

# **oTree (1)**

**2017.7.14 @ KU**

# 목표

- 대상: 경제학 실험을 하고자 하는 연구자
- 목표
  - oTree로 실험 구현
- 본 슬라이드는 다음 사이트에서 최신 버전을 유지하고 있음
  - <https://github.com/z0nam/oTreeBasic>
- 준비물: 네트워크가 되는 PC (Windows 10 이상 권장)

# 주제 (1차)

- 실험 플랫폼 개관
- oTree
- 예제: 죄수의 딜레마 게임 (1차)
  - 매뉴얼의 두번째 예제
  - <http://otree.readthedocs.io/en/latest/tutorial/part1.html>
- 과제: [codeacademy.com](https://www.codecademy.com)에서 다음 코스 마스터하기
  - <https://www.codecademy.com/learn/python> (필수)
  - <https://www.codecademy.com/learn/learn-html-css> (옵션)
  - <https://www.codecademy.com/learn/learn-javascript> (옵션)

# 이후 계획

- 2차:
  - 심화예제: 설문 결과에 따라 조를 배정하는 최후통첩게임
  - [https://github.com/z0nam/gntech\\_game](https://github.com/z0nam/gntech_game)
  - 과제: 게임을 설계하여 otree 프로젝트 만들기
- 3차:
  - 웹서버에 내가 만든 게임을 올려보기

# 사회과학 실험 플랫폼

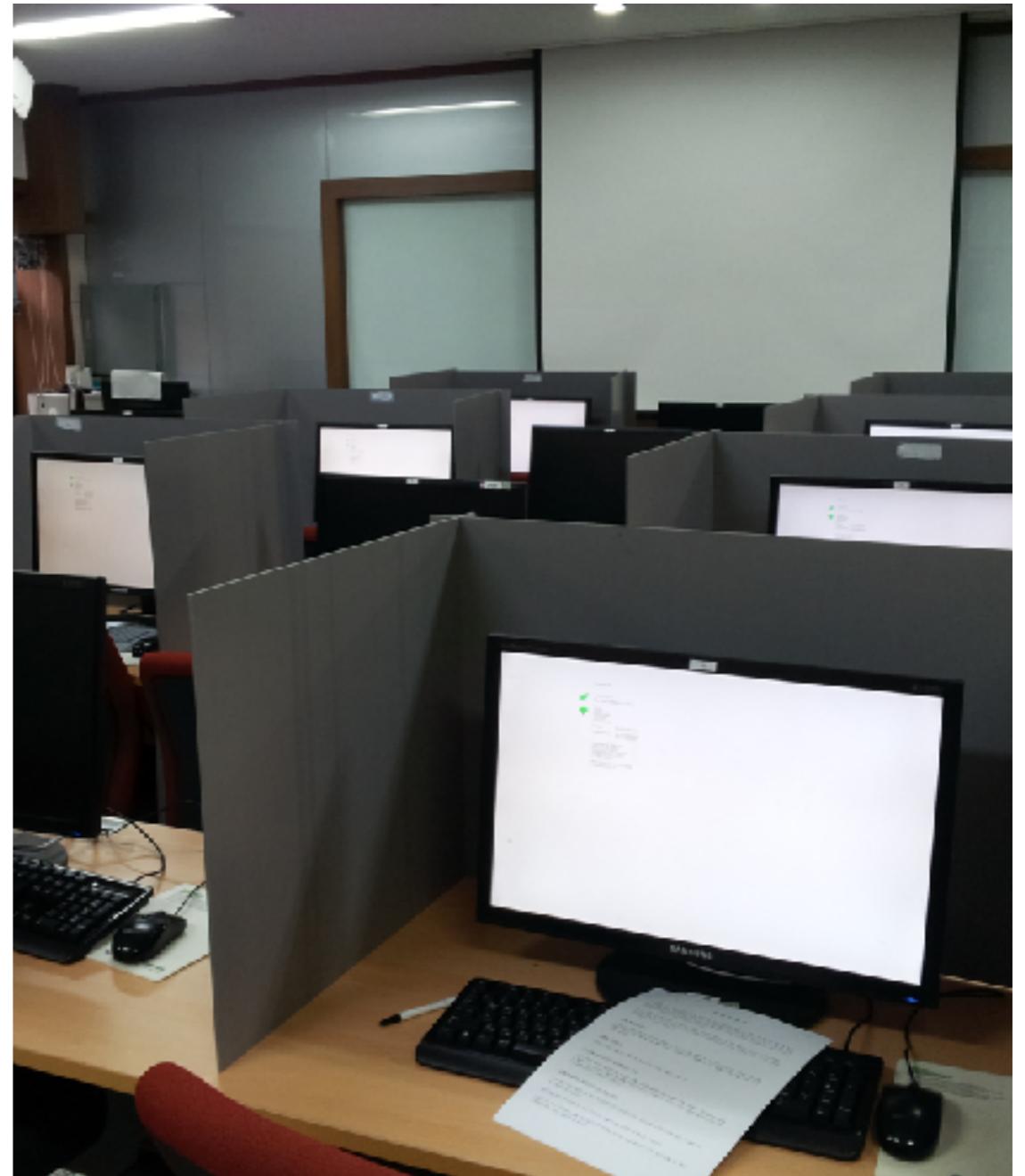
	PRO	CON	ex
종이	간편	시공간적 제약	종이설문지
웹폼	대량설문 가능	Fixed form	Google Forms
zTree	랩실험 가능	구현한계	
oTree	웹실험 가능	django 지식필요	
MTurk	대량웹실험	피실험자 통제 불가능	
전용앱	유연함	개발 어려움, 구현불안정	

# 실험 플랫폼의 요건

- 참가자의 응답에 따라 실험 설계자가 원하는 메시지가 정확하게 전달되어야 함
  - fixed form (구글폼, 종이 등)으로는 불가능
- 실제 실험상황에서 모니터링, 보상 지급 등이 실시간으로 가능해야 함
  - 네트워크 필요
- 참가자들이 가능한한 쉽게 접근할 수 있어야 함
  - zTree는 PC가 설치되어 있는 환경에서만 가능

# 현재의 기술기반

- 지금까지 가장 많이 사용하고 있는 사회과학 실험 플랫폼은 zTree임.
  - 주로 랩실험으로 PC실에서 실행함
- paper → PC → mobile device
- 대부분의 피실험자들은 wifi 접속 가능한 mobile device를 가지고 있음
  - 20대 스마트폰 보급률 ≈ 100% (2015)
  - 웹실험은 스마트폰으로 실행 가능



# oTree

- <http://www.otree.org/>
  - 소스코드: <https://github.com/oTree-org/oTree>
  - 이름은 윈도우 기반 플랫폼 zTree에서 따온 것으로 보임
  - 웹기반 오픈 소스 실험 플랫폼
    - 행동실험
    - 설문
      - 복잡한 구조를 가지는 경우 적합



# oTree citation

- If you publish research conducted using oTree, you are required by the oTree license to cite this paper.
- Citation:
  - Chen, D.L., Schonger, M., Wickens, C., 2016. oTree - An open-source platform for laboratory, online and field experiments. *Journal of Behavioral and Experimental Finance*, vol 9: 88-97

# oTree 구현을 위한 준비물

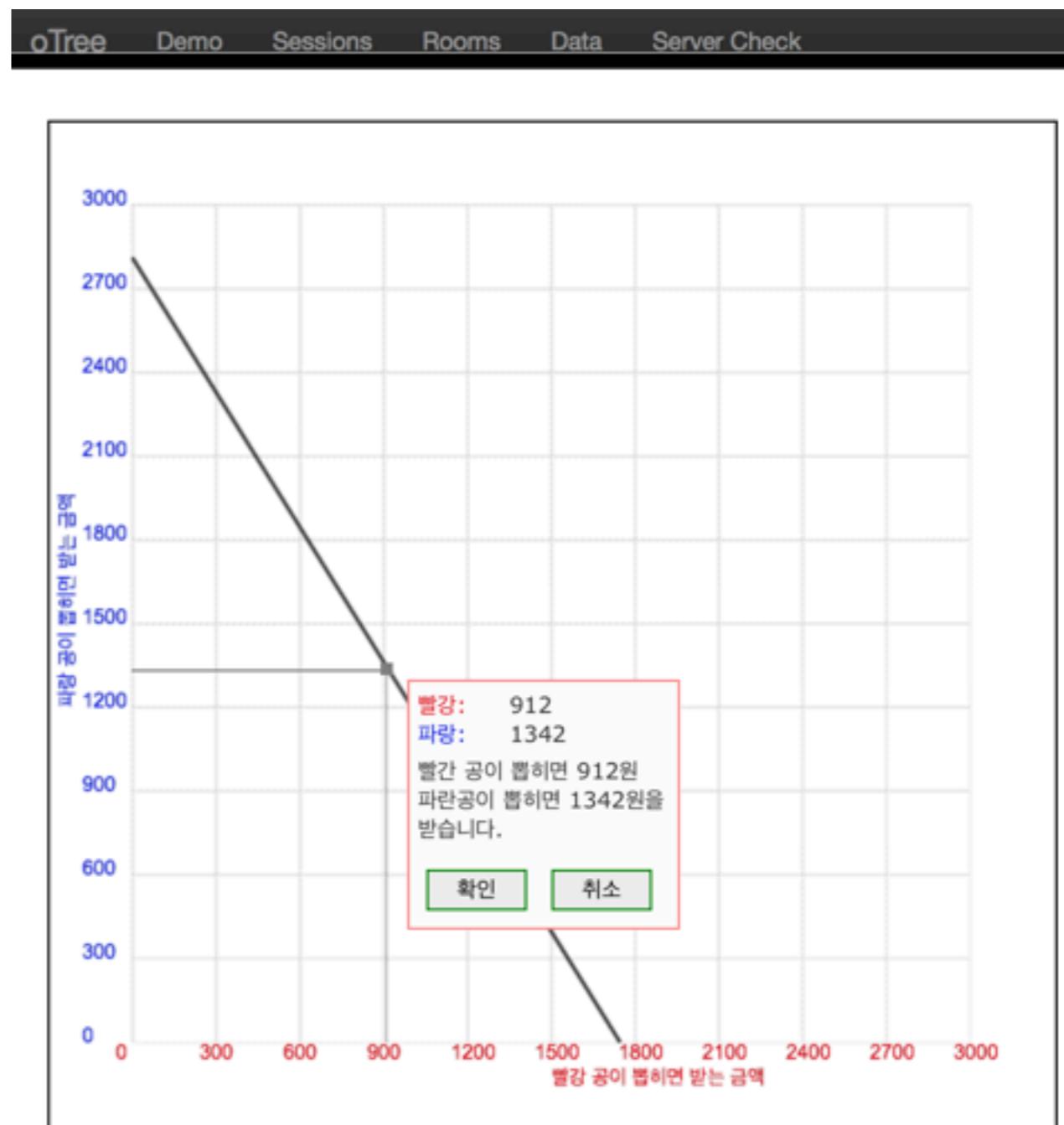
- 네트워크가 가능한 PC
  - Windows: 10 이상 권장 (powershell)
  - Linux, MacOS..
  - Python3, database (MySQL, MariaDB, postgre 등), otree-core 가 설치되어 있거나 설치할 수 있어야 함

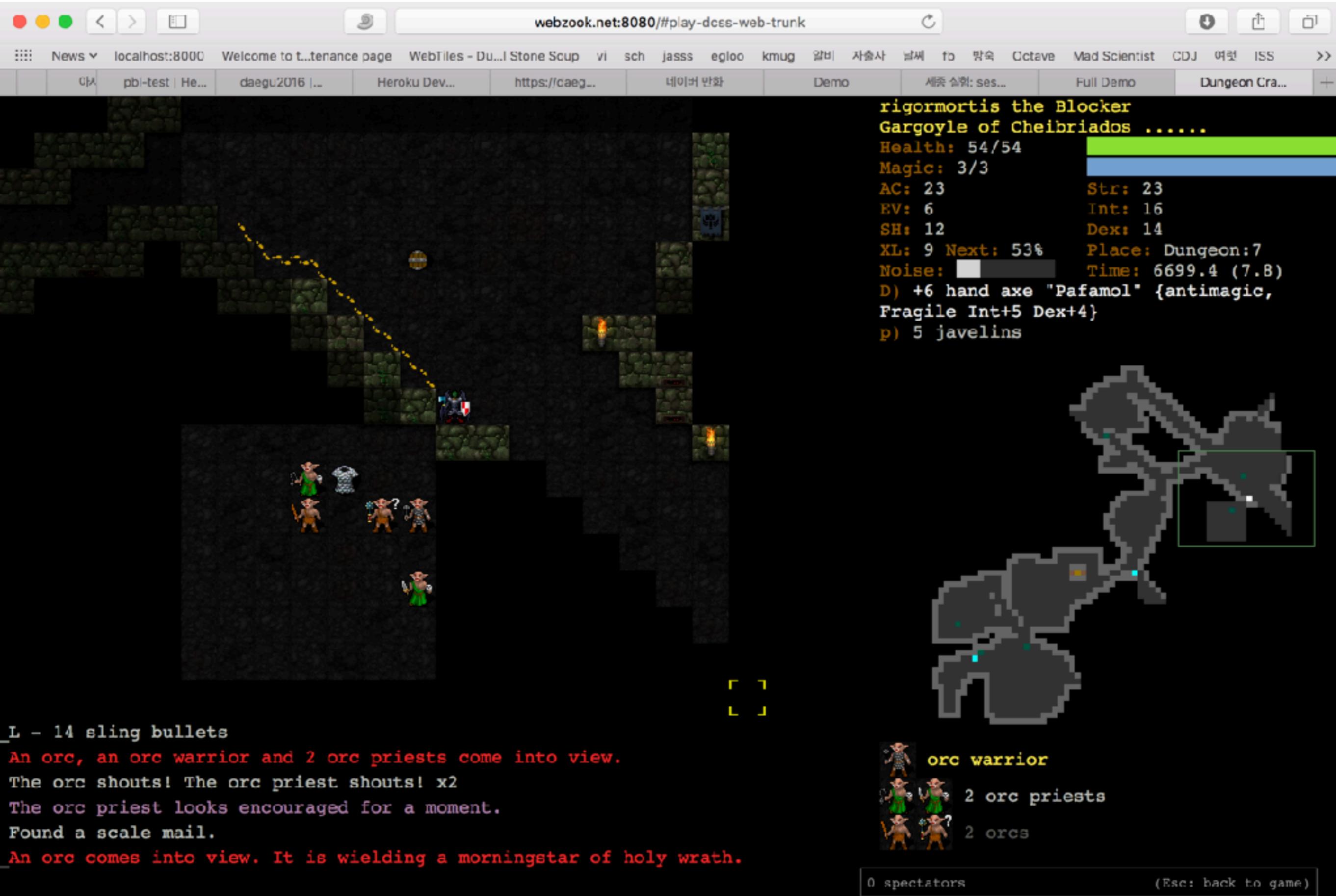
# oTree를 위한 사전지식

- oTree는 Python 웹서버 프레임워크인 django (장고)로 구현되어 있음. 따라서
  - Python 언어를 사용할 수 있어야 함
  - django 프레임워크를 이해하는 것이 필요함
  - Python과 database가 설치된 웹서버 운영이 가능해야 함
  - Command Line Interface (CLI) 소위 ‘도스창’ 사용법
- oTree는 실험에 공통적으로 쓰이는 많은 부분들을 구현해두어 필요한 프로그래밍을 최소화는 것을 목표로 하고 있지만 최소한의 Python, django 에 대한 지식은 필요함

# 선택 지식

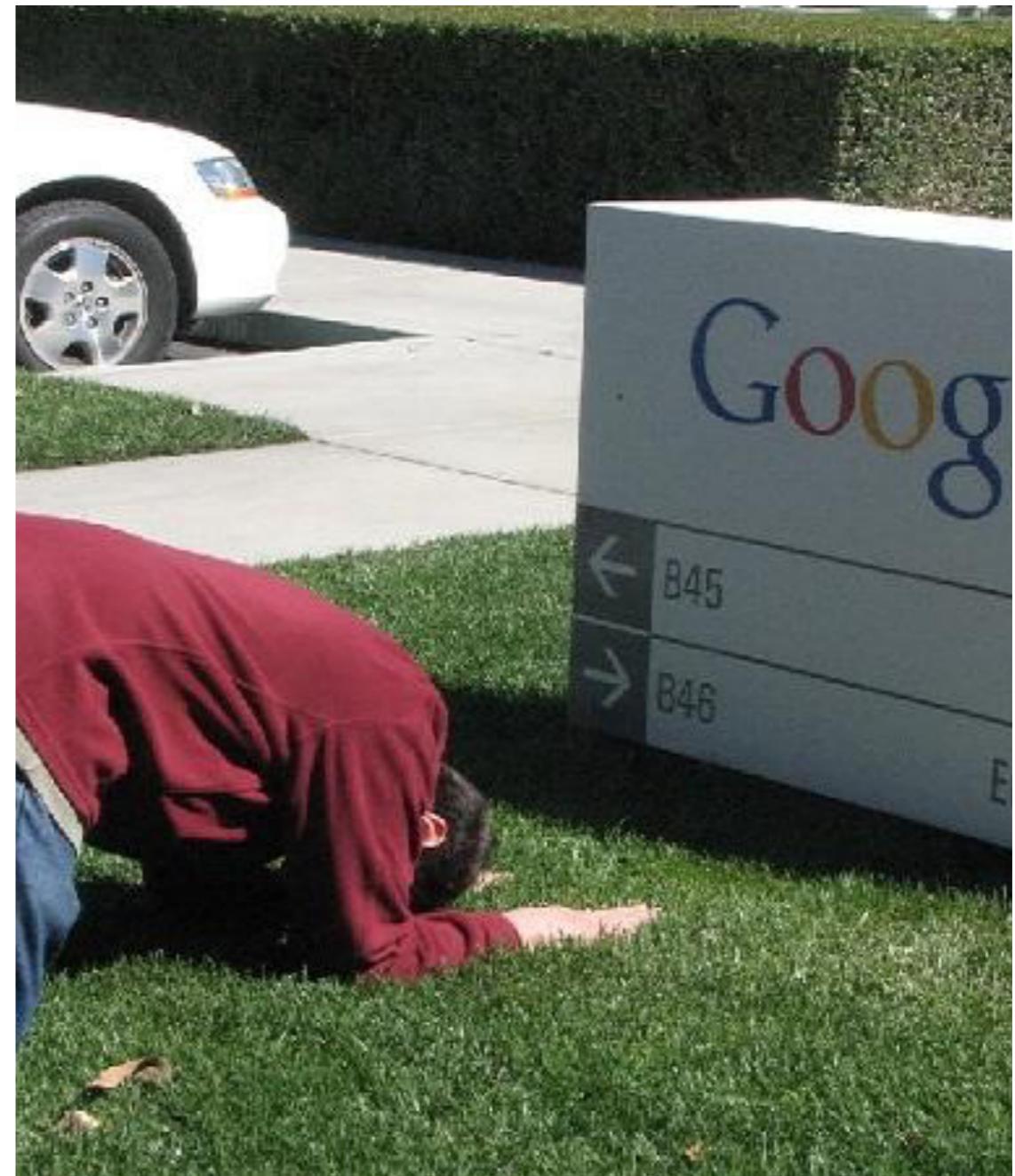
- Javascript (자바 아님)
  - 로컬 (브라우저)에서 즉각적으로 대응할 수 있게 해줌
  - 간단한 컴퓨터 게임 구현도 가능
  - <https://www.codecademy.com/learn/javascript>
- HTML, CSS
  - 브라우저의 가장 기본적 언어
  - <https://www.codecademy.com/learn/learn-html-css>





# 매뉴얼 및 첫걸음을 위한 링크

- oTree
  - <http://otree.readthedocs.io/en/latest/>
- Python
  - <https://www.python.org/about/gettingstarted/>
  - <https://www.codecademy.com/learn/python> (완전 초보일 경우 강력추천)
- django
  - <https://www.djangoproject.com/start/>
  - <https://docs.djangoproject.com/en/1.11/intro/tutorial01/>
- Google God, stackoverflow



# 실습시 유의사항

- \$ 으로 시작하는 부분은 CLI 명령어임 (터미널이나 커맨드창 등에서 실행)
  - \$ 를 뺀 나머지 부분을 입력 (bash 기준)
  - [[ ]] 로 둘러싸여 있는 부분은 자신이 정해서 입력하라는 뜻 (대괄호까지 입력하면 안됨)
- 폴더 위치, 표현 형식 (C:\tmp\, /usr/local/bin ) 등은 다를 수 있으므로 항상 유의
- 매뉴얼대로 돌아가지 않는 경우는 대체로 해당 명령어의 경로가 설정되지 않았거나 설치되지 않았을 때가 많음

# 설치

- OS에 따라 조금씩 다름
- Python3 → otree-core 설치
  - otree-core 설치시 필요한 패키지 (django 등)는 자동 설치됨
  - 개인 테스트용으로는 서버/클라이언트 모두 개인 PC를 사용
    - 실제 실험시에는 웹서버를 사용해야 함
  - 개인 테스트용 database는 SQLite를 사용 (otree와 함께 설치됨)
  - 실제 실험시에는 웹서버에 database가 설치되어 있어야 함.

# 설치

/

- Python3 설치
  - 깔려있는지 알고 싶다면: \$ python3 -V
  - <https://www.python.org/downloads/>
- oTree 설치
  - \$ pip3 install -U otree-core
  - otree manual “Installing oTree” 참고

#update

# 에디터, 통합개발환경(IDE)

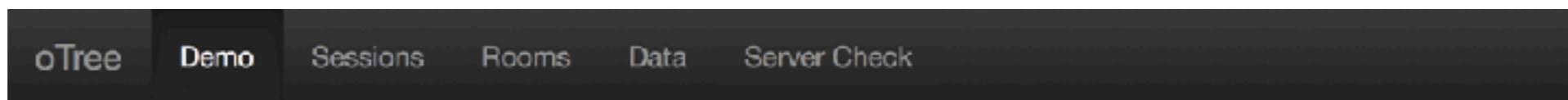
- notepad 로도 되지만, 불편함
- IDE
  - 강력한 기능, 하지만 내부 동작 메커니즘을 잘 이해 못할 수도 있음
  - pyCharm, eclipse ..
- editor
  - 프로그래밍용 notepad 정도로 생각하시면 되겠음
  - ATOM, notepad++ ..
- VIM, EMACS.. : 쉘에서 작동하는 에디터. 진입장벽 높지만 익숙해지면 생산성은 가장 높음  
(bash 에 기본 설치되어 있음)

# The first oTree Project

- \$ cd [[oTree 설치할 폴더]]
- \$ otree startproject [[만들 oTree 프로젝트이름(foo)]]
- \$ cd foo
- \$ otree resetdb
- \$ otree runserver
- 제대로 되었다면 브라우저 주소창에 localhost:8000 넣어볼 것.

cd .. / #upper directory

# The first oTree project



## Demo

- [oTree on GitHub](#) .
- [oTree homepage](#) .

Here are various games implemented with oTree. These games are all open source, and you can modify them as you wish.

You can add entries to this list in [settings.py](#) .

Public Goods	Participants: 3
Trust Game	Participants: 2
Guess 2/3 of the Average	Participants: 3
Survey	Participants: 1
Quiz	Participants: 1
Quiz and Public Goods	Participants: 2

# Main menu

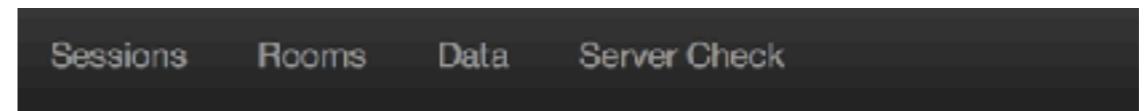
oTree   Demo   Sessions   Rooms   Data   Server Check

- oTree: intro page
- Demo: == intro page
- Sessions: 세션 모니터 페이지      *설정 Set*
- Rooms: 참가자 관리 페이지
- Data: 실험 응답 데이터 관리 페이지
- Server Check: 서버 상태에 대한 간단한 상태보고

# 전체 실험 절차

# Sample Games

- 샘플 게임 설치를 허가할 경우  
oTree에서 제공하는 샘플 게임들이  
구현 가능한 상태로 기본 설치됨.
- Demo 의 Prisoners' Dilemma  
를 클릭해보자.



er's Dilemma: session [eb41gi5f](#) (c

tion [Links](#) [Monitor](#) [Data](#) [\\$ Payment](#)

porary links for testing and demonstration. To launch a real study, either create a session or create a session through the [sessions page](#).

open full-screen mode, the session-wide link, or the single-use links.

mode

en mode.

de link

v link in up to 2 browser tabs.

<http://localhost:8000/join/q96ubq7upv/>

links

in its own browser tab.

<http://localhost:8000/InitializeParticipant/dxi5cufa/>

# demo game

## Prisoner's Dilemma: session **wjix952v (demo)**

 Description

 Links

 Monitor

 Data

 Payments

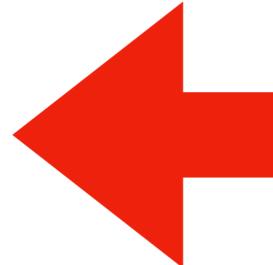
 New

Below are temporary links for testing and demonstration. To launch a real study, either create persistent links by setting up a [room](#), or create a session through the [sessions page](#).

You can either open full-screen mode, the session-wide link, or the single-use links.

### Full screen mode

[Play in full screen mode.](#)



### Session-wide link

Open the below link in up to 2 browser tabs.

<http://localhost:8000/join/p72wu7j3hq/>

### Single-use links

Open each link in its own browser tab.

---

P1 <http://localhost:8000/InitializeParticipant/9cefmls1/>

P2 <http://localhost:8000/InitializeParticipant/icf4z42e/>

# Full screen mode

Prisoner's Dilemma: session [wjix952v \(demo\)](#)

The screenshot shows a web-based experiment interface. At the top, there is a navigation bar with tabs: Description, Links, Monitor (which is highlighted in blue), Data, Payments, and New. A red arrow points from the Korean text "관리자 화면" (Administrator View) to the Monitor tab. Below the navigation bar is a table with the following data:

ID in session	Code	Label	Page	App	Round	Page name	Status	Time on page
P1	9cefmls1		1/5 pages	prisoner	1	Introduction	Playing	<1 min ago
P2	icf4z42e		1/5 pages	prisoner	1	Introduction	Playing	<1 min ago

Below the table, there are two side-by-side sections labeled "Introduction". Each section has a red speech bubble pointing towards it, labeled "P1 화면" and "P2 화면" respectively. At the bottom of each "Introduction" section is a yellow bar with the text "Time left to complete this page: ⏳ 1:02".

# Monitor Screen

Prisoner's Dilemma: session [fmp0e7xo](#) (demo)

>Description    Links    **Monitor**    Data    Payments    New

ID in session	Code	Label	Page	App	Round	Page name	Status	Time on page
P1	0ch5o0ku		5/5 pages	payment_info	1	PaymentInfo	Playing	~2 hr ago
P2	kmooou66h		5/5 pages	payment_info	1	PaymentInfo	Playing	~2 hr ago

Advance slowest user(s)

식별코드

현재 진행상황

진행단계

가장 낮은 진행유저를 다음 단계로 강제진행

# 실험데이터의 저장 (xlsx, csv)

Prisoner's Dilemma: session fmp0e7xo (demo)

Description

Links

Monitor

Data

Payments

New

ID in session	Prisoner					
	Player			Group	Subsession	
	ID in group	Role	Decision			
P1	1		Cooperate	200 points	1	1
P2	2		Cooperate	200 points	1	1

You can download data in Excel or CSV format [here](#).



# 주의사항

- SQLite를 쓰는 경우 서버 호스팅 정책(특히 heroku)에 따라 데이터가 증발할 수 있으니 실험 끝나면 즉시 데이터 다운받아두는 습관을 기르자.
- SQLite는 테스트용으로만 사용할 것.
  - Postgres, MySQL 등을 사용할 것을 강력히 권장함.  
→ 안정성↑

## Data Export

### Citation requirement

If you publish research conducted using oTree, you are required by the oTree license to cite [this paper](#).

### Citation:

Chen, D.L., Schonger, M., Wckans, C., 2016. oTree - An open-source platform for laboratory, online and field experiments. Journal of Behavioral and Experimental Finance, vol 9: 88-97

### All apps

[Excel](#) | [CSV](#)

Data for all apps in one file. There is one row per participant; different apps and rounds are stacked horizontally. This format is useful if you want to correlate participants' behavior in one app with their behavior in another app.

### Per-app

These files contain a row for each player in the given app. If there are multiple rounds, there will be multiple rows for the same participant. This format is useful if you are mainly interested in one app, or if you want to correlate data between rounds of the same app.

App	Data	Documentation
Ultimatum	<a href="#">Excel</a>   <a href="#">CSV</a>	<a href="#">CSV</a>
Payment Info	<a href="#">Excel</a>   <a href="#">CSV</a>	<a href="#">CSV</a>
Prisoner	<a href="#">Excel</a>   <a href="#">CSV</a>	<a href="#">CSV</a>

Time spent on each page

[Download](#)

왜 까다롭게  
NASH?

# Result File

	A	B	C	D	E	F	G	H	I	J	
1	participant_id_in_session	participant_code	participant_label	participant_is_bot	participant_index_in_pages	participant_max_page_index	participant_current_app_name	participant_count_number	participant_current_page_name	participant_ip_address	participant_time
2	1	dx5wfa		0	0	5					
3	2	20pe2ll1		0	0	5					
4	1	mlbcem3		0	0	5					
5	2	k5dy76lk		0	0	5					
6	1	9cefmls1		0	2	5	prisoner		1 Decision	127.0.0.1	2017-07-13 0
7	2	ic4n42e		0	2	5	prisoner		1 Decision	127.0.0.1	2017-07-13 0
8	1	0jh5o0ku		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
9	2	kmoou66h		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
10	1	0c5M4jz7		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
11	2	9zqeruck		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
12	3	t9o0ephr		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
13	4	6ao4tks		0	5	5	payment_info		1 PaymentInfo	127.0.0.1	2017-07-13 1
14											
15											

- 앱의 모든 결과가 저장되어 있음.
- 이것을 원하는 구조로 가공하여 통계 패키지로 export할 것.

# Time Spent Table

*Time  
Nash*

	A	B	C	D	E	F	G	H	I	J
1	session id	participant id in session	participant code	page index	app name	page name	time stamp	seconds on page	subsession pk	auto submitted
2	3		1 Scefmls1		1 prisoner	Introduction	1499932728	100	3	TRUE
3	3		2 icf4z42e		1 prisoner	Introduction	1499932728	100	3	TRUE
4	4		1 ikseazc5		1 ultimatum	Introduction	1499942824	31	1	FALSE
5	4		1 ikseazc5		2 ultimatum	Offer	1499942842	18	1	FALSE
6	4		1 ikseazc5		3 ultimatum	WaitForProposer	1499942852	10	1	FALSE
7	4		1 ikseazc5		6 Ultimatum	ResultsWaitPage	1499942874	22	1	FALSE
8	4		1 ikseazc5		7 ultimatum	Results	1499942889	15	1	FALSE
9	4		2 xbp5eb9g		1 ultimatum	Introduction	1499942852	59	1	FALSE
10	4		2 xbp5eb9g		3 ultimatum	WaitForProposer	1499942852	0	1	FALSE
11	4		2 xbp5eb9g		4 ultimatum	Accept	1499942874	22	1	FALSE
12	4		2 xbp5eb9g		6 Ultimatum	ResultsWaitPage	1499942874	0	1	FALSE
13	4		2 xbp5eb9g		7 ultimatum	Results	1499942890	16	1	FALSE
14	5		1 Cch5oCku		1 prisoner	Introduction	1499942963	22	4	FALSE
15	5		1 Cch5oCku		2 prisoner	Decision	1499942973	10	4	FALSE
16	5		1 Cch5oCku		3 prisoner	ResultsWaitPage	1499942977	4	4	FALSE
17	5		1 Cch5oCku		4 prisoner	Results	1499942986	9	4	FALSE
18	5		2 kmoou66h		1 prisoner	Introduction	1499942971	30	4	FALSE
19	5		2 kmoou66h		2 prisoner	Decision	1499942976	5	4	FALSE
20	5		2 kmoou66h		3 prisoner	ResultsWaitPage	1499942976	0	4	FALSE
21	5		2 kmoou66h		4 prisoner	Results	1499942984	8	4	FALSE
22	6		1 Ck5f4jz7		1 prisoner	Introduction	1499950724	13	5	FALSE
23	6		1 Ck5f4jz7		2 prisoner	Decision	1499950728	4	5	FALSE
24	6		1 Ck5f4jz7		3 prisoner	ResultsWaitPage	1499950750	22	5	FALSE
25	6		1 Ck5f4jz7		4 prisoner	Results	1499950757	7	5	FALSE
26	6		2 Sqgoruck		1 prisoner	Introduction	1499950731	19	5	FALSE
27	6		2 Sqgoruck		2 prisoner	Decision	1499950749	18	5	FALSE
28	6		2 Sqgoruck		3 prisoner	ResultsWaitPage	1499950750	1	5	FALSE

# 참가자 보상지급

Prisoner's Dilemma: session fmp0e7xo (demo)

Description

Links

Monitor

Data

Payments

New

PDF generated: July 13, 2017, 12:35 p.m.

## Session

Session type

prisoner

Session code

fmp0e7xo

Experimenter name

참가사례비

게임보상

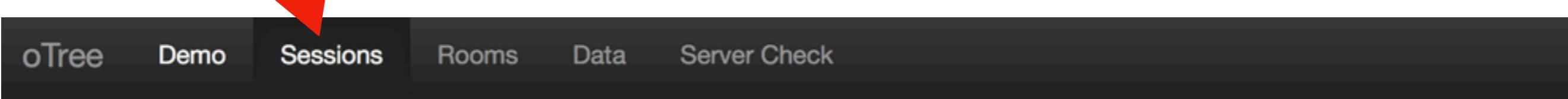
최종보상

## Participants

Participant code	Participant label	Participation fee	Payoff (bonus)	Total	Note
0ch5o0ku		\$0.00	\$0.00	\$0.00	
1mnn000166h		\$0.00	\$0.00	\$0.00	

# 세션 진행

## (데모 아닌 진짜 1 단위의 실험)



### oTree Sessions



No sessions yet. Click the button to create a new session.

+ Create new session

**Session:** 시작부터 최종보상지급까지 진행되는 한 단위의 실험

# 세션 설정

## Create a new session

Session config :

Prisoner's Dilemma

실행할 세션 (사전에 정의함)

Number of participants:

4

Must be a multiple of 2

Create

참가자의 수

2의 배수

Configure session 

### App sequence

**Prisoner** This is a one-shot "Prisoner's Dilemma". Two players are asked separately whether they want to cooperate or defect. Their choices directly determine the payoffs.

**Payment** This application provides a webpage instructing participants how to get paid. Examples are given for **Info** the lab and Amazon Mechanical Turk (AMT).

# Monitor Screen

oTree   Demo   Sessions   Rooms   Data   Server Check

## Prisoner's Dilemma: session [w7hsod1a](#)

Description

Links

Edit

Monitor

Data

Payments

You can either use the session-wide link, persistent links, or single-use links.

### Session-wide link

If it is impractical to distribute distinct URLs to each participant, you can give the following start URL to all 4 participants.

<http://localhost:8000/join/hy7fu040xn/>

Note: unlike the other link modes, this does not prevent the same person from playing twice.

### Persistent links

If you want to give your participants permanent links that don't change, you should create your session in a [room](#).

### Single-use links

Below are single-use links, which you can distribute to your participants. Each link has a unique code for the participant.

P1 <http://localhost:8000/InitializeParticipant/0k5f4jz7/>

P2 <http://localhost:8000/InitializeParticipant/9qqorucx/>

P3 <http://localhost:8000/InitializeParticipant/t9o0ephr/>

P4 <http://localhost:8000/InitializeParticipant/6jaoifks/>

참가 링크

(참가자 수만큼 생성됨)

<Object>

parent

1  
↓  
child

# 설정 메모

oTree Demo Sessions Rooms Data Sessions

Description

Links

Edit

Monitor

Label :

대그룹 PD게임 (고려대 경제학과 3학)

For internal record-keeping

Experimenter name :

조남운

For Internal record-keeping

Comment :

oTree 강좌를 위한 샘플 게임 [4인]

고려대학교

2017.7.14 19:00 시험

세션 메모

참가사례비

포인트의 환율

Sef: 4 2017.7.14

Participation fee :

50.00

\$

Real world currency per point :

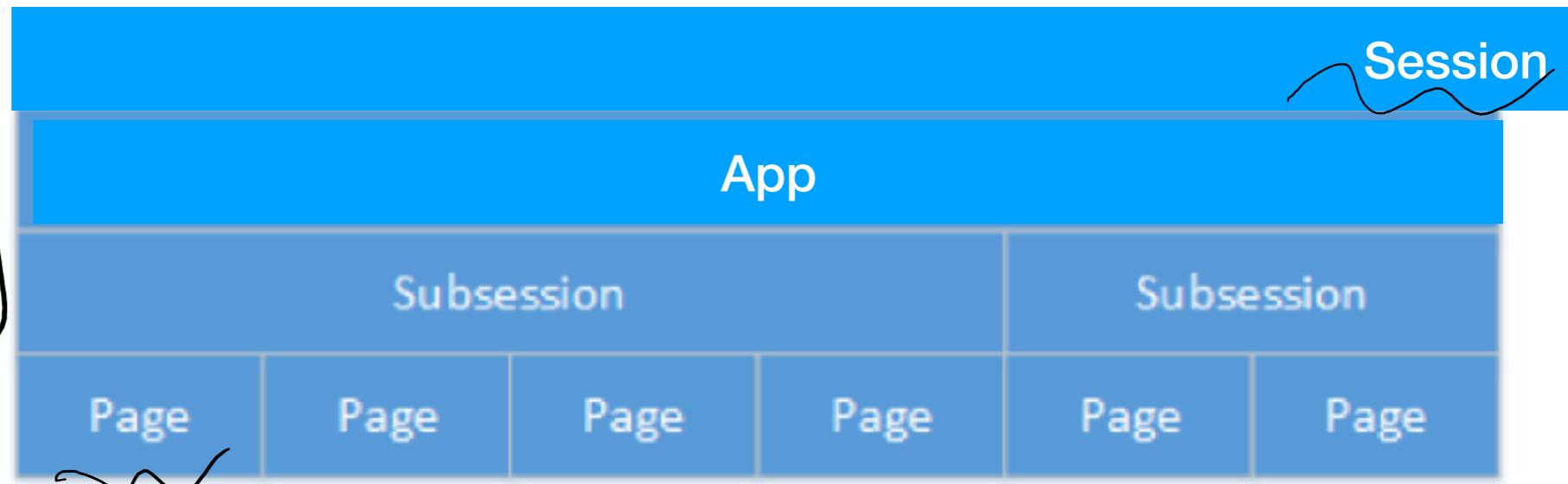
0.5

Next

# oTree의 기본 개념들

- Session

↓ Child



- App

- Subsession

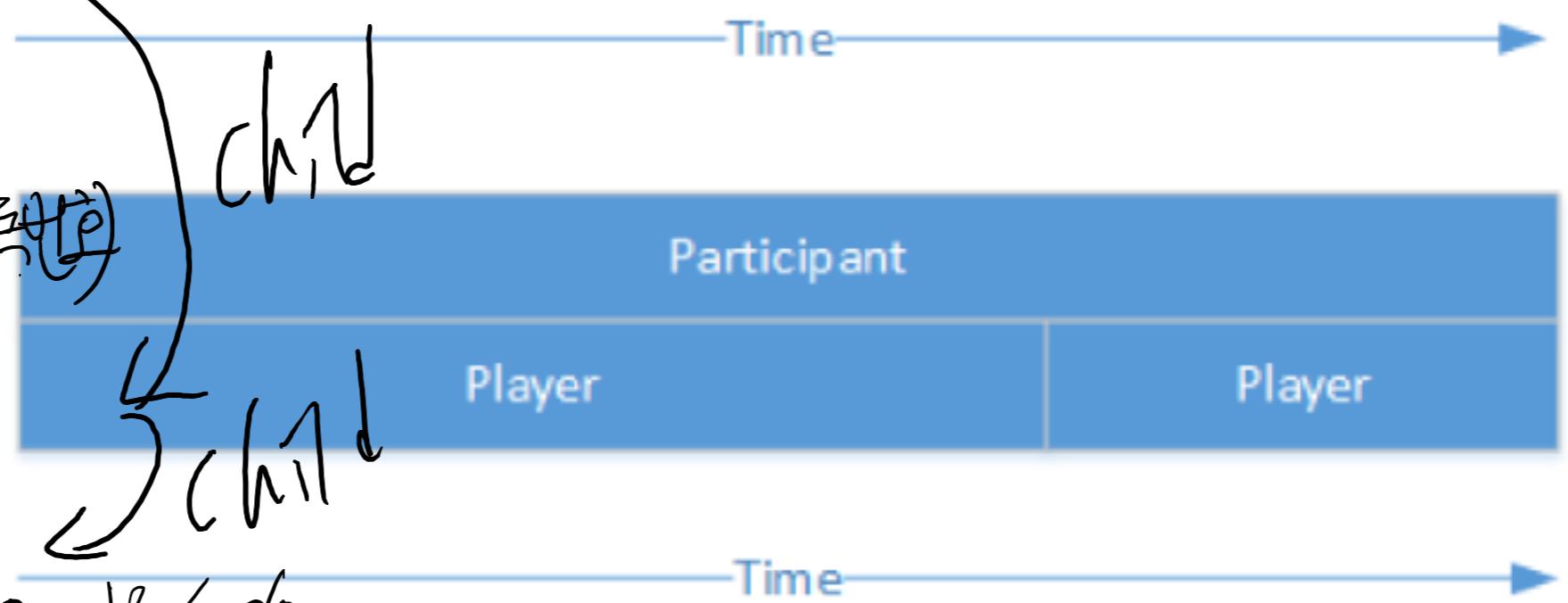
- Page

- Participant (주체)

((( • Group : player의 그룹

- Player

→ 각 게임의 player는 다른 그룹



# Session, App

- Session: 가장 큰 단위

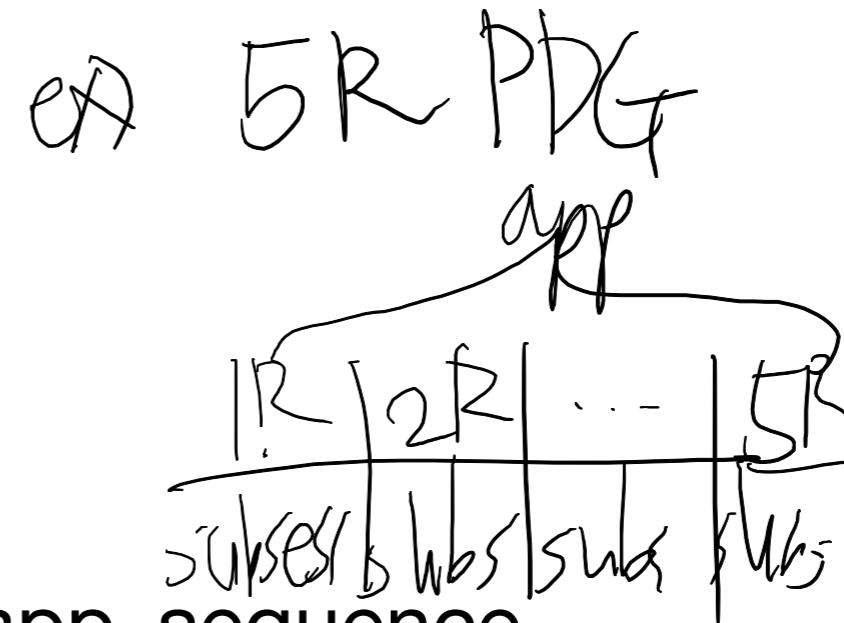
- 일련의 App으로 구성

- SESSION\_CONFIGS → app\_sequence

- 예) 2명씩 3그룹 PD 3회 실시 → 상위 50%는 제안자, 하위 50%는 응답자로 Ultimatum 5회 실시 → 설문

- 위 실험 전체가 Session

- Session = PD + Ult + Questionary (Apps)



# App, Subsession

3 Subs . 5 Subs

- 2명씩 3그룹 PD 3회 실시 → 상위 50%는 제안자, 하위 50%는 응답자로 Ultimatum 5회 실시 → 설문  
↓  
1 Sub
- PD App: PD (1라운드) → PD (2R) → PD (3R)
- 각 라운드가 Subsession에 해당

# Subsession, Page

- PD (n라운드) : Subsession
  - 전략설정화면 → 결과보여주기 화면
  - 각 화면들이 Page에 해당

# 주의

1

- 위 구분은 절대적인 것이 아님!
- “원한다면” 동일한 게임을 하나의 Session에 하나의 App으로 두고 Subsession의 연쇄로 둘 수도 있음
- 예: 2명씩 3그룹 PD 3회 실시 → 상위 50%는 제안자, 하위 50%는 응답자로 Ultimatum 5회 실시 → 설문
  - 위 Session을 하나의 App으로 두고 PD(1R) → PD(2R) → PD(3R) → Ult(1R) → ... → Ult(5R) → 설문 이라는 Subsession 집합으로도 설정할 수 있는 것임.

# 권장사항

- 재사용의 측면에서 가능한한 이질적인 게임은 App으로 구분하는  
것이 편함
- 다른 Session에서 App 단위로 불러 사용할 수 있기 때문
- 특히 설문 같은 모듈은 더더욱!

# Participant, Group, Player

- Participant

- 참가자 (ex: 조남운, 이남형, 최은철, 강영준..)
- 보상지급의 주체

- Group

- 게임 내 Player의 집합

- Player

- Subsession 내의 플레이어들

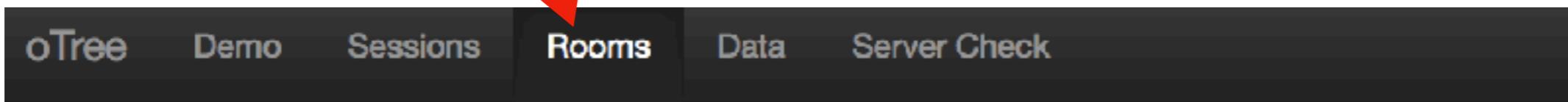
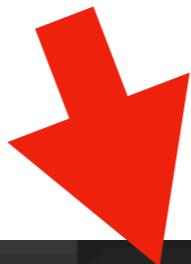
- 주의: 다른 Subsession 의 1번째 플레이어들은 다른 Participant일 수 있음. Group도 마찬가지.

A	B	C
participant.id_in_session	participant.code	participant.lab
1	dxi5cufa	
2	20pe2ll1	
1	m1lxcem3	
2	k5dy76ik	
1	9cefmls1	
2	icf4z42e	
1	0ch5o0ku	
2	kmoou66h	
1	0k5f4jz7	
2	9qqorucx	
3	t9o0ephr	
4	6jaoifks	

# Rooms

- 참가자 풀을 여러 세션에 걸쳐 유지하고 싶은 경우가 있을 수 있음
  - 예: 게임이론 강의 내내 일련의 행동실험을 유지
- Room에 Participant 정보를 축적하고 지속적으로 유지관리할 수 있음
  - 이렇게 할 경우 참가자별 URL이 아닌 id/pw로 로그인하여 실험을 수행
- Sessions가 아닌 Rooms로 세션을 실행

# Rooms로 게임실행



## Rooms



Current rooms:

Econ 101 class

[Room for live demo \(no participant labels\)](#)

---

See docs about rooms [here](#).

# Room 설정: Session과 동일

## Room: Econ 101 class

### Create a new session

Session config :

Prisoner's Dilemma

Number of participants:

36

Must be a multiple of 2

[Create](#)

### Configure session

You can make more properties configurable by adding them to your session config in `settings.py`.

participation\_fee

0.0

real\_world\_currency\_per\_point

0.0

### App sequence

**Prisoner** This is a one-shot "Prisoner's Dilemma". Two players are asked separately whether they want to cooperate or defect. Their choices directly determine the payoffs.

**Payment Info** This application provides a webpage instructing participants how to get paid. Examples are given for the lab and Amazon Mechanical Turk (AMT).

# App의 핵심 3요소

- Model: models.py
  - data structure
- View: views.py
  - action structure
- Template: html files
  - user interface

핵심 3요소

# oTree 개관

# Project

- 가장 큰 단위
- Session의 집합
- App들을 정의내리고 앱들의 조합으로 Session을 구성하는 것이라고 생각하면 되겠음
- 예) Starcraft: Project
  - 자유의 날개: Session
    - 게임1, 3으로 구성 (각 게임이 App)
  - 공허의 유산: Session
    - 게임 3, 4로 구성

```
foo3
├── Procfile
├── README.md
└── _static
    └── global
        ├── instructions.css
        └── matrix.css
└── _templates
    └── global
        ├── MTurkPreview.html
        └── Page.html
├── manage.py
├── requirements.txt
├── requirements_base.txt
└── requirements_server.txt
└── settings.py
```

# App

- \$ otree startapp [[foo2]]
- 위 명령을 실행하면 otree의 단일 앱 실행을 위해 필요한 최소 필요 파일들이 설치됨
  - 이중 가장 중요한 파일들은 붉은 색으로 강조함.

```
foo2
└── __init__.py
└── __builtin__
    └── __init__.py
└── migrations
    └── __init__.py
└── models.py
└── templates
    └── foo2
        ├── MyPage.html
        └── Results.html
└── tests.py
└── views.py
```

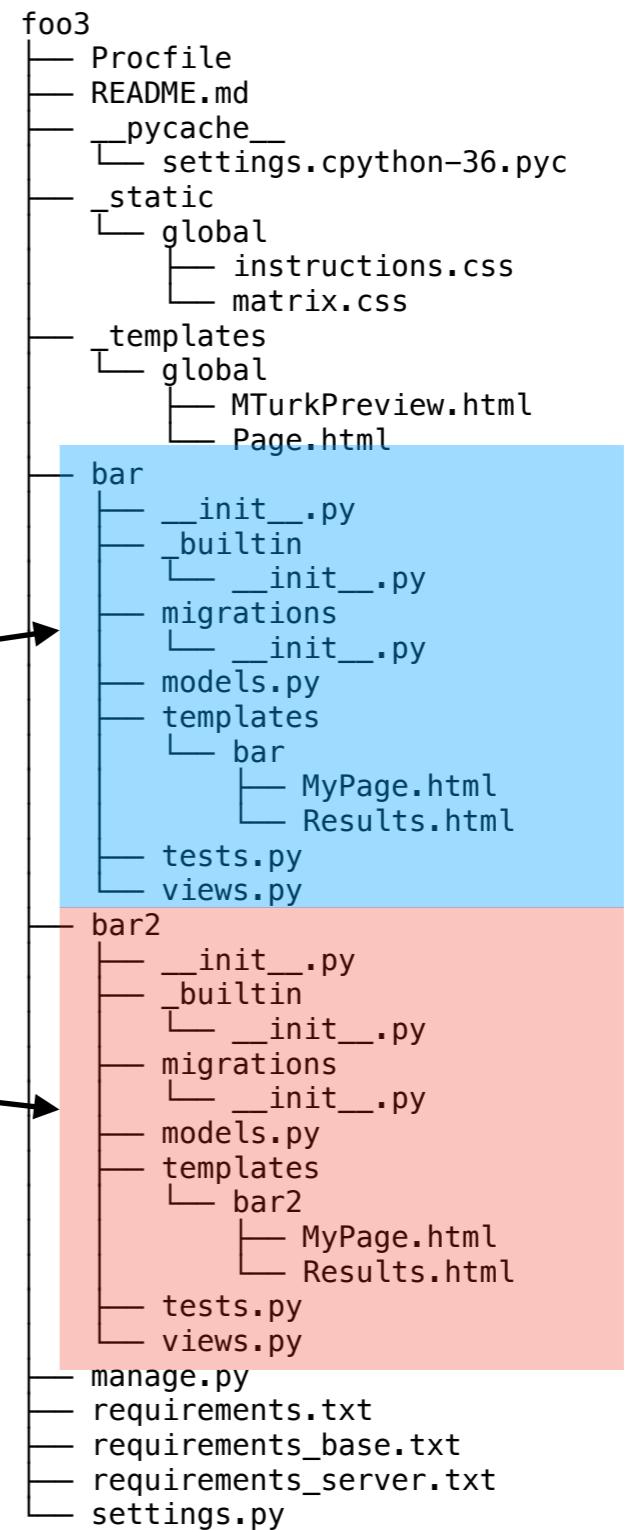
4 directories, 8 files

# 전체 구조

- foo3 project with

- bar app

- bar2 app



# Object Hierarchy

in oTree

- 모든 object들은 위계구조를 가지고 있음



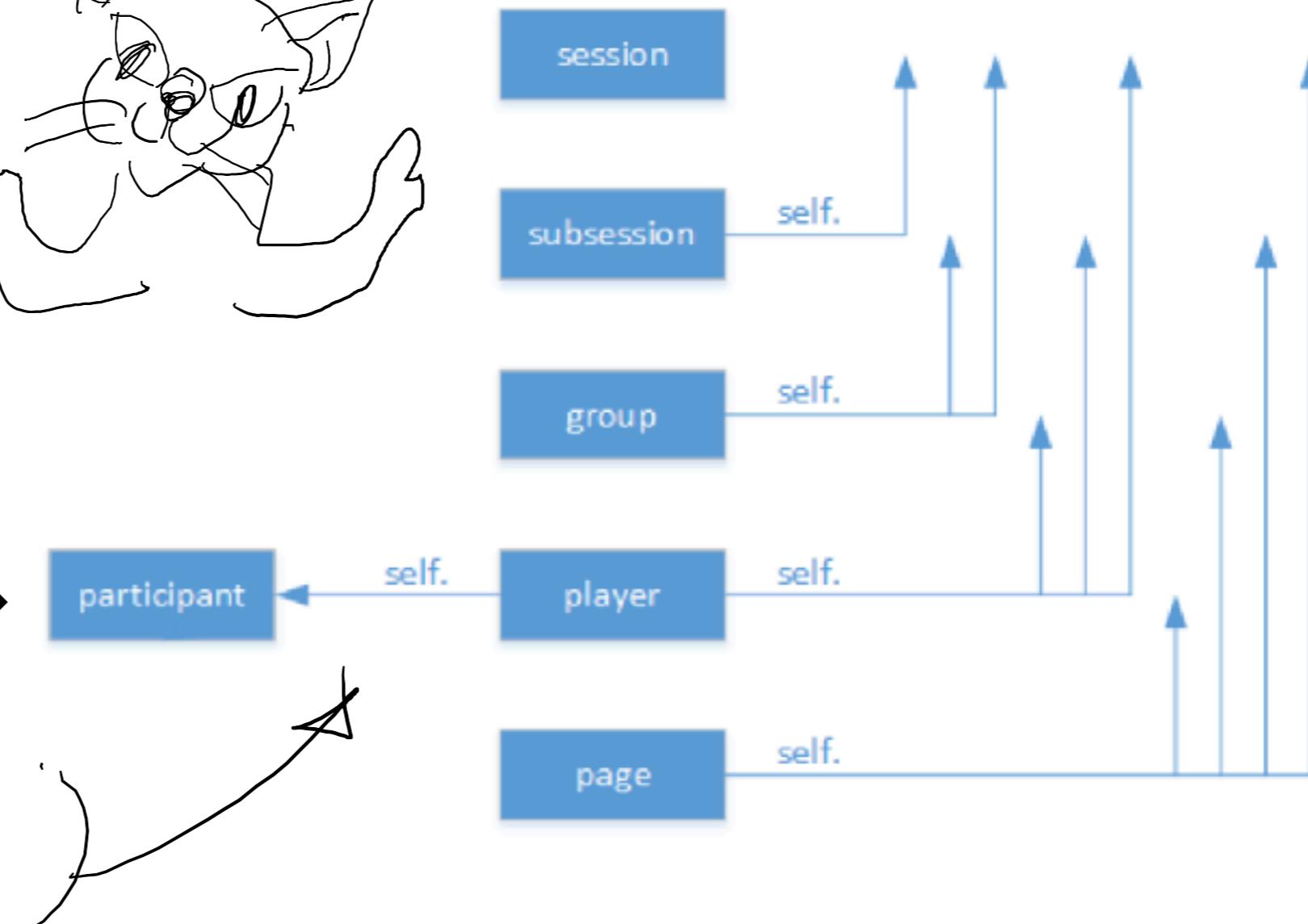
- 상위 위계구조: parent

- 하위 위계구조: child

- 자기 자신: self

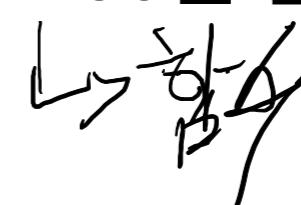
- session → subsession → group → player → page

- 한편 player는 participant  
에도 속해 있음

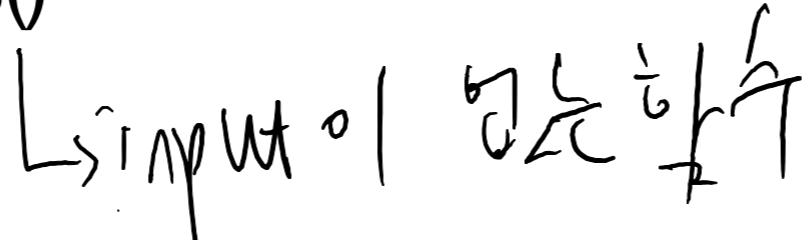


# Getting parent object

- 자신의 상위 객체 (parent object)를 불러야 할 경우가 있음
  - 형식: self.[[상위 객체]]
  - 예1: PD 3라운드에서 2번째 그룹 플레이어3의 participant object를 부르고 싶다 --> model.py Player 부분에서: self.participant,  

  - 예2: 플레이어 객체가 자신이 속해있는 page object를 부르고 싶다  
→ self.page
- 부모는 하나이므로 특별히 부모를 부르기 위한 method를 쓸 필요가 없음  


# Getting child object

- 자식은 한놈이 아닌 경우가 대부분이므로 통상적으로 자식 전체를 부른 뒤, 일처리를 함.
  - 자식 전부를 호출하는 method가 필요함
- 예: 그룹에서 플레이어 전부를 부르고 싶다 → [[in group]]  
self.get\_players()  


# models.py

- 실험 실행에 필요한 변수들의 구조를 정의
    - 여기에서 정의된 항목들이 database table의 column이 되는 것임 (실제로 \$ otree resetdb 명령이 수행하는 작업임)
      - Class → member objects, member methods (or functions) 으로 구성
    - Subsession class, Group class, Player class를 정의해야 함
      - 그 외 필요한 class가 있다면 여기에서 정의함
- Excel  
child  
must  
in model.py

# 정의 방법

- 가령 Player에 name (문자열), age (양의 정수), is\_student (이진) 속성이 필요 하다면 오른쪽과 같이 정의
- 문자열 형태일 경우:  
칼럼이름 = models.CharField()
- 양의 정수일 경우: 칼럼이름 =  
models.PositiveIntegerField()
- 이진일 경우:  
칼럼이름 = models.BooleanField()
- models. 가 붙는 이유: models 클래스  
에 정의되어 있는 함수들이기 때문  
(django)

세부 사항은 매뉴얼 참고

name	age	is_student
John	30	False
Alice	22	True
Bob	35	False
...		

Here is how to define the above table structure:

```
class Player(BasePlayer):  
    ...  
    name = models.CharField()  
    age = models.PositiveIntegerField()  
    is_student = models.BooleanField()
```

of django's member object.

# models → Constants

- 앱 내에서 사용할 상수들을 정의
- 예: PGG → MPCR, PD: payoff values
- 상수이므로 변하지 않음을 강제 → tight !!
- 만일 잠시라도 변해야 하는 값이라면 변수로 선언해야 함

error 방지를 위한

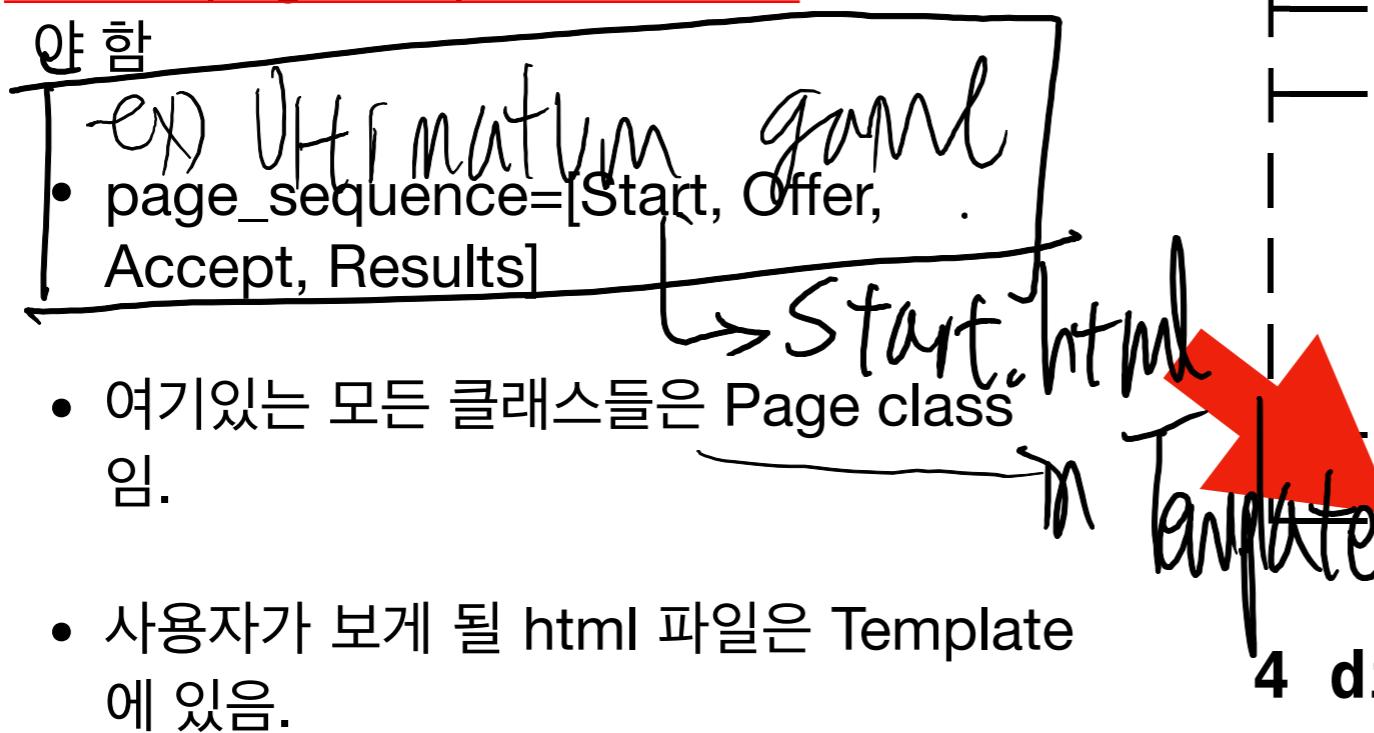
# views.py

- Page들을 관리함

- 예: 그룹 사람들이 다 할 때까지 waiting.html 을 보여주고 있다가 다 하면 payoff 계산 → result.html 보여주기.

- 반드시 page\_sequence가 정의되어 있어

야 함

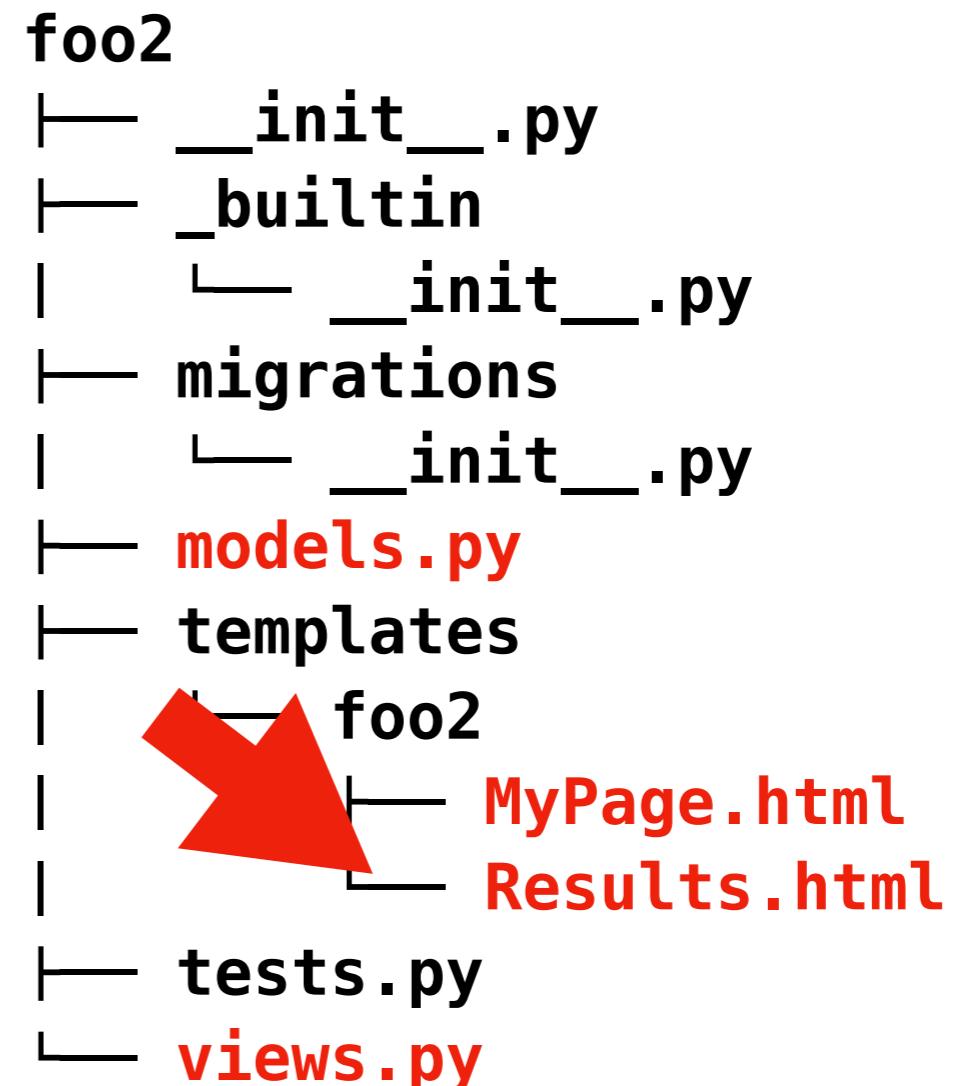


```
foo2
└── __init__.py
└── __builtin__
    └── __init__.py
└── migrations
    └── __init__.py
└── models.py
└── templates
    └── foo2
        ├── MyPage.html
        └── Results.html
└── tests.py
└── views.py
```

4 directories, 8 files

# Template

- 참가자 브라우저에 뿌리게 될 화면  
들의 html 파일들
- django 의 Template 문법 사용
  - [https://  
docs.djangoproject.com/en/  
1.8/ref/templates/language/](https://docs.djangoproject.com/en/1.8/ref/templates/language/)
  - 실질적 연산은 views.py 에서 하  
되, 불러오고 싶은 값들은  
Template에서 직접 부를 수 있  
음



4 directories, 8 files

# Public Good Game

첫번째 otree 프로젝트 만들어보기

# PGG

- 3명이 한 그룹을 이룸
- 그룹내 각 참가자들은 100만큼의 가상화폐를 지급받음
- MPCR = 100% (즉, 2배가 된다는 것을 의미)
  - efficiency factor = 2
- 참가자들은 자신의 endowment를 자신이 보유하는 금액과 기여할 금액으로 쪼갬
- 그룹내 모든 기여금을 합하여 efficiency factor 를 곱하여  $1/n$  으로 나눠줌 → *free-riding*

# Big Picture

- Preparation
- models.py
- Template/\*.html
- views.py

# Upgrade oTree

- 실험 진행중에는 신중할 것 (업그레이드 되어 의도치 않은 현상이 발생할 수 있음)
- \$ pip3 install --upgrade otree-core
- \$ otree resetdb

#same as  
pip3 install --upgrade otree-core

# 앱 만들기

참고) Upper class  
CD.

- 우선, 자신이 oTree 프로젝트를 설치할 폴더로 이동할 것
- otree startproject [[프로젝트 이름]]
  - \$ otree startproject otree\_exercise
  - Include sample games? ⇒ no!
- 프로젝트 폴더로 이동 (requirements\_base.txt 가 있는 폴더임)
  - otree startapp [[앱이름]]
  - \$ otree startapp pgg

# models.py 편집

- 에디터에서 [[앱이름]]/models.py 를 연다
- 필수 class들은 이미 적혀 있음
  - pass: 정의만 하고 아무 일도 하지 않는다는 의미
- BaseConstants, BaseSubsession..
  - otree-core에 미리 정의되어 있는 기본 값들을 상속받음.

```
from otree.api import (
    models, widgets, BaseConstant
    Currency as c, currency_range
)

author = 'Your name here'

doc = """
Your app description
"""

class Constants(BaseConstants):
    name_in_url = 'pgg'
    players_per_group = None
    num_rounds = 1

class Subsession(BaseSubsession):
    pass
```

# class Constants

→ 상속 from Base Constants

```
class Constants(BaseConstants):
```

```
    name_in_url = 'pgg' # url에 보일 이름
```

```
    players_per_group = 3 # 그룹당 플레이어수
```

```
    num_rounds = 1 # 실행 총 라운드수
```

```
    endowment = c(100) # endowment, 동화 단위  
    efficiency_factor = 2 # MPCR
```

Tip: 주석은 꼼꼼히 달아주는 것이 좋음

# class Player

```
class Player(BasePlayer):
    contribution = models.CurrencyField(min=0,
                                          max=Constants.endowment)
```

- contribution: 기여금의 양
  - 자신이 보유할 금액은 endowment - contribution 으로 계산하면 되기 때문에 별도의 변수를 둘 필요는 없음
    - 하지만 명시적으로 보고자 하면 두어도 됨.
- CurrencyField: 정수로 두어도 되지만 보상 지급을 위해 otree에서 정의한 게임화폐단위를 쓸 것임을 의미
  - min= .. max= ... 게임내 설정값. (method 정의에 포함되어 있을 때에만 작동함. 해당 method 참고)
  - Constants.endowment :: Constants class 의 member인 endowment 값을 부른 것임.

*keep=Constants.endowment  
- contribution*

# class Group [[1]]

`class Group(BaseGroup):`

```
total_contribution = models.CurrencyField()  
individual_share = models.CurrencyField()
```

- 그룹 차원의 계산을 위해 필요한 변수들의 정의
  - `total_contribution` : 그룹내 기여금의 총합
  - `individual_share` : 그룹내 기여금 \* `efficiency_factor` / n
  - 이 역시 원한다면 별도의 변수 정의 없이 코딩 가능하지만 여기에서는 명시적으로 보기 위해 별도 변수로 지정한 것이라 볼 수 있음.

# class Group [[2]]

```
class Group(BaseGroup):
```

```
    total_contribution = models.CurrencyField()  
    individual_share = models.CurrencyField()
```

```
def set_payoffs(self):
```

```
    self.total_contribution =  
        sum([p.contribution for p in self.get_players()])  
    self.individual_share =  
        self.total_contribution *
```

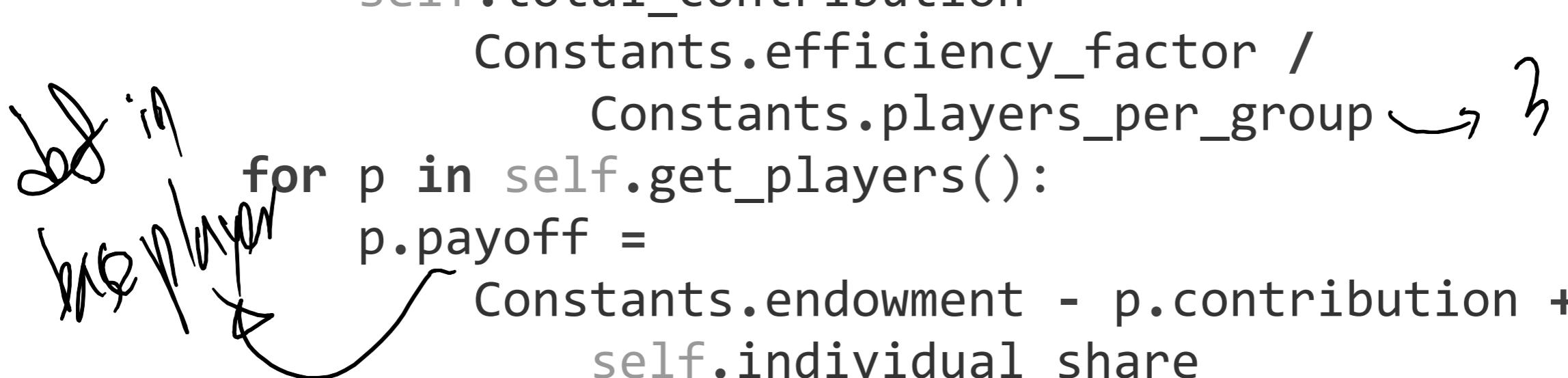
```
        Constants.efficiency_factor /
```

```
        Constants.players_per_group ↗
```

```
    for p in self.get_players():
```

```
        p.payoff =
```

```
        Constants.endowment - p.contribution +  
        self.individual_share
```





- <http://otree.readthedocs.io/en/latest/tutorial/part1.html>
- 위 링크를 따라서 step by step으로 따라해봅시다.

# 과제, 잊지 마세요!

- 과제: [codeacademy.com](https://www.codecademy.com)에서 다음 코스 마스터하기
  - <https://www.codecademy.com/learn/python> (필수)
  - <https://www.codecademy.com/learn/learn-html-css> (옵션)
  - <https://www.codecademy.com/learn/learn-javascript> (옵션)

# 수고하셨습니다!