

1. 이번 과제에서는 모델이 최대한 적은 정보를 가질수록 좋습니다. 모델에 너무 많은 정보를 저장하게 되면, 언제부터 정보가 잘못 저장된 시점을 추적하기 힘들어집니다. 이번 과제에 필요한 모델은 아래 정도로 충분합니다.

```
1 from django.db import models
2
3 class Room(models.Model):
4     player1 = models.ForeignKey('auth.User', null=True, related_name='room_as_player1',
on_delete=models.CASCADE)
5     player2 = models.ForeignKey('auth.User', null=True, related_name='room_as_player2',
on_delete=models.CASCADE)
6
7 class History(models.Model):
8     player = models.ForeignKey('auth.User', on_delete=models.CASCADE)
9     room = models.ForeignKey(Room, related_name='history', on_delete=models.CASCADE)
10    place_i = models.IntegerField()
11    place_j = models.IntegerField()
```

2. ModelSerializer 이외의 다른 클래스를 상속해서 serializer를 구현할 수 있습니다. BaseSerializer는 자기만의 독창적인 serializer를 구현할 수 있도록 해줍니다. 아래는 /rooms/id/players 를 나타낼 수 있는 serializer입니다.

```
29 class PlayerSerializer(serializers.BaseSerializer):
30     def to_representation(self, obj):
31         l1 = ([ obj.player1.id ] if obj.player1 else [])
32         l2 = ([ obj.player2.id ] if obj.player2 else [])
33         return l1 + l2
34
35     class Meta:
36         model = Room
```

<http://www.django-rest-framework.org/api-guide/serializers/#baseserializer> 를 참조해봅시다.

3. View를 구현할 수 있는 방법은 여러 가지가 있었습니다. Django REST에서 제공하는 클래스를 상속하는 방법으로 할 수가 있고, Response를 반환하는 함수를 직접 구현할 수도 있습니다. URL에 따라서 적절한 방법을 선택해서 구현해봅시다. 아래는 /rooms/id/players 에 대해서 GET/POST를 수행하는 view입니다.

```
25 @api_view(['GET', 'POST'])
26 def room_players_list(request, pk):
27     try:
28         room = Room.objects.get(pk = pk)
29     except Room.DoesNotExist:
30         return Response(status = status.HTTP_404_NOT_FOUND)
31
32     serializer = PlayerSerializer(room)
33     if request.method == 'GET':
34         return Response(serializer.data)
35     elif request.method == 'POST':
36         if room.player1 == None:
37             room.player1 = request.user
38             room.save()
39         elif room.player2 == None:
40             room.player2 = request.user
41             room.save()
42     else:
43         return Response(status=status.HTTP_403_FORBIDDEN)
44     return Response(status=status.HTTP_201_CREATED)
```

