클라이언트 서버 프로그래밍

**RMI 수강신청 프로젝트 보고서**

학과: 융합소프트웨어학부 응용소프트웨어전공

학번: 60201976

이름: 장채은

목차

1. **서론**2
2. 프로젝트 목적2
3. 프로젝트 순서3
4. **본론4**
5. 공통코드5
6. 로그인/회원가입5
7. 학생 정보5
8. 강의 정보5
9. 수강신청5
10. 프로젝트 실행 결과5
11. 결론**1**

장 제목 입력(수준 3)6

1. 서론
2. 프로젝트 목적

목적을 설명하기에 앞서 프로젝트에서 사용한 기술 및 간단한 개념을 설명하고 넘어가고자 한다. Client는 사용자 서비스 화면을 제공하며, 사용자가 작성한 정보를 서버에 전달하여 서비스를 받는 역할을 한다. Server에서는 Data와 Client 연결 및 서비스 제공 코드를 작성한다. 또한 프로그램이 예외가 발생하여 생기는 에러 상황을 방지해주고, Client에서 요청한 서비스를 제공한다. 클라이언트와 서버가 소통하는 프로그램의 경우 하나의 컴퓨터 안에서 서비스를 제공하고 받을 수 있지만, 서비스를 제공을 하는 Server와 제공을 받는 Client의 경우 다른 환경(위치 등) 일 수 있기 때문에 Client와 Server, Data로 각각의 프로젝트 파일로 생성하였다. RMI는 순수 자바 기반의 분산 프로그래밍(여러 개로 나누는)을 지원하는 라이브러리기 때문에 각각의 Client와 Server의 소통이 가능하도록 도와준다.

따라서, 수강신청 프로젝트는 Client와 Server, Data가 서로 요청과 응답을 받으면서 Client와 Server간의 상호관계(데이터 요청 및 제공 등)를 알아보는 목적을 갖고 RMI 라이브러리를 이용하여 진행했다.

1. 프로젝트 순서

보고서 순서는 다음과 같다. 수강신청 프로그램의 모든 서비스 기능은 Sequence Diagram을 통해 시각적으로 설명하고자 한다. 이후, 프로젝트 Source Code 확인을 통해 서비스의 유지 보수 및 기능 & 기술에 대해 자세히 알아보며, 프로젝트 실행 결과를 CMD 화면으로 보여준다.

1. 본론

본격적인 기능 설명에 들어가기에 앞서, 사용자가 서비스를 사용할 때, 명확한 이유의 예외 결과를 알려주는 것이 중요하다고 생각했기 때문에 수강신청 프로그램에서 예외 처리에 힘을 쏟았다는 점을 강조한다.

서비스는 크게 로그인/회원 가입, 학생 정보, 강의 정보, 수강 신청을 구현했다. 서비스에 대해 자세히 설명하기 전에 공통적으로 사용하는 코드부터 설명하고자 한다.

1. 공통 코드

* Course.java, Student.java Reservation.java는 Client, Service, Data에서 동일한 코드를 사용한다.
* Server의 성공/실패 여부를 Enumeration으로 받아오기 때문에 동일한 예외처리 Class를 갖고 있다.
* Log도 Log File 생성 이외의 Class는 Client, Server, Data 모두 동일하다
* 각각의 메소드는 Log에 학번을 넣어 File에 저장하는 코드를 항상 갖고 있다.

🡪 ***LOG***.info(logUser);

Course.java [Client/Server/Data]

텍스트이(가) 표시된 사진

자동 생성된 설명

Reservation.java [Client/Server/Data]

텍스트이(가) 표시된 사진

자동 생성된 설명

Student.java [Client/Server/Data]

텍스트이(가) 표시된 사진

자동 생성된 설명

Student.java의 경우 Data 프로젝트 내의 소스 파일과 Client의 파일이 살짝 다르다. Student의 비밀번호는 File에 저장되어 toString()으로 불러오면 안 되기 때문에 Data에서는 허용하되, Client, Server에서는 허용되지 않도록 아래처럼 변경해주었다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Data에서는 File을 초기화 해준 다음, 다시 List 값을 복사하기 때문에 password 부분이 필요하다. 따라서, Client와 Server에서는 기존의 Student.java toString() 함수에서 this.password 부분을 제외시켜준 것이다.

StringReturnException.java [Client/Server]

Enum으로 된 Exception Class이다.

텍스트이(가) 표시된 사진

자동 생성된 설명

StringReturnResponse<T>.java [Client/Server]

Server에서 서비스 성공/실패 여부를 다음 객체를 통해 return을 해준다. 이에, Client와 Server 모두 위 Class를 저장하고 있어야 한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

Constructor를 두개 만들어서 실패한 경우 StringReturnException, 성공한 경우 Result 값을 return 하여 Client에서 확인하도록 설계했다.

LogFormat.java [Client/Server/Data]

텍스트이(가) 표시된 사진

자동 생성된 설명

Log에는 학번, 시간, Class의 이름과 Method의 이름이 필요하다. 따라서 Log Formatter를 상속받아 사용하고 싶은 코드로 변경하는 작업이 필요하다.

NullDataException.java

Student, Course List의 File이 없는 경우 처리하는 Exception이다. String ReturnException을 사용하기 때문에 따로 Exception을 상속받아 사용하는 코드는 많이 없지만, 상속받은 Exception 코드를 경험해보고자 사용해보았다.

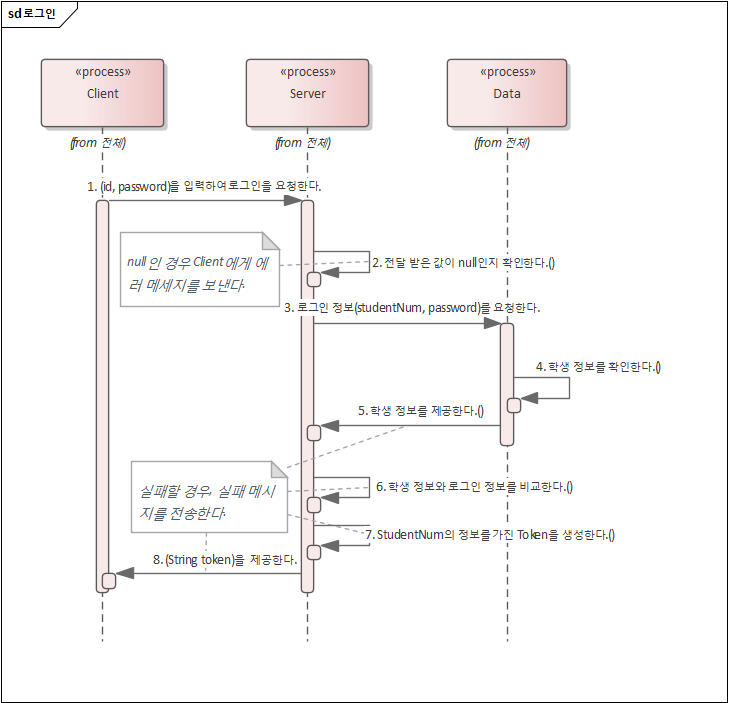
텍스트이(가) 표시된 사진

자동 생성된 설명

1. 로그인/회원가입

회원 가입과 로그인의 경우 사용자가 직접 자신의 정보를 작성하여 Client가 서버에 전달하기 때문에 값이 전달되지 않는 경우 데이터에 값을 작성하기 이전에 예외처리를 진행해야 한다. 로그인, 회원가입 이후, 수강신청 서비스를 제공받을 수 있기 때문에 각 서비스에 접근하여 로그인한 사용자와 접근하려는 사용자가 일치하는 지 확인하는 Token도 생성해준다.

로그인



로그인 EA는 다음과 같다. EA 시나리오 순으로 아래에 코드를 작성할 것이다.

1. [Client: Client.java] – main

텍스트이(가) 표시된 사진

자동 생성된 설명

프로그램을 실행하면 Log를 실행해줘야 하며, 그 이후 login/sign-up이 될 때까지 loop문을 돌리면 된다.

1. [Client: Client.java] – logConfigurationMethod

텍스트이(가) 표시된 사진

자동 생성된 설명

로그 File을 생성하는 Method로 (1)- 에서의 LogFormat을 File에 저장하도록 한다. Console이 아닌, File에 저장할 것이므로 FileHandler를 사용해야한다.

1. [Client: Client.java] – login

텍스트이(가) 표시된 사진

자동 생성된 설명

로그인의 경우 학번과 비밀번호를 Server에게 전달한 후, login이 되었다면 학번을 Return 해주고 아니면 null을 return 해준다. 연결된 server에서 return result는 token이므로, token을 저장해준다. server에 연결 될 경우 method parameter로 전달해 줘야한다.

1. [Server: Server.java] – login

텍스트이(가) 표시된 사진

자동 생성된 설명

Client에서 받은 정보가 null인지 확인하며, Login이 되었다면 token을 생성해준다. 이 과정 속에서 문제가 발생하게 되면 바로 Client에 에러 메시지를 전송한다. 이 때, 에러 메시지는 앞서 설명한 Exception class 자체를 return 한다.

1. [Server: AES128.java] – AES128, encrypt

텍스트이(가) 표시된 사진

자동 생성된 설명

Token 생성의 경우 Spring의 JWT와 완전히 동일한 메커니즘으로 생성을 하지 못했다. 하지만, Token 내에 User의 정보(학번)가 저장되어 있기 때문에 암호화를 통해 Token을 생성하여 Client에게 전달해준다.

Token과 학번의 정보는 암호화된 코드들로 비교할 예정이기 때문에 복호화가 필요가 없다.

1. [Server: Server.java] – createToken

텍스트이(가) 표시된 사진

자동 생성된 설명

Login의 extract method로, AES128를 사용하여 학번 정보를 암호화한다.

1. [Data: Server.java] – checkLogin

텍스트이(가) 표시된 사진

자동 생성된 설명

Data에서 전달받은 학번과 비밀번호를 가진 학생이 있는지 확인한다.

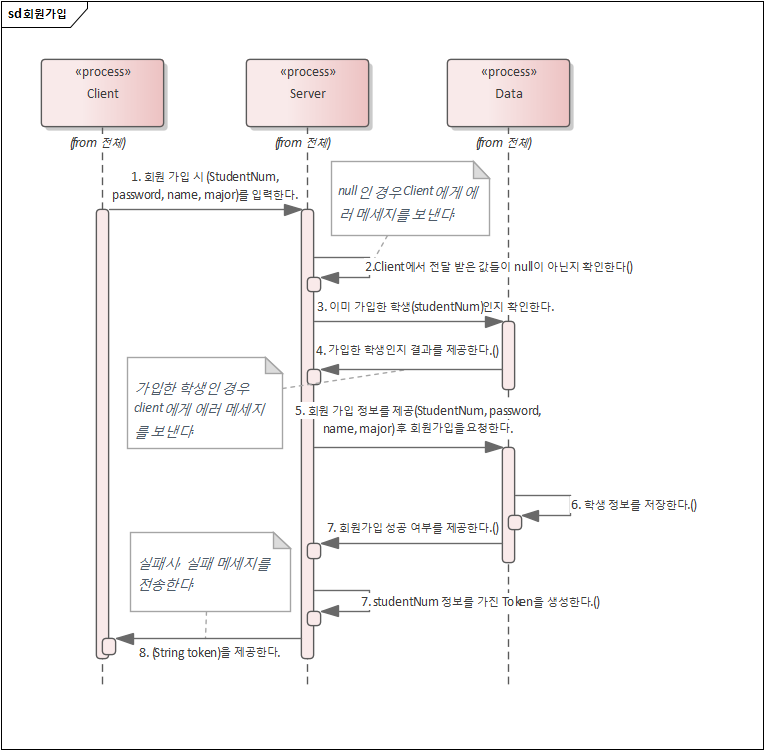
1. [Data: StudentList.java] – checkStudent

텍스트이(가) 표시된 사진

자동 생성된 설명

StudentList에 저장된 정보와 입력 받은 데이터를 확인하여 로그인 여부를 Server에 전달한다.

회원가입



회원가입 EA는 다음과 같다. EA 시나리오 순으로 아래에 코드를 작성할 것이다.

1. [Server: Server.java] – signUP

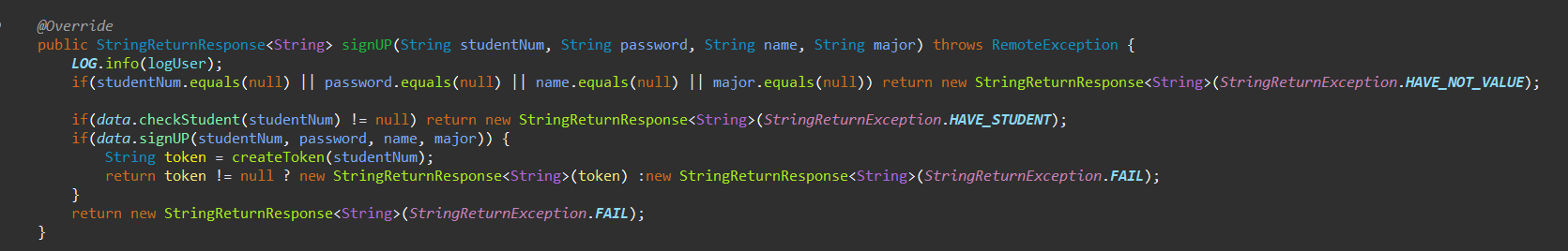
텍스트이(가) 표시된 사진

자동 생성된 설명

로그인의 경우 학번과 비밀번호를 Server에게 전달한 후, login이 되었다면 학번을 Return 해주고 아니면 null을 return 해준다. 연결된 server에서 return result는 token이므로, token을 저장해준다. token의 경우 server method parameter로 전달해 줘야한다.

Log와 관련된 내용은 로그인 부분에서 언급했으므로 넘어가도록 한다.

1. [Server: Server.java] – signUP



Client에서 받은 정보가 null인지 확인하며, 회원가입이 되었다면 token을 생성해준다. 이 과정 속에서 문제가 발생하게 되면 바로 Client에 에러 메시지를 전송한다. 이 때, 에러 메시지는 앞서 설명한 Exception class 자체를 return 한다.

로그인과 달리, 회원가입이므로 이미 가입되어 있는 학번은 다시 가입할 수 없다. 따라서, Data에 학번 중복조회가 필요하다. 이 또한 문제가 발생하게 되면 바로 Client에 에러 메시지를 전송한다.

Token 생성 또한 위 로그인 단계에서 작성했기 때문에 따로 코드를 첨부하지 않고 넘어간다.

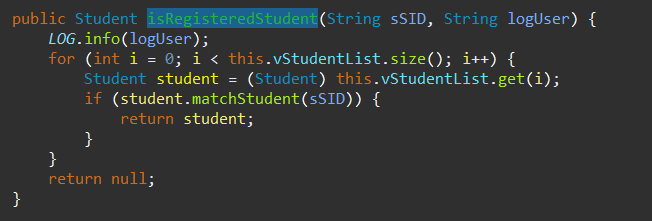
1. [Data: Server.java] – checkStudent

텍스트이(가) 표시된 사진

자동 생성된 설명

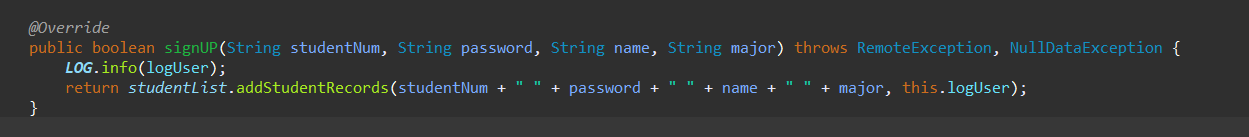
전달받은 데이터가 학생 데이터에 저장이 되어 있는지 아닌지를 확인하는 메소드이다.

1. [Data: StudentList.java] – isRegisterdStudent



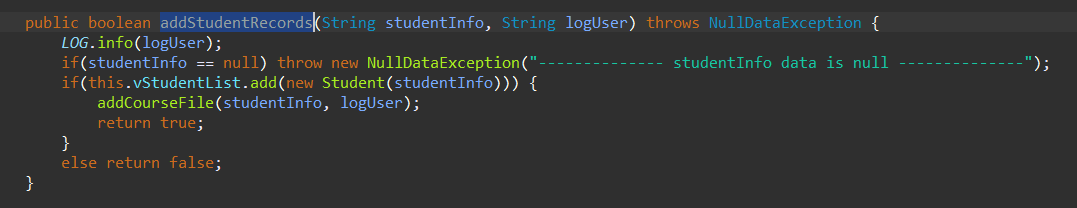
checkStudnet와 다른 점은 학번만 비교한다는 점이다. 학번만 비교하여 이미 학생 정보가 저장되어 있다면 null을 return 한다.

1. [Data: Server.java] – signUP



Server로부터 정보를 받아와 File 및 ArrayList에 저장하고자 한다. 실제로 저장하는 method는 studentList 내부에 저장되어 있다.

1. [Data: StudentList.java] – addStudentRecords



ArrayList에 회원가입 한 학생 정보를 저장하고, 실제 File에도 저장을 해줘야한다. 실제 File에 저장하기 위해 extract method인 addCourseFile을 생성했다. List와 File에 모두 저장이 되었다면 return true를 해준다.

1. [Data: StudentList.java] – addCourseFile

텍스트이(가) 표시된 사진

자동 생성된 설명

BufferedWriter를 사용하여 기존에 저장된 TextFile에 정보를 붙여 저장해둔다.

로그인과 회원가입이 완료되면 학번을 main에 return을 해준다. return의 값이 null 이 아니라면 수강 신청 서비스로 이동하며 Log를 하기 위한 User 설정을 return 한 학번으로 저장한다. 또한 서버와 Data에도 학번을 전달해준다. 이와 관련된 코드는 다음과 같다.

[Client: Client.java] – main

텍스트이(가) 표시된 사진

자동 생성된 설명

어디에서든 사용할 수 있도록 속성을 저장한 후, Server에 학번 정보를 넘겨준다.

[Server: Server.java] – sendServerStudentForLog

텍스트이(가) 표시된 사진

자동 생성된 설명

Client와 마찬가지로 어디서든 사용할 수 있는 속성을 저장하고, Data에 학번 정보를 넘겨준다.

[Data: Data.java] – sendServerStudentForLog

텍스트이(가) 표시된 사진

자동 생성된 설명

1. 학생 정보

학생 리스트

학생 등록

학생 삭제

1. 강의 정보

강의 리스트

강의 등록

강의 삭제

1. 수강 신청

수강 신청

수강 신청 리스트

1. 학생 정보

학생 리스트 학생 삭제

학생 등록

1. 강의 정보
2. 수강 신청
3. 코드 설명
4. 실행 결과
5. 결론

이번 프로젝트에서 가장 아쉬웠던 점은 Token 생성 기능이었다. 프로젝트에서 Spring의 Json Web Token과 완전히 동일한 구조의 Token을 생성하고 싶었지만, 완전한 기능을 만들지 못했다는 점이 아쉬웠다. 하지만, 이번 프로젝트에서 생성한 Token은 암호화를 도와주는 AES를 활용하여 외부에 studentNum이 노출되지 않으면서도 복호화 시, 학생의 정보를 확인할 수 있도록 하여 최대한 비슷하게 구현했다.

또한, RMI로만 실습을 진행하고 grpc를 사용해보지 못했던 점이 아쉬웠다. 현업에서는 RMI보다 grpc를 더욱더 선호하고 많이 사용한다고 하기에, RMI를 완전히 익히면 grpc도 사용해보고자 한다.

RMI 수강신청 프로젝트는 프로젝트 개발 및 EA, 보고서까지 작성하면서 Client와 Server 간의 정보 요청 및 제공의 메커니즘을 명확하게 알게 된 좋은 프로젝트였다. Client와 Server의 통신에 관심을 갖고 인터넷을 검색하더라도 글로만 작성되어 있는 경우가 더러 있어 쉽게 이해할 수 없는 개념들도 많았지만 이번 프로젝트를 통해 많은 것들을 알게 되었다. 또한, 하나의 프로젝트 생성으로 연결된 것이 아닌 RMI Registry를 통해 연결되어 사용자의 요청 서비스가 화면에 보이게 된다는 점이 신기했다.

교수님 과의 멘토링 활동은 method 이름을 명확하게 사용해야하는 점과 같은 피드백을 다시 되새기며 프로젝트에 임해 여태까지의 다른 코드보다 훨씬 더 깔끔하다라는 생각을 갖게 되는 시간이었다.