2019.12.06



design pattern

term project

member: 김상범, 양준호, 장동재

**Category**

1. **What is Jsoup?**
2. **Design and Implementation suervey of Jsoup**
3. **What to do?**
   1. **Summary**
   2. **Details**
4. **Test**
   1. **Proceeding to existing code**
   2. **Procedding to new code**
      1. **Wrong host name**
      2. **Wrong protocol**
5. **Github**
   1. **Address of team Github**
   2. **Project history screenshot**

**1. What is Jsoup?**

**‘**Jsoup’은 HTML 문서에 저장된 데이터를 구문 분석, 추출 및 조작하도록 설계된 JAVA의 Open Source Library로, 단순한 HTML document parsing을 넘어 웹 사이트에 대한 Request, Response를 모두 처리할 수 있다. 덕분에 일부 특별한 경우가 아닌 이상 브라우저로 사이트를 이용하는 상황을 그대로 재현해낼 수 있는데, 이는 즉, 대부분의 사이트에서 원하는 정보만 뽑아내는 ‘Viewer’를 만들 수 있다는 말이다.

Jsoup은 HTML 문서를 읽어 들인 후에 그 문서를 DOM (Document Object Model) 객체로 변환한다. 그리고 Jsoup의 selector api를 이용해서 특정 Element에 접근할 수 있고, 해당 element의 정보를 읽거나 수정할 수 있다.

Jsoup의 주요 요소는 크게 다섯 가지로 볼 수 있다.

|  |  |
| --- | --- |
| Class name | Description |
| Document | Jsoup 얻어온 결과 HTML 전체 문서 |
| Element | Document의 HTML 요소 |
| Elements | Element가 모인 자료형. for나 while 등 반복문 사용이 가능하다. |
| Connection | Jsoup의 connect 혹은 설정 메소드들을 이용해 만들어지는 객체, 연결을 하기 위한 정보를 담고 있다. |
| Response | Jsoup가 URL에 접속해 얻어온 결과. Document와 다르게 status 코드, status 메시지나 charset같은 헤더 메시지와 쿠키 등을 가지고 있다. |

Jsoup을 이용해 하는 작업은 크게, Connection 객체를 통해 URL에 접속하고, Response객체에서 세션ID같은 쿠키와 HTML Document를 얻어낸 후, Document의 Element들을 파싱하는 과정으로 나누어진다고 볼 수 있다.

**2. Design and Implementation survey of Jsoup**

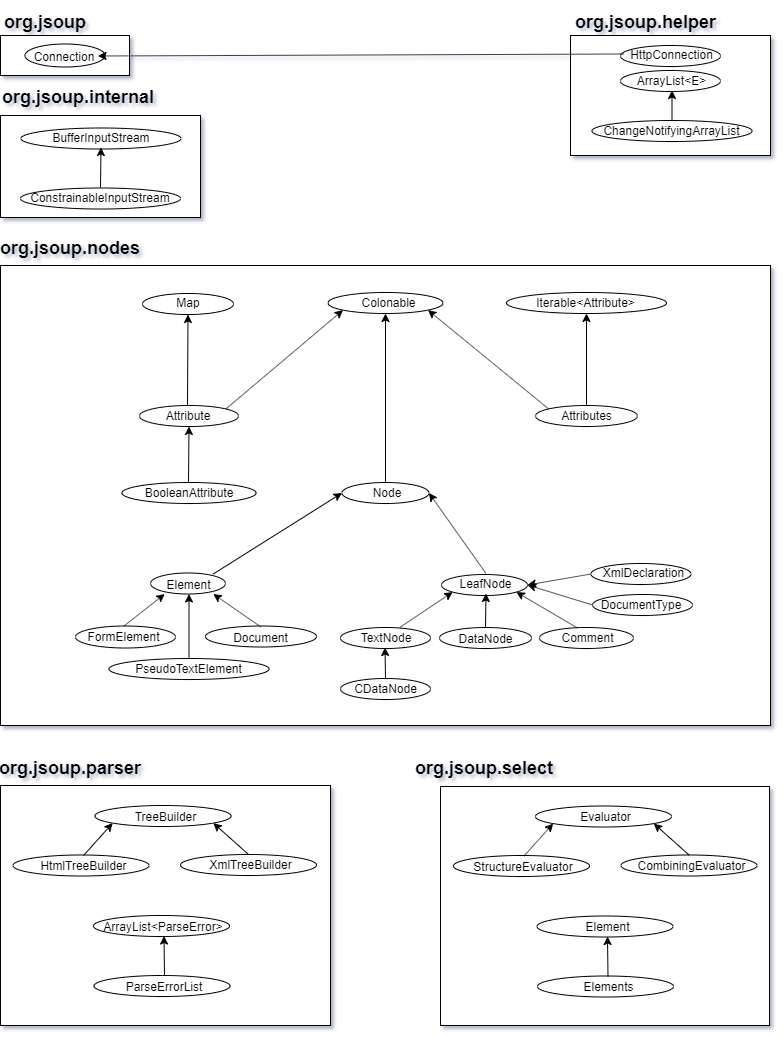


Figure 1) Packages in jsoup and relationships of each class

Jsoup의 세분화 조사에 앞서 Jsoup 내 존재하는 패키지들과 각 패키지별 클래스들의 관계를 대략적으로 그려보았다. Figure 1에 있는 패키지들 외에 org.jsoup.safety 패키지가 존재하지만, 내부 클래스 간 관계가 명확하지 않아 제외하고 나머지 패키지들을 파악했다. 우리가 초점을 맞출 부분은 opg.jsoup.helper의HttpConnection 이며, Parsing을 진행할 때, org.jsoup.parser 패키지의 HtmlTreeBuilder 클래스와 XmlTreeBuilder 클래스에서 인터페이스 함수들을 HTML과 XML에 따라 다르게 Override하는 것을 보아 기존 적용 되어있는 Design Pattern은 Decorator Pattern으로 판단했다.

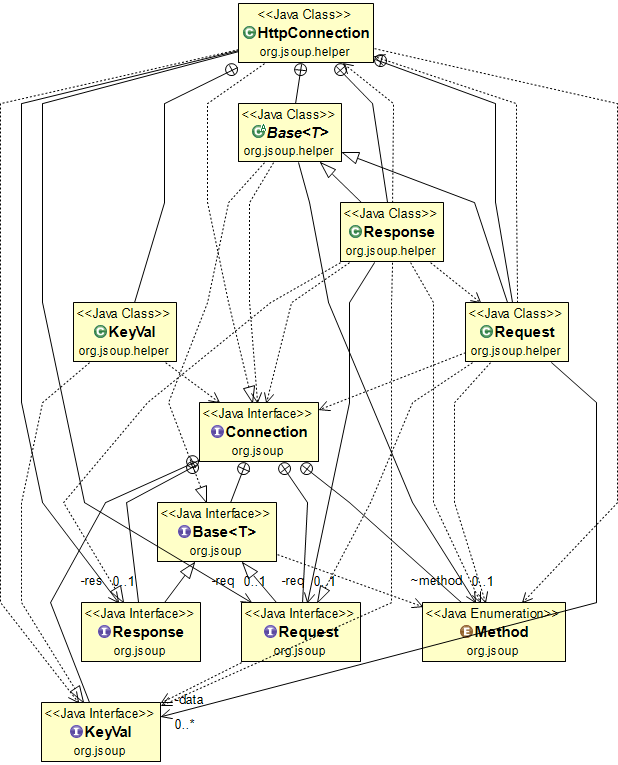


Figure 2) Class Diagram

Figure 2는 HttpConnection와 관계있는 클래스들을 보여주고 있다.

**3. What to do ?**

**A. Summary**

앞서 얘기했 듯, 우리는 HttpConnection 클래스에 초점을 맞추었다. Jsoup을 이용하여 Parsing을 진행할 때, 입력 받은 URL주소가 완벽하지 않을 경우 불러오지 못하는 것을 알고 해당 문제에 대한 개선을 진행했다.

간략하게 개선한 사항에 대해 설명하자면, 기존의 코드에서는 URL이 string으로만 받아지는데, URL형태로 받아져도 코드를 수정하지 않고 받을 수 있도록 interface를 만들어 Decorator Pattern을 적용했다. 또한 string으로 URL을 받을 때, 예를 들면, URL ‘http://www.cgv.co.kr/movies/’ 에서 통신 protocol인 (http)가 정확히 입력되지 않았을 경우 원하는 페이지인 cgv로의 연결이 되지 않는데, URL프로토콜 부분의 철자가 틀려도 수정이 되어 연결하는 기능도 추가했다.

**B. Details**

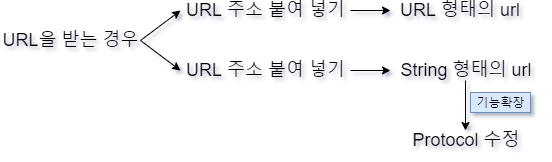


Figure 3) 2 cases of getting URL

우선, URL을 받는 경우를 2가지로 나누었다.

1. URL 복사 붙여 넣기
2. URL의 정확한 protocol을 모르고 사용자가 쓰는 경우

번의 경우 URL형태의 url이 나오게 되고, 번의 경우 String 형태의 url이 나오게 된다. 이어서 번에서 나온 url의 protocol 을 고쳐주는 기능을 추가했다.

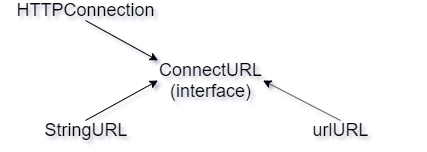


Figure 4) Class and Interface

Figure 3의 case 1의 경우엔 urlURL 클래스를 만들어서 처리할 수 있도록 했고, case 2는 StringURL 클래스를 만들어서 처리할 수 있도록 했다. 또한 StringURL 과 urlURL 클래스를 묶는 ConnectURL 인터페이스를 추가했다. 위 과정을 통해 기존에 String 형태로만 URL을 사용하는 것이 아닌 URL 형태의 url도 받을 수 있게 했다. 또한 protocol이 완벽하지 않아도 url로 연결이 되게 했다. 사용한 Design Pattern은 Decorator Pattern으로 이를 이용해 다양한 형태의 URL이 입력되어도 하나의 인터페이스를 통해서 다형성을 가지게 되었다.

**4. Test**

**A. Proceeding to existing code**

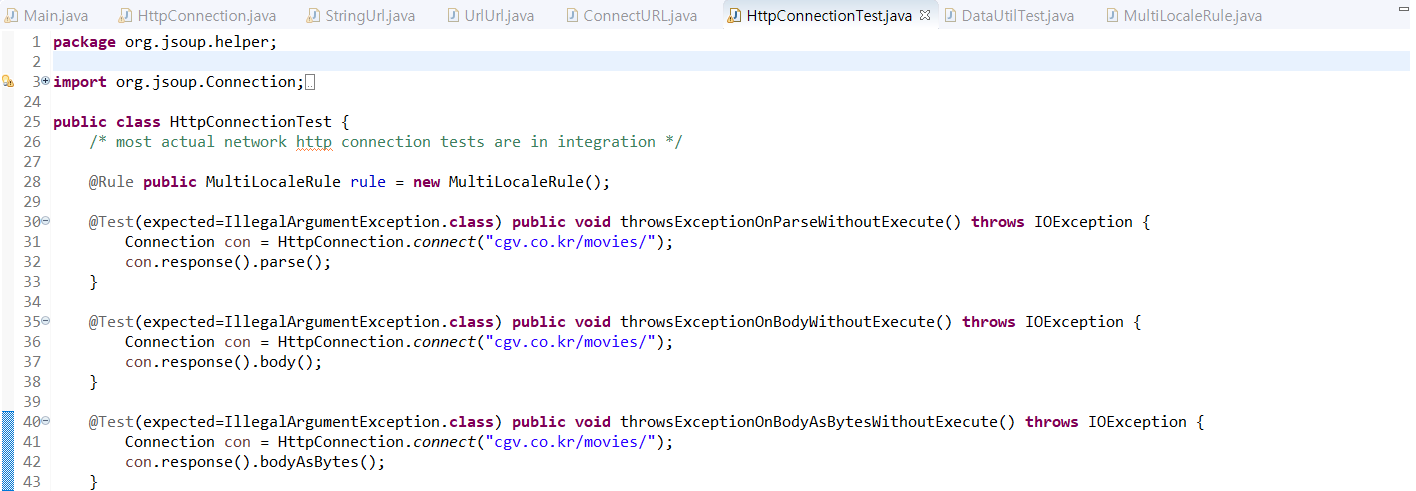


Figure 5.1) Test code

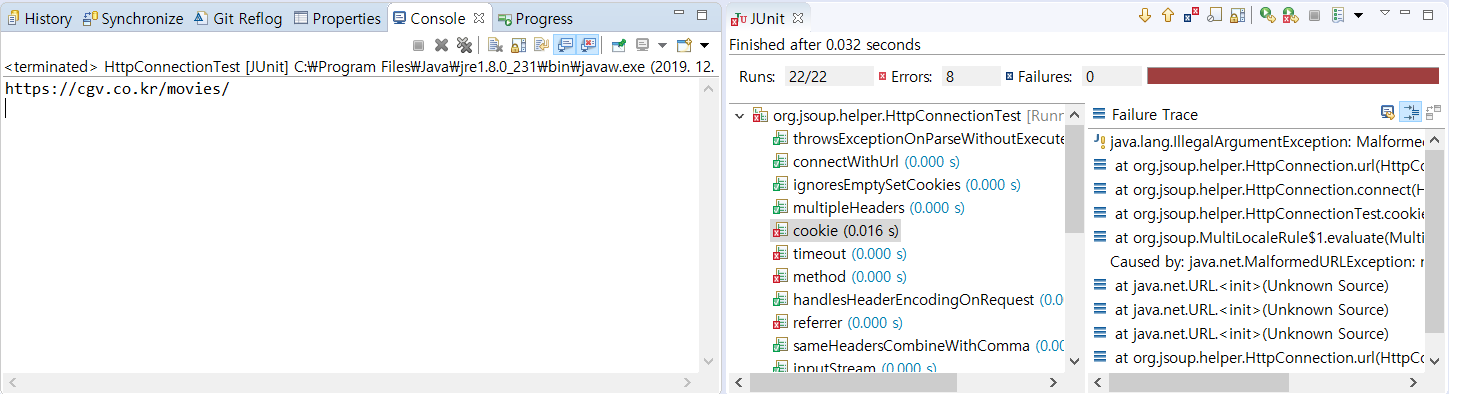


Figure 5.2) Result

**B. Proceeding to New code**

i. Wrong host name



Figure 6.1) Test code

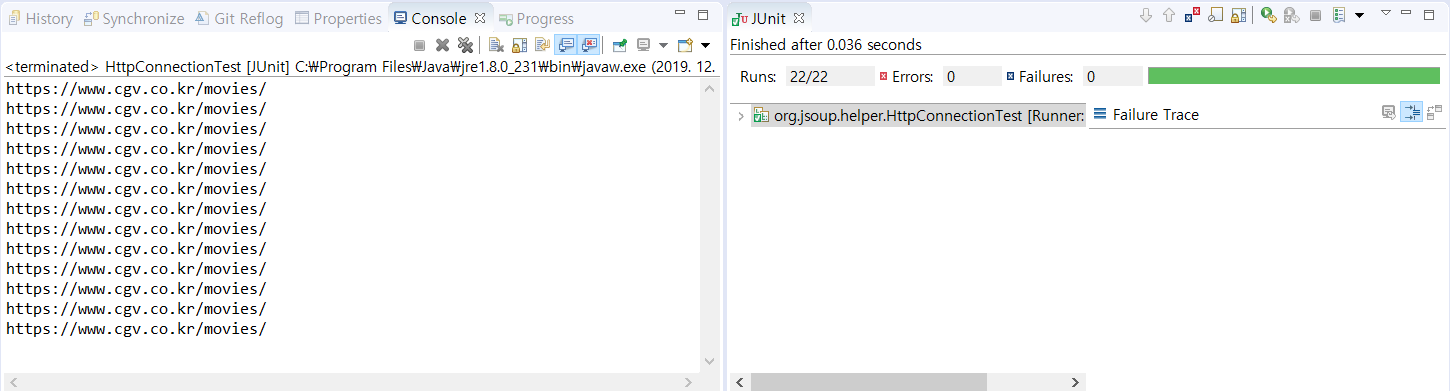


Figure 6.2) Result

ii. Wrong protocol

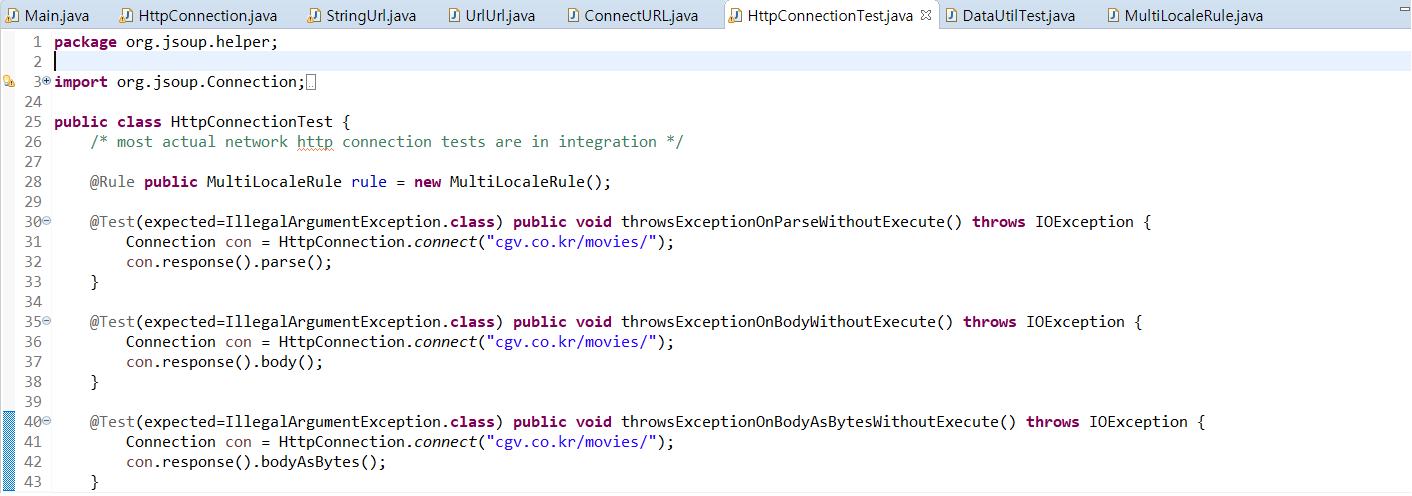


Figure 7.1) Test code

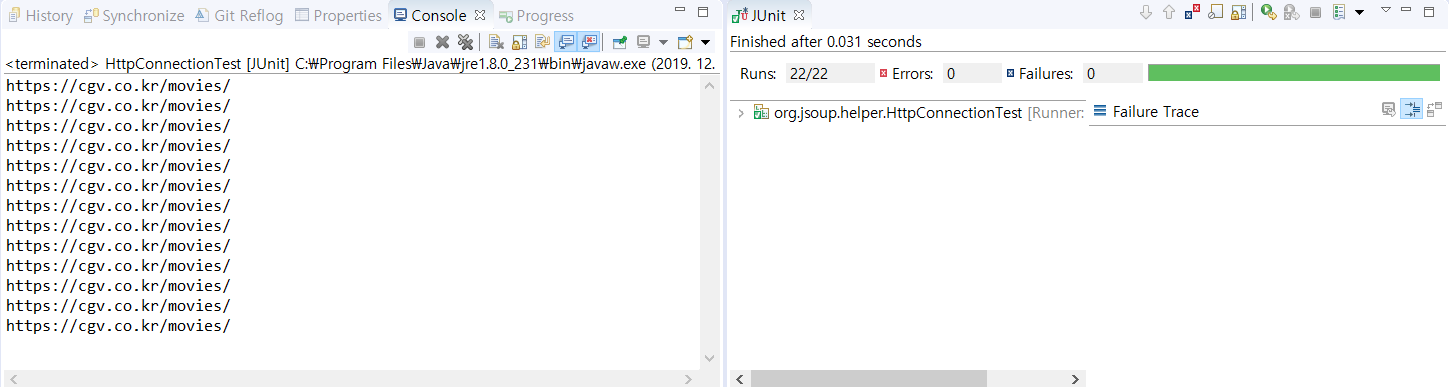


Figure 7.2) Result

**5. Github**

A. Github Address : <https://github.com/plm3332/DesignPattern/>

B. Project Progress History.

