소프트웨어 개발의 원리 및 실제 - 실습 과제 3

Mar. 29

- **과제는 두 가지입니다**
  과제 A. Django REST 를 이용한 채무관계 기록 백엔드 서버 개발 (퍼미션 추가)
  과제 B. React / Redux 를 이용한 오목 게임 개발

A. Django REST 과제 – 채무관계 기록 백엔드 개발 (퍼미션 추가)

1) 설명

과제 2번에 이어서, 사람 간의 채무관계를 기록하는 백엔드 서버에 퍼미션 기능을 추가해 본다.

새로운 채무를 생성(`POST debts/`)할 때에는 현재 유저가 자동으로 **borrower**로 등록되도록 하고, 나머지 채무 정보는 모두 POST로부터 직접 받는다.

기존의 채무를 삭제(`DELETE debts/id/`) 할 때에는, 현재 유저가 반드시 해당 채무의 **lender**와 같아야 한다.

특정 채무를 보는 것 (`GET debts/id/`)은 현재 유저가 해당 채무의 **lender** 혹은 **borrower**와 같아야 한다.

특정 유저 정보를 보는 것 (`GET users/id/`, `GET usersum/id/`)은 **현재 유저가 해당 유저**와 동일해야 한다.

`debt_admin`이란 이름을 가진 특수한 유저가 있어서, 이 유저들은 위의 연산(`POST debts/`, `GET debts/id/`, `DELETE debts/id/`, `GET users/id/`, `GET usersum/id/`)이 모두 가능해야 한다. 채점시 `debt_admin` 유저는 스크립트가 자동으로 생성해 줄 것이다. `debt_admin` 유저는 다른 유저와 이름만 다를 뿐, 특별한 권한(예 : Django 관리자 권한)을 갖거나 하지는 않는다.

기존의 채무를 수정(`PUT debts/id/`) 하는 것은 반드시 `debt_admin` 유저만 가능하다.

전체 채무 목록(`GET debts/`) / 유저 목록(`GET users/`) / usersum 목록(`GET usersum/`) 을 보는 것은 반드시 `debt_admin`이란 이름을 가진 유저만이 가능하다.

B. React / Redux 를 이용한 오목 게임 개발

<- restart 버튼

-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	**O**	-	-	-	-
-	-	-	-	**X**	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-	-

status label ->

보드

1. 보드 크기 19 × 19
2. 아무것도 들어가있지 않은 칸은 – 로 표시(처음 시작 시 모든 칸은 -)
3. 홀수 번째가 O, 짝수 번째 X
4. 바둑판의 줄무늬 및 테두리는 보이지 않아도 된다.
5. 보드의 칸 하나하나마다 id 가 있어야 하며 이때 id 이름은 y 축 좌표와 x 축 좌표를 붙여 쓴 형태이다.
   ex) 0_0, 0_1, 0_2, ...
   1_0, 1_1, 1_2, ... 좌상단의 id 가 0_0

Status label

1. Label 안에 표시되는 내용은 4 가지 - Next O, Next X, O win, X win
2. 각 상황에 맞게 label 내에 표시되는 내용이 변한다.
   A. 다음 턴의 차례를 표시 할 때 - Next O, Next X
   B. 게임의 승자가 정해질 시 승자를 표시할 때 - O win, X win
3. Status label 의 id 는 status_label

RESTART 버튼

1. 누르면 판과 status label 이 초기화 상태로 돌아가야한다.
2. 이 후 다시 새로운 게임 진행이 가능해야 한다.
3. Restart 버튼의 id 는 restart

기타사항

1. 게임의 승자가 정해진 이후에는 보드에 바둑돌을 더 놓을 수 없다.
2. 보드의 각 칸, restart 버튼, status label 은 모두 위처럼 id 를 가져야 한다.

제출 및 채점

숙제 A.

채점은 수업 및 실습 시간에 공지된 바와 같이, 학생들이 제출한 bitbucket 리포지토리를 사용해 자동 채점을 할 것이다. **자동 채점은 4 월 5 일 수요일 00 시 00 분부터 시작한다.**

초기환경 세팅은 숙제 2 를 참조한다. 제출은 자신의 bitbucket 개인 리포지토리를 이용한다.
~/debtpages 를 아래와 같이 통째로 복사한 후, **assn03_A** 라는 branch 로 push 한다

```
(env) swpp@ip-172-31-30-158:~/swpp-assn201701$ cp -r ../debtpages/ .
(env) swpp@ip-172-31-30-158:~/swpp-assn201701$ ls
debtpages ..... (다른파일,ex. mysite)
(env) swpp@ip-172-31-30-158:~/swpp-assn201701$ ls debtpages/
debtpages  debtpages_rest  manage.py  .... (다른 파일)
```

채점에 사용할 스크립트는 이번주 중에 공지한다.

숙제 B.

채점은 수업 및 실습 시간에 공지된 바와 같이, 학생들이 제출한 bitbucket 리포지토리를 사용해 자동 채점을 할 것이다. **자동 채점은 4 월 8 일 토요일 00 시 00 분부터 시작한다.**

초기환경 세팅은 숙제 2 를 참조한다. 제출은 자신의 bitbucket 개인 리포지토리를 이용한다.
~/omak 를 아래와 같이 통째로 복사한 후, **assn03_B** 라는 branch 로 push 한다

```
(env) swpp@ip-172-31-30-158:~/swpp-assn201701$ cp -r ../omak/ .
(env) swpp@ip-172-31-30-158:~/swpp-assn201701$ ls
 omak  ..... (다른파일,ex. mysite)
```

채점에 사용할 스크립트는 이번주 중에 공지한다.

이번 숙제부터는 채점 환경에서 테스트가 원활하게 돌지 않은 경우, 0 점 처리됩니다.

Appendix. Django REST 과제에서 구현해야 할 페이지의 예시

127.0.0.1:8000/debts/	127.0.0.1:8000/debts/id (id는 숫자)
채무 관계의 리스트	채무 관계 id의 정보

Django REST framework

Debt List

Debt List

```
GET /debts/
```

```
HTTP 200 OK
Allow: GET, POST, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 11,
        "created": "2017-03-21T19:51:49.152749Z",
        "amount": 352,
        "borrower": 76,
        "lender": 79
    },
    {
        "id": 12,
        "created": "2017-03-21T19:51:49.163834Z",
        "amount": 263,
        "borrower": 78,
        "lender": 77
    },
    {
        "id": 13,
        "created": "2017-03-21T19:51:49.175327Z",
        "amount": 808,
        "borrower": 78,
        "lender": 75
    },
    {
        "id": 14,
        "created": "2017-03-21T19:51:49.186390Z",
        "amount": 518,
        "borrower": 81,
        "lender": 78
    },
    {
        "id": 15,
        "created": "2017-03-21T19:51:49.197827Z",
        "amount": 393,
```

Django REST framework

Debt List / Debt Detail

Debt Detail

```
GET /debts/12/
```

```
HTTP 200 OK
Allow: PUT, GET, DELETE, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 12,
    "created": "2017-03-21T19:51:49.163834Z",
    "amount": 263,
    "borrower": 78,
    "lender": 77
}
```

users/ 에 접속할 시 나오는 데이터	**users/id 에 접속할 시 나오는 데이터**
전체 유저의 목록	아이디가 id인 유저의 정보

Django REST framework

User List

User List

```
GET /users/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "username": "admin",
        "debts_as_borrower": [],
        "debts_as_lender": []
    },
    {
        "id": 74,
        "username": "test1",
        "debts_as_borrower": [],
        "debts_as_lender": [
            15
        ]
    },
    {
        "id": 75,
        "username": "test2",
        "debts_as_borrower": [
            18
        ],
        "debts_as_lender": [
            13
        ]
    },
    {
```

Django REST framework

User List / User Detail

User Detail

```
GET /users/74/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 74,
    "username": "test1",
    "debts_as_borrower": [],
    "debts_as_lender": [
        15
    ]
}
```

127.0.0.1:8000/usersum/	127.0.0.1:8000/usersum/id
전체 유저의 목록, 단 빌린 돈과 빌려준 돈을 보여 줘야 함	아이디가 id인 유저의 정보, 단 빌린 돈과 빌려준 돈을 보여줘야 함

Django REST framework

User Sum List

User Sum List

GET /usersum/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept


```
[
    {
        "username": "admin",
        "id": 1,
        "lended_money": 0,
        "borrowed_money": 0
    },
    {
        "username": "test1",
        "id": 74,
        "lended_money": 393,
        "borrowed_money": 0
    },
    {
        "username": "test2",
        "id": 75,
        "lended_money": 808,
        "borrowed_money": 846
    },
    {
        "username": "test3",
        "id": 76,
        "lended_money": 0,
        "borrowed_money": 352
    },
    {
        "username": "test4",
        "id": 77,
        "lended_money": 2151,
        "borrowed_money": 605
```


Django REST framework

User Sum List / User Sum Detail

User Sum Detail

GET /usersum/75/

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept


```
{
    "id": 75,
    "borrowed_money": 846,
    "username": "test2",
    "lended_money": 808
}
```