

# oTree (3/3)

고려대 실험경제 세미나  
2017.8.24 @ 연세대학교

# 일러두기

- 본 문서는 아래 버전에 의거하여 작성되어 있음
  - oTree: 1.4.11 – Django: 1.8.8
- 2019년 1월 현재 oTree는 2.1.35로 상당히 많은 부분이 바뀌어 있음

# 주제

- Production
- 실제 실험을 진행하기 위해 필요한 것들
  - server setting 하기
  - production server 준비하기
- Rooms
- 그 외..
  - 실험 진행과 관련한 실질적인 준비들
    - IRB
    - 모집절차

# 준비

- 지난시간의 PGG 소스
- 실습: heroku.com 계정
- 본 슬라이드는 아래 링크에 공개되어 있음:
- <https://github.com/z0nam/oTreeBasic>

# Production

# Server Setup

- 다양한 옵션이 있음
  - Heroku ← 본 세미나는 이 방법을 사용
  - Docker
  - Win/MacOS/Linux Server
  - <http://otree.readthedocs.io/en/latest/server/intro.html>

# Heroku?

- <https://www.heroku.com>
- 상용 클라우드 호스팅 솔루션
- oTree 는 heroku 를 사용할 경우 매우 간단히 세팅을 할 수 있도록 되어 있음.
- 사용량에 따른 과금
  - 소용량의 트래픽/CPU/Data 는 무료

# heroku 세팅 절차

- heroku 가입
- heroku CLI 설치
- git setting
- Heroku app 준비
- oTree 업그레이드
- requirements\_base.txt 수정
- git push
- turn on timeout worker Dyno



# Heroku 가입

- <https://www.heroku.com>
- 자신의 아이디와 비밀번호를 기억해둘 것.



## ELEMENTS

### Powerful platform, unparalleled ecosystem

Don't reinvent the wheel. Heroku's 150+ third-party add-ons, 1000+ open source buildpacks, and 3000+ ready-to-deploy Heroku Buttons provides a rich ecosystem of pre-integrated extensions and services.

[SIGN UP FOR FREE](#)

[Explore Heroku Elements](#)



# Heroku CLI 설치

- Heroku 구동을 위한 CLI (Command Line Interface) 설치
  - Heroku 는 기본적으로 CLI 기반으로 동작함.
- <https://devcenter.heroku.com/articles/heroku-cli>
- \$ heroku login
  - 앞에서 가입했던 아이디와 비밀번호로 로그인.

# Login

```
$ heroku login
```

Enter your Heroku credentials:

Email: foo@bar.com

Password: \*\*\*\*

Logged in as foo@bar.com

# git 저장소 초기화

- git이란: 소스코드 유지 관리 및 배포를 위한 버전 관리 시스템
  - 초보자용 입문링크: <https://rogerdudler.github.io/git-guide/index.ko.html>
- Heroku 는 git 으로 유지 관리함
- 따라서 oTree의 소스를 git 저장소로 관리해야 함
- \*\* 직접 코드를 짜는 일이 많은 경우에는 git을 알아두면 좋음

# git 저장소 초기화

- 자신의 oTree 프로젝트 루트 폴더로 이동
  - settings.py 파일이 있는 폴더임
- \$ git init
- 주의: 다른 폴더에서 git init을 할 경우 그 폴더 및 하부내용 모두가 git으로 관리됨
  - 잘못 만든 경우: 해당 폴더의 “.git” 폴더를 삭제할 것
- 텅 빈 git 저장소가 준비됨. (아직 git 저장소에 내용이 들어 있지 않음)

# Heroku app 준비

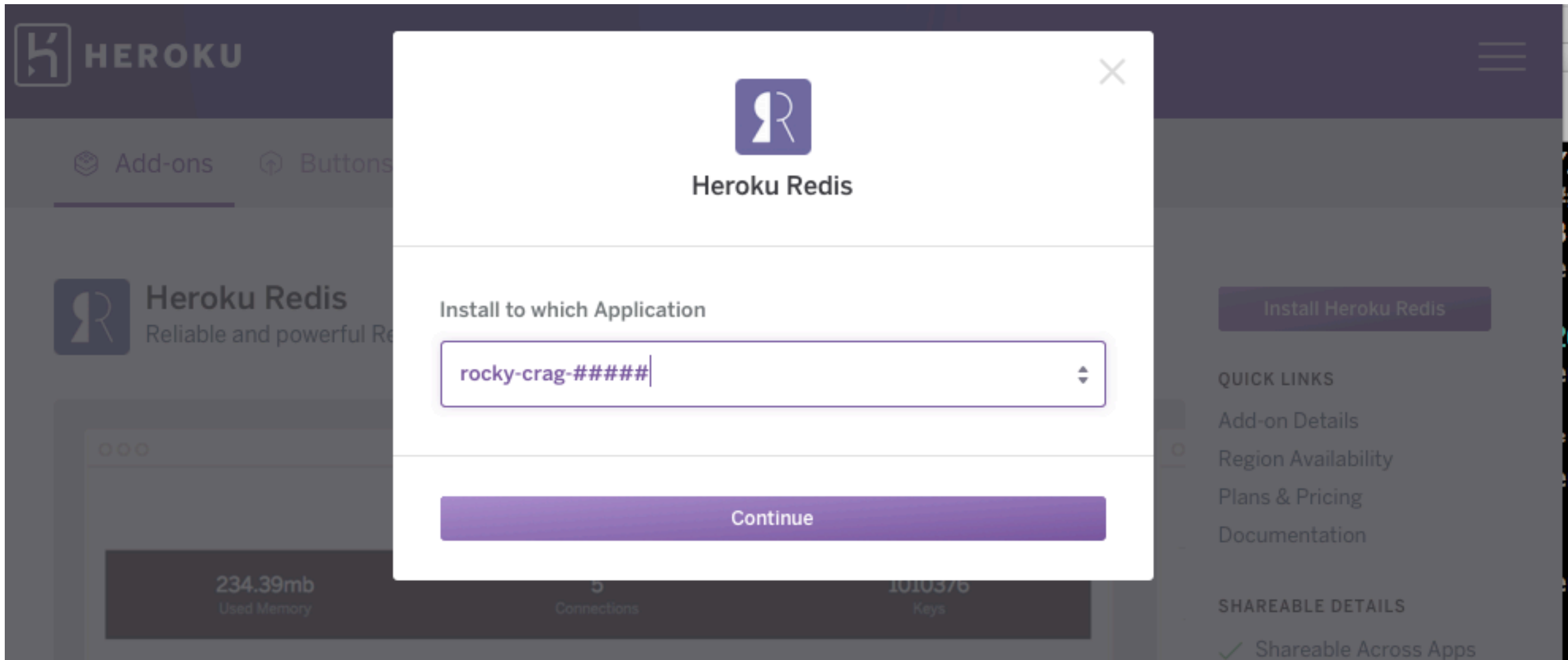
- 이전 Heroku 측에 oTree 프로젝트를 받아들일 준비를 해야 함.
- 동일 폴더에서 oTree 프로젝트를 동기화할 Heroku app 생성
- `$ heroku create`

Creating app... done,  rocky-crag-#####

<https://rocky-crag-#####.herokuapp.com/> | <https://git.heroku.com/rocky-crag-#####.git>

- 위 url 은 oTree 프로젝트의 이름임. (변경 가능)

# Redis add-on 설치



- <https://elements.heroku.com/addons/heroku-redis>
- Hobby Dev 버전 선택

# oTree upgrade

- 주의: 실제 운용시에는 신중해야 함
  - 업그레이드로 인해 기존 코드가 오작동할 가능성이 있으므로 준비가 완전히 되었을 때에는 이 절차를 건너뛸 것
- 그 외의 일반적 상황에서는 가능한한 최신 버전을 유지하는 것이 좋음.
- `$ pip3 install -U otree-core`



# requirements\_base.txt

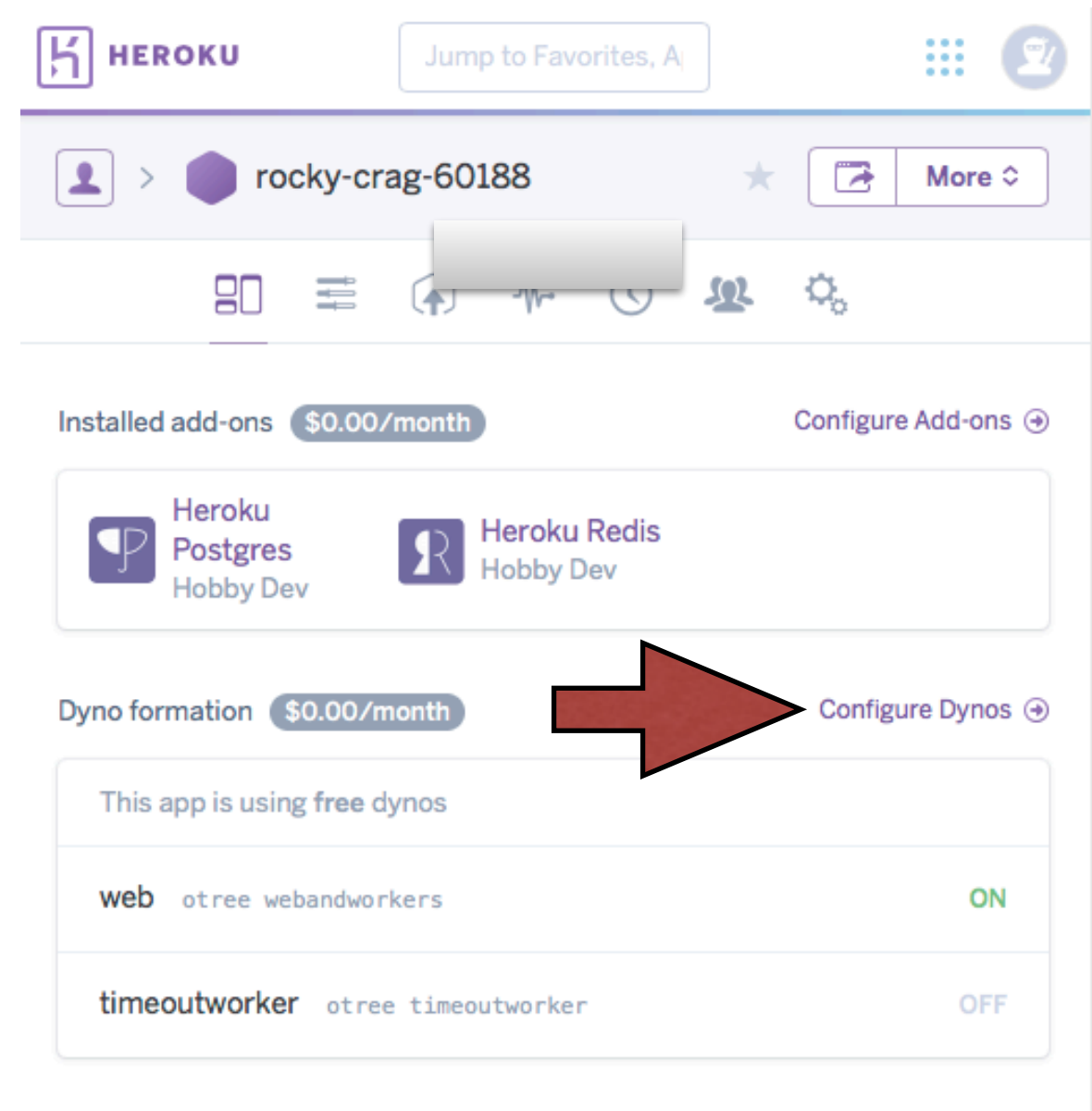
- 프로젝트 루트에 있음
- oTree와 django의 현재 버전을 확인할 것
  - `$ otree --version`
  - oTree: 1.4.11 – Django: 1.8.8
- requirements\_base.txt를 열어 위 정보에 따라 수정
  - `otree-core>=1.4.11`
  - `Django==1.8.8`
- Heroku 서버는 이 정보를 읽고 필요한 패키지를 직접 설치함 (사용자가 서버에 설치하지 않음)

# local → heroku

- git 연동 ⇒ heroku 에 push
  - \$ git add .
    - 현재 폴더와 그 하위폴더 전체를 git에 올릴 준비를 하는 것 (루트에서 해야 함)
  - \$ git commit -am "First commit"
    - 로컬 git 에 현재 상태를 세팅 (default branch 이름: master)
    - "First commit" 자리에는 소스 수정 내역 등을 간단히 기록
  - \$ git push heroku master
    - (remote) heroku branch에 (local) master brance를 푸시 (업로드) 함
- 차후 소스에 수정이 있을 때 위 과정을 통해 업데이트할 수 있음.

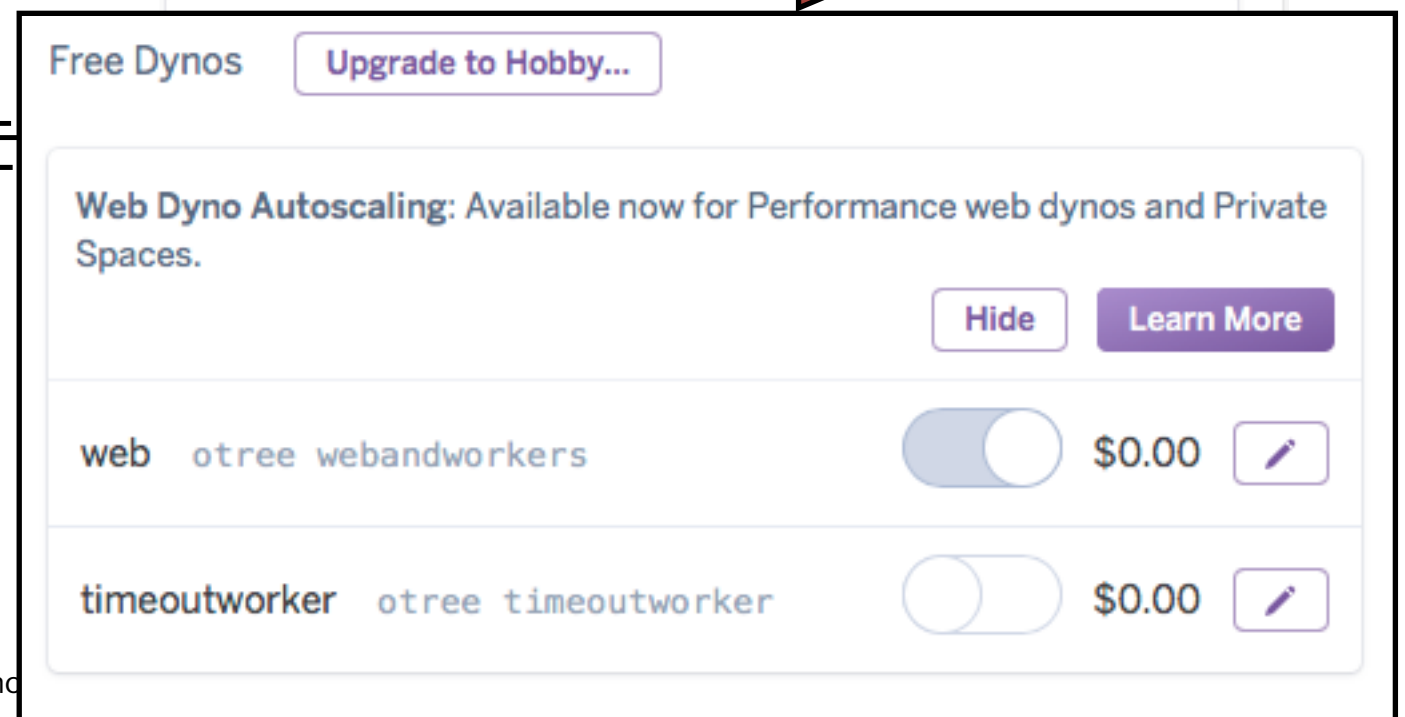
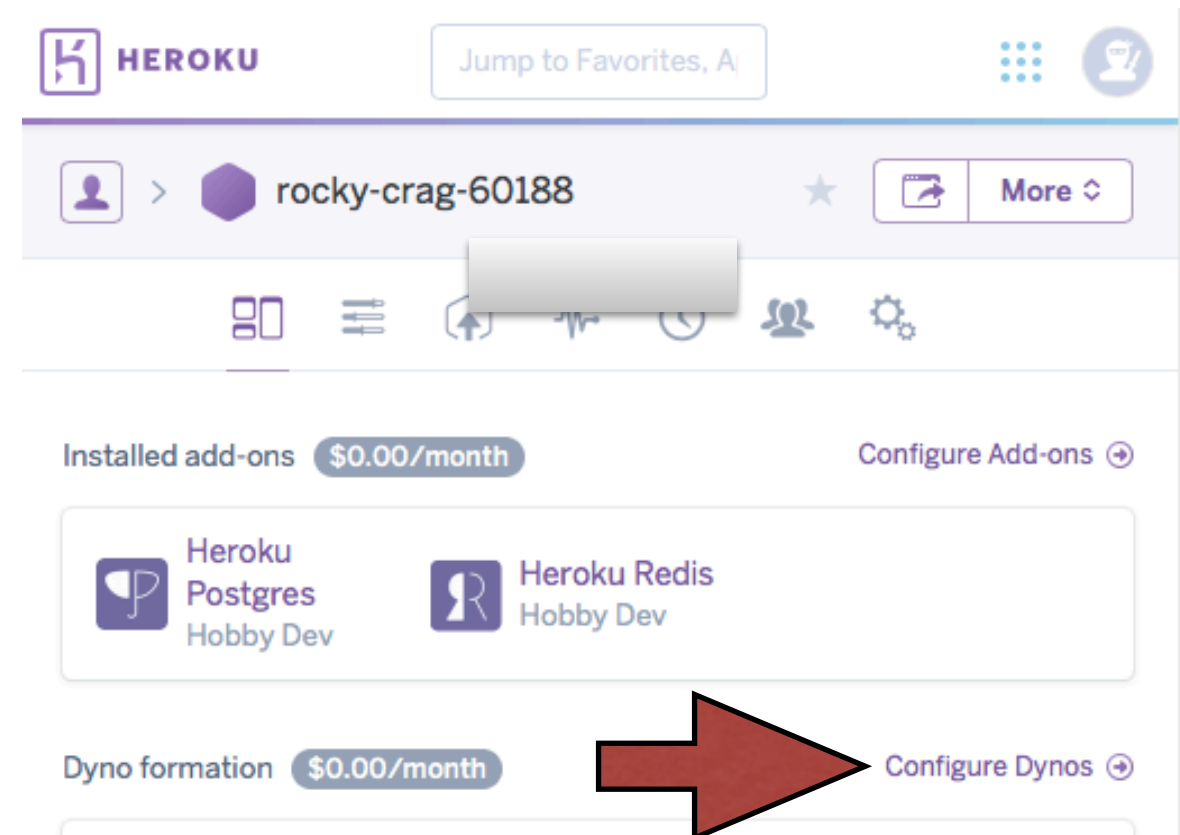
# timeout dyno 켜기

- <https://dashboard.heroku.com/apps/rocky-crag-60188>
- 실험 라운드에 타임아웃이 있을 경우 필요함
- 최초 설정시 크레딧 카드 등록이 필요
- 타임아웃 기능을 쓰지 않는다면 켜지 않아도 됨.



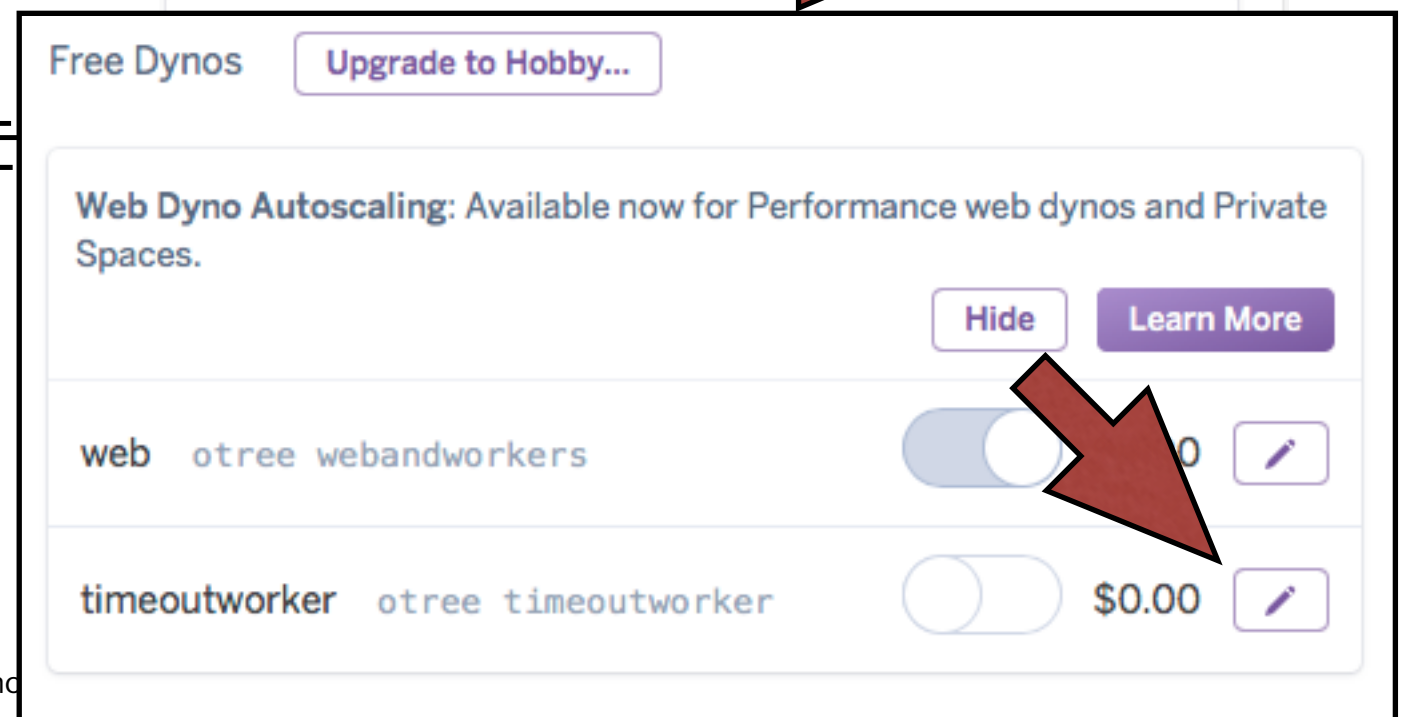
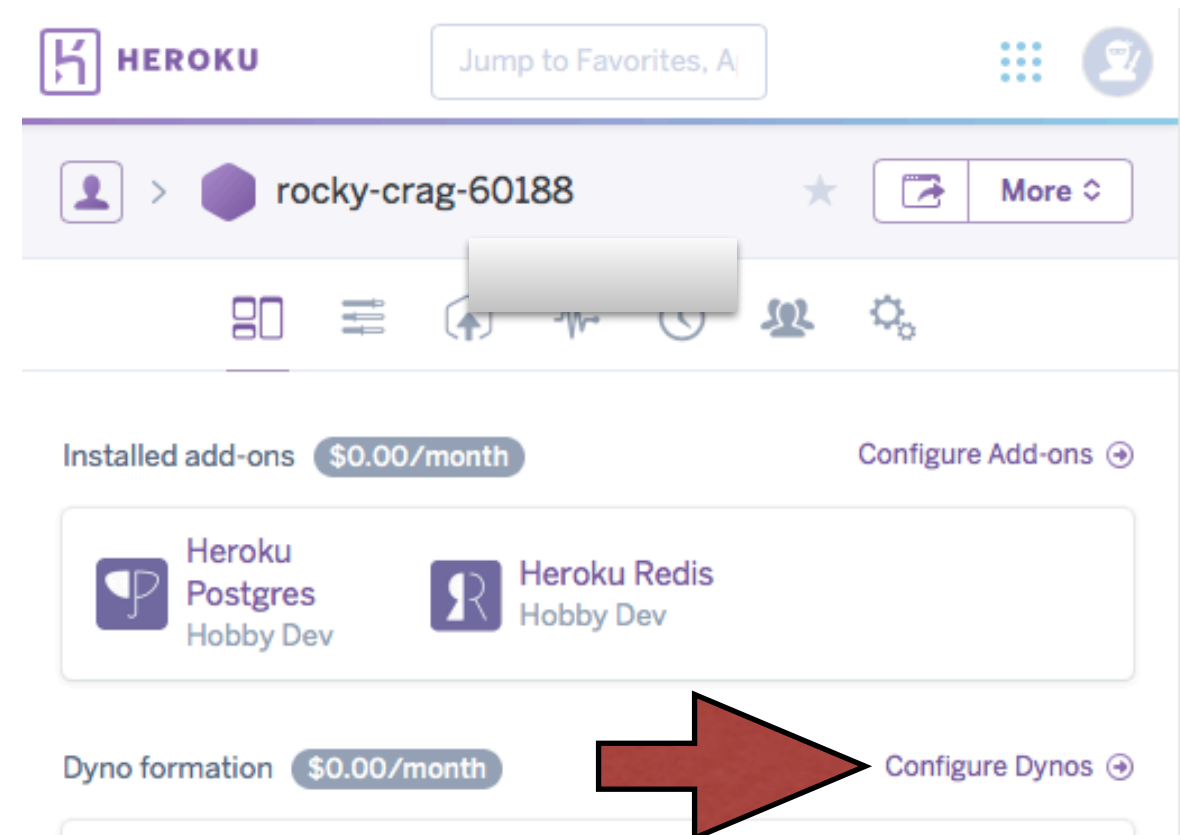
# timeout dyno 켜기

- <https://dashboard.heroku.com/apps/rocky-crag-60188>
- 실험 라운드에 타임아웃이 있을 경우 필요함
- 최초 설정시 크레딧 카드 등록이 필요
- 타임아웃 기능을 쓰지 않는다면 켜지 않아도 됨.



# timeout dyno 켜기

- <https://dashboard.heroku.com/apps/rocky-crag-60188>
- 실험 라운드에 타임아웃이 있을 경우 필요함
- 최초 설정시 크레딧 카드 등록이 필요
- 타임아웃 기능을 쓰지 않는다면 켜지 않아도 됨.



# Server Control

# otree resetdb (on heroku)

- heroku [명령어..]
  - 다양한 명령어들이 있음
  - 자세한 것은 \$ heroku help
- \$ heroku run otree resetdb
  - timeout 에러가 뜨는 경우 (서울대의 경우 heroku 포트가 막혀 있음) 테더링 등의 방법으로 우회할 것
- \$ heroku open
  - 혹은 브라우저에서 앞에서 heroku create로 생성된 `https://rocky-crag-#####.herokuapp.com` 주소를 입력하여 들어갈 수도 있음

# Production Server 준비



# Server Check

- `https://your_url.herokuapp.com/server_check/`
- 상단 메뉴에서 “Server Check” 클릭
- Production을 위한 최소 필요 조건이 간단히 나열되어 있음
  - Green == Good
  - Red == Bad

oTree

## Server Readiness Checks

For details on how to fix any issues highlighted below, see [here](#).

- You have a recent version of oTree (1.4.11).
- You are using a proper database (Postgres, MySQL, etc).
- DEBUG mode is on** You should only use DEBUG mode during development and testing of your app. Before launching a study, you should switch DEBUG mode off. To turn off DEBUG mode, run:  
`heroku config:set OTREE_PRODUCTION=1`
- You are using a server other than `runserver`.
- Sentry not configured** Sentry can send you the details of each server error by email. This is necessary because once you have turned off `DEBUG` mode, you will no longer see Django's yellow error pages; you or your users will just see generic "500 server error" pages. oTree offers a free Sentry service; you can find the sign-up link in the oTree documentation.
- No password protection** To prevent unauthorized server access, you should set the environment variable `OTREE_AUTH_LEVEL`.
- Your database appears to be synced.
- The timeoutworker is running

# 환경변수 세팅

- `$ heroku config:set OTREE_PRODUCTION=1`
  - 디버깅 메시지들을 숨기는 등의 작업을 함
  - 이후 에러 트래킹을 위해서는 Sentry를 신청하던지 ([http://otree.readthedocs.io/en/latest/server/next\\_steps.html#sentry](http://otree.readthedocs.io/en/latest/server/next_steps.html#sentry)) heroku logs 등을 사용
- `$ heroku config:set OTREE_AUTH_LEVEL=DEMO`
  - 데모 버전 (아무나 들어와서 데모를 시행할 수 있음)

# Password Protection

- 실제 실험에서는 보안을 위해 oTree Admin 페이지를 암호로 보호해야 함.
- `$ otree config:set OTREE_AUTH_LEVEL=STUDY`
- Admin username: settings.py 에서
  - `ADMIN_USERNAME = 'your_admin_username'`
  - default username: 'admin'
  - settings.py 수정했을 경우 git commit, push 잊지 말 것.
- Admin PW: `heroku config:set OTREE_ADMIN_PASSWORD='blahblah'`

# 성공적으로 세팅한 경우

## Admin Login

The password is defined in your `settings.py` file.

Log in

Sentry는 무료이므로  
실제 실험시에는 신청해 둘 것

oTree Demo Sessions Rooms Data **Server Check** Logout

### Server Readiness Checks

For details on how to fix any issues highlighted below, see [here](#).

- You have a recent version of oTree (1.4.11).
- You are using a proper database (Postgres, MySQL, etc).
- DEBUG mode is off
- You are using a server other than `runserver`.
- Sentry not configured** Sentry can send you the details of each server error by email. This is necessary because once you have turned off `DEBUG` mode, you will no longer see Django's yellow error pages; you or your users will just see generic "500 server error" pages. oTree offers a free Sentry service; you can find the sign-up link in the oTree documentation.
- Password protection is on. Your app's `AUTH_LEVEL` is `STUDY`.
- Your database appears to be synced.
- The timeoutworker is running

# 기타 사항들

- 기타 사항들은 매뉴얼 참고
  - Papertrail, DB backup 등등.
  - <http://otree.readthedocs.io/en/latest/server/heroku.html>

# 부하 테스트

- 최소 작동은 full screen mode로도 가능하지만 실제 참가자수만큼의 테스트는 힘들
- oTree의 Bot 기능을 사용
  - <http://otree.readthedocs.io/en/latest/bots.html#bots>
  - 이 부분은 자세한 설명을 생략

# 실제 실험 단계

# 참가자 접속 방법

- Room 기능을 사용
  - 가장 보편적 (추천)
- Single-use link
  - 개인별 1회용 링크 (세션마다 새로 생성됨)
- Session-wide link
  - 세션 자체 단일 링크로 모든 참가자가 접속
    - 중복접속을 구분할 수 없으므로 비추천



# Room

- <http://otree.readthedocs.io/en/latest/rooms.html#rooms>
- 사전에 정의된 참가자 그룹 리스트라고 생각하면 됨
- 참가자가 사전에 결정되어 있거나 여러 번에 걸쳐 참가하는 경우 (가령 수업용 실험) 등에 유용
- 표준적 랩실험의 경우 이 방법을 사용할 것을 강력히 권장함.

# Rooms의 생성

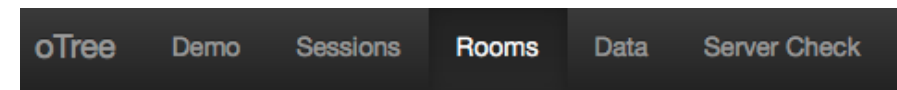
- settings.py 를 수정
  - 오른쪽과 같은 코드를 쓰면 됨
  - ROOM\_DEFAULTS: 이 프로젝트 전체의 공통사항 (지금은 없으므로 {} )
  - name: 그룹(room)내부 이름 (영문자,숫자,"\_"만 가능)
  - display\_name: 출력될 room 이름
  - participant\_label\_file: 참가자 레이블 (id) 리스트
- git commit, push 잊지말것!

```
ROOM_DEFAULTS = {}

ROOMS = [
    {
        'name': 'econ101',
        'display_name': 'Econ 101 class',
        'participant_label_file': 'econ101.txt',
    },
    {
        'name': 'econ_lab',
        'display_name': 'Experimental Economics Lab',
    },
]
```

# participant\_label\_file

- 프로젝트 루트에 생성
  - 파일명 = ROOM 디렉터리  
리의  
participant\_label\_file 과  
동일해야함.
- username list 라고 생각하  
면 됨
  - 숫자, 문자, 밑줄문자  
(underscore: “\_”) 만 가  
능함
- git commit, push 잊지 말  
것.



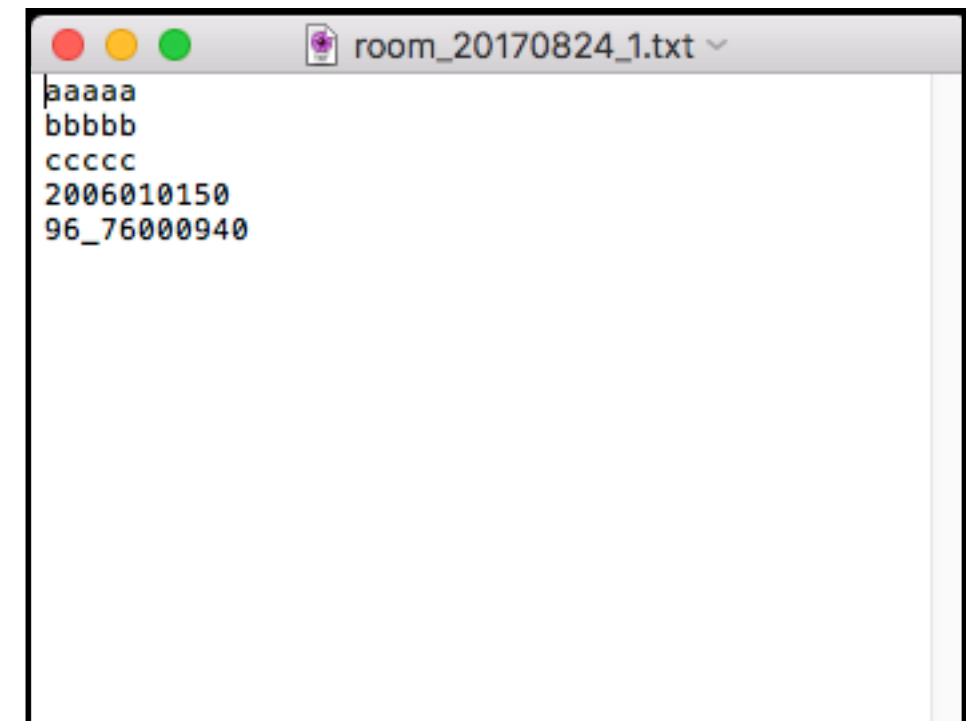
## Rooms

Current rooms:

공공재게임 2017.8.24 1차

공공재게임 레이블 파일 없는 버전

See docs about rooms [here](#).



# 참가자 제공 URL

- 참가자 개별 URL
  - 참가자마다 다른 URL
- Room-wide URL
  - 참가자 공통 URL
  - 진입시 각자의 participant\_label을 입력 받음.

**Participant-specific URLs**

These URLs contain the labels, so participants don't have to enter their label manually.

Show/Hide

- [http://localhost:8000/room/pgg\\_20170824\\_1/?participant\\_label=aaaaa](http://localhost:8000/room/pgg_20170824_1/?participant_label=aaaaa)
- [http://localhost:8000/room/pgg\\_20170824\\_1/?participant\\_label=bbbbb](http://localhost:8000/room/pgg_20170824_1/?participant_label=bbbbb)
- [http://localhost:8000/room/pgg\\_20170824\\_1/?participant\\_label=ccccc](http://localhost:8000/room/pgg_20170824_1/?participant_label=ccccc)
- [http://localhost:8000/room/pgg\\_20170824\\_1/?participant\\_label=2006010150](http://localhost:8000/room/pgg_20170824_1/?participant_label=2006010150)
- [http://localhost:8000/room/pgg\\_20170824\\_1/?participant\\_label=96\\_76000940](http://localhost:8000/room/pgg_20170824_1/?participant_label=96_76000940)

**Welcome**

Please enter your participant label.

Next

# Admin 화면 (예시)

Create a new session

Session config:

----- ▾

Number of participants:

Create

---

7 participants present

Show/Hide

Alice Bob Evan Gary Helen Jim  
Louis

5 participants not present

Show/Hide

Charlie Danielle Fiona Ian Katie

Welcome


Please enter your participant label:

Alice

Next

Please wait

Waiting for your session to begin



# 실험 진행

- 리허설을 통해 다양한 상황을 살펴볼 것.

# 실험 종료후

- 최종 실험이 끝나면 payment info 등의 정보가 남음
  - 참가자들에게 해당 금액 지급
- 일단 현장에서 csv, xls 로 다운받아둘 것.
- DB backup은 heroku site에서 가능
  - 자세한 설명은 매뉴얼 참고

# 현실에서의 실험 진행



# 랩실험의 전체 과정

- IRB 승인
- 참가자 모집
- 리허설
- 랩에 모이기
- 실험 진행
- 실험 종료

# IRB (Institutional Review Board) 승인

- 연구윤리위원회
  - 각 대학마다 있음
  - 설문, 실험 등의 사회과학 연구도 IRB를 요구하는 추세임
  - 실험을 주관하는 (실험 책임 연구자 소속) 대학의 IRB 승인을 받으면 됨.
  - 상당히 까다로우므로 최소 반년 전부터 준비하길 권장
- 서울대: <http://irb.snu.ac.kr>
- 고려대: [http://rms.korea.ac.kr/nsys/nrpt/content.do?menu\\_pgm\\_id=ORSC60.NRPT1029Q.00](http://rms.korea.ac.kr/nsys/nrpt/content.do?menu_pgm_id=ORSC60.NRPT1029Q.00) (문의 필요)

# 참가자 모집

- 학내 모집: 학과 게시판 협조, 대학 커뮤니티 (고파스, snulife 등등 ) 등의 구인구직 게시판
  - 심리학과 등에서 올린 글들을 검색하여 참고할 것
  - 구글 forms 등을 통한 링크 사용하면 편리함.
- 설문업체를 통한 방법
  - 온라인 설문업체의 온라인 패널을 사용할 수도 있음 (유료)

# 주의사항

- 20명 실험을 생각한다면, 당일 결원을 생각하여 25% 정도 (5명 정도) 더 모집할 것
  - 20명 이상 오면 참가사례비 주고 돌려보내면 됨
  - 상황에 따라 판단해야 함
- 사전 확인 문자, 메일 등을 적극적으로 활용할 것
  - 당일 오전에는 참가 가능 여부를 회신받아서 최소 필요 인원이 참가 가능한지 확인해야 함.

# 리허설

- 실험 전에 production server가 제대로 기능하는지 가능한한 실제 실험과 유사한 환경에서 전체 진행해볼 것
- 예상 시간을 넘길 경우 중간 이탈이 발생할 수 있으므로 특히 예상 시간을 최대한 보수적으로 파악해야 함.

# 랩에 집결

- 모바일이던 PC실이던 사전에 정한 장소에 모일 경우 (랩실험), 혼자서는 진행하기 어려울 때가 많음
- 실험 진행을 보조할 인원을 확보하길 권장
- 자리 안내, url 안내 등.
- IRB의 지도를 받았을 경우 실험 설명은 IRB에 제출한 버전과 동일해야 함
  - dry reading
  - 객관성 확보를 위해 동영상을 사용하는 경우도 있음

# 실험 진행, 종료

- 종료시간을 철저히 지킬 수 있도록 주의할 것
- 종료후 설문 작성시간, 보상 지급 시간이 걸리므로 그것까지 감안해야 함
  - 현장에서는 참가사례비만 주고 차등 보상은 계좌이체로 할 수도 있음.
- 영수증 받아야 함 (증빙)

수고하셨습니다!