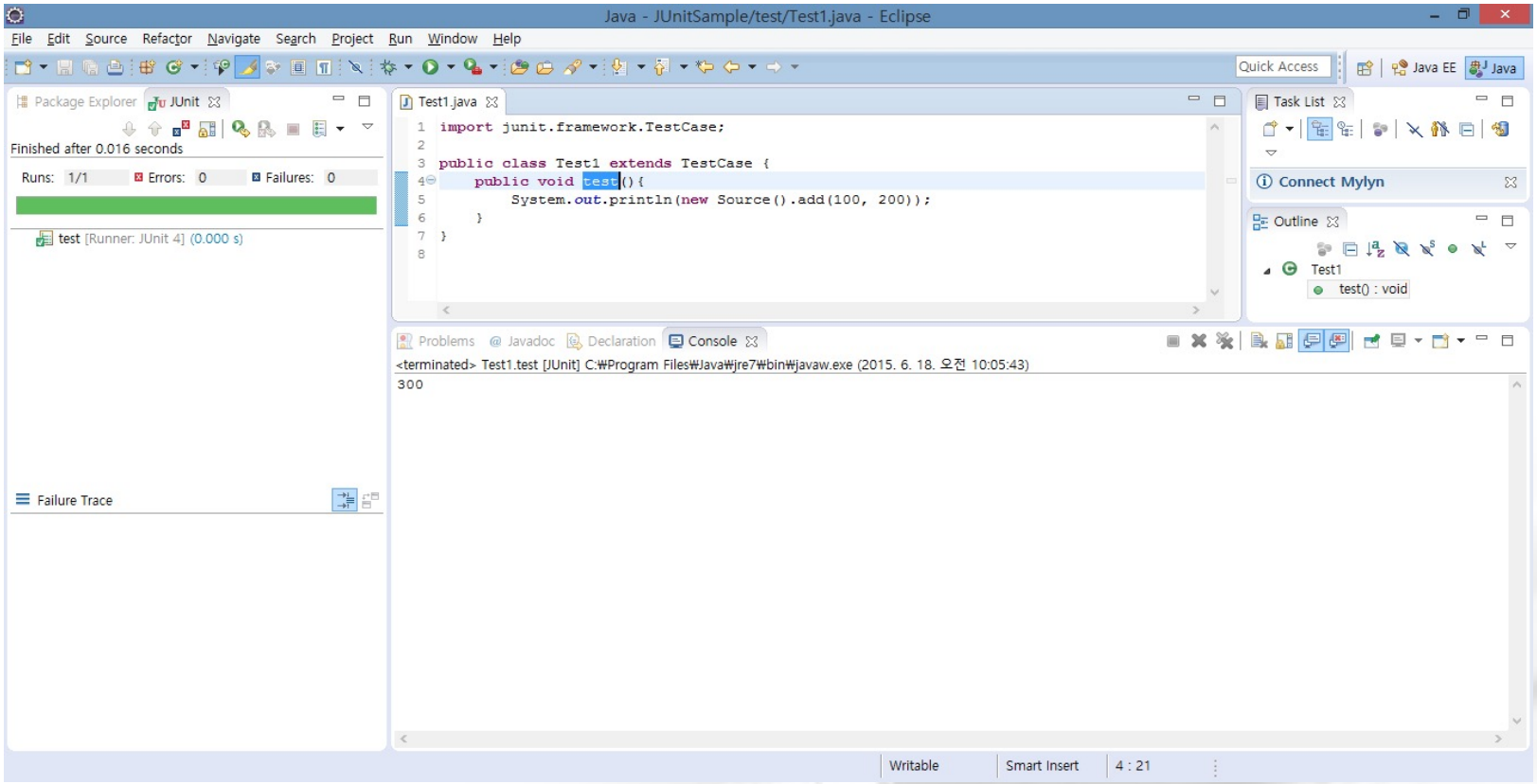
JUnit

❖ Object 클래스로부터 상속받는 클래스를 생성하고 테스트를 위한 메서드를 작성

```
import org.junit.Test;

public class Test2 {
	@Test
	public void test(){
		System.out.println(new Source().add(100, 200));
	}
}
```

JUnit

❖ Test 메서드를 선택하고 Run As – JUnit Test를 선택

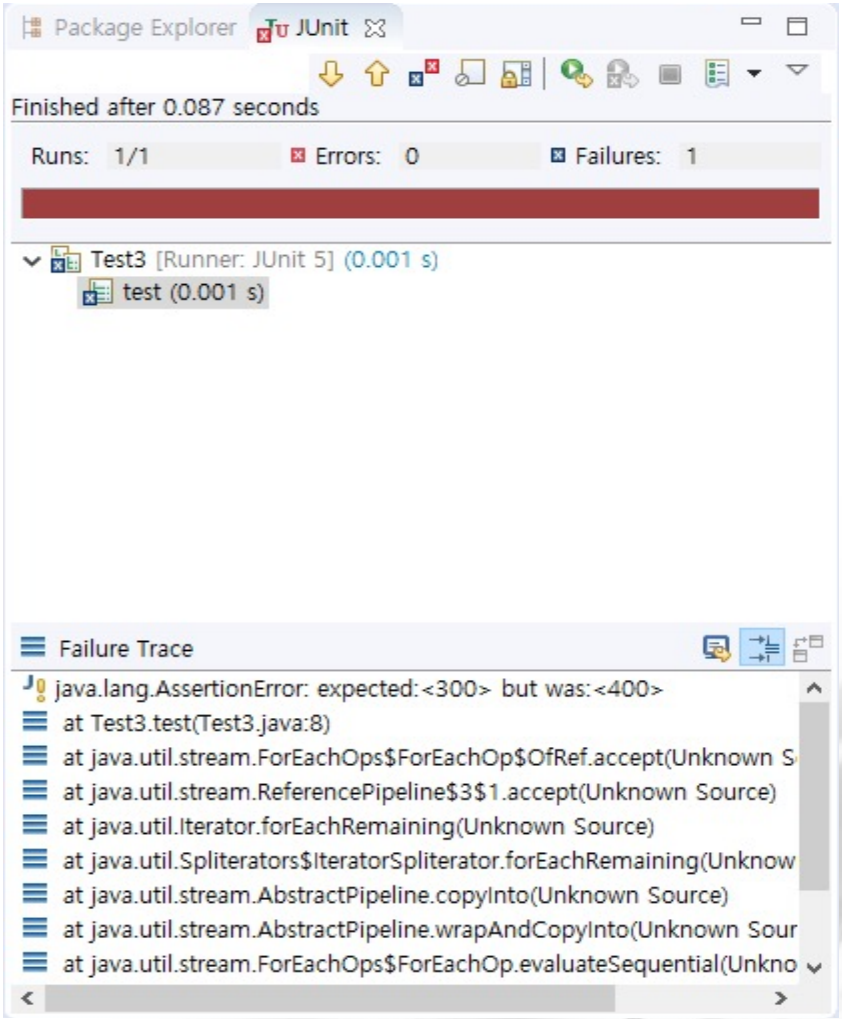


JUnit

❖ Object 클래스로부터 상속받는 클래스를 생성하고 테스트를 위한 메서드를 작성

```
import org.junit.Assert;
import junit.framework.TestCase;

public class Test3 extends TestCase {
	public void test(){
		Integer result = new Source().add(100, 200);
		Integer i = 400;
		Assert.assertEquals(result, i);
	}
}
```

JUnit


Package Explorer
JUnit
Finished after 0.087 seconds
Runs: 1/1
Errors: 0
Failures: 1
Test3 [Runner: JUnit 5] (0.001 s)
test (0.001 s)
Failure Trace
java.lang.AssertionError: expected:<300> but was:<400>
at Test3.test(Test3.java:8)
at java.util.stream.ForEachOps$ForEachOp$OfRef.accept(Unknown S
at java.util.stream.ReferencePipeline$3$1.accept(Unknown Source)
at java.util.Iterator.forEachRemaining(Unknown Source)
at java.util.Spliterators$IteratorSpliterator.forEachRemaining(Unknow
at java.util.stream.AbstractPipeline.copyInto(Unknown Source)
at java.util.stream.AbstractPipeline.wrapAndCopyInto(Unknown Sour
at java.util.stream.ForEachOps$ForEachOp.evaluateSequential(Unkno

Apache Commons

Apache Commons

- Apache Commons
  - ✓ Apache Commons는 자바 프로그램을 만드는 데 공통으로 사용할 수 있는 재사용 가능한 컴포넌트를 정리한 라이브러리 집합
  - ✓ 오픈소스의 소프트웨어 개발 프로젝트를 지원하는 비영리 단체인 Apache 소프트웨어 재단이 제공
  - ✓ 컬렉션 및 문자열의 처리, 파일의 입출력 등 에서 빈번하게 발생하는 정형적인 처리를 API로 정리
  - ✓ 이러한 편리한 라이브러리를 얼마나 잘 알고 활용할 수 있는지는 프로그램의 생산성, 나아가 품질과 밀접한 관계가 있음

Commons Lang

- Commons Lang
  - ✓ Commons Lang에는 자바의 java.lang 패키지의 내용을 확장한 유틸리티를 포함
  - ✓ http://commons.apache.org/proper/commons-lang/ 에서 다운로드 후 다운로드한 JAR 파일을 클래스 패스에 등록함으로써 라이브러리를 사용할 수 있음
  - ✓ Maven을 사용하는 경우는 pom.xml의 <dependencies> 태그에 다음과 같이 쓰면 라이브러리를 사용할 수 있음

```
<!-- https://mvnrepository.com/artifact/org.apache.commons/commons-lang3 -->
<dependency>
        <groupId>org.apache.commons</groupId>
        <artifactId>commons-lang3</artifactId>
        <version>3.11</version>
</dependency>
```

Commons Lang

- Commons Lang – CommonsLangMain.java

```
import org.apache.commons.lang3.StringUtils;
import org.apache.commons.lang3.builder.EqualsBuilder;
import org.apache.commons.lang3.builder.HashCodeBuilder;
import java.io.Serializable;
import org.apache.commons.lang3.SerializationUtils;

class DTO implements Serializable{
	private static final long serialVersionUID = 1L;

	private int num;
	private String name;
	public int getNum() {
		return num;
	}
	public void setNum(int num) {
		this.num = num;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
```

Commons Lang

- Commons Lang

```
//hashCode 메서드와 equals 메서드 재정의
/*
@Override
public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + ((name == null) ? 0 : name.hashCode());
        result = prime * result + num;
        return result;
}
```

Commons Lang

- Commons Lang

```
@Override
public boolean equals(Object obj) {
        if (this == obj)
                return true;
        if (obj == null)
                return false;
        if (getClass() != obj.getClass())
                return false;
        DTO other = (DTO) obj;
        if (name == null) {
                if (other.name != null)
                        return false;
        } else if (!name.equals(other.name))
                return false;
        if (num != other.num)
                return false;
        return true;
}
 */
```

Commons Lang

- Commons Lang

```
        @Override
        public int hashCode() {
                return HashCodeBuilder.reflectionHashCode(this);
        }
        @Override
        public boolean equals(Object obj) {
                EqualsBuilder builder = new EqualsBuilder();
                DTO otherDTO  = (DTO)obj;

                builder.append(getNum(), otherDTO.getNum());
                builder.append(getName(), otherDTO.getName());
                return builder.isEquals();
        }

        @Override
        public String toString() {
                return "DTO [num=" + num + ", name=" + name + "]";
        }

}
```

Commons Lang

- Commons Lang

```
public class CommonsLangMain {

        public static void main(String[] args) {
                String text = null;

                //빈 문자열 확인
                if (text == null || text.length() == 0) {
                        System.out.println("text는 비었다");
                }
                //isEmpty 메서드의 부정형인 isNotEmpty 메서드
                //공백만의 문자열도 빈 것으로 판정하는 isBlank 메서드
                if (StringUtils.isEmpty(text)) {
                        System.out.println("text는 비었다");
                }
```

Commons Lang

- Commons Lang

```
                DTO dto1 = new DTO();
                dto1.setNum(1);
                dto1.setName("아담");

                DTO dto2 = new DTO();
                dto2.setNum(2);
                dto2.setName("류시아");

                System.out.println(dto1.hashCode());
                System.out.println(dto1.equals(dto2));

                //깊은 복사
                DTO dto3 = SerializationUtils.clone(dto1);
                System.out.println(dto3);

        }
}
```

로그 출력

로그 출력

❖ 로그 와 레벨

- ✓ 애플리케이션의 규모가 클수록 애플리케이션의 동작이 복잡해져 문제가 발생하기 쉬움
- ✓ 문제가 발생했을 때 조사하기 쉽도록 애플리케이션에는 동작 상황을 로그 파일에 출력하는 처리를 추가
- ✓ 동작 상황을 로그 파일에 출력한다 라고는 했으나 출력하는 내용에 따라 레벨(중요도)이 다른데 메서드가 호출될 때 인수의 값을 확인하고 싶은 경우와 오류가 발생한 것을 기록하는 경우는 오류가 발생한 것을 기록하는 쪽이 훨씬 더 레벨이 높음
- ✓ 레벨을 대부분 다음의 4단계로 나눔

레벨	이용 상황	예
ERROR	오류가 발생하고 작업을 계속할 수 없는 경우에 출력한다.	필수 데이터의 취득 불가, 외부 시스템과의 통신 오류
WARN	오류가 발생했지만 계속 처리 할 수 있는 경우에 출력한다.	설정 파일이 없으면 기본값 사용
INFO	정상 동작 중에 상태 전이/처리가 실시된 경우에 출력한다.	외부 시스템과의 통신 시작/ 종료, 일련의 업무 처리 종료
DEBUG	동작 확인을 위한 상세한 값을 출력한다.	데이터 업데이트 시의 값

- ❑ ERROR보다 더 치명적인 오류를 나타내는 FATAL
- ❑ DEBUG보다 더 자세한 정보를 출력하는 TRACE

로그 출력

- 로깅 설정
  - ✓ 애플리케이션 안에서는 다양한 레벨의 로그를 출력하는 코드를 작성하게 되는데 애플리케이션의 실행 상황에 따라 출력되는 로그 레벨을 제한하는 경우가 많음
  - ✓ 개발 시의 디버깅을 목적으로 한다면 더 자세한 정보를 빈번하게 출력할 필요가 있으며 반대로 운용 시에 대량의 로그를 발생시키면 중요한 정보가 묻혀 문제를 찾아내기가 어렵게 되며 또한 성능의 악화와 리소스의 소비 증대로 인해 애플리케이션의 동작 자체에 영향을 끼칠 우려도 있음
  - ✓ 용도를 근거로 어느 레벨까지 로그를 출력할 것인지를 설정에 따라 전환할 수 있도록 하는 것이 로깅 라이브러리
  - ✓ 로깅 라이브러리 중에서도 최근 자주 사용되고 있는 것이 SLF4J와 Logback의 조합
  - ✓ SLF4J는 다양한 로깅 라이브러리를 포장하여 통합적으로 취급할 수 있도록 한 라이브러리
  - ✓ Logback은 로깅 라이브러리 중에 하나이며 지금까지 일반적이었던 Log4j의 후속 라이브러리에 해당
  - ✓ Log4j보다 높은 성능으로 리소스를 절약할 수 있는 것이 특징
  - ✓ SLF4J는 다음 사이트에서 다운로드

    http://www.slf4j.org/
  - ✓ Logback은 다음 사이트에서 다운로드

    http://logback.qos.ch/

로그 출력

- 로깅 설정
  - ✓ Maven에서 SLF4j와 Logback을 사용할 경우 pom.xml의 설정

```
<dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>slf4j-api</artifactId>
        <version>1.7.21</version>
</dependency>
<dependency>
        <groupId>ch.qos.logback</groupId>
        <artifactId>logback-classic</artifactId>
        <version>1.1.7</version>
</dependency>
```

로그 출력

- 기본 설정
  - ✓ SLF4J + Logback을 사용하여 로그를 출력하려면 다음 두 가지 처리를 수행
    - ❑ 설정 파일(logback.xml)의 작성
    - ❑ 로그 출력 코드의 기술
  - ✓ 설정 파일(logback.xml)의 작성
    - ❑ 먼저 로그 출력의 동작을 결정하는 설정 파일을 작성
    - ❑ 클래스 경로를 통하는 디렉터리 밑에 파일 이름이 'logback.xml'인 XML 파일을 만든 후 콘솔에 로그를 출력하기 위한 설정

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
 <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
 <encoder>
 <pattern>%date [%thread] %-5level %logger{35} - %message%n</pattern>
 </encoder>
 </appender>
 <root level="INFO">
 <appender-ref ref="STDOUT" />
 </root>
</configuration>
```

로그 출력

- ❖ 기본 설정
  - ✓ 설정 파일(logback.xml)의 작성
    - ❑ 모든 설정은 configuration 태그 안에 기술
    - ❑ appender 태그에서 로그를 출력할 위치를 정의
    - ❑ 위의 예에서는 콘솔에 출력하는 정의를 실시하고 있으며 그 정의에 'STDOUT'이라는 이름을 붙임
    - ❑ pattern 태그에는 출력하는 로그의 포맷을 기술하고 있다.
      - ○ %d 등의 기호는 출력하는 로그 문자열에 포함된 파라미터
      - ○ 지정할 수 있는 주요 파라미터

파라미터	설명
%date	로그를 출력한 날짜와 시간
%thread	스레드명
%level	로그 레벨(위의 예에서는 '-5'가 들어 있는데, 이것은 로그 레벨의 출력폭을 5문자 고정으로 하고 있음을 나타낸다.)
%logger	로그를 출력한 클래스 이름
%line	로그를 출력한 행 번호
%message	로그를 출력하는 코드에서 지정한 로그 메시지
%n	줄 바꿈

로그 출력

- 기본 설정
  - ✓ 로그 출력
    - ❑ 로그를 출력하려면 먼저 SLF4J의 LoggerFactory의 getLogger 메서드를 사용하여 클래스의 Logger 객체를 취득
    - ❑ Logger에는 로그 레벨에 따른 메서드(error/warn/info/debug/trace)가 준비되어 있기 때문에 용도에 따라 나누어 호출
    - ❑ 예

```
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
public class LogbackSample {
        private static final Logger logger =
LoggerFactory.getLogger(LogbackSample.class);
         public static void main(String[] args) {
                    logger.info("애플리케이션을 실행하였다");
        }
}
```

로그 출력

- 기본 설정
  - ✓ 파일 출력
    - ❑ logback.xml 파일의 appender 수정

```
<?xml version="1.0" encoding="UTF-8" ?>
<configuration>
 <appender name="FILE"
 class="ch.qos.logback.core.rolling.RollingFileAppender">
 <file>logs/logtest.log</file>
 <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
 <fileNamePattern>logs/logtest.%d{yyyy-MM-dd}.log</fileNamePattern>
 <maxHistory>30</maxHistory>
 </rollingPolicy>
 <encoder>
 <pattern>%date [%thread] %-5level %logger{35} - %message%n</pattern>
 </encoder>
 </appender>
 <root level="INFO">
 <appender-ref ref="FILE" />
 </root>
</configuration>
```

로그 출력

- 기본 설정
  - ✓ 패키지 별로 출력 설정
    - ❑ logback.xml 파일의 appender 수정

```
<?xml version="1.0" encoding="UTF-8" ?>
<configuration>
(중간 생략)
<logger name="jp.co.acroquest.java.log.dao" level="DEBUG" />
<root level="INFO">
<appender-ref ref="FILE" />
</root>
</configuration>
```

로그 출력

- 기본 설정
  - ✓ 동적 설정
    - ❑ logback.xml 파일의 appender 수정

```
<?xml version="1.0" encoding="UTF-8" ?>
<configuration scan="true" scanPeriod="60 seconds">
(중간 생략)
</configuration>
```

CSV

CSV

- CSV(Comma-Separated Values)
  - ✓ 항목을 쉼표(,)로 구분하여 나열한 표 형식의 텍스트 데이터로 확장자는 일반적으로 csv
  - ✓ CSV는 첫 번째 줄에 정의 항목의 명칭을 작성하고(필수는 아님) 두 번째 행부터는 데이터를 작성
  - ✓ 각 항목은 쉼표(다른 구분자 사용 가능 – 공백, 탭, 엔터 등)로 구분되어 있으며 항목 안에 쉼표가 있는 경우는 항목을 큰따옴표(")로 감싸서 작성
  - ✓ Microsoft Excel에서도 열 수 있으며 검색 및 편집이 용이
  - ✓ 일정한 구분자가 아니고 일정한 간격이면 fwf(fixed width file)이라고 함
  - ✓ 생성 시 고려할 사항
    - ❑ 필드 안에 쉼표, 큰따옴표, 줄 바꿈을 포함하는 경우에는 큰따옴표(")로 묶는 등 이스케이프 처리가 필요
    - ❑ 빈 줄, 주석 행은 건너뛰는 처리가 필요
    - ❑ 인코딩, 줄 바꿈 코드의 취급이 필요

CSV

- CSV 읽기
  - ✓ CSV 데이터는 항목이 구분자로 구분되어 있기 때문에 자바에서 읽어 들일 때 '구분자로 split 메서드를 실행하면 된다'고 생각하기 쉬운데 실제로는 그렇게 간단하지가 않음
  - ✓ 예를 들어 항목 자체에 쉼표가 포함될 수 있기 때문에 단순히 쉼표를 구분자로 split 메서드를 실행했을 경우 하나의 항목을 2개로 분할할 가능성도 있으며 그 경우 큰 따옴표 기호로 문자열을 둘러싸는 것이 일반적인 대처 방안이지만 그러면 이번에는 문자열 안에 큰따옴표가 들어 있을 때의 대처가 필요하게 되는데 이것만으로도 복잡하지만 그 외에도 다음과 같은 사항을 생각하면 필요한 프로그램의 양은 점점 늘어남
    - ❑ 첫 번째 행은 헤더 행으로 하고 싶음
    - ❑ 들어 있는 값이 올바른지 체크하고 싶음
    - ❑ 각 행의 내용을 자바의 객체에 각각 넣고 싶음
    - ❑ 쓰기도 하고 싶음
  - ✓ CSV 처리가 쉽게 보이기도 하지만 내용이 복잡해서 CSV 데이터를 읽어주는 라이브러리를 사용하는데 그 라이브러리 중의 하나가 Super CSV

volleyball.csv

```
name,age,birth,email,nickname
배유나,32,1989-11-30,bae@xxx.co.kr,"배구천재,웃음보따리"
문정원,28,1992-03-04,moon@xxx.co.kr,"문라이트"
```

CSV 읽기

- NoLibraryCSVMain

```
import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.InputStreamReader;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
```

CSV 읽기

❖ NoLibraryCSVMain

```
public class NoLibraryCSVMain {

        public static void main(String[] args) {
                //파일의 내용을 문자열로 읽기 위한 스트림을 생성
                try(BufferedReader br =
                                        new BufferedReader(
                                                new InputStreamReader(
                                                        new FileInputStream(

"./volleyball.csv")))){
                                //데이터를 저장할 리스트
                                List<Player> list = new ArrayList<>();
                                //첫번째 줄은 데이터로 사용하지 않기 위해서 만든 변수
                                boolean flag = false;
```

CSV 읽기

- NoLibraryCSVMain

```
//줄단위로 읽기
while(true) {
        String line = br.readLine();
        if(line == null) {
                break;
        }
        //첫번째 줄의 데이터는 파싱하지 않음
        if(flag == false) {
                flag = true;
                continue;
        }
        //한 줄을 읽어서 ,로 분할
        String [] ar = line.split(",");
        //한 줄의 데이터를 저장할 객체를 생성
        Player player = new Player();
        player.setName(ar[0]);
        player.setAge(Integer.parseInt(ar[1]));

        String birthday = ar[2];
        SimpleDateFormat sdf =
                new SimpleDateFormat("yyyy-MM-dd");
        Date date = sdf.parse(birthday);
        player.setbirth(date);
```

CSV 읽기

❖ NoLibraryCSVMain

```
                                        //객체를 리스트에 추가
                                        list.add(player);
                                }

                                for(Player player : list) {
                                        System.out.printf("%s₩n", player);
                                }

                        }catch(Exception e) {
                                System.out.printf(
                                        "파일읽기예외:%s₩n", e.getMessage());
                        }
                }
        }
```

pom.xml

- pom.xml 파일에 SuperCsv를 사용하기 위한 의존성을 추가 – dependencies 태그 안에 추가

```
<dependency>
        <groupId>net.sf.supercsv</groupId>
        <artifactId>super-csv</artifactId>
        <version>2.4.0</version>
</dependency>
```

Player.java

```
import java.util.Date;

public class Player {
		private String name;
		private int age;
		private Date birth;
		private String email;
		private String nickname;

		public Player() {
			super();
			// TODO Auto-generated constructor stub
		}
		public Player(String name, int age, Date birth, String email, String nickname) {
			super();
			this.name = name;
			this.age = age;
			this.birth = birth;
			this.email = email;
			this.nickname = nickname;
		}
```

Player.java

```
public String getName() {
        return name;
}
public void setName(String name) {
        this.name = name;
}
public int getAge() {
        return age;
}
public void setAge(int age) {
        this.age = age;
}
public Date getbirth() {
        return birth;
}
public void setbirth(Date birth) {
        this.birth = birth;
}
public String getEmail() {
        return email;
}
public void setEmail(String email) {
        this.email = email;
}
```

Player.java

```
	public String getNickname() {
		return nickname;
	}
	public void setNickname(String nickname) {
		this.nickname = nickname;
	}

	@Override
	public String toString() {
		return "Player [name=" + name + ", age=" + age + ", birth=" + birth + ",
email=" + email + ", nickname="
				+ nickname + "]";
	}
}
```

SuperCsvMain.java

```
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;

import org.supercsv.cellprocessor.ParseDate;
import org.supercsv.cellprocessor.ParseInt;
import org.supercsv.cellprocessor.constraint.NotNull;
import org.supercsv.cellprocessor.ift.CellProcessor;
import org.supercsv.io.CsvBeanReader;
import org.supercsv.io.ICsvBeanReader;
import org.supercsv.prefs.CsvPreference;
import org.supercsv.cellprocessor.Optional;
```

SuperCsvMain.java

```
public class SuperCsvMain {

	public static void main(String[] args) {
		// 제약조건을 설정
		CellProcessor[] processors = new CellProcessor[] { new NotNull(), // 이
름은 필수
				new ParseInt(new NotNull()), // 정수로 변환
				new ParseDate("yyyy-MM-dd"), // Date 로 변환,
				new Optional(), new Optional() };

		// 데이터를 저장할 List 생성
		List<Player> list = new ArrayList<>();
		// 파일 경로를 생성
		Path path = Paths.get("volleyball.csv");
```

SuperCsvMain.java

```
                        // 읽기
                        try (ICsvBeanReader beanReader = new
CsvBeanReader(Files.newBufferedReader(path),
                                                CsvPreference.STANDARD_PREFERENCE)) {
                                // 헤더 가져오기 - 첫번째 행의 데이터
                                String[] header = beanReader.getHeader(true);
                                System.out.println(Arrays.toString(header));
                                String[] header1 = {"name", "age", "birth", "email","nickname"};
                                // 두번째 행 부터 읽어오기
                                Player player = null;
                                // 줄 단위로 읽어서 player 로 만든 후
                                // list에 추가
                                while ((player = beanReader.read(Player.class, header1,
processors)) != null) {
                                        list.add(player);
                                }
                                for (Player temp : list) {
                                        System.out.printf("%s₩n", temp);
                                }
                        } catch (Exception e) {
                                System.out.printf("파일 읽기 예외:%s₩n", e.getMessage());
                                e.printStackTrace();
                        }
                }
}
```

JSON Parsing

JSON Parsing

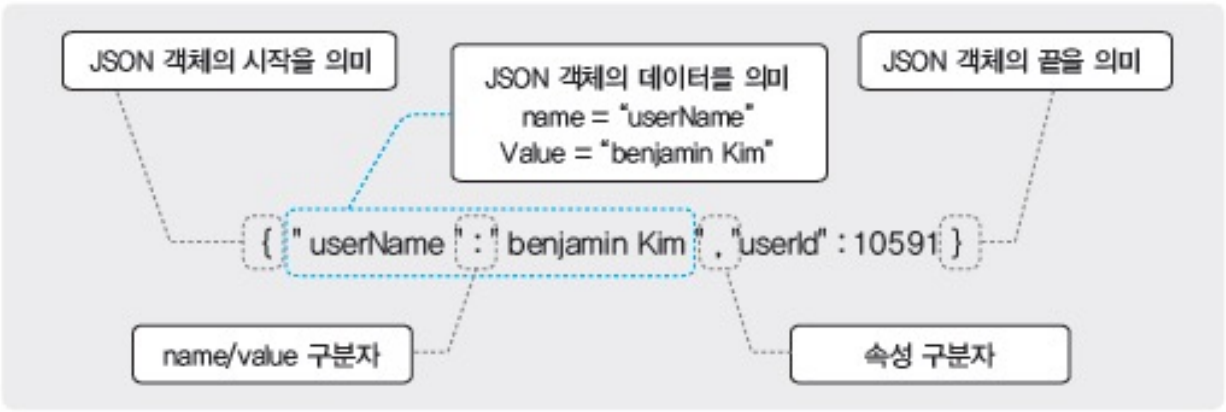
- JSON(JavaScript Object Notation)
  - ✓ 속성-값 쌍(attribute-value pairs and array data types or any other serializable value) 또는 "키-값 쌍"으로 이루어진 데이터 오브젝트를 전달하기 위해 인간이 읽을 수 있는 텍스트를 사용하는 개방형 표준 포맷
  - ✓ JSON을 사용함으로써 얻을 수 있는 이점
    - 데이터 크기가 작음
    - 사람이 읽고 쓰기에 용이
    - JSON은 순수 텍스트로 구성되어 있고 데이터 교환을 위한 표준 표기 방법으로 이기종 시스템 간에 데이터 교환 방식으로 플랫폼에 종속적이지 않음
  - ✓ JSON Library - json

```
<dependency>
        <groupId>org.json</groupId>
        <artifactId>json</artifactId>
        <version>20201115</version>
</dependency>
```

JSON Parsing

- JSON(JavaScript Object Notation)
  - ✓ JSON 규칙
    - ❑ JSON 객체
      - JSON 객체의 시작과 끝은 중괄호 ({ })로 표기
      - JSON 객체의 속성은 Name/Value 형태
      - 속성 Name/Value는 콜론(:)을 기준으로 구분
      - JSON 객체의 속성은 콤마를 사용해서 구분
      - JSON 객체의 속성 name은 객체 내에서 유일
      - String형 데이터는 쌍 따옴표를 사용

사용예 : { “NAME” : “Benjamin Kim”, “URL” : “www.facebook.com”}



JSON Parsing

- JSON(JavaScript Object Notation)
  - ✓ JSON 규칙
    - ❑ JSON 배열
      - JSON 배열의 시작과 끝은 대괄호를 사용
      - JSON 배열의 멤버 변수들은 콤마를 사용해서 구분
      - JSON 배열의 구성 아이템으로 JSON 객체도 넣을 수 있음
      - JSON 객체의 value 부분에는 JSON 객체를 넣을 수 있음

```
사용예: [ "Benjamin Kim", "Hamzani Ali", "Lodoss" ]
        [ 1, 2, 3, 5, 8, 13, 21]
        [ { "name": "Benjamin" , "age" : 33 } , { "name" : "Hamzani" , "age" : 30 } ]
```

JSON Parsing

❖ json Library의 주요 클래스들

클래스	설명
org.json.simple Class JSONObject	JSON 객체를 추상화한 클래스로, java.util.HashMap 클래스를 상속받고 있으므로 대부분의 메소드가 HashMap 클래스로부터 상속받고 있다.
org.json.simple Class JSONArray	JSON 배열을 추상화한 클래스로, java.util.ArrayList 클래스를 상속하고 있으므로 메소드 사용 방법은 대부분 우리가 이미 배운 ArrayList와 거의 흡사하다.
org.json.simple.parser Class JSONParser	JSON 데이터를 파싱하는 기능을 구현한 클래스다.
org.json.simple Class JSONValue	JSON 데이터를 다루기 위한 몇 가지 메소드들을 제공한다.
org.json.simple.parser Class ParseException	JSONParser 클래스를 사용해서 파싱할 때 발생할 수 있는 예외 사항을 추상화한 클래스다.

❖ JSONObject 나 JSONArray에는 get으로 시작하는 메서드를 이용해서 데이터를 추출하는 메서드가 존재하며 JSONObject는 key를 JSONArray는 index를 매개변수로 대입해야 함

JSON Parsing

- JSON 라이브러리를 사용할 수 있도록 pom.xml 파일의 dependencies 태그 안에 아래 코드를 추가

```
<dependency>
        <groupId>org.json</groupId>
        <artifactId>json</artifactId>
        <version>20201115</version>
</dependency>
```

JSON Parsing

```
import org.json.JSONArray;
import org.json.JSONObject;

public class JSONMain {
	public static void main(String[] args) {
		//JSON 배열 파싱
		String json = "[100, 500, 300, 200, 400]";
		JSONArray ar = new JSONArray(json);
		for (int i = 0; i < ar.length(); i++) {
			System.out.print(ar.get(i) + "₩t");
		}
		System.out.println();
		//JSON 객체 파싱
		json = "{color: ₩"red₩",value: ₩"#f00₩"}";
		JSONObject obj = new JSONObject(json);
		System.out.println("색상:" + obj.getString("color"));
		System.out.println("색상:" + obj.getString("value"));
	}
}
```

Open API

- Open API
  - 오픈 API(Open Application Programming Interface, Open API, 공개 API)는 누구나 사용할 수 있도록 공개된 API를 말하며 개발자에게 사유 응용 소프트웨어나 웹 서비스에 프로그래밍적인 권한을 제공
  - 하나의 웹 사이트에서 자신이 가진 기능을 이용할 수 있도록 공개한 프로그래밍 인터페이스가 오픈 API 라고 정의하기도 함
  - 네이버 지도, 구글 맵, 오픈 스트리트 맵 등이 대표적인 예
  - 지도 서비스 및 다양한 서비스들에서 시도되고 있으며 누구나 접근하여 사용할 수 있다는 장점이 있음
  - 메타블로그도 오픈 API를 사용하여 만든 예이며 최근에 코로나 사태로 라이브코로나, 코로나 맵 등의 서비스가 생겼는데, 네이버 클라우드 플랫폼의 API 지원을 받고 있음

영화정보 Open API

- http://cyberadam.cafe24.com/movie/list

{"count":4262,"list":[{"movieid":4264,"title":"바이러스 격리구역
","subtitle":"Infected","pubdate":"2013","director":"필립 매시즈위츠|","actor":"딜라란 마틴|보 린
튼|유지니아 쿠즈미나|아드리안 부|아디아 딘|니나 케이트|","genre":"공포
","rating":10.0,"thumbnail":"REP_02310042540_1_1_0138.jpg","link":"https://movie.naver.com/
movie/bi/mi/basic.nhn?code=104605"},{"movieid":4263,"title":"쥬라기공원 : 다이노어택
","subtitle":"Jurassic Attack","pubdate":"2013","director":"안소니 팽크하우저|","actor":"조던 로슨
|코린 네멕|베론 웰스|게리 스트레치|","genre":"SF/판타지,공포,액션/모험
","rating":10.0,"thumbnail":"REP_02310042015_1_1_0235.jpg","link":"https://movie.naver.com/
movie/bi/mi/basic.nhn?code=103735"},{"movieid":4262,"title":"영웅본색 2
HD","subtitle":null,"pubdate":null,"director":null,"actor":null,"genre":"드라마,액션/모험
","rating":10.0,"thumbnail":"REP_02310042053_1_1_0219.jpg","link":"http://swiftapi.rubypaper.c
o.kr:2029/hoppin/detailView?movieId=P00000529234"},{"movieid":4261,"title":"페인킬러 엑스
","subtitle":"Painkiller","pubdate":"2012","director":"베레니카 매시즈위츠|","actor":"마이크 파프|
엘로디 하라|론 푸실로|딜라란 마틴|","genre":"액션/모험,스릴러
","rating":0.0,"thumbnail":"REP_02310041823_1_1_0240.jpg","link":"https://movie.naver.com/m
ovie/bi/mi/basic.nhn?code=104607"...]}

영화정보 Open API

- 영화 제목과 평점만 추출해서 출력
  바이러스 격리구역:10.0
  쥬라기공원 : 다이노어택:10.0
  영웅본색 2 HD:10.0
  페인킬러 엑스:0.0
  스페셜 포스 : 특수부대 전랑:0.0
  스파이 : 작전명 태풍:10.0
  데몬킹스 (자막):10.0
  감독 미카엘 하네케:10.0
  댐999 : 거대한 물폭탄:0.0
  데몬킹스 (더빙):5.5

영화정보 Open API

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

import org.json.JSONArray;
import org.json.JSONObject;

public class JSONMovie {

	public static void main(String[] args) {

		// 웹의 문자열 가져오기
		String json = "";
		try {
			// 다운로드 받을 주소 생성
			String addr = "http://cyberadam.cafe24.com/movie/list";
			URL url = new URL(addr);
			// 연결 객체 생성
			HttpURLConnection con = (HttpURLConnection)
url.openConnection();
			// 옵션 설정
			// 최대 30초 동안 연결을 시도
			con.setConnectTimeout(30000);
```

영화정보 Open API

```
                    // 웹에서 문자열을 읽어올 스트림 생성
                    BufferedReader br = new BufferedReader(new
InputStreamReader(con.getInputStream()));
                    // 문자열을 일시적으로 저장할 객체 생성
                    StringBuilder sb = new StringBuilder();
                    // 줄 단위로 읽어서 sb에 추가
                    while (true) {
                              String line = br.readLine();
                              if (line == null) {
                                        break;
                              }
                              sb.append(line + "₩n");
                    }
                    // 문자열 변수에 대입
                    json = sb.toString();
                    // System.out.printf("%s₩n", json);

          } catch (Exception e) {
                    System.out.printf("다운로드 예외:%s₩n", e.getMessage());
                    e.printStackTrace();
          }
```

영화정보 Open API

```
                try {

                        JSONObject data = new JSONObject(json);
                        JSONArray movies = data.getJSONArray("list");
                        for (int i = 0; i < movies.length(); i = i + 1) {
                                JSONObject item = movies.getJSONObject(i);
                                System.out.printf("%s:%s₩n", item.getString("title"),
item.getDouble("rating"));
                        }

                } catch (Exception e) {
                        System.out.printf("JSON 파싱 예외:%s₩n", e.getMessage());
                        e.printStackTrace();
                }
        }
}
```

Open API

- Kakao Open API
  - ✓ 개발 가이드: https://developers.kakao.com/docs/latest/ko/daum-search/dev-guide#search-book
  - ✓ URL: https://dapi.kakao.com/v3/search/book?target=title?query=검색할 도서명
  - ✓ Header: Authorization: KakaoAK kkkkkkkkkkkkkkkkkkkkkkkkkkkkkkkk
  - ✓ REST API AppKey: 06fab290c9f4eb6f130c09796d57bc30
  - ✓ 파라미터가 한글인 경우 인코딩 : java.net.URLEncodier(String data, String enctype)

Kakao Open API

{"channel":{"result":"10","title":"Search Daum Open API","totalCount":"97","description":"Daum Open API search result","item":[{"author_t":"이일
민","sale_price":"67500","cover_s_url":"http://t1.daumcdn.net/thumb/R72x100/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FBOK0001898558228A?
moddttm=20150421074557","sale_yn":"Y","pub_date":"20120921","link":"http://book.daum.net/detail/book.do?bookid=BOK0001898558228A","barcode":"BOK0001898558228A","etc_author":"","status_des":"정
상판매","author":"이일민","title":"토비의 스프링 3.1 세트","category":"컴퓨터/IT ","translator":"","pub_nm":"에이콘출판","description":"요약 [토비의 스프링 3.1]은 스프링을 처음 접하거나 스프
링을 경험했지만 스프링이 어렵게 느껴지는 개발자부터 스프링을 활용한 아키텍처를 설계하고 프레임워크를 개발하려고 하
는...","isbn":"89607773433","ebook_barcode":"","isbn13":"9788960773431","cover_l_url":"http://t1.daumcdn.net/thumb/R110x160/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FBOK0001898558228A?
moddttm=20150421074557","list_price":"75000"},{"author_t":"오창훈","sale_price":"29700","cover_s_url":"http://t1.daumcdn.net/thumb/R72x100/?
fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FKOR9788960770942?moddttm=20150409094051","sale_yn":"Y","pub_date":"20090820","link":"http://book.daum.net/detail/book.do?
bookid=KOR9788960770942","barcode":"KOR9788960770942","etc_author":"","status_des":"정상판매","author":"오창훈","title":"오픈 API를 활용한 매쉬업 가이드","category":"컴퓨
터/IT ","translator":"","pub_nm":"에이콘출판","description":"웹 생태계를 더욱 풍요롭게 해주는 매쉬업 애플리케이션. 블로그, 차트, 지도, 이미지, 동영상 API의 효과적인 활용법과 실용 예제를 배우
고, API 사용자 인증, 파일 전송, 배포 등...","isbn":"8960770949","ebook_barcode":"","isbn13":"9788960770942","cover_l_url":"http://t1.daumcdn.net/thumb/R110x160/?
fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FKOR9788960770942?moddttm=20150409094051","list_price":"33000"},{"author_t":"주영아 이광
호","sale_price":"32400","cover_s_url":"http://t1.daumcdn.net/thumb/R72x100/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FBOK0001846128711N?
moddttm=20150412073414","sale_yn":"Y","pub_date":"20120630","link":"http://book.daum.net/detail/book.do?bookid=BOK0001846128711N","barcode":"BOK0001846128711N","etc_author":"이광
호","status_des":"정상판매","author":"주영아","title":"jQuery Mobile inside","category":"컴퓨터/IT ","translator":"","pub_nm":"스마트미디어","description":"네이티브 앱을 대체하기 위해서 각광
받고 있는 웹앱 기술. "웹"이라는 주제가 분명 Java나 Objective-C같은 네이티브 개발 기술보다는 쉽게 생각되어 질 수도 있지만, 막상 뚜껑을 열어보
면...","isbn":"8981360122","ebook_barcode":"","isbn13":"9788981360122","cover_l_url":"http://t1.daumcdn.net/thumb/R110x160/?fname=http%3A%2F%2Ft1.daumcdn.net%2Fbook%2FBOK0001846128711N?

Kakao Open API

- 도서 이름을 입력받아서 입력받은 도서로 검색한 후 도서 제목과 저자 그리고 가격을 조회

조회할 도서이름(종료는 그냥 엔터):java

제목	가격	저자
명품 JAVA Programming(개정판 4판)	33000	황기태 김효수
Java	40000	Savitch Mock
Java	89340	Herbert Schildt
Java	59560	Herbert Schildt
Java	59560	Herbert Schildt
Java	56080	Herbert Schildt
Java	54220	Schildt Herbert
How to Program(Java)	48000	Paul Deitel Harvey Deitel
Java의 정석(3판)	30000	남궁성
HTML5 + CSS3 + Javascript 웹 프로그래밍(명품)	28000	황기태

조회할 도서이름(종료는 그냥 엔터):

Kakao Open API

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.UnsupportedEncodingException;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.Scanner;

import org.json.JSONArray;
import org.json.JSONObject;
```

Kakao Open API

```
class JsonThread extends Thread {
        public void run() {
                // 도서이름을 입력받기 위한 스캐너 생성
                Scanner sc = new Scanner(System.in);
                while (true) {
                        //도서이름 입력받기
                        System.out.print("조회할 도서이름(종료는 그냥 엔터):");
                        String bookname = sc.nextLine();
                        if(bookname.trim().length() == 0) {
                                break;
                        }
                        try {
                                //도서제목이 한글인 경우를 대비해서 인코딩
                                bookname = URLEncoder.encode(bookname, "utf8");
                                //다운로드 받은 문자열을 임시로 저장할 변수
                                StringBuilder sBuffer = new StringBuilder();
                                //다운로드 받은 문자열 전체를 저장할 변수
                                String json = "";
                                //파싱한 결과를 출력하기 위한 인스턴스
                                ArrayList<Map<String, Object>> data = new
ArrayList<>();
```

Kakao Open API

```
try {
        //다운로드 받을 url 만들기
        String urlAddr =
"https://dapi.kakao.com/v3/search/book?target=title&query=" + bookname;
        URL url = new URL(urlAddr);
        //url에 연결하는 객체 만들기
        HttpURLConnection conn = (HttpURLConnection)
url.openConnection();

        //연결 객체가 제대로 만들어져 있다면
        if (conn != null) {
                //다운로드 옵션 설정
                conn.setConnectTimeout(20000);
                conn.setUseCaches(false);
                //헤더 설정
                conn.addRequestProperty("Authorization",
"KakaoAK 06fab290c9f4eb6f130c09796d57bc30");
```

Kakao Open API

```
                                        //서버에서 정상 응답을 하면
                                        if (conn.getResponseCode() ==
HttpURLConnection.HTTP_OK) {
                                                //서버에서 문자열을 읽어올 스트림을 생성
                                                InputStreamReader isr = new

InputStreamReader(conn.getInputStream());
                                                BufferedReader br = new

BufferedReader(isr);
                                                //스트림에서 한 줄 씩 읽어서 sBuffer에 추

가
                                                while (true) {
                                                        String line = br.readLine();
                                                        if (line == null) {
                                                                break;
                                                        }
                                                        sBuffer.append(line);
                                                }
                                                //전부 읽었으므로 서버와 연결 해제
                                                br.close();
                                                conn.disconnect();
                                        }
                                }
```

Kakao Open API

```
                    //읽어온 문자열을 json에 추가
                    json = sBuffer.toString();
                    //System.out.println(json);
                            } catch (Exception e) {
                                    System.out.println("가져오기 실패:" +
e.getMessage());
                            }
```

Kakao Open API

```
//JSON 파싱
try {
		//처음에는 객체 형태로 가져오므로 객체 형태로 파싱
		JSONObject obj = new JSONObject(json);
		//documents 키에 해당하는 데이터를 배열로 가져오기
		JSONArray documents = obj.getJSONArray("documents");
		//배열의 데이터를 하나씩 읽으면서 수행
		for (int i = 0; i < documents.length(); i++) {
			//하나의 객체 가져오기
			JSONObject book = documents.getJSONObject(i);
			//하나의 객체를 저장하기 위한 Map 생성
			Map<String, Object> map = new HashMap<>();
			//제목과 저자는 바로 문자열로 가져오기
			map.put("title", book.getString("title"));
			map.put("price", book.getInt("price"));
			//저자는 다시 배열이므로 배열을 순회해서 list에 추가
			JSONArray authors = book.getJSONArray("authors");
			List<String>authorList = new ArrayList<>();
			for (int j = 0; j < authors.length(); j++) {
				authorList.add(authors.getString(j));
			}
			map.put("author", authorList);
			//하나의 객체를 전체 리스트에 저장
			data.add(map);
		}
}
```

Kakao Open API

```
                        catch (Exception e) {
                                System.out.println("파싱 실패:" + e.getMessage());
                        }

                        //데이터 출력
                        System.out.printf("%40s%10s%30s₩n", "제목", "가격", "저자");
                        for(Map<String, Object> temp : data) {
                                System.out.printf("%40s%10d", temp.get("title"),
temp.get("price"));
                                List<String>list = (List<String>)temp.get("author");
                                for(String imsi : list) {
                                        System.out.printf("%20s ", imsi);
                                }
                                System.out.println();
                        }
                } catch (UnsupportedEncodingException e) {
                        System.out.println("인코딩 에러");
                        System.out.println(e.getMessage());
                        e.printStackTrace();
                }
}
                sc.close();
}
}
```

Kakao Open API

```
public class JSONBookMain {
        public static void main(String[] args) {
                JsonThread th = new JsonThread();
                th.start();
        }
}
```

JSON Writing

- JSON Writing
  - ✓ jackson-databind 라이브러리의 ObjectMapper 클래스 이용
  - ✓ 의존성

```
<dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-core</artifactId>
        <version>2.7.4</version>
</dependency>
<dependency>
        <groupId>com.fasterxml.jackson.core</groupId>
        <artifactId>jackson-databind</artifactId>
        <version>2.7.4</version>
</dependency>
```

JSON Writing

- JSONCreate.java

```
import java.io.File;
import java.util.Arrays;
import java.util.List;
import com.fasterxml.jackson.databind.ObjectMapper;
public class JsonCreate {
        static class Employee {
                private String name;
                private int age;
                private List<String> licenses;

                public String getName() {
                        return name;
                }
                public void setName(String name) {
                        this.name = name;
                }
                public int getAge() {
                        return age;
                }
                public void setAge(int age) {
                        this.age = age;
                }
```

JSON Writing

- JSONCreate.java

```
                public List<String> getLicenses() {
                        return licenses;
                }
                public void setLicenses(List<String> licenses) {
                        this.licenses = licenses;
                }

                @Override
                public String toString() {
                        return "Employee [name=" + name + ", age=" + age + ",
licenses=" + licenses + "]";
                }
        }
```

JSON Writing

- JSONCreate.java

```
        public static void main(String[] args) {
                Employee employee1 = new Employee();
                employee1.setName("아담");
                employee1.setAge(50);
                employee1.setLicenses(Arrays.asList("OCP","정보처리 기사"));

                Employee employee2 = new Employee();
                employee2.setName("류시아");
                employee2.setAge(48);
                employee2.setLicenses(Arrays.asList("CCNA","사무자동화 산업기사"));

                File file = new File("employee.json"); // 쓰기 대상의 JSON 데이터 파일

                ObjectMapper mapper = new ObjectMapper();
                try {
                        mapper.writeValue(file, Arrays.asList(employee1, employee2));
                } catch (Exception e) {
                        // TODO Auto-generated catch block
                        e.printStackTrace();
                }
        }
}
```

XML Parsing

XML Parsing

- XML
  - XML은 웹 서비스의 기본 데이터 포맷으로 서버와 클라이언트의 통신 수단
  - XML을 활용하면 구조화된 데이터를 저장한 문서를 만들 수 있으며 보다 자유로운 문서 표현이 가능
  - 트리(Tree) 형태로 구조화 될 수 있어 프로그램에서 데이터를 읽고 쓰기에 매우 적합
  - 데이터가 문서에 저장되므로 어떤 프로그램이든지 쉽게 내용을 확인할 수 있기 때문에 개발 언어 혹은 OS에 매우 독립적인 형식
  - HTML로 문서의 구조나 정보를 표현하는 것에는 한계
  - 대부분의 오픈 소스 라이브러리나 프레임워크에서는 프로그램 외부에서 기능에 대한 정의 및 설정을 XML 파일로 조정할 수 있음
- RSS(Really Simple Syndication, Rich Site Summary): 자주 바뀌는 내용을 제공하기 위해 사용되는 표준 웹 피드 포맷으로 xml 형식을 주로 이용

XML Parsing

- XML Parsing
  - ✓ 서버는 클라이언트의 요청을 받아들여 처리하고 그 결과를 XML로 리턴하며 클라이언트는 XML을 분석하여 처리 결과를 얻음
  - ✓ XML 자체는 단순한 텍스트 포맷이지만 규칙이 엄격
  - ✓ XML 파서는 크게 DOM, SAX 두 가지로 구분되며 DOM은 트리 형식으로 문서를 읽어서 전체 구조를 파악한 후 정보를 구하는 방식이고 SAX는 순차적으로 문서를 읽으면서 정보를 차례대로 읽는 방식
  - ✓ DOM은 메모리를 많이 사용하지만 성능이 좋고 SAX는 느리지만 메모리를 거의 사용하지 않음

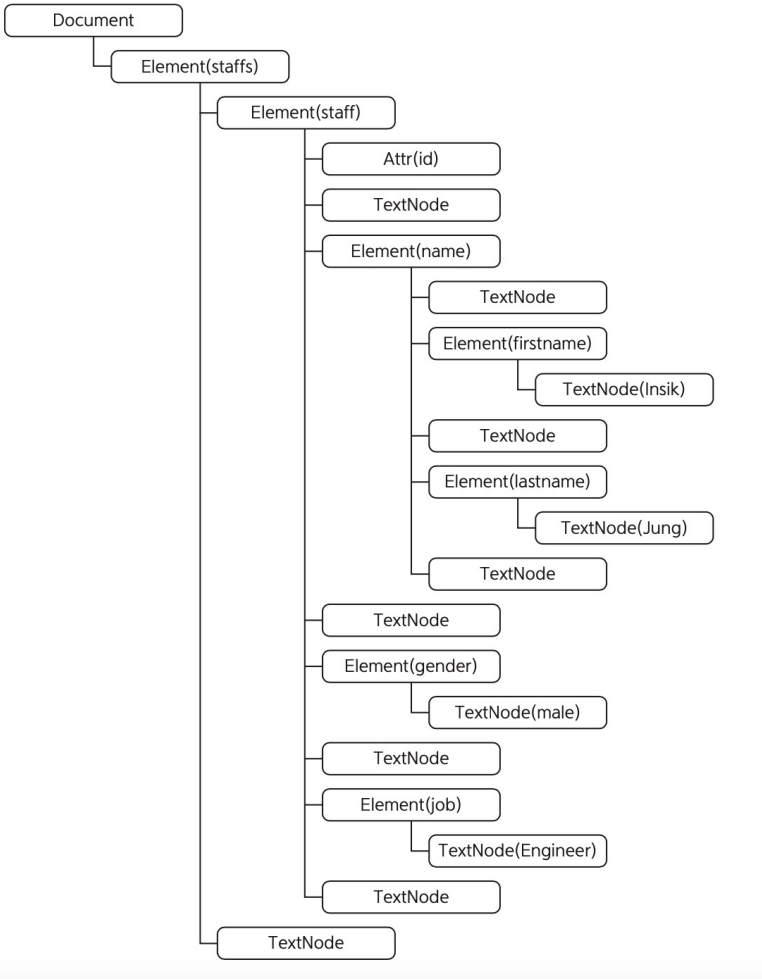
XML Parsing

- DOM 파서
  - ✓ DOM(Document Object Model)은 XML 문서를 객체의 트리 구조로 읽어들이는 API
  - ✓ xml를 트리 구조로 표시한 예
  - ✓ 요소
    - ❑ Document: XML 문서 전체
    - ❑ Element: 태그
    - ❑ Attr: 속성
    - ❑ TextNode: 태그로 묶인 문자열

```
<?xml version="1.0" encoding="UTF-8"?> <staffs>
<staff id="1024"> <name>
<firstname>InSik</firstname>
<lastname>Jung</lastname> </name>
<gender>male</gender>
<job>Engineer</job> </staff>
</staffs>
```

XML Parsing

- DOM 파서



XML Parsing

- DOM 파서
  - ✓ XML 문서를 Parsing하기 위해서는 DocumentBuilderFactory 객체를 생성
  - ✓ DocumentBuilderFactory 클래스는 추상 클래스이므로 직접 생성할 수 없으며 newInstance라는 정적 메서드로 생성
  - ✓ 객체를 생성한 후 주석 무시 여부, 네임 스페이스 인식 여부, 유효성 점검 여부 등의 속성을 설정
  - ✓ 속성을 맞춘 후 newDocumentBuilder 메서드를 호출하면 DocumentBuilder 객체가 리턴
    - ❑ static DocumentBuilderFactory newInstance()
    - ❑ DocumentBuilder newDocumentBuilder()
  - ✓ DocumentBuilder도 추상 클래스이므로 반드시 이 과정을 거쳐야만 생성할 수 있음
    - ❑ Document parse(InputStream stream [, String systemId])
    - ❑ Document parse(String uri)
    - ❑ Document parse(File file)
  - ✓ Parse 메서드는 스트림을 분석하여 메모리에 트리 형태로 전개해 놓고 이후 Document 객체의 메서드로 요소들을 빠른 속도로 읽을 수 있음
    - ❑ Element Document.getDocumentElement()
  - ✓ XML 문서는 유일한 루트 엘리먼트 하나를 가지는데 getDocumentElement 메서드는 루트 엘리먼트를 구함 – 여기까지는 전부 동일

XML Parsing

- DOM 파서
  - ✓ 다음 메서드는 태그 명과 일치하는 엘리먼트를 찾아 Node의 배열인 NodeList 객체를 리턴
    - NodeList Element.getElementsByTagName(String tagname)
  - ✓ NodeList에는 개수를 구하는 getLength 메서드와 순서 값으로부터 노드를 찾는 item 메서드가 제공됨
  - ✓ Node 객체를 구했으면 다음 메서드로 노드의 정보를 구함
    - String getNodeName()
    - short getNodeType()
    - String getNodeValue()
  - ✓ 각각 노드의 이름, 타입, 값이며 다음 노드를 중심으로 주변 노드를 찾는 메서드
    - Node getFirstChild()
    - Node getLastChild()
    - Node getNextSibling()
    - Node getPreviousSibling()
    - Node getParentNode()
    - NodeList getChildNodes()

XML Parsing

- DOM 파서
  - ✓ Item 엘리먼트의 값을 읽으려면 getFirstChild로 첫 번째 자식을 구하고 getNodeValue로 자식의 값을 읽는데 이유는 DOM은 엘리먼트 안의 문자열도 하나의 객체로 취급해서 문자열을 엘리먼트의 자식으로 취급하기 때문
  - ✓ 항목 여러 개를 읽어야 하는 경우에는 NodeList의 배열을 순회해야 하며 엘리먼트의 속성을 읽을 때는 다음 메서드를 사용
    - NamedNodeMap Node.getAttributes ()
  - ✓ 엘리먼트 와는 달리 속성은 순서가 없으므로 어떤 속성이 먼저 조사될지는 알 수 없음
  - ✓ XML 스팩은 속성의 순서에 의미를 부여하지 않으며 NamedNodeMap은 순서가 없는 속성을 저장

기상청 날씨

http://www.kma.go.kr/weather/forecast/mid-term-xml.jsp?stnId=109

```
<wid>
  <header>
    <title>서울,경기도 육상주간예보</title>
    <tm>201403110600</tm>
    <wf>
      <![CDATA[
        기압골의 영향으로 19일에 비가 오겠고, 그 밖의 날은 고기압의 가장자리에 들어 가끔 구름많겠습니다.<br />기온은
        />강수량은 평년(강수량 : 1~2mm)과 비슷하거나 조금 많겠습니다.
      ]]>
    </wf>
  </header>
  <body>
    <location city="11B10101" province="11B00000" wl_ver="3">
      <province code="11B00000">서울 · 인천 · 경기도</province>
      <city code="11B10101">서울</city>
      <data>
        <numEf>2</numEf>
        <tmEf>2014-03-13</tmEf>
        <wf>구름많음</wf>
        <tmn>3</tmn>
        <tmx>10</tmx>
        <reliability>보통</reliability>
      </data>
      <data>
        <numEf>3</numEf>
        <tmEf>2014-03-14</tmEf>
        <wf>구름많음</wf>
        <tmn>-2</tmn>
        <tmx>9</tmx>
        <reliability>보통</reliability>
      </data>
```

기상청 날씨

http://www.kma.go.kr/weather/forecast/mid-term-xml.jsp?stnId=109

서울

2020-05-23:흐리고 비
2020-05-24:구름많음
2020-05-25:맑음
2020-05-26:맑음
2020-05-27:맑음
2020-05-28:구름많음

인천

2020-05-23:흐리고 비
2020-05-24:구름많음
2020-05-25:맑음
2020-05-26:맑음
2020-05-27:맑음
2020-05-28:구름많음

수원

2020-05-23:흐리고 비
2020-05-24:구름많음
2020-05-25:맑음
2020-05-26:맑음
2020-05-27:맑음
2020-05-28:구름많음

기상청 날씨

❖ 데이터를 받아와서 Parsing

```
import java.io.BufferedReader;
import java.io.ByteArrayInputStream;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;
import java.util.List;

import javax.xml.parsers.DocumentBuilder;
import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;
import org.w3c.dom.Element;
import org.w3c.dom.Node;
import org.w3c.dom.NodeList;
```

기상청 날씨

```
public class KMAParsing {
	public static void main(String[] args) {
		//기상청 날씨 데이터 가져오기
		String xml = "";
		try {
			//URL 만들기
			String addr =
				"http://www.weather.go.kr/weather/forecast/mid-
term-xml.jsp?stnId=109";
			URL url = new URL(addr);
			//연결 생성 및 옵션 설정
			HttpURLConnection con =
(HttpURLConnection)url.openConnection();
			con.setUseCaches(false);
			con.setConnectTimeout(30000);
			//데이터 읽을 수 있는 스트림 생성
			BufferedReader br = new BufferedReader(new
InputStreamReader(con.getInputStream()));
```

기상청 날씨

```
                //한 줄 씩 읽어서 StringBuilder에 저장
                StringBuilder sb = new StringBuilder();
                while(true) {
                        String line = br.readLine();
                        if(line == null) {
                                break;
                        }
                        sb.append(line + "₩n");
                }
                //데이터 옮기기
                xml = sb.toString();
                //연결 끊기
                br.close();
                con.disconnect();

}catch(Exception e) {
                System.out.printf(
                        "다운로드 예외:%s₩n",
                        e.getMessage());
}
```

기상청 날씨

```
try {
        //Parsing을 수행할 객체 생성
        DocumentBuilderFactory factory =
DocumentBuilderFactory.newInstance();
        DocumentBuilder documentBuilder =
factory.newDocumentBuilder();
        //Parsing을 수행할 데이터 만들기
        InputStream is = new ByteArrayInputStream(xml.getBytes());
        //메모리에 전부 펼치기
        Document document = documentBuilder.parse(is);
        //루트 찾기
        Element element = document.getDocumentElement();
        //city 태그의 내용을 찾아서
        //List에 저장
        List<String>cities = new ArrayList<>();
        //city 태그 전부 가져오기
        NodeList cityList = element.getElementsByTagName("city");
        for(int i=0; i<cityList.getLength(); i=i+1) {
                Node item = cityList.item(i);
                Node city = item.getFirstChild();
                cities.add(city.getNodeValue());
        }
```

기상청 날씨

```
//날짜를 가져와서 전부 저장하기
List<String> dateList = new ArrayList<>();
//tmEf 태그의 값 가져오기
NodeList tmef = element.getElementsByTagName("tmEf");
for(int i=0; i<tmef.getLength();i=i+1) {
                Node item = tmef.item(i);
                Node t = item.getFirstChild();
                dateList.add(t.getNodeValue());
}
```

기상청 날씨

```
//wf 태그의 값 저장하기
List<String>weatherList = new ArrayList<>();
NodeList wfList =
            element.getElementsByTagName("wf");
for(int i=0; i<wfList.getLength(); i=i+1) {
            //첫번째 데이터를 사용하지 않기 위해서
            //첫번째 데이터인 경우는 다음으로 넘기기
            if(i==0) {
                        continue;
            }
            Node item = wfList.item(i);
            Node wf = item.getFirstChild();
            weatherList.add(wf.getNodeValue());
}
```

기상청 날씨

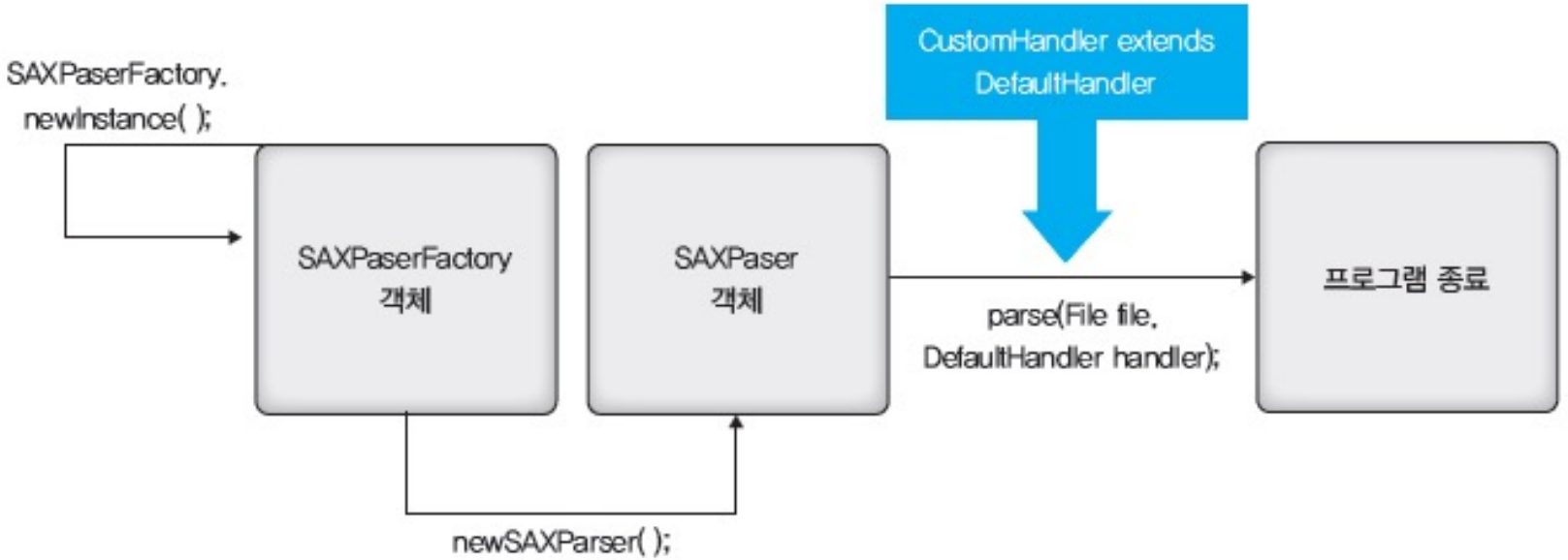
```
                    //데이터 출력
                    int i = 0;
                    for(String city : cities) {
                            System.out.printf("%s₩n", city);
                            for(int j=0; j<6; j=j+1) {
                                    System.out.printf("₩t%s:%s₩n",

        dateList.get(i*6+j),weatherList.get(i*6+j));
                            }
                            i = i + 1;
                    }
            }catch(Exception e) {
                    System.out.printf(
                                    "XML Parsing 예외:%s₩n",
                                    e.getMessage());
            }
        }
}
```

XML Parsing

- SAX(Simple API for XML) Parsing
  - ✓ SAX 파서는 이벤트 기반의 Parsing을 지원
  - ✓ 수행 시간이 빠름
  - ✓ 메모리 효율성이 높음
  - ✓ 구조화된 객체가 따로 필요
  - ✓ 순차적 Parsing을 지원

XML Parsing

❖ SAX 파서 관련 주요 클래스와 메서드



XML Parsing

- SAX 파서 관련 주요 클래스와 메서드
  - ✓ javax.xml.parsers.SAXParserFactory 클래스
    - ❑ XML 문서를 Parsing하는 SAXParser 객체를 만들기 위한 Factory 클래스

사용법 : public static SAXParserFactory newInstance( )

public static SAXParserFactory newInstance(String factoryClassName,ClassLoader classLoader)

SAXParserFactory [객체 이름] = SAXParserFactory.newInstance( );

사용예 : `SAXParserFactory factory = SAXParserFactory.newInstance();`

- ❑ newSAXParser( ) : 현재 SAXParserFactory 객체에 설정된 값을 바탕으로 SAXParser 객체 생성

```
public abstract SAXParser newSAXParser() throws ParserConfigurationException,
SXException
```

XML Parsing

- SAX 파서 관련 주요 클래스와 메서드
  - ✓ javax.xml.parsers.SAXParser 클래스
    - ❑ XML 문서를 Parsing하기 위한 클래스
    - ❑ SAXParser 객체를 만드는 방법

```
사용법 : SAXParser [객체 이름] = [SAXParserFactory 객체].newSAXParser( );
사용예 : SAXParserFactory factory = SAXParserFactory.newInstance();
         SAXParser parser = factory.newSAXParser();
```

    - ❑ SAXParser 객체는 SAXParserFactory 클래스에서 제공한 newSAXParser( ) 메서드를 사용해서 생성

parse	• 선언부 : public void parse(File f, DefaultHandler dh) throws SAXException, IOException • 설명 : parse( ) 메소드는 여러 형태로 오버로딩되어 있으며 SAXException과 IOException 예외를 던질 수 있다. 아래의 parse( ) 메소드 선언부에서는 예외 선언을 생략했으니 참고하도록 하자. 매개변수 File f와 DefaultHandler 객체를 사용하여 파싱한다. 매개변수 f와 XML 문자를 읽을 때마다 이벤트가 발생하며 이벤트에 따라서 DeafultHandler 객체의 메소드가 SAXParser 객체 내부에서 호출된다. 이벤트마다 발생하는 DefaultHandler의 메소드는 따로 설명한다.
	• 선언부 : public void parse(InputSource is, DefaultHandler dh) public void parse(InputStream is, DefaultHandler dh) public void parse(InputStream is, DefaultHandler dh, String systemId) • 설명 : XML의 데이터를 읽을 때 Input 스트림 계열인 경우 사용하면 된다.
	• 선언부 : public void parse(String uri, DefaultHandler dh) • 설명 : XML의 데이터가 다른 시스템이나 네트워크상에 존재하고 있어 데이터의 위치가 URI로 표기되는 경우 이를 사용한다.

XML Parsing

- SAX 파서 관련 주요 클래스와 메서드
  - ✓ org.xml.sax.helpers.DefaultHandler 클래스
    - ❑ DefaultHandler 클래스는 필요한 데이터를 받아온 후 ValueObject에 그 데이터를 설정하는 클래스
    - ❑ DefaultHandler 클래스를 사용하는 방법

사용법 : **DefaultHandler [객체 이름] = new DefaultHandler( );**

사용예 :
```
DefaultHandler handler = new DefaultHandler();
SAXParserFactory factory = SAXParserFactory.newInstance();
SAXParser parser = factory.newSAXParser();
parser.parse(new File("D:/source.xml", handler);
```

    - ❑ DefaultHandler 클래스는 기본 생성자를 사용해서 인스턴스
    - ❑ DefaultHandler 클래스에서 제공하는 메서드들은 모두 SAXException을 예외로 던짐

메소드	설명
characters	• 선언부 : public void characters(char[ ] ch, int start, int length) • 설명 : XML 요소 내부에 데이터를 파싱할 때 이 characters 메소드가 내부적으로 호출된다. 즉, <name>Benjamin</name>이라는 내용을 SAXParser 객체가 파싱한다면 name 태그 요소 내부의 Benjamin이라는 값을 파싱할 때 characters( ) 메소드를 호출한다.
startDocument	• 선언부 : public void startDocument( ) • 설명 : XML 문서를 맨 처음 파싱할 때 startDocument( ) 메소드가 내부적으로 호출된다.
endDocument	• 선언부 : public void endDocument( ) • 설명 : startDocument( ) 메소드와 반대로 XML 문서 파싱이 끝나면 이 endDocument( ) 메소드가 내부적으로 호출된다.

XML Parsing

❖ SAX 파서 관련 주요 클래스와 메서드

✓ org.xml.sax.helpers.DefaultHandler 클래스

startElement	• 선언부 : public void startElement(String uri, String localName, String qName, Attributes attributes) • 설명 : XML의 시작 요소를 파싱할 때마다 startElement 메소드가 호출된다. 〈name〉Benjamin〈/name〉 XML 데이터를 파싱한다면 〈name〉 태그를 파싱할 때 startElement( ) 메소드가 호출된다. 이 때 startElement( ) 메소드를 호출하면서 요소의 모든 정보값을 매개변수로 넘겨준다. 매개변수 uri는 XML의 Namespace URI를 의미한다. Namespace는 xmlns라는 속성 이름을 사용하며 〈name xmlns='www.gilbut.co.kr'〉이라는 태그를 파싱하면 uri 매개변수로는 'www.gilbut.co.kr'이라는 값이 넘어온다. localname 매개변수는 접두사를 제외한 노드의 이름이다. 〈name:first〉 요소의 경우 localName은 first이다. 접두사를 포함한 노드의 이름은 qName 매개변수로 넘어온다. 또한 노드의 속성값들은 Attributes 객체로 넘어온다.
endElement	• 선언부 : public void endElement(String uri, String localName, String qName) • 설명 : startElement 메소드와 반대로 XML의 종료 요소를 파싱할 때 endElement( ) 메소드가 호출된다. 〈name〉Benjamin〈/name〉 XML 데이터를 파싱한다면 〈/name〉 태그를 파싱할 때 endElement( ) 메소드가 호출된다.
error	• 선언부 : public void error(SAXParseException e) • 설명 : SAXParser 객체로부터 회복 가능한 에러가 발생하면 error( ) 메소드가 호출된다.

XML Parsing

❖ http://www.khan.co.kr/rss/rssdata/sk_entertainment.xml – 스포츠 경향 연예 RSS

```
▼<rss xmlns:dc="http://purl.org/dc/elements/1.1/" version="2.0">
  ▼<channel>
    <title>스포츠경향:연예</title>
    <link/>
    <description>스포츠경향 RSS 서비스 | 연예</description>
    <lastBuildDate>2021-11-14T09:52:01+09:00</lastBuildDate>
    <copyright>Copyright (C)1996 Kyunghyang Shinmun, All right reserved</copyright>
    <webMaster>webmaster@khan.co.kr</webMaster>
    <language>ko</language>
    <image/>
   ▼<item>
    ▼<title>
       <![CDATA[ 서경석 "1년에 한 번 하는 호텔식사" ]]>
     </title>
     <link>http://sports.khan.co.kr/news/sk_index.html?
     art_id=202111130000013&sec_id=540101&utm_campaign=rss_btn_click&utm_source=khan_rss&utm_mediu
    ▼<description>
       <![CDATA[ 서경석 SNS개그맨 서경석이 근황을 팬들에게 전했다.<br /> <br /> 서경석은 지난 11일 SNS에 서울 시내 호
       #11월11일 1년에 한 번 ··· ]]>
     </description>
     <dc:date>2021-11-13T00:00:01+09:00</dc:date>
    ▼<author>
       <![CDATA[ 손봉석 기자 paulsohn@kyunghyang.com ]]>
     </author>
    ▼<category>
       <![CDATA[ ]]>
     </category>
    </item>
```

XML Parsing

- 스포츠 경향 신문사 실시간 연예 기사 제목 및 링크와 기사 요약 출력

  서경석 “1년에 한 번 하는 호텔식 사”:http://sports.khan.co.kr/news/sk_index.html?art_id=202111130000013&sec_id=540101&utm_campaign=rss_btn_click&utm_source=khan_rss&utm_medium=rss&utm_content=sk_entertainment

  서경석 SNS개그맨 서경석이 근황을 팬들에게 전했다.

  서경석은 지난 11일 SNS에 서울 시내 호텔에서 가족과 함께 한 저녁식사 모습이 담긴 사진을 공개했다.

  서경석은 “#11주년 #결기 #11월11일 1년에 한 번 …

  [종합] ‘지헤중’ 송혜교, 멜로퀸의 귀환… 숨멎 엔딩 8.4%:http://sports.khan.co.kr/news/sk_index.html?art_id=202111130902003&sec_id=540201&utm_campaign=rss_btn_click&utm_source=khan_rss&utm_medium=rss&utm_content=sk_entertainment

  SBS 방송 캡처송혜교와 장기용의 멜로 케미가 폭발했다.

XML Parsing

```
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;

import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import org.xml.sax.helpers.DefaultHandler;
```

XML Parsing

```
class KhanHandler extends DefaultHandler {
        // 파싱할 데이터를 저장할 객체
        List<Map<String, String>> list = new ArrayList<>();

        // 파싱에 사용할 임시 변수
        Map<String, String> map = null;
        String content = null;

        public KhanHandler(List<Map<String, String>> list) {
                this.list = list;
        }

        // 문서의 시작
        public void startDocument() {
                //System.out.println("문서 시작");
        }

        // 문서의 종료
        public void endDocument() {
                //System.out.println("문서 종료");
        }
```

XML Parsing

```
// 시작 태그 인식했을 때 처리
public void startElement(String url, String name, String elementName, Attributes
attrs) throws SAXException {
        if (elementName.equals("item")) {
                map = new HashMap<>();
        }
}

// 시작태그와 끝태그 사이의 내용을 인식 했을 때 처리
public void characters(char[] str, int start, int len) throws SAXException {
        if (content == null) {
                content = new String(str, start, len);
        } else {
                content = content + new String(str, start, len);
        }
}
```

XML Parsing

```
        // 끝 태그를 인식 했을 때 처리
        public void endElement(String url, String localName, String name) {
                if (name.equals("item")) {
                        list.add(map);
                        map = null;
                }
                if (map != null) {
                        if (name.equals("title")) {
                                map.put("title", content);
                        } else if (name.equals("link")) {
                                map.put("link", content);
                        } else if (name.equals("description")) {
                                map.put("description", content);
                        }
                }
                content = null;
        }
}
```

XML Parsing

```
public class KhanParsing {
	public static void main(String[] args) {
		try {
			// 파싱한 결과 저장할 인스턴스
			List<Map<String, String>> list = new ArrayList<>();
			// 파싱 수행할 객체 만들기
			SAXParserFactory parserFactory =
SAXParserFactory.newInstance();
			SAXParser parser = parserFactory.newSAXParser();
			// 파싱 수행을 위임

	parser.parse("http://www.khan.co.kr/rss/rssdata/sk_entertainment.xml", new
KhanHandler(list));
```

XML Parsing

```
                                        //데이터 출력
                                        for(Map<String, String> map : list) {
                                                System.out.println(map.get("title").trim() + ":" +
map.get("link").trim());

                System.out.println(map.get("description").trim().replaceAll("<(/)?([a-zA-
Z]*)(₩₩s[a-zA-Z]*=[^>]*)?(₩₩s)*(/)?>", ""));
                                                System.out.println();
                                        }
                        } catch (Exception e) {
                                        System.out.printf("XML 파싱 예외:%s₩n", e.getMessage());
                                        e.printStackTrace();
                        }
                }
}
```

XML Parsing

- StAX
  - ✓ StAX(Streaming API for XML)는 자바 6부터 표준으로 통합된 API
  - ✓ StAX는 이벤트 구동형으로 처리를 수행한다는 점에서 SAX와 비슷하지만 SAX는 푸시형, StAX는 풀형이라는 차이가 있는데 '푸시'란 API 쪽에서 코드를 호출하는 것을 말하고, '풀'이란 이벤트를 끌어내는 동작
  - ✓ 파서가 이벤트를 던지는 것이 아니라 프로그램이 이벤트가 있는지의 여부를 판단하여 처리
  - ✓ 이벤트가 있으면 파서에서 취득한다는(=이벤트를 끌어내는 움직임) 점에서 풀형
  - ✓ SAX는 구문 분석이 시작되면 코드 측에서 제어할 수 없지만 이에 반해 StAX는 구문 분석의 제어가 가능하기 때문에 '구문 분석의 중간에 중지한다' 등의 처리도 실현 가능
  - ✓ StAX는 'Cursor API'와 'Event Iterator API'라는 두 종류 의 API가 준비되어 있으며 각각 XML 파싱의 방법이 다름

XML Parsing

- XPath
  - ✓ XPath(Xml Path language)는 XML 파일 중에서 조건에 맞는 부분만을 추출하는 형태로 처리하는 API
  - ✓ XML 파일 안의 특정 부분을 나타내는 문자열(위치 경로)을 지정함으로써 조건에 맞는 요소를 직접 가져올 수 있음
  - ✓ 미리 XML 파일의 구조를 알고 있으며, 파일 안의 일부 값을 추출하려는 경우에 유용한 API
  - ✓ XPath에서는 태그의 위치를 '/'로 구분
  - ✓ '@'를 사용하여 속성을 지정하거나 'text()'를 사용하여 태그로 둘러싸인 문자열을 취득하거나 대괄호([])를 사용하여 조건을 지정할 수도 있음

XML Parsing

- JAXB
  - ✓ JAXB(Java Architecture for XML Binding)는 지금까지 소개해온 API와는 조금 다른데 XML 파일을 읽어들이기보다 XML 파일과 자바의 객체를 연결하는 동작을 수행
  - ✓ JAXB를 이용함으로 써 XML과 자바 객체를 상호 변환할 수 있음
  - ✓ XML Schema를 읽어서 자바 오브젝트로 만드는 언마샬링(Unmarshaling)과 반대로 자바 오브젝트를 XML로 변환하는 마샬링(Marshalling)을 가능하도록 해주는 API

XML Parsing

- pom.xml 파일에 의존성 추가

```
<dependency>
        <groupId>javax.xml.bind</groupId>
        <artifactId>jaxb-api</artifactId>
        <version>2.1</version>
</dependency>
<dependency>
        <groupId>com.sun.xml.bind</groupId>
        <artifactId>jaxb-impl</artifactId>
        <version>2.2.11</version>
</dependency>

<dependency>
        <groupId>com.sun.xml.bind</groupId>
        <artifactId>jaxb-core</artifactId>
        <version>2.2.11</version>
</dependency>
```

XML Parsing

❖ Person 클래스 생성하고 작성

```
import javax.xml.bind.annotation.XmlAccessType;
import javax.xml.bind.annotation.XmlAccessorType;
import javax.xml.bind.annotation.XmlType;

@XmlAccessorType(XmlAccessType.FIELD)
@XmlType(name = "", propOrder = { "name", "email" })
public class Person {
    private String name;
    private String email;

        public Person() {
                    super();
        }

        public Person(String name, String email) {
                    super();
                    this.name = name;
                    this.email = email;
        }
```

XML Parsing

❖ Person 클래스 생성하고 작성

```
        public String getName() {
                return name;
        }
        public void setName(String name) {
                this.name = name;
        }
        public String getEmail() {
                return email;
        }
        public void setEmail(String email) {
                this.email = email;
        }

        @Override
        public String toString() {
                return "Person [name=" + name + ", email=" + email + "]";
        }
}
```

XML Parsing

- PersonReport 클래스 생성하고 작성

```
import java.util.List;
import javax.xml.bind.annotation.*;

@XmlAccessorType(XmlAccessType.FIELD)
@XmlRootElement(name = "PersonList")
public class PersonReport {
	@XmlElement(name = "Person")
	private List<Person> list;

	public PersonReport() {
	}
	public PersonReport(List<Person> list) {
		this.list = list;
	}

	public List<Person> getList() {
		return list;
	}
}
```

XML Parsing

- PersonMarshalling 클래스 생성하고 작성한 후 실행

```
import java.util.ArrayList;
import java.util.List;

import javax.xml.bind.JAXBContext;
import javax.xml.bind.Marshaller;

public class PersonMarshalling {

    public void marshalTest() {
        try {
            Person person1 = new Person("cyberadam", "ggangpae1@gmail.com");
            Person person2 = new Person("군계", "itstudy@kakao.com");
            List<Person> list = new ArrayList<>();
            list.add(person1);
            list.add(person2);

            PersonReport report = new PersonReport(list);

            //JAXBContext 생성 & marshaller 생성
            JAXBContext context = JAXBContext.newInstance(PersonReport.class);
            Marshaller marshaller = context.createMarshaller();
            //보기 좋게 출력해주는 옵션
            marshaller.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
```

XML Parsing

- PersonMarshalling 클래스 생성하고 작성한 후 실행

```
            //표준 출력으로 결과를 보여준다.
            marshaller.marshal(report, System.out);
        } catch (Exception e) {
            System.out.println(e.getLocalizedMessage());
        }
    }

    public static void main(String[] args) {
        new PersonMarshalling().marshalTest();
    }
}
```

XML Parsing

- person.xml 파일을 생성하고 작성

```
<?xml version="1.0" encoding="UTF-8"?>
<PersonList>
    <Person>
        <name>cyberadam</name>
        <email>ggangpae1@gmail.com</email>
    </Person>
    <Person>
        <name>군계</name>
        <email>itstudy@kakao.com</email>
    </Person>
</PersonList>
```

XML Parsing

- PersonUnmarshalling 클래스 생성하고 작성한 후 실행

```
import java.io.File;
import java.util.List;

import javax.xml.bind.JAXBContext;
import javax.xml.bind.Unmarshaller;

public class PersonUnmarshalling {
	public static void main(String[] args) {
		try {
			// XML 파일을 읽어온다.
			File file = new File("person.xml");

			// JAXBContext 생성 & unmarshaller 생성
			JAXBContext context =
JAXBContext.newInstance(PersonReport.class);
			Unmarshaller unmarshaller = context.createUnmarshaller();

			// XML을 Person 오브젝트로 변환한다.
			PersonReport personReport =
(PersonReport)unmarshaller.unmarshal(file);
			List<Person> list = personReport.getList();
```

XML Parsing

❖ PersonUnmarshalling 클래스 생성하고 작성한 후 실행

```
                    for(Person person : list) {
                        System.out.println(person);
                    }
                } catch (Exception e) {
                    // ... handle exception
                }
            }
        }
```

HTML Parsing

HTML Parsing

- HTML 태그를 지칭하는 방법
  - ✓ Tag: 브라우저가 해석해서 화면 랜더링을 하기 위한 HTML의 규칙으로 하나의 파일에서 중복될 수 있음
  - ✓ Class: 동일한 디자인을 적용하기 위해서 Tag들을 그룹화 한 것으로 CSS에서 많이 사용
  - ✓ ID: JavaScript에서 하나의 태그를 구분하기 위해 설정하는 것으로 하나의 파일에서 유일 무이함
  - ✓ Name: Client의 내용을 Server에서 사용하기 위해서 만든 이름으로 중복될 수 있음
  - ✓ Xpath(XML Path Language): XML 문서의 특정 요소나 속성에 접근하기 위한 경로를 지정하는 것으로 중복될 수 없음

HTML Parsing

- HTML Parsing
  - ✓ 안드로이드나 모바일 프로그래밍을 하다보면 웹 화면에서는 보이지만 Oepn API를 제공하지 않는 사이트의 데이터를 사용해야 하는 경우가 있는데 이러한 경우에는 HTML을 읽어서 원하는 데이터를 추출
  - ✓ HTML Parsing을 수행해주는 라이브러리는 여러 가지 존재
  - ✓ jsoup 라이브러리
    - ❑ 파이썬의 BeautifulSoup와 유사한 형태의 라이브러리
    - ❑ jsoup은 HTML 문서를 읽어들인 후에 그 문서를 DOM 객체로 변환
    - ❑ jsoup의 selector api를 이용해서 특정 Element에 접근을 할 수 있고 해당 Element의 정보를 읽거나 수정할 수 있음
    - ❑ jsoup은 jquery의 selector와 비슷한 selector api를 제공하기 때문에 쉽게 사용 가능

Jsoup

- Jsoup
  - ✓ HTML형식의 string을 넘겨주면 자바에서 사용할 수 있는 DOM 객체로 만들어 주는 parser의 역할
  - ✓ 웹 페이지를 읽어 들이는 기능까지 하는 라이브러리는 아님
  - ✓ Jsoup.parse(String url, int timeoutMillisecons) api를 이용하면 URL로 부터 웹 페이지를 읽어와서 DOM 객체로 변환해 주긴 하지만 jsoup은 네트워크 라이브러리는 아니여서 해당 api를 사용하는 것은 비추천
  - ✓ URLConnection 같은 네트워크 전용 API로 HTML을 읽어온 다음에 읽어온 String을 jsoup으로 변환시키는 방법을 사용하는 것이 바람직
  - ✓ 읽어온 HTML을 jsoup document 객체로 변환시키는 방법

    String html = ...(html 문서)

    Document doc = Jsoup.parse(html);

Jsoup

- 전통적인 방식의 Navigating api
  - ✓ getElementById(), getElementsByTag(), getElementsByClass(), getElementsByAttribute() 와 같은 메서드를 이용하여 특정 Element를 조회
  - ✓ getElementById 만 리턴 타입이 Element 이고 나머지는 Elements
  - ✓ id는 하나의 Element만 가지고 있는 고유 속성이기 때문에 단일 Element를 리턴
  - ✓ Elements는 List 인터페이스를 구현했기 때문에 일반 리스트 처럼 사용 가능
  - ✓ Elements는 List이긴 하지만 jquery 객체처럼 Element에 있는 대부분의 메서드를 가지고 있고 해당 메서드를 호출하면 Elements 내부에 있는 모든 Element에서 해당 메서드를 호출

Jsoup

- jquery 방식의 Navigating api
  - ✓ jquery처럼 $('.class#id') 와 비슷하게 사용할 수 있는 메서드가 jsoup에 존재하는데 select 메서드

    ```
    Document doc = Jsoup.parse(html);
    Elements elements = doc.select(".class #id");
    ```

  - ✓ select 메서드는 string 인자를 하나를 받고, 해당 인자는 jquery에서 사용되는 selector와 비슷한 css query이고 리턴 값은 Elements
  - ✓ Selector에 id 값을 넣어도 애초에 select 메서드의 리턴 타입은 Elements 이므로 Elements를 반환
  - ✓ Element를 1개 가져오고 싶다면 elements.first()
  - ✓ select 메서드는 getElementById와 getElementByTag와 getElementByClass 메서드를 여러번 호출해야 하는 일을 한번에 할 수 있게 해줌
- Element의 메서드
  - ✓ 태그 사이의 문자열을 리턴받고자 하는 경우는 text()
  - ✓ 태그 안의 속성 값을 리턴받고자 하는 경우는 attr(String attribute)
  - ✓ Element를 찾는 메서드 사용 가능

한겨레 웹 사이트의 메인 기사

- pom.xml 파일에 jsoup 라이브러리 사용을 위한 의존성을 추가

```
<dependency>
        <groupId>org.jsoup</groupId>
        <artifactId>jsoup</artifactId>
        <version>1.11.3</version>
</dependency>
```

한겨레 웹 사이트의 메인 기사

- HaniHTMLParsingMain

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;

import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;
```

한겨레 웹 사이트의 메인 기사

❖ HaniHTMLParsingMain

```
public class HaniHTMLParsing {
        public static void main(String[] args) {
                //hani.co.kr의 html 받아오기
                String html = "";
                try {
                        String addr = "https://www.hani.co.kr/";
                        URL url = new URL(addr);
                        HttpURLConnection con =
                                (HttpURLConnection)
                                        url.openConnection();
                        con.setConnectTimeout(30000);

                        BufferedReader br =
                                new BufferedReader(
                                        new InputStreamReader(
                                                con.getInputStream()));
                        StringBuilder sb = new StringBuilder();
                        while(true) {
                                String line = br.readLine();
                                if(line == null) {
                                        break;
                                }
                                sb.append(line + "₩n");
                        }
```

한겨레 웹 사이트의 메인 기사

- HaniHTMLParsingMain

```
                    html = sb.toString();
                    br.close();
                    con.disconnect();

                    //System.out.printf("%s₩n", html);


          }catch(Exception e) {
                    System.out.printf(
                              "다운로드 예외:%s₩n", e.getMessage());
          }
```

한겨레 웹 사이트의 메인 기사

❖ HaniHTMLParsingMain

```
//HTML 파싱을 수행하는 부분
try {
        //HTML 문자열을 파싱할 수 있도록 DOM으로 생성
        Document document =
                Jsoup.parse(html);
        //태그로 찾아오기
        /*
        Elements elements =
                document.getElementsByTag("a");
        for(int i=0; i<elements.size(); i=i+1) {
                Element element = elements.get(i);
                System.out.printf("%s₩n",
                        element.text());
        }
        */
```

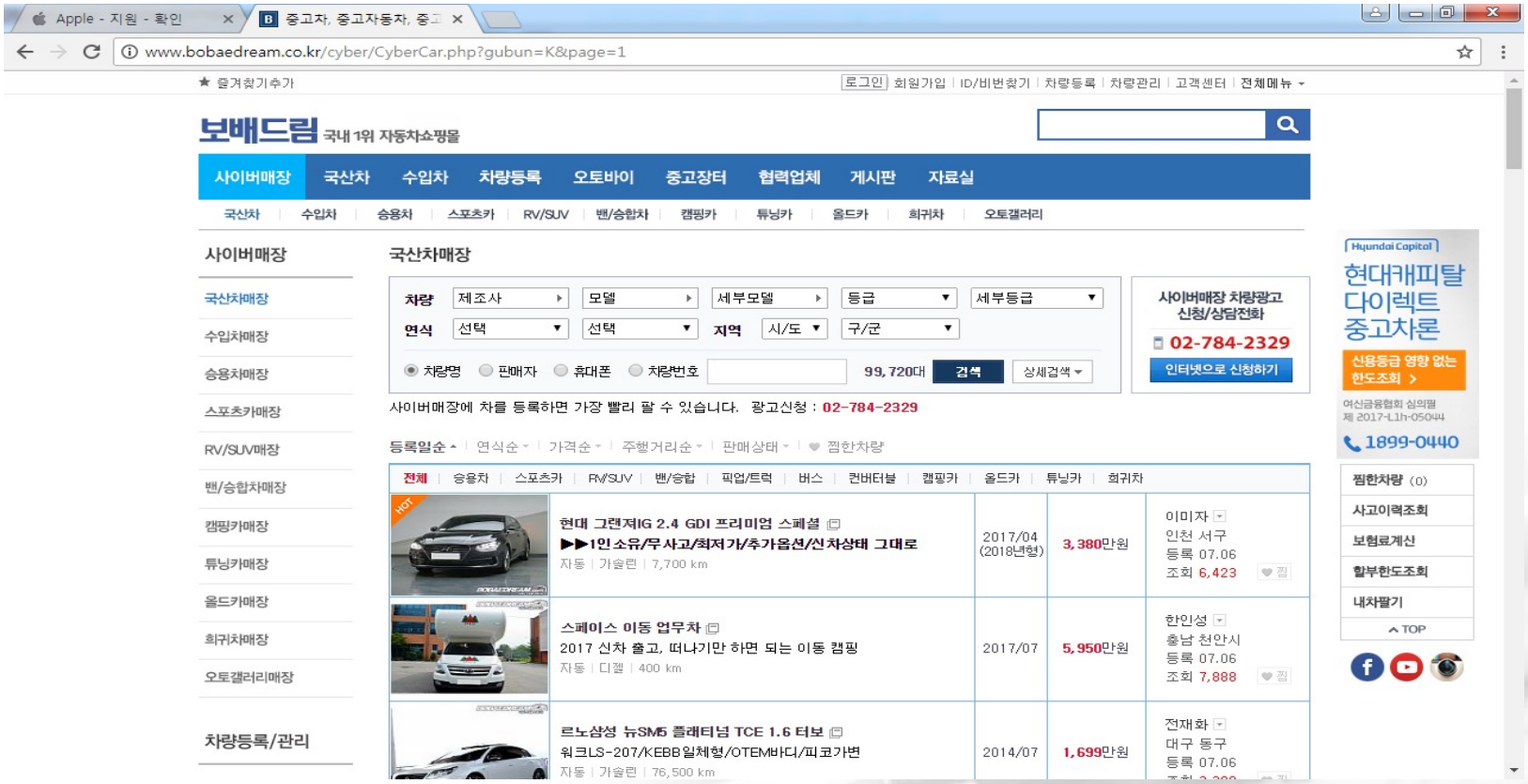
한겨레 웹 사이트의 메인 기사

❖ HaniHTMLParsingMain

```
                    Elements elements =
                            document.select(
                            "#main-top > div.main-top > div.main-top-article >
h4 > a");
                    for(int i=0; i<elements.size(); i=i+1) {
                            Element element =
                                    elements.get(i);
                            //태그 안의 문자열을 찾아오기
                            System.out.printf("%s₩n",
                                    element.text());
                            //태그 내의 속성 값을 찾아오기
                            System.out.printf("%s₩n",
                                    element.attr("href"));
                    }

            }catch(Exception e) {
                    System.out.printf(
                            "HTML 파싱 예외:%s₩n", e.getMessage());
            }

        }

}
```

보배드림

- https://www.bobaedream.co.kr/cyber/CyberCar.php?gubun=K&page=1



보배드림

❖ 보배드림 페이지에서 자동차 정보 가져오기

{price=9,600, title=현대 싼타페TM 2.0 T-GDi 2WD 인스퍼레이션}
{price=3,500, title=기아 올 뉴 K3 1.6 T-GDI GT 플러스}
{price=2,150, title=현대 싼타페DM 2.0 VGT 2WD 익스클루시브 스페셜}
{price=1,550, title=현대 올 뉴 마이티 2.5톤 슈퍼캡 장축고상}
{price=4,100, title=기아 봉고3 1톤 더블캡 초장축}
{price=1,380, title=르노삼성 뉴 SM5 LE}
{price=480, title=캠핑카마스터 하이루프 캠핑카}
{price=5,000, title=제네시스 더 올 뉴 G80 3.5 T-GDi AWD 스포츠 팩}
{price=6,890, title=현대 e-카운티 캠핑카}
{price=4,500, title=현대 제네시스 BH380 로얄 VIP 팩}
{price=1,590, title=현대 제네시스 BH460}
{price=999, title=제네시스 G80 3.3 T-GDi AWD 스포츠}
{price=3,700, title=현대 뉴 제네시스 G330 모던 AWD}
{price=1,890, title=현대 제네시스 쿠페 380 GT P}

보배드림

❖ BobaeHTMLParsing

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import java.util.ArrayList;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;

import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;
```

보배드림

❖ BobaeHTMLParsing

```
public class BobaeHTMLParsing {
        public static void main(String[] args) {
                StringBuilder sBuffer = new StringBuilder();
                String html = "";
                try {
                        // 데이터를 다운로드 받을 주소 생성
                        String urlAddr =
"https://www.bobaedream.co.kr/cyber/CyberCar.php?gubun=K&page=1";

                        URL url = new URL(urlAddr);
                        // 주소와 연결
                        HttpURLConnection conn = (HttpURLConnection)
url.openConnection();
```

보배드림

❖ BobaeHTMLParsing

```
                    if (conn != null) {
                        conn.setConnectTimeout(20000);
                        conn.setUseCaches(false);

                        InputStreamReader isr = new
InputStreamReader(conn.getInputStream());
                        BufferedReader br = new BufferedReader(isr);
                        // 줄 단위로 읽어서 sBuffer에 저장
                        while (true) {
                            String line = br.readLine();
                            if (line == null) {
                                break;
                            }
                            sBuffer.append(line);
                        }
                        br.close();
                        conn.disconnect();
                    }
                    // 전부 읽었으면 String으로 변환
                    html = sBuffer.toString();
                } catch (Exception e) {
                    System.out.println("다운로드 중 에러 발생");
                }
```

보배드림

❖ BobaeHTMLParsing

```
// html 문자열을 메모리에 펼침
Document doc = Jsoup.parse(html);
// 펼쳐진 데이터에서 클래스 이름이 carinfo 인 데이터 찾아오기
Elements root = doc.getElementsByClass("mode-cell title");
//System.out.println(root);

// 클래스가 price 인 데이터 찾아오기
Elements prices = doc.getElementsByClass("price");
//System.out.println(prices);
```

보배드림

❖ BobaeHTMLParsing

```
// Parsing한 결과를 저장할 리스트 생성
List<Map<String, String>> list = new ArrayList<Map<String, String>>();
Map<String, String> map = null;
for (int i = 1; i < root.size(); i++) {
        Element element = root.get(i);
        // carinfo 클래스 안에 있는 요소 중에서 클래스가 title 인 데이터
전부 찾아오기
        Elements titles = element.getElementsByTag("a");
        // price를 순회하면서 em 태그의 데이터 찾아오기
        Element price = prices.get(i+1);
        try {
                Elements strong = price.getElementsByTag("strong");

                // 하나의 데이터를 저장할 맵을 생성
                map = new LinkedHashMap<String, String>();
                // title 클래스의 첫번째 데이터의 문자열을 title이라는
키에 저장

                map.put("title", titles.get(0).text());
                map.put("price", strong.get(0).text());
        }catch(Exception e) {
                map.put("price", "계약");
        }
        list.add(map);
}
```

보배드림

❖ BobaeHTMLParsing

```
                for (Map<String, String> temp : list) {
                        System.out.println(temp);
                }
        }
}
```

selenium

- Selenium
  - ✓ 웹 앱을 테스트하는데 이용하는 프레임워크
  - ✓ Webdriver라는 API를 통해 운영체제에 설치된 브라우저를 제어
  - ✓ 브라우저를 동작시켜서 JavaScript를 이용해 비동기적으로 가져온 데이터나 프레임에 속한 콘텐츠 들을 가져올 수 있음
  - ✓ Selenium은 실제 웹 브라우저가 동작하기 때문에 JS로 렌더링이 완료된 후의 DOM 결과물에 접근이 가능
  - ✓ Selenium 설치
    - ❑ 직접 다운로드 받아서 설치
      - ❑ https://www.seleniumhq.org/ 에서 java 용 jar 파일을 다운로드
    - ❑ Maven 이용

```
<dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>3.141.59</version>
</dependency>
```

selenium

- Selenium
  - ✓ 크롬 드라이버 다운로드 한 후 압축 해제

    https://chromedriver.chromium.org/downloads

크롬을 이용한 naver 접속

❖ naver 접속하기

```
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;

class SeleniumTest{
        // WebDriver
        private WebDriver driver;
        // Properties
        public static final String WEB_DRIVER_ID = "webdriver.chrome.driver";
        //자신의 크롬 드라이버 위치로 설정
        public static final String WEB_DRIVER_PATH =
"/Users/adam/Documents/chromedriver";
        // 크롤링 할 URL
        private String base_url;
        public SeleniumTest() {
        super();
        //System Property SetUp
        System.setProperty(WEB_DRIVER_ID, WEB_DRIVER_PATH);

        //Driver SetUp
        driver = new ChromeDriver();
        base_url = "https://www.naver.com";
    }
```

크롬을 이용한 naver 접속

❖ naver 접속하기

```
        public void crawl() {
                try {
                        // get page (= 브라우저에서 url을 주소창에 넣은 후 request 한 것
과 같다)
                        driver.get(base_url);
                        System.out.println(driver.getPageSource());

                } catch (Exception e) {
                        e.printStackTrace();
                } finally {
                        //driver.close();
                }
        }
}
```

크롬을 이용한 naver 접속

❖ naver 접속하기

```
public class SeleniumMain {
        public static void main(String[] args) {
                SeleniumTest selTest = new SeleniumTest();
                selTest.crawl();
        }
}
```

selenium

- 메서드
  - ✓ get(url): url에 접속
  - ✓ 페이지의 element에 접근하는 api
    - ❑ WebElement driver.findElement(By.id 또는 xpath(id 또는 xpath))
    - ❑ WebElement driver.findElements(By.tagName 또는 name 또는 class(tag 또는 name 이나 class 이름))
  - ✓ Element를 찾아서 sendKeys('값')을 호출하면 선택한 Element에 값을 입력
  - ✓ Element를 찾아서 click() 이나 submit()을 호출하면 버튼을 누른 것과 동일한 효과 수행
  - ✓ close(): 브라우저 종료
  - ✓ switchTo().frame(프레임 Element) : 프레임 전환
  - ✓ 크롬 창을 출력하지 않고 가져오기

```
ChromeOptions options = new ChromeOptions();
options.addArguments("headless");
//Driver SetUp
driver = new ChromeDriver(options);
```

selenium

- 프레임 접근
  - ✓ Selenium 을 활용하여 크롤링 하다보면 눈에는 원하는 정보가 보이고 크롬의 개발자 도구에서도 원하는 부분이 다 보이는데 파이썬으로 접근이 안되는 경우는 대부분 프레임 구조로 만들어 져 있는 경우
  - ✓ 소스 코드로 frame 확인

```
# 프레임 살펴보기
iframes = driver.find_elements_by_css_selector('iframe')
for iframe in iframes:
    print(iframe.get_attribute('name'))
```

  - ✓ 프레임으로의 전환

```
driver.switchTo().frame(프레임)
```

selenium

❖ 다음 카페에 접속해서 댓글 작성하기

```
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;

class SeleniumDaum {
        // WebDriver
        private WebDriver driver;
        // Properties
        public static final String WEB_DRIVER_ID = "webdriver.chrome.driver";
        public static final String WEB_DRIVER_PATH =
"/Users/adam/Documents/chromedriver";

        public SeleniumDaum() {
                super();
                // System Property SetUp
                System.setProperty(WEB_DRIVER_ID, WEB_DRIVER_PATH);

        ChromeOptions options = new ChromeOptions();
        options.addArguments("headless");
        //Driver SetUp
        driver = new ChromeDriver();
        }
```

selenium

❖ 다음 카페에 접속해서 댓글 작성하기

```
public void crawl() {
        try {

    driver.get("https://logins.daum.net/accounts/signinform.do?url=https%3A%2F%2Fwww.daum.net%2F");
                Thread.sleep(30000);

    driver.findElement(By.xpath("//*[@id=₩"id₩"]")).sendKeys("ggangpae3");

    driver.findElement(By.xpath("//*[@id=₩"inputPwd₩"]")).sendKeys("cyberadam");
                driver.findElement(By.xpath("//*[@id=₩"loginBtn₩"]")).click();
                Thread.sleep(30000);
                driver.get("http://cafe.daum.net/samhak7/_memo");
                Thread.sleep(3000);
                //프레임이 있는 경우 프레임으로 이동
                driver.switchTo().frame(driver.findElement(By.id("down"

    driver.findElement(By.xpath("//*[@id=₩"primaryContent₩"]/div/div[1]/div[2]/div/div/div[1]/textarea")).sendKeys("더조은에서 연습");
                Thread.sleep(3000);

    driver.findElement(By.xpath("//*[@id=₩"primaryContent₩"]/div/div[1]/div[2]/div/div/div[2]/div[2]/div/button")).click();
```

selenium

❖ 다음 카페에 접속해서 댓글 작성하기

```
                        } catch (Exception e) {
                                e.printStackTrace();
                        } finally {
                                // driver.close();
                        }
                }
        }
```

selenium

❖ 다음 카페에 접속해서 댓글 작성하기

```
public class DaumCafeMacro {
        public static void main(String[] args) {
                SeleniumDaum selTest = new SeleniumDaum();
                selTest.crawl();
        }
}
```

selenium

- ❖ 자바스크립트 코드 실행
  - ✓ 일반적인 형태로 로그인을 시도하는 경우에 로봇 입력화면이 출력될 수 있는데 이런 경우에는 자바스크립트 코드를 이용해서 직접 로그인을 시도하면 됨

    JavascriptExecutor 자바스크립트실행변수 = (JavascriptExecutor)WebDriver변수;

    자바스크립트실행변수.executeScript(" 자바스크립트 코드", "매개변수")

selenium

❖ 네이버 중고나라 카페 게시물에 댓글 달기

```
import org.openqa.selenium.By;
import org.openqa.selenium.JavascriptExecutor;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;

public class NaverCafeMacro {

        public static void main(String[] args) {
                try {
                // Properties
                final String WEB_DRIVER_ID = "webdriver.chrome.driver";
                final String WEB_DRIVER_PATH = "/Users/adam/Documents/chromedriver";

                // System Property SetUp
                System.setProperty(WEB_DRIVER_ID, WEB_DRIVER_PATH);

                ChromeOptions options = new ChromeOptions();
                options.addArguments("headless");
                // WebDriver
                WebDriver driver = new ChromeDriver();
```

selenium

❖ 네이버 중고나라 카페 게시물에 댓글 달기

```
String id = "네이버아이디";
String password = "네이버비밀번호";

driver.get("https://nid.naver.com/nidlogin.login");
Thread.sleep(10000);

JavascriptExecutor js = (JavascriptExecutor)driver;
js.executeScript("document.getElementsByName('id')[0].value=₩'" + id +
"₩'");
js.executeScript("document.getElementsByName('pw')[0].value=₩'" +
password + "₩'");

driver.findElement(By.xpath("//*[@id=₩"log.login₩"]")).click();

Thread.sleep(10000);
driver.get("https://cafe.naver.com/joonggonara");


Thread.sleep(10000);
driver.get("https://cafe.naver.com/joonggonara/617588810");
Thread.sleep(3000);
driver.switchTo().frame(driver.findElement(By.id("cafe_main")));
```

selenium

❖ 네이버 중고나라 카페 게시물에 댓글 달기

```
                String reply = "매크로를 이용한 댓글";
                Thread.sleep(3000);

        driver.findElement(By.xpath("//*[@id=₩"app₩"]/div/div/div[2]/div[2]/div[5]/div[2]/
div[1]/textarea")).sendKeys(reply);
                Thread.sleep(3000);

        driver.findElement(By.xpath("//*[@id=₩"app₩"]/div/div/div[2]/div[2]/div[5]/div[2]/
div[2]/div[2]/a")).click();
                }catch(Exception e) {
                        System.out.println("댓글 작성 에러");
                        System.out.println(e.getMessage());
                        e.printStackTrace();
                }

        }

}
```

selenium

❖ 유투브 스크롤

```
import org.openqa.selenium.By;
import org.openqa.selenium.Keys;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;
import org.openqa.selenium.chrome.ChromeOptions;

public class YoutubeScroll {
	public static void main(String[] args) {
		try {
			// Properties
			final String WEB_DRIVER_ID = "webdriver.chrome.driver";
			final String WEB_DRIVER_PATH =
"/Users/adam/Documents/chromedriver";

			// System Property SetUp
			System.setProperty(WEB_DRIVER_ID, WEB_DRIVER_PATH);

			ChromeOptions options = new ChromeOptions();
			options.addArguments("headless");
			// WebDriver
			WebDriver driver = new ChromeDriver();
```

selenium

❖ 유투브 스크롤

```
        driver.get("https://www.youtube.com/results?search_query=%EB%A7%88%EB%A7%8
    8%EB%AC%B4");
                                Thread.sleep(5000);
                                WebElement body = driver.findElement(By.tagName("body"));

                                int num_of_pagedowns = 20;
                                for (int i = 0; i < num_of_pagedowns; i = i + 1) {
                                        body.sendKeys(Keys.PAGE_DOWN);
                                        Thread.sleep(3000);
                                }

                                String html = driver.getPageSource();
                                System.out.println(html);

                        } catch (Exception e) {
                                System.out.println("유투브 스크롤 실패");
                                System.out.println(e.getMessage());
                                e.printStackTrace();
                        }
                }
        }
```

암호화

- 암호화
  - ✓ 보안을 위해서 데이터를 암호화 하는 경우가 있음
  - ✓ 암호화된 데이터를 원래대로 복원하는 작업이 복호화
  - ✓ 암호화 방식은 복호화 가능 여부에 따라 복호하는 불가능하고 비교만 가능한 방식과 복호화가 가능한 방식 2가지
  - ✓ 암호화는 많은 자바라이브러리들이 기능을 제공

암호화

- 복호화가 불가능하고 비교만 가능한 암호화 방식
  - ✓ jbcrypt
    - ❑ mavenrespository.com에서 jbcrypt를 검색해서 의존성 추가

```
<dependency>
    <groupId>org.mindrot</groupId>
    <artifactId>jbcrypt</artifactId>
    <version>0.4</version>
</dependency>
```

    - ❑ 암호화

      String 변경되서 저장되는 문자열= BCrypt.hashpw(변경할 문자열, BCrypt.gensalt());

    - ❑ 암호화된 데이터 체크

```
boolean isValidPassword = BCrypt.checkpw(비교할 문자열1, 비교할 문자열2);
```

암호화

- pom.xml 파일에 추가

```
<dependency>
        <groupId>org.mindrot</groupId>
        <artifactId>jbcrypt</artifactId>
        <version>0.4</version>
</dependency>
```

암호화

❖ main 메서드를 소유한 클래스를 만들고 작성

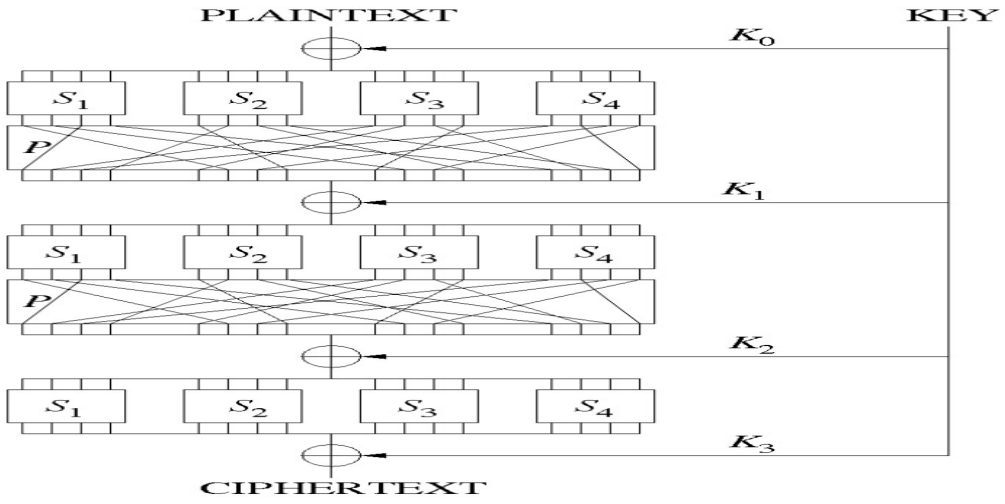
```
import org.mindrot.jbcrypt.BCrypt;

public class JbcryptMain {

	public static void main(String[] args) {
		String encryptString= BCrypt.hashpw("cyberadam", BCrypt.gensalt());
		System.out.printf("암호화된 문자열:%s₩n", encryptString);
		Boolean result = BCrypt.checkpw("cyberadam", encryptString);
		if(result) {
			System.out.printf("두 개의 문자열은 일치합니다.₩n");
		}else {
			System.out.printf("두 개의 문자열은 일치하지 않습니다.₩n");
		}
	}
}
```

암호화

- 복호화가 가능한 암호화 방식
  - ✓ 자바에서는 대칭키 알고리즘을 사용하여 데이터를 암호화/복호화할 때 javax.crypto.Cipher 클래스를 사용
  - ✓ 이 클래스의 인스턴스는 정적 메서드인 Cipher.getInstance()를 호출하여 가져올 수 있는데, 호출 시 사용할 알고리즘, 운용 모드, 패딩 방식을 인자로 넘겨주어야 함
  - ✓ 혼돈과 확산
    - ❑ 안전한 암호문은 공격자가 이를 보고 원본 메시지나 암호화에 사용된 키를 유추할 수 없어야 함
    - ❑ 혼돈(confusion)은 암호문으로부터 키를 알아낼 수 없게 하는 성질
    - ❑ 확산(diffusion)은 암호문으로부터 원문을 알아낼 수 없게 하는 성질
    - ❑ 혼돈이란 키의 비트 하나만 바꿔도 암호문 전체가 바뀌도록 하는 성질로 안전한 암호는 공격자가 아무리 많은 평문-암호문 쌍을 알고 있어도 그 속에서 키의 패턴을 발견할 수 없도록 해야 하며 확산은 원문의 비트를 하나만 바꿔도 암호문 전체가 바뀌도록 하는 성질로 서로 다른 원문이 비슷한 내용을 담고 있더라도 각각의 암호문은 완전히 다른 값을 가져야 한다는 것

암호화

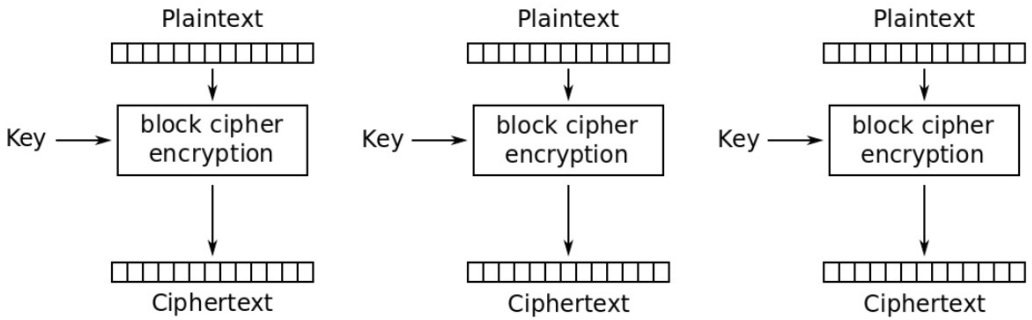
- 복호화가 가능한 암호화 방식
  - ✓ 암호 알고리즘
    - 암호 알고리즘에서는 혼돈과 확산을 달성하기 위해 Substitution과 Permutation을 이용
    - Substitution은 문자를 다른 문자로 바꾸는 것이고, Permutation은 문자들의 순서를 바꾸는 것
    - Substitution과 Permutation을 한 번 수행하는 것이 암호 알고리즘의 기본 수행 단위
    - Substitution-Permutation을 연속하여 수행하도록 이어 놓은 것이 SPN(Substitution Permutation Network)
    - SPN에서 한 번의 Substitution-Permutation 수행을 라운드 또는 레이어



Substitution Permutation Network

암호화

- 복호화가 가능한 암호화 방식
  - ✓ 운용 모드: 데이터를 블록으로 나누어 처리하고 합치는 것, 그것이 암호화에서 운용 모드가 하는 역할이며 대표적인 운용 모드로는 ECB(Electronic Code Book)와 CBC(Cipher Block Chaining)가 있음
    - ❑ ECB
      - ○ ECB는 단순히 블록 단위로 처리한 결과를 이어붙이는 방법

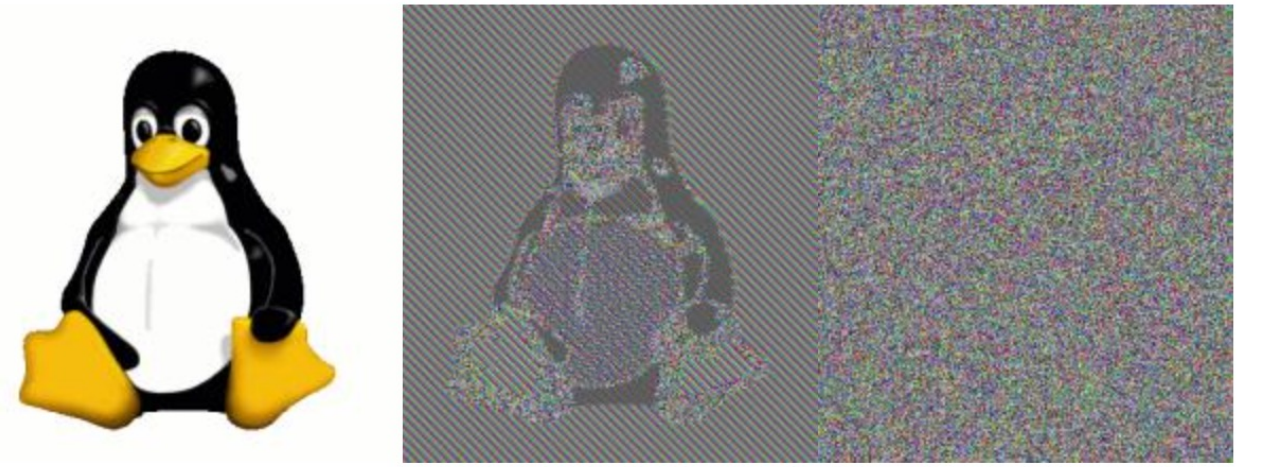


Electronic Codebook (ECB) mode encryption

Electronic Code Book

암호화

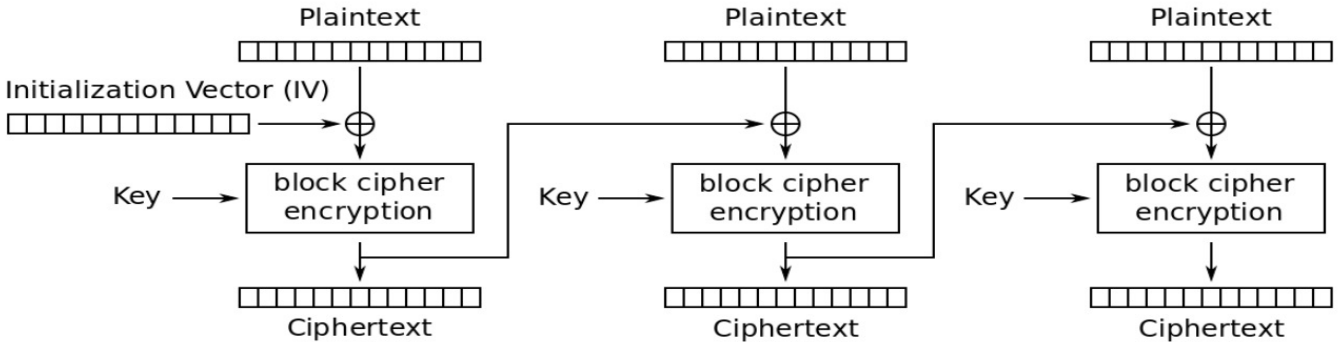
- 복호화가 가능한 암호화 방식
  - ✓ 운용 모드
    - ❑ ECB
      - ○ ECB는 단순하지만 같은 값을 갖는 원문 블록은 같은 암호 블록을 출력하기 때문에 원문의 패턴이 그대로 드러남
      - ○ 암호문에서 원문을 유추할 수 있음을 의미하기 때문에 ECB는 확산의 성질을 달성하지 못함



원본 ECB 안전한 모드

암호화

- ❖ 복호화가 가능한 암호화 방식
  - ✓ 운용 모드
    - ❑ CBC
      - ○ 원문 블록을 그대로 암호화하지 않고 직전에 암호화된 블록과 XOR 연산을 한 다음에 암호화를 수행하는 방식으로 같은 내용을 갖는 원문 블록이라도 전혀 다른 암호문을 갖음
      - ○ 첫번째 블록은 직전 암호문이 없으므로 XOR 연산 대상이 없는데 이를 위해 CBC 모드는 초기화 벡터(initialization vector)를 입력받아야 하며 초기화 벡터는 원문 블록을 XOR하는데 쓰이기 때문에 당연히 블록 사이즈와 동일한 크기여야 함



Cipher Block Chaining (CBC) mode encryption

Cipher Block Chaining

암호화

- 복호화가 가능한 암호화 방식
  - ✓ 패딩
    - AES와 같은 16바이트 크기의 블록 암호 알고리즘을 사용하는데 원문의 크기가 16바이트의 배수가 아니라면 마지막 블록은 16바이트보다 작은 크기가 되는데 이 때 마지막 블록의 빈 부분을 채워주는 방식을 패딩이라고 함
    - 원문의 길이가 L 바이트이면 마지막 블록은 L mod 8의 크기를 갖게 되는데 그럼 패딩 크기는 8 – (L mod 8)가 되므로 자바에서 사용하는 PKCS5는 단순히 패딩 크기의 값을 갖는 바이트를 크기만큼 반복

8 - (L mod 8)	패딩 바이트
1	01
2	02 02
…	…
8	08 08 08 08 08 08 08 08

암호화

- 복호화가 가능한 암호화 방식
  - ✓ CryptoUtil.java
    - ❑ CBC 방식으로 암호화 하도록 만든 클래스
    - ❑ encrypt 메서드를 이용해서 평문을 암호화 할 수 있음
    - ❑ decrypt 메서드를 이용해서 암호화된 문장으로부터 평문을 리턴 받음

암호화

- 암호화 와 복호화
  - ✓ pom.xml 파일에 의존성 추가

```
<dependency>
        <groupId>commons-codec</groupId>
        <artifactId>commons-codec</artifactId>
        <version>1.15</version>
</dependency>
```

  - ✓ 프로젝트에 CryptoUtil.java 파일을 복사

암호화

❖ 암호화 와 복호화

✓ main 메서드를 소유한 클래스를 만들고 실행

```
public class EncryptMain {
        public static void main(String[] args) {
                CryptoUtil cryptoUtil = new CryptoUtil();
                String msg = "adam";
                try {
                        String encrypt = cryptoUtil.encrypt(msg);
                        System.out.printf("암호화된 문자열:%s₩n", encrypt);

                        String decrypt = cryptoUtil.decrypt(encrypt);
                        System.out.printf("복호화된 문자열:%s₩n", decrypt);

                }catch(Exception e) {
                        System.out.printf("예외:%s₩n", e.getMessage());
                        e.printStackTrace();
                }
        }
}
```

EMAIL 전송

CommonsEmail

- CommonsEmail
  - ✓ 자바를 사용해서 이메일을 보낼 수 있는 각종 API를 제공하는 이메일에 관련된 컴포넌트
  - ✓ CommonsEmail 라이브러리는 이메일을 전송하는 기능만 제공할 뿐 받는 기능은 제공하지 않음
  - ✓ CommonEmail 컴포넌트를 이용하여 이메일을 보내기 위해서는 3개의 라이브러리 파일 필요 – 'activation. jar', 'common_email_1.2.jar' 그리고 'mail.jar' 파일
    - ❑ CommonEmail : http://commons.apache.org/email/download_email.cgi
    - ❑ Javamail : http://www.oracle.com/technetwork/java/index-138643.html
    - ❑ JAF : http://www.oracle.com/technetwork/java/jaf11-139815.html
    - ❑ 다운로드한 zip 파일의 압축을 해제한 뒤에 build path에 추가

CommonsEmail

- SMTP 프로토콜
  - ✓ SMTP 프로토콜을 지원하는 Email 서버가 필요
  - ✓ 리눅스나 유닉스 환경의 서버를 운영하고 있다면 SendMail과 같은 데몬을 설치하여 Mail 서버 구축
  - ✓ 윈도우 환경의 서버를 운영하고 있다면 윈도우에서 제공하는 메일 서버를 설치해서 구성
  - ✓ gmail 계정이 있으면 접근 가능 한 SMTP 서버의 주소

SMTP Server address : smtp.gmail.com

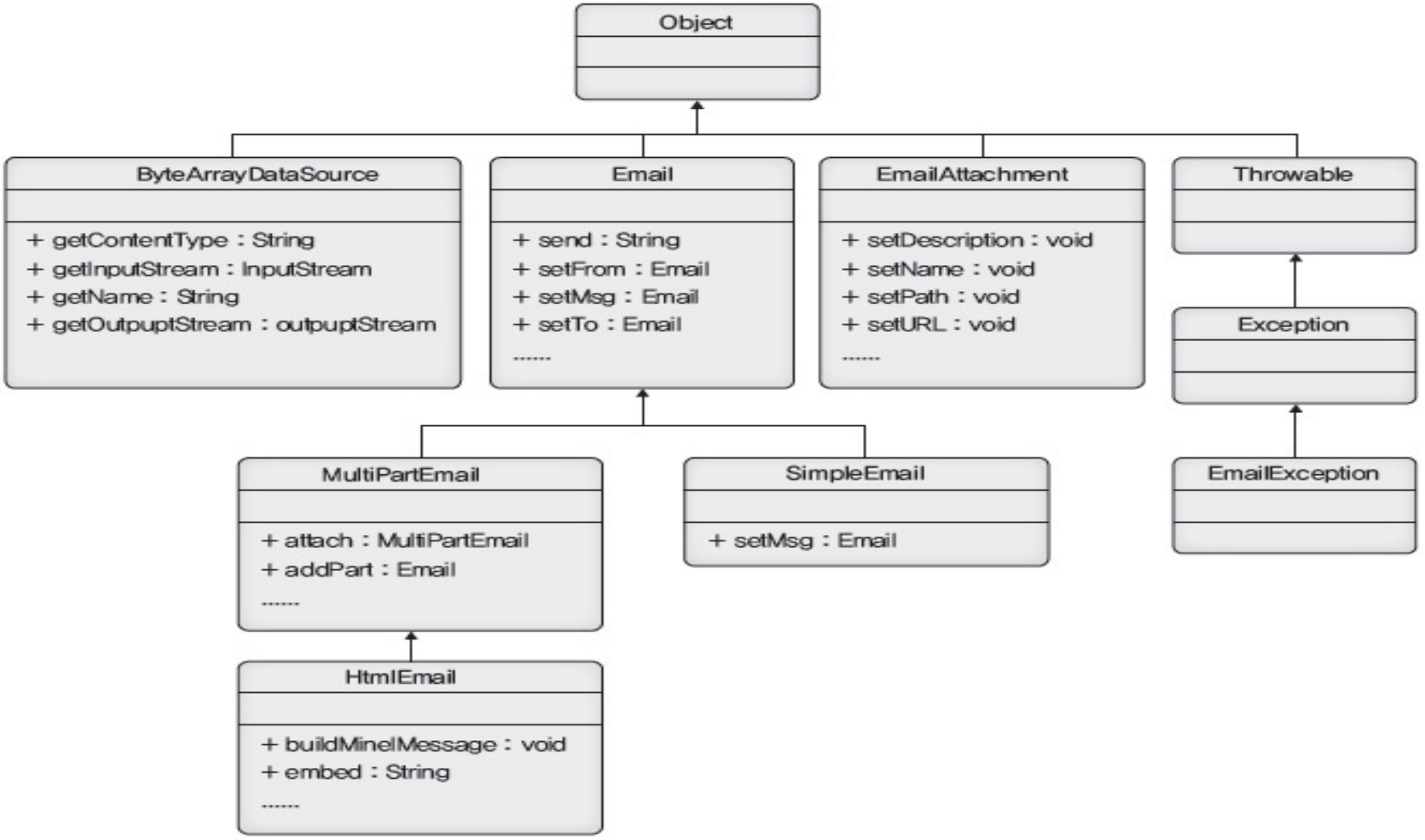
SMTP Server port : 587

SMTP Server account : 여러분의 google 가입 계정(아이디와 패스워드)

  - ✓ naver 계정이 있으면 접근 가능한 SMTP 서버의 주소

POP 서버명 : pop.naver.com	SMTP 서버명 : smtp.naver.com	POP 포트 : 995, 보안연결(SSL) 필요
SMTP 포트 : 465, 보안 연결(SSL) 필요	아이디 : ggangpae11	비밀번호 : 네이버 로그인 비밀번호

Commons Email API



Commons Email API

❖ Email 클래스

- ✓ Email 클래스: 이메일을 보내기 위한 여러 정보들(받는 사람, 보내는 사람, 제목 그리고 메시지)을 설정할 수 있는 API와 Email 전송 API를 제공
- ✓ Email 클래스들의 상위 클래스이므로 Email 클래스에서 제공되는 메서드들은 다른 하위 클래스들에게 상속
- ✓ Email 클래스는 abstract 키워드로 선언된 클래스이므로 Email 클래스를 사용하여 직접 인스턴스 생성 불가
- ✓ Email 추상 클래스를 구현한 SimpleEmail이나 MultiPartEmail 클래스의 객체를 생성해서 사용

메소드	설명
addBcc	• 선언부 : public Email addBcc(String email) public Email addBcc(String email, String name) public Email addBcc(String email, String name, String charset) • 설명 : BCC란 숨은 참조를 의미한다. 숨은 참조에 포함된 사람도 이메일을 전송받으며 그 사람들은 단지 이메일을 참고할 뿐이다. 위의 메소드들을 사용하면 숨은 참조를 추가할 수 있다. 매개변수 email은 이메일 주소를 의미하며 name은 그 사람의 이름을 의미한다. charset은 문자열 인코딩 셋을 의미한다.
addCc	• 선언부 : public Email addCc(String email) public Email addCc(String email, String name) public Email addCc(String email, String name, String charset) • 설명 : Cc란 참조를 의미한다. Cc에 포함된 사람은 이메일을 단지 참고하면 된다. 참조될 사람의 이메일을 addCc( ) 메소드를 사용하여 추가한다.
addHeader	• 선언부 : public void addHeader(String name, String value) • 설명 : SMTP 프로토콜의 헤더에 특정 값을 추가할 때 사용한다. 매개변수 name은 헤더 이름이며 value는 그 헤더의 값을 의미한다.

Commons Email API

❖ Email 클래스

addReplyTo	• 선언부 : public Email addReplyTo (String email) public Email addReplyTo (String email, String name) public Email addReplyTo (String email, String name, String charset) • 설명 : 이메일을 읽은 뒤 응답을 위해 받는 사람들을 추가할 때 사용하는 메소드다.
addTo	• 선언부 : public Email addTo (String email) public Email addTo (String email, String name) public Email addTo (String email, String name, String charset) • 설명 : 이메일을 받는 사람의 이메일 주소를 넣는다. 여러 사람에게 보낼 수 있으므로 계속 메소드를 호출해도 무방하다.
send	• 선언부 : public String send( ) • 설명 : Email 객체에 여러 정보를 설정하고 그 데이터들을 전송할 때 호출하는 메소드다. 이 메소드가 호출되면 실제 이메일이 전송된다.

Commons Email API

❖ Email 클래스

setAuthentication	• 선언부 : public void setAuthentication (String username, String password) • 설명 : SMTP 서버와 연결할 때 SMTP 서버에 접속하기 위한 인증을 받아야 한다. 이때 setAuthentication( ) 메소드를 사용하여 SMTP 계정 정보인 사용자 아이디와 암호를 설정하면 된다. setAuthentication( ) 메소드에 의해서 설정된 계정은 send( ) 메소드에 의해서 SMTP 서버 인증을 받을 때 사용된다.
setFrom	• 선언부 : public Email setFrom(String email) public Email setFrom(String email, String name) public Email setFrom(String email, String name, String charset) • 설명 : 보내는 사람의 이메일 주소를 넣는다. 보내는 사람은 항상 한 명뿐이므로 addTo( ) 메소드와 다르게 setFrom( ) 메소드는 계속 호출하더라도 마지막에 호출된 값에 의해서 보낸 사람의 정보가 설정된다.
setHostName	• 선언부 : public void setHostName(String aHostName) • 설명 : STMP 서버의 Host 이름이나 IP 주소를 입력하도록 한다.

Commons Email API

❖ Email 클래스

setSmtpPort	• 선언부 : public void setSmtpPort(int aPortNumber) • 설명 : SMTP 기본 포트는 465번이지만 SMTP 서버의 설정에 따라서 port 번호가 다를 수 있다. 이 메소드를 사용해서 포트를 변경하지 않으면 Email 클래스는 기본 포트 465번을 이용하여 SMTP 서버에 연결하려고 한다.
setSSL	• 선언부 : public void setSSL(boolean ssl) • 설명 : 메일 클라이언트와 메일 서버 사이에 데이터를 암호화하기 위해서 사용한다.
setSubject	• 선언부 : public Email setSubject(String subject) • 설명 : 보내고자 하는 메일의 제목을 입력한다.
setMsg	• 선언부 : public static Email set msg(String msg) • 설명 : 메일 본문을 사용한다. 개행 문자(₩n)를 사용해서 줄바꿈을 한다.
setDebug	• 선언부 : public void setDebug(boolena d) • 설명 : 디버깅 관련 정보를 출력하는 메소드다. true인 경우 디버깅 정보가 출력된다.

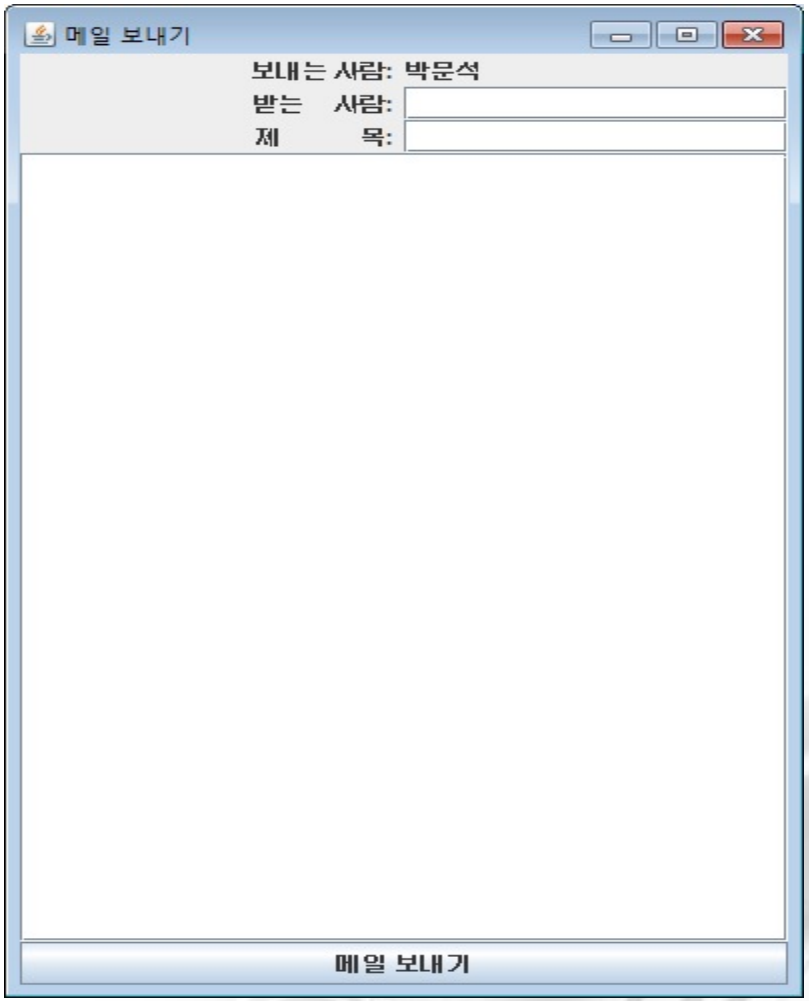
Commons Email API

- 텍스트 전송을 위한 SimpleEmail 클래스
  - ✓ SimpleEmail 클래스 : 특별한 추가 기능 없이 순수한 텍스트 기반의 이메일을 전송하고자 할 때 사용
  - ✓ SimpleEmail 인스턴스 생성

```
사용법 : SimpleEmail [객체 이름] = new SimpleEmail();
사용예 : SimpleEmail email = new SimpleEmail();
```

  - ✓ SimpleEmail 클래스는 public으로 선언된 생성자들이 존재하므로 new 키워드를 사용하여 객체를 인스턴스
  - ✓ 매개변수가 없는 기본 생성자만 제공

텍스트 메일 보내기



텍스트 메일 보내기

- pom.xml 파일에 EMAIL 관련 라이브러리를 위한 의존성을 설정

```
<dependency>
        <groupId>org.apache.commons</groupId>
        <artifactId>commons-email</artifactId>
        <version>1.5</version>
</dependency>
```

텍스트 메일 보내기

- GUI 만들기

```
import java.awt.GridLayout;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

텍스트 메일 보내기

❖ GUI 만들기

```
public class TextMailView extends JFrame {
        private static final long serialVersionUID = 1L;

        public TextMailView(){
                setTitle("메일 보내기");
                setBounds(100,100,400,600);
                setDefaultCloseOperation(EXIT_ON_CLOSE);
                JPanel northPanel = new JPanel();
                northPanel.setLayout(new GridLayout(3,2));
                JLabel lblSend = new JLabel("보내는 사람:  ", JLabel.RIGHT);
                JLabel sendEmail = new JLabel("박문석");
                JLabel lblReceive = new JLabel("받는    사람:  ", JLabel.RIGHT);
                JTextField receiveEmail = new JTextField(30);
                JLabel lblTitle = new JLabel("제          목:  ", JLabel.RIGHT);
                JTextField txtTitle = new JTextField(30);
                northPanel.add(lblSend);
                northPanel.add(sendEmail);
                northPanel.add(lblReceive);
                northPanel.add(receiveEmail);
                northPanel.add(lblTitle);
                northPanel.add(txtTitle);
                add("North", northPanel);
```

텍스트 메일 보내기

- GUI 만들기

```
        JTextArea content = new JTextArea(20,20);
        JScrollPane sp = new JScrollPane(content);
        add(sp);

        JPanel southPanel = new JPanel();
        JButton btnSend = new JButton("메일 보내기");
        southPanel.add(btnSend);
        add("South", btnSend);


        setVisible(true);
    }
}
```

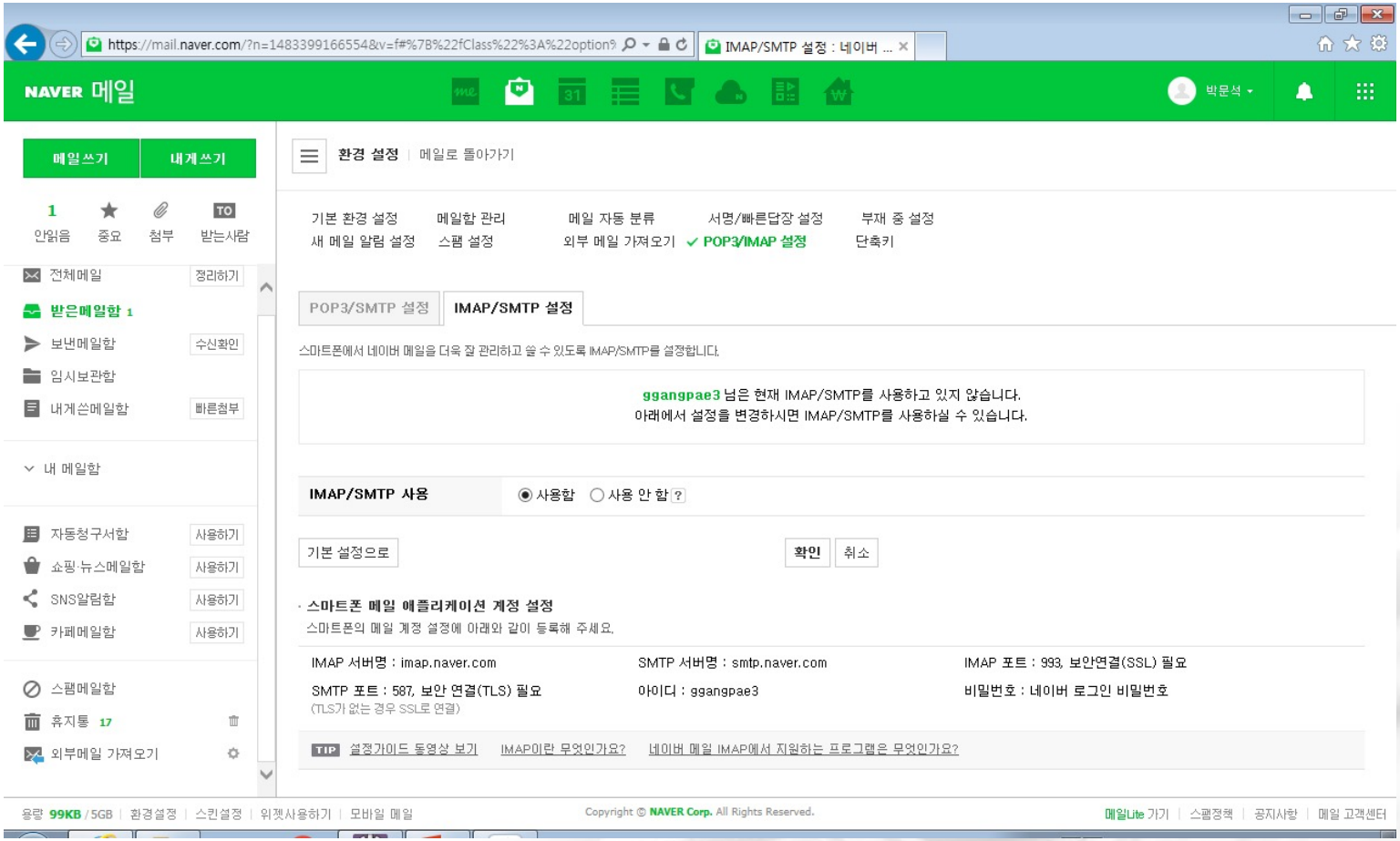
텍스트 메일 보내기

- Main 만들기

```
public class TextMailMain {
        public static void main(String[] args) {
                new TextMailView();
        }
}
```

텍스트 메일 보내기

❖ 네이버 메일인 경우 환경설정 메뉴에서 SMTP 설정



텍스트 메일 보내기

❖ EventHandler 만들기

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.JOptionPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;

import org.apache.commons.mail.SimpleEmail;

public class TextMailEventHandler implements ActionListener {
	private JTextField txtReceiveEmail;
	private JTextField txtTitle;
	private JTextArea txtContent;

	public TextMailEventHandler(JTextField txtReceiveEmail, JTextField txtTitle, JTextArea
txtContent) {
		super();
		this.txtReceiveEmail = txtReceiveEmail;
		this.txtTitle = txtTitle;
		this.txtContent = txtContent;
	}
```

텍스트 메일 보내기

❖ EventHandler 만들기

```
@Override
public void actionPerformed(ActionEvent e) {
        String receiveEmail = txtReceiveEmail.getText().trim();
        if (receiveEmail.length() == 0) {
                JOptionPane.showMessageDialog(null, "받는 분의 이메일 주소가
없습니다.");
                return;
        }
        String title = txtTitle.getText().trim();
        if (title.length() == 0) {
                title = "무제";
        }
        String content = txtContent.getText().trim();
        if (content.length() == 0) {
                content = "냉무";
        }

        SimpleEmail simpleEmail = new SimpleEmail();
        // Smtp 서버 연결 설정.
        simpleEmail.setHostName("smtp.naver.com");
        simpleEmail.setSmtpPort(587);
        simpleEmail.setAuthentication("ggangpae11", "cyberadam");
        String rt = "";
```

텍스트 메일 보내기

❖ EventHandler 만들기

```
try {
    // Smtp SSL, TLS 설정
    simpleEmail.setSSLOnConnect(true);
    simpleEmail.setStartTLSEnabled(true);
    simpleEmail.setCharset("utf-8");

    // 받는 사람 설정
    simpleEmail.addTo(receiveEmail, "받는 이", "utf-8");
    // 보내는 사람 설정
    simpleEmail.setFrom("ggangpae11@naver.com", "박문석", "utf-
8");

    // 제목 설정
    simpleEmail.setSubject(title);

    // 본문 설정
    simpleEmail.setMsg(content);
    rt = simpleEmail.send();
    System.out.println(rt);
```

텍스트 메일 보내기

❖ EventHandler 만들기

```
                                    JOptionPane.showMessageDialog(null, "메일 보내기 성공");
                                    txtReceiveEmail.setText("");
                                    txtTitle.setText("");
                                    txtContent.setText("");
                            } catch (Exception exception) {
                                    JOptionPane.showMessageDialog(null, exception.getMessage(),
"메일 보내기 실패", JOptionPane.ERROR_MESSAGE);
                                    ;
                            }

                    }

}
```

텍스트 메일 보내기

- TextMailView의 생성자에서 메일 보내기 버튼 이벤트 핸들러 설정

```
TextMailEventHandler handler = new TextMailEventHandler(receiveEmail, txtTitle, content);
btnSend.addActionListener(handler);
```

첨부 파일 이메일

❖ 메시지와 파일을 함께 전송하는 MultiPartEmail 클래스

- ✓ MultiPartEmail 클래스 : 메일을 전송할 때 여러 개의 파일을 첨부해서 보낼 수 있는 기능을 제공해주는 클래스
- ✓ Multi-Part 전송 : 첨부 파일을 전송할 때 메시지와 파일을 여러 개로 분할해서 전송하는 방식
- ✓ MultiPartEmail 인스턴스 생성

```
사용법 : MultiPartEmail [객체 이름] = new MultiPartEmail();
사용예 : MultiPartEmail email = new MultiPartEmail();
```

첨부 파일 이메일

❖ 메시지와 파일을 함께 전송하는 MultiPartEmail 클래스

메소드	설명
attach	• 선언부 : public MultiPartEmail attach(EmailAttachment attachment) public MultiPartEmail attach(DataSource ds, String name, String description) public MultiPartEmail attach(DataSource ds, String name, String description, String disposition) public MultiPartEmail attach(URL url, String name, String description) • 설명 : 파일을 첨부하기 위한 메소드들이다. attach( ) 메소드가 호출되면 첨부 파일은 MultiPartEmail 객체의 속성으로 설정된다. 그리고 send( ) 메소드 호출 시 전송된다. 또한 다양한 형태의 첨부 파일을 처리할 수 있도록 attach( ) 메소드들이 오버라이딩되어 있다. 보통은 CommonsEmail 컴포넌트에서 제공하는 EmailAttachment 객체를 이용해서 attach( ) 메소드를 호출한다. 매개변수 중 name은 첨부 파일의 이름을 의미하며 description은 설명, 그리고 disposition은 어떤 성격의 첨부 파일인지 의미한다.

첨부 파일 이메일

❖ 첨부 파일을 위한 EmailAttachment 클래스

- ✓ EmailAttachment 클래스 : 이메일에 첨부 파일을 추가해서 보낼 때 사용
- ✓ 첨부 파일을 추상화한 클래스로서 반드시 MultiPartEmail 클래스와 같이 사용
- ✓ EmailAttachment 클래스의 인스턴스 생성

```
사용법 : EmailAttachment [객체 이름] = new EmailAttachment();
사용예 : EmailAttachment attachment= new EmailAttachment();
```

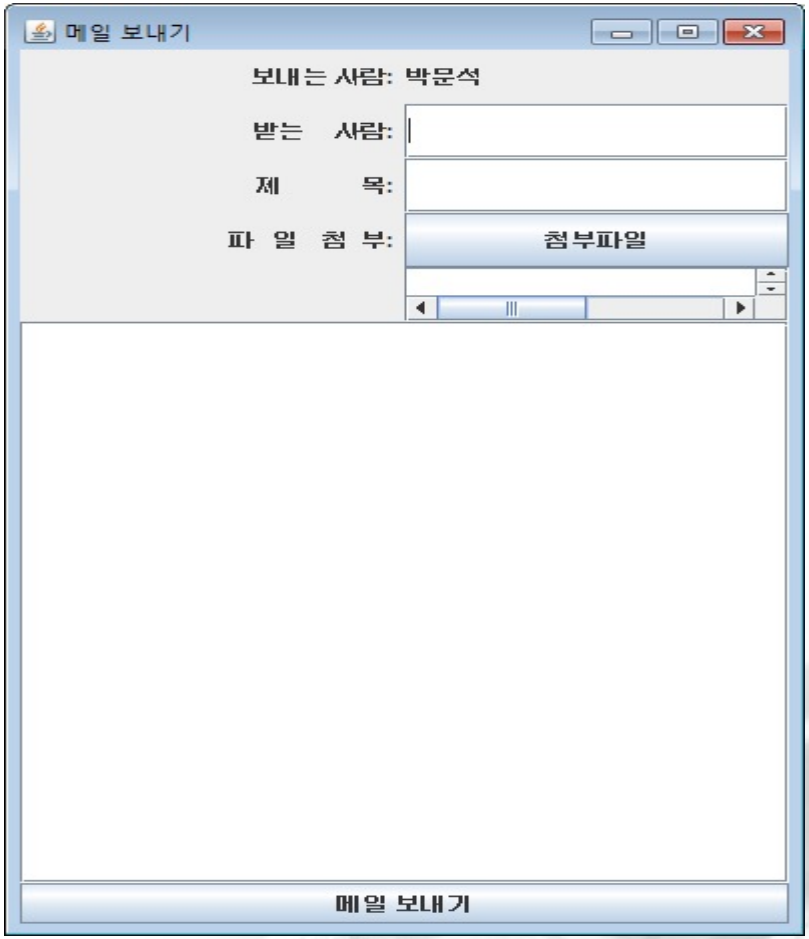
메소드	설명
getDescription/ setDescrption	• 선언부 : public String getDescription( ) public void setDescription(String desc) • 설명 : 첨부 파일에 대한 설명을 EmailAttachment 객체에 설정(set)하거나 설정한 것을 받아오는(get) 메소드
getDisposition/ setDisposition	• 선언부 : public String getDisposition( ) public void setDisposition(String aDisposition) • 설명 : 첨부 파일에 대한 성격을 설정(set)하거나 설정된 것을 받아오는(get) 메소드
getName/ setName	• 선언부 : public String getName( ) public void setName(String aName) • 설명 : EmailAttachment 객체에 이름을 설정하거나 받아오는 메소드

첨부 파일 이메일

❖ 첨부 파일을 위한 EmailAttachment 클래스

getPath/ setPath	• 선언부 : public String getPath( ) public void setPath(String aPath) • 설명 : EmailAttachment 객체에 Path를 지정하면 Path에 위치한 파일이 첨부 파일이 된다. 실제 첨부 파일을 설정하는 메소드이므로 EmailAttachment 클래스에서 가장 중요한 메소드다.
getURL/ setURL	• 선언부 : public URL getURL( ) public void setURL(URL aUrl) • 설명 : EmailAttachment 객체에 첨부할 파일이 위치한 URL을 설정하거나 설정된 값을 받아오는 메소드다. 위의 get/setPath 메소드와 같은 역할을 하지만 경로를 지정하는 방식에 따른 매개변수의 타입이 다르다.

첨부 파일 이메일



첨부 파일 이메일

- FileMailView 클래스를 생성하고 생성자 작성

```
import java.awt.GridLayout;
import java.beans.EventHandler;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JScrollPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;
```

첨부 파일 이메일

❖ FileMailView 클래스를 생성하고 생성자 작성

```
public class FileMailView extends JFrame{
        private static final long serialVersionUID = 1L;

        public FileMailView(){
                setTitle("메일 보내기");
                setBounds(100,100,400,600);
                setDefaultCloseOperation(EXIT_ON_CLOSE);

                JPanel northPanel = new JPanel();
                northPanel.setLayout(new GridLayout(5,2));
                JLabel lblSend = new JLabel("보내는 사람:  ", JLabel.RIGHT);
                JLabel sendEmail = new JLabel("박문석");

                JLabel lblReceive = new JLabel("받는      사람:  ", JLabel.RIGHT);
                JTextField receiveEmail = new JTextField(30);

                JLabel lblTitle = new JLabel("제              목:  ", JLabel.RIGHT);
                JTextField txtTitle = new JTextField(30);

                JLabel lblFile = new JLabel("파   일   첨   부:  ", JLabel.RIGHT);
                JButton btnFile = new JButton("첨부파일");
```

첨부 파일 이메일

❖ FileMailView 클래스를 생성하고 생성자 작성

```
JTextArea txtFiles = new JTextArea(2, 30);
JScrollPane sp1 = new JScrollPane(txtFiles);


northPanel.add(lblSend);
northPanel.add(sendEmail);
northPanel.add(lblReceive);
northPanel.add(receiveEmail);
northPanel.add(lblTitle);
northPanel.add(txtTitle);
northPanel.add(lblFile);
northPanel.add(btnFile);
northPanel.add(new JLabel());
northPanel.add(sp1);

add("North", northPanel);

JTextArea content = new JTextArea(20,20);
JScrollPane sp = new JScrollPane(content);
add(sp);
```

첨부 파일 이메일

❖ FileMailView 클래스를 생성하고 생성자 작성

```
                        JPanel southPanel = new JPanel();
                        JButton btnSend = new JButton("메일 보내기");
                        southPanel.add(btnSend);
                        add("South", btnSend);

                        FileMailEventHandler handler = new FileMailEventHandler(receiveEmail,
txtTitle, content, txtFiles);
                        btnSend.addActionListener(handler);
                        btnFile.addActionListener(handler);
                        setVisible(true);
                }

}
```

첨부 파일 이메일

- FileMailEventHandler 클래스를 작성하고 작성

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.File;

import javax.swing.JFileChooser;
import javax.swing.JOptionPane;
import javax.swing.JTextArea;
import javax.swing.JTextField;

import org.apache.commons.mail.EmailAttachment;
import org.apache.commons.mail.MultiPartEmail;
```

첨부 파일 이메일

❖ FileMailEventHandler 클래스를 작성하고 작성

```
public class FileMailEventHandler implements ActionListener {
        private JTextField txtReceiveEmail;
        private JTextField txtTitle;
        private JTextArea txtContent;
        private JTextArea txtFiles;

        public FileMailEventHandler(JTextField txtReceiveEmail, JTextField txtTitle, JTextArea
txtContent,
                                    JTextArea txtFiles) {
                super();
                this.txtReceiveEmail = txtReceiveEmail;
                this.txtTitle = txtTitle;
                this.txtContent = txtContent;
                this.txtFiles = txtFiles;
        }
```

첨부 파일 이메일

❖ FileMailEventHandler 클래스를 작성하고 작성

```
@Override
public void actionPerformed(ActionEvent e) {
        if (e.getActionCommand().equals("메일 보내기")) {
                String receiveEmail = txtReceiveEmail.getText().trim();
                if (receiveEmail.length() == 0) {
                        JOptionPane.showMessageDialog(null, "받는 분의 이
메일 주소가 없습니다.");
                        return;
                }
                String title = txtTitle.getText().trim();
                if (title.length() == 0) {
                        title = "무제";
                }
                String content = txtContent.getText().trim();
                if (content.length() == 0) {
                        content = "냉무";
                }
```

첨부 파일 이메일

❖ FileMailEventHandler 클래스를 작성하고 작성

```
MultiPartEmail email = new MultiPartEmail();
// Smtp 서버 연결 설정.
email.setHostName("smtp.naver.com");
email.setSmtpPort(587);
email.setAuthentication("ggangpae11", "cyberadam");
String rt = "";
try {
        // Smtp SSL, TLS 설정
        email.setSSLOnConnect(true);
        email.setStartTLSEnabled(true);

        email.setCharset("utf-8");

        if (txtFiles.getText().length() != 0) {
                String[] fileNames =
txtFiles.getText().split(",");

                for (String file : fileNames) {
                        EmailAttachment attach = new
EmailAttachment();

                        attach.setName("");
                        attach.setPath(file);
                        email.attach(attach);
                }
        }
```

첨부 파일 이메일

❖ FileMailEventHandler 클래스를 작성하고 작성

```
                                        // 받는 사람 설정
                                        email.addTo(receiveEmail, "받는 이", "utf-8");
                                        // 보내는 사람 설정
                                        email.setFrom("ggangpae11@naver.com", "박문석",
"utf-8");

                                        // 제목 설정
                                        email.setSubject(title);
                                        // 본문 설정
                                        email.setMsg(content);
                                        rt = email.send();
                                        System.out.println(rt);
                                        JOptionPane.showMessageDialog(null, "메일 보내기
성공");
                                        txtReceiveEmail.setText("");
                                        txtTitle.setText("");
                                        txtContent.setText("");

                                } catch (Exception exception) {
                                        JOptionPane.showMessageDialog(null,
exception.getMessage(), "메일 보내기 실패", JOptionPane.ERROR_MESSAGE);
                                }
                        }
```

첨부 파일 이메일

❖ FileMailEventHandler 클래스를 작성하고 작성

```
                else {
                        JFileChooser fc = new JFileChooser();
                        fc.setMultiSelectionEnabled(true);
                        int result = fc.showOpenDialog(null);
                        if (result == JFileChooser.APPROVE_OPTION) {
                                File[] f = fc.getSelectedFiles();
                                for (File file : f) {
                                        if (txtFiles.getText().trim().length() == 0) {
                                                txtFiles.setText(file.getPath());
                                        } else {
                                                String origin =
txtFiles.getText();

                                                txtFiles.setText(origin + "," +
file.getPath());
                                        }
                                }
                        }
                }
        }
}
```

첨부 파일 이메일

❖ FileMailMain 클래스를 생성하고 작성한 후 실행

```
public class FileMailMain {

        public static void main(String[] args) {
                new FileMailView();
        }

}
```

HTML 이메일

- HTML 형식의 HtmlEmail 클래스
  - ✓ HtmlEmail 클래스 : 이메일을 HTML 형식으로 전송하고자 할 때 사용
  - ✓ MuliPartEmail 클래스를 상속받고 있으므로 첨부 파일 전송까지 가능
  - ✓ HtmlEmail 클래스는 기본 생성자만 제공

HTML 이메일

❖ HTML 형식의 HtmlEmail 클래스

메소드	설명
embed	• 선언부 : public String embed(File file) public String embed(File file, String cid) public String embed(String urlString, String name) public String embed(URL url, String name) public String embed(DataSource dataSource, String name) public String embed(DataSource dataSource, String name, String cid) • 설명 : Email 본문에 이미지 등의 파일을 포함시킬 때 사용하는 메소드다. 이미지 파일의 유형에 따라서 여러 매개변수들을 받을 수 있도록 오버라이딩되어 있다.
setHtmlMsg	• 선언부 : public HtmlEmail setHtmlMsg(String aHtml) • 설명 : 메일 본문에 HTML 태그를 포함한 메시지를 남기기 위해서 사용한다. 앞서 살펴본 setMsg과 같은 역할을 하지만 HTML 태그를 포함할 때 에러를 방지하는 역할을 한다.
setTextMsg	• 선언부 : public HtmlEmail setTextMsg(String aText) • 설명 : HTML 태그가 포함되지 않는 순수 텍스트 메시지를 메일 본문에 포함하기 위해서 사용하는 메소드다.

HTML 이메일

❖ HTMLMailMain

```
import java.io.File;

import org.apache.commons.mail.EmailException;
import org.apache.commons.mail.HtmlEmail;

public class HTMLMailMain {

        public static void main(String[] args) {
                HtmlEmail htmlEmail = new HtmlEmail();

                // Smtp 서버 연결 설정.
                htmlEmail.setHostName("smtp.naver.com");
                htmlEmail.setSmtpPort(587);
                htmlEmail.setAuthentication("ggangpae11", "cyberadam");
                // 이미지 파일 생성
                File logoFile = new File("red3.jpg");

                String rt = "Failure";
```

HTML 이메일

```
try {
        // Smtp SSL, TLS 설정
        htmlEmail.setSSLOnConnect(true);
        htmlEmail.setStartTLSEnabled(true);
        htmlEmail.setCharset("utf-8");

        // 받는 사람 설정
        htmlEmail.addTo("ggangpae1@gmail.com", "박문석", "utf-8");
        // 보내는 사람 설정
        htmlEmail.setFrom("ggangpae11@naver.com", "박문석", "utf-
8");
        // 제목 설정
        htmlEmail.setSubject("html 메일 보내기");

        // HTML 테그가 포함된 메일 본문을 StringBuffer 객체를 사용하여
처리
        StringBuffer sb = new StringBuffer();
        sb.append("<html><body>");
        sb.append("안녕하세요 박문석 입니다. <br />");
        sb.append("<img
src=cid:").append(htmlEmail.embed(logoFile)).append(">");
        sb.append("</body></html>");
        htmlEmail.setHtmlMsg(sb.toString());
```

HTML 이메일

```
                    rt = htmlEmail.send();
                    System.out.println(rt);
                } catch (EmailException e) {
                    System.out.println(e.getMessage());
                }

            }

}
```