Introduction & Architecture

Remixed from material by Ali Seyhun Saral & Philipp Chapkovski

Programming experiment in oTree

Organization:

- 24.02.2021 – 26.02.2021, 9 a.m. – 1 p.m. lecture
- Two problem sets for the afternoon

If you want to be graded:

- Contact me & Sign-up in LSF
- You need to take two seminars to fulfil the MW86 module requirements
- Grading: 10% participation, 90% final project

Programming experiment in oTree

Final project

- Replicate experiment in oTree from a published paper & provide small documentation (2-3 pages)
- **Make an appointment with me to discuss the experiment in advance**
- Deadline: 31.03.2021, 11:59 p.m.

Resources:

- **Course page:** https://github.com/ToFeWe/DICE-otree-class
- Webex-link via mail
- Documentation: https://otree.readthedocs.io/en/self/
- Alis community book: https://otreecb.netlify.app/intro.html
- Live experiments: https://dice-otree-course.herokuapp.com/room/live_demo/

What is oTree?

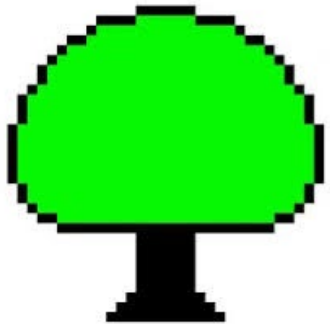
- oTree is a platform/software package to run online/lab experiments.
- Participants interact by using their browsers
- oTree runs on a web server
  - It runs on your computer for development (local installation)
  - It runs on a physical server (Server setup)
  - It runs on cloud services (Heroku, oTree hub)
- A friendly visual version is available (needs paid subscription)

Lets start with an example

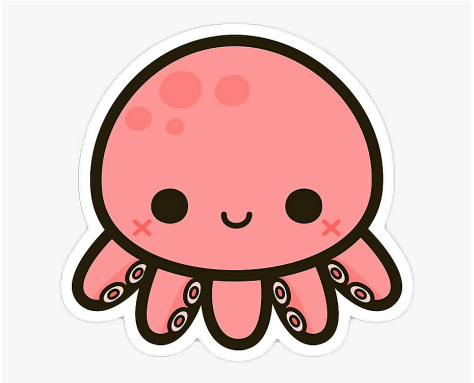
https://dice-otree-course.herokuapp.com/room/live_demo/

One framework to rule them all

How it is different from z-Tree?

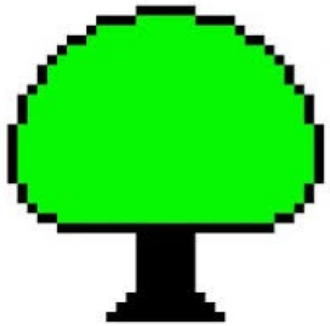


z-Tree

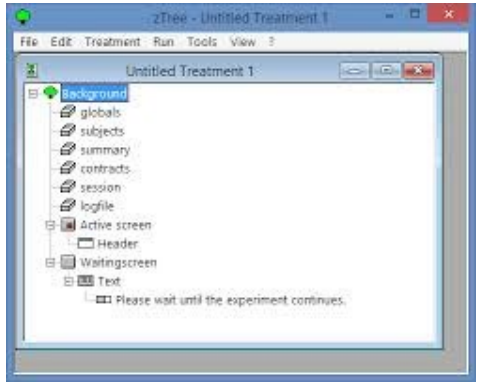


oTree

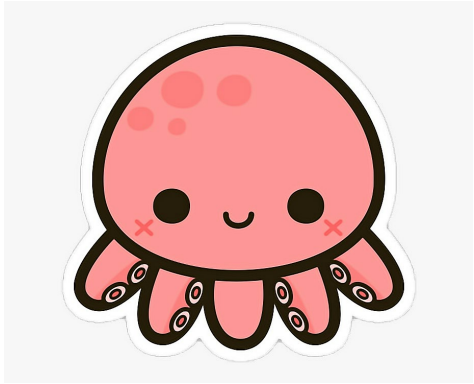
How it is different from z-Tree?



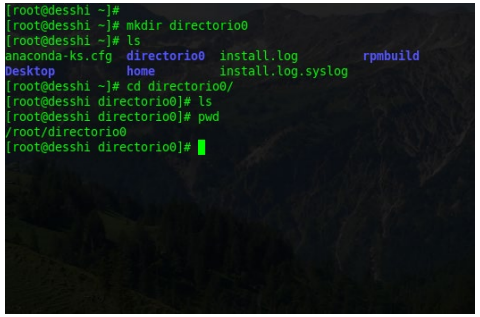
z-Tree

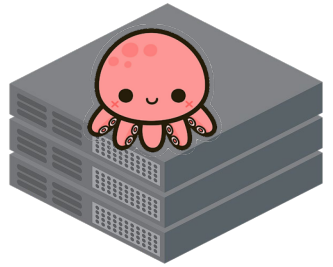


Click & Run



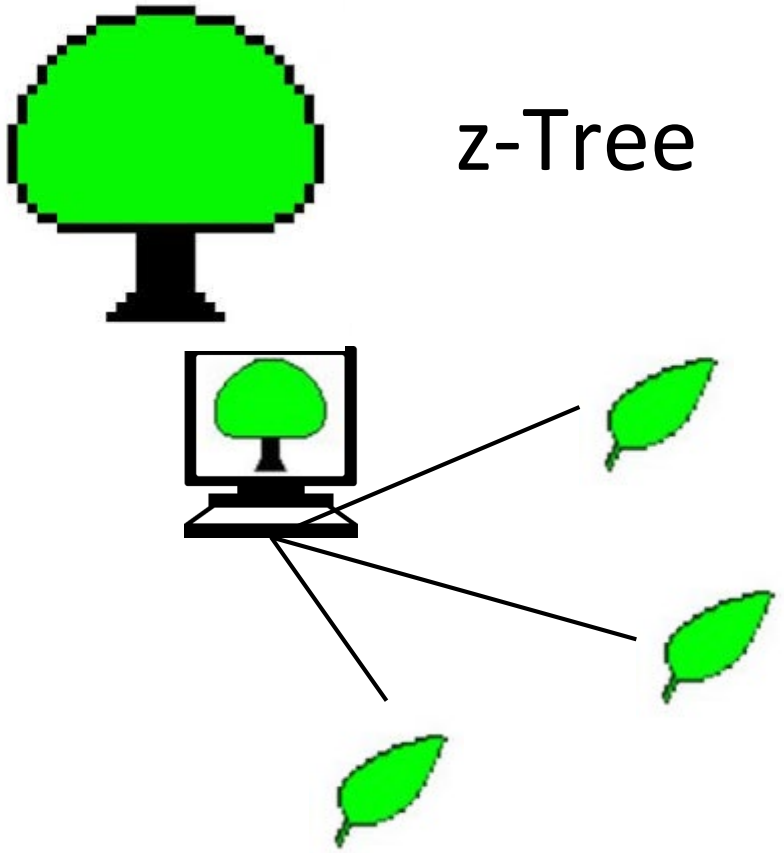
oTree





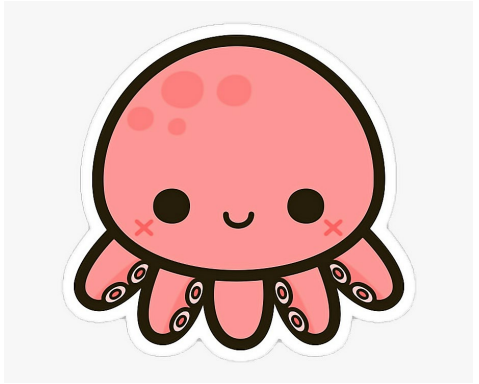
Server Setup

How it is different from z-Tree?

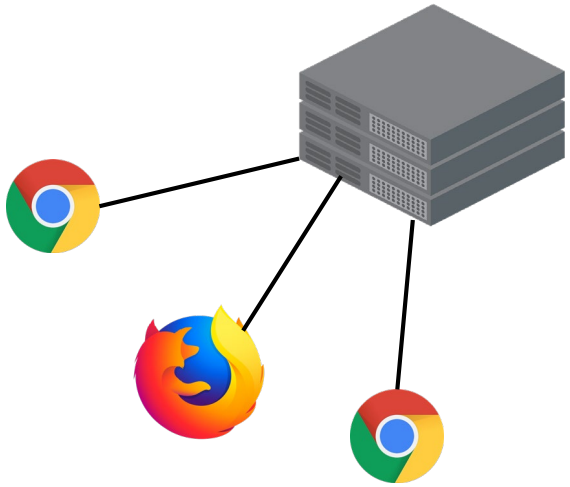


z-Tree

Clients require software

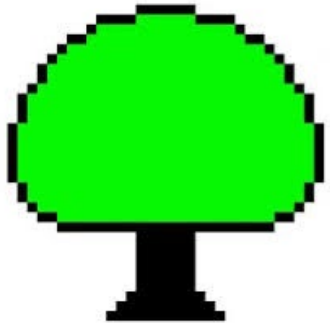


oTree

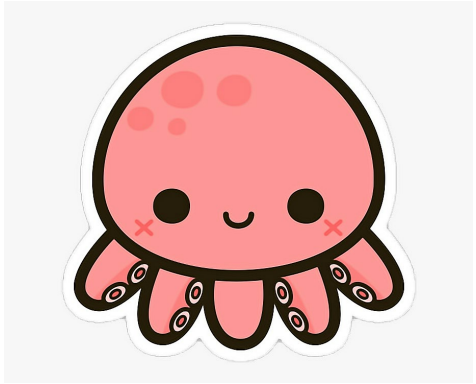


Clients connect with browser

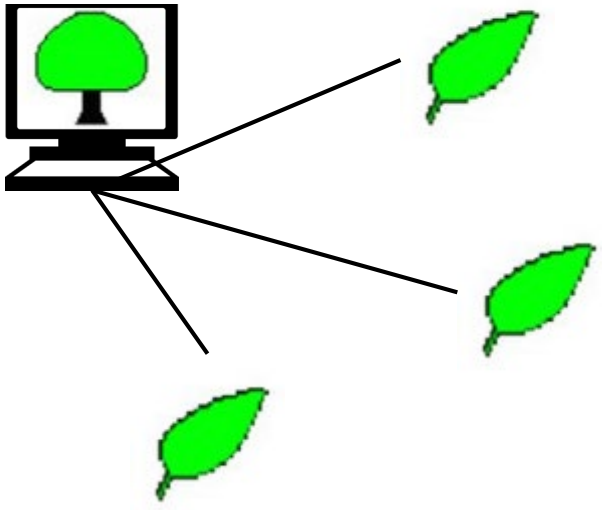
How it is different from z-Tree?



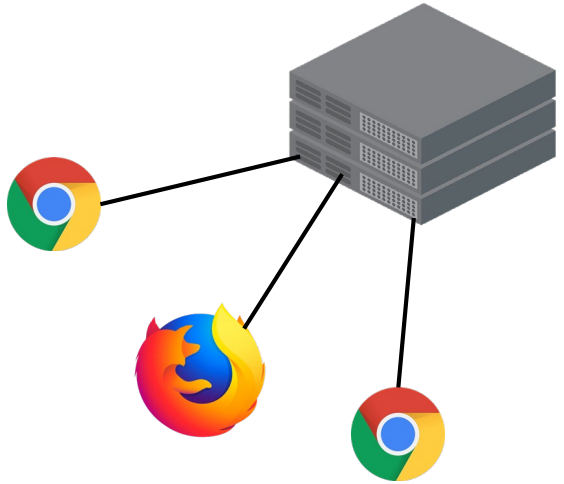
z-Tree



oTree

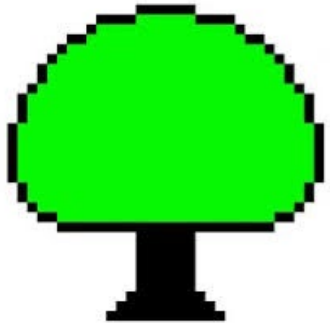


Local



z-Tree unleashed
by CLER

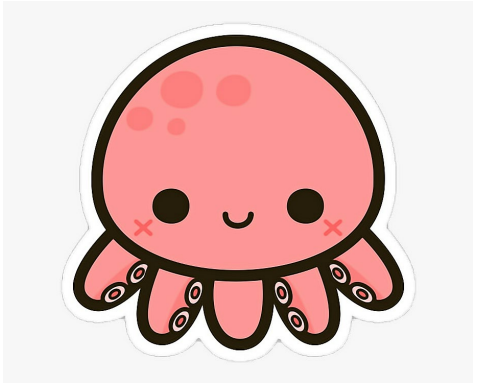
How it is different from z-Tree?



z-Tree

+

```
a = 10 + b;
c = abs(b);
d = subjects.find();
```

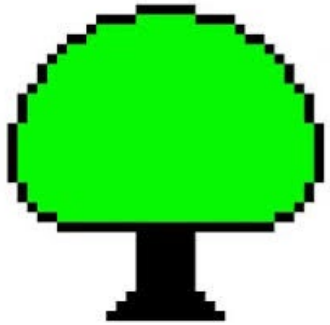


oTree

+

+

How it is different from z-Tree?

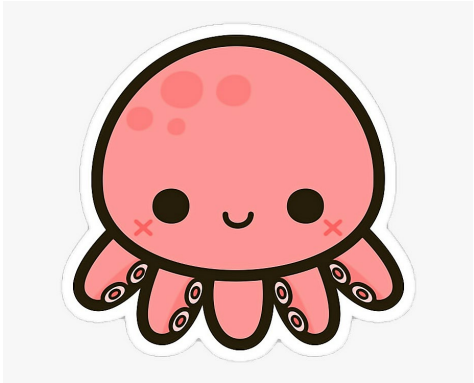


z-Tree

+

```
a = 10 + b;
c = abs(b);
d = subjects.find();
```

Single file per experiment



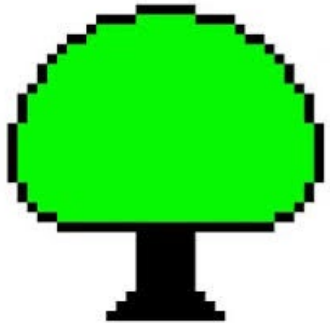
oTree

+

+

Set of files per experiment (app)

How it is different from z-Tree?

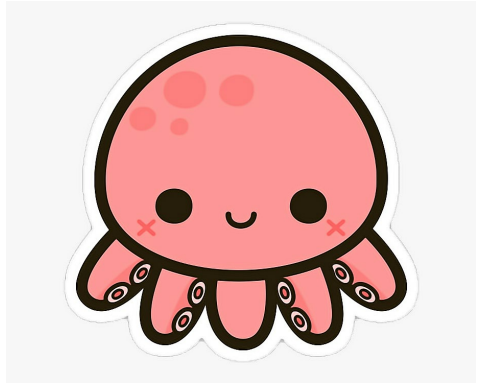


z-Tree

+

```
a = 10 + b;
c = abs(b);
d = subjects.find();
```

Simple to start, hard to scale



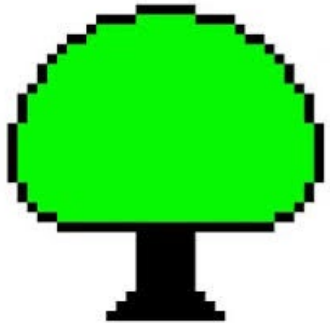
oTree

+

+

Hard to start, simple to scale

How it is different from z-Tree?



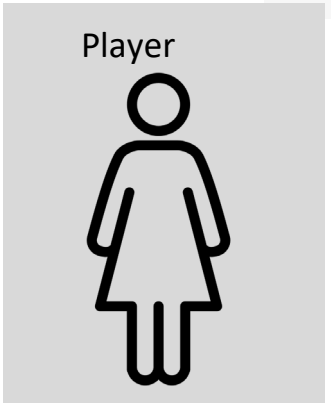
z-Tree

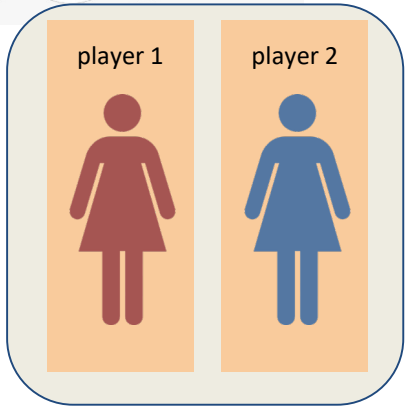
subjects table

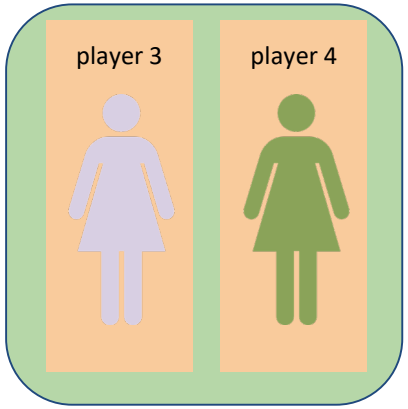
Period	Subject	Group	Profit	TotalProfit	Participate
1	1	1	0	0	1
1	2	1	0	0	1
1	3	1	0	0	1

Data in table format

oTree





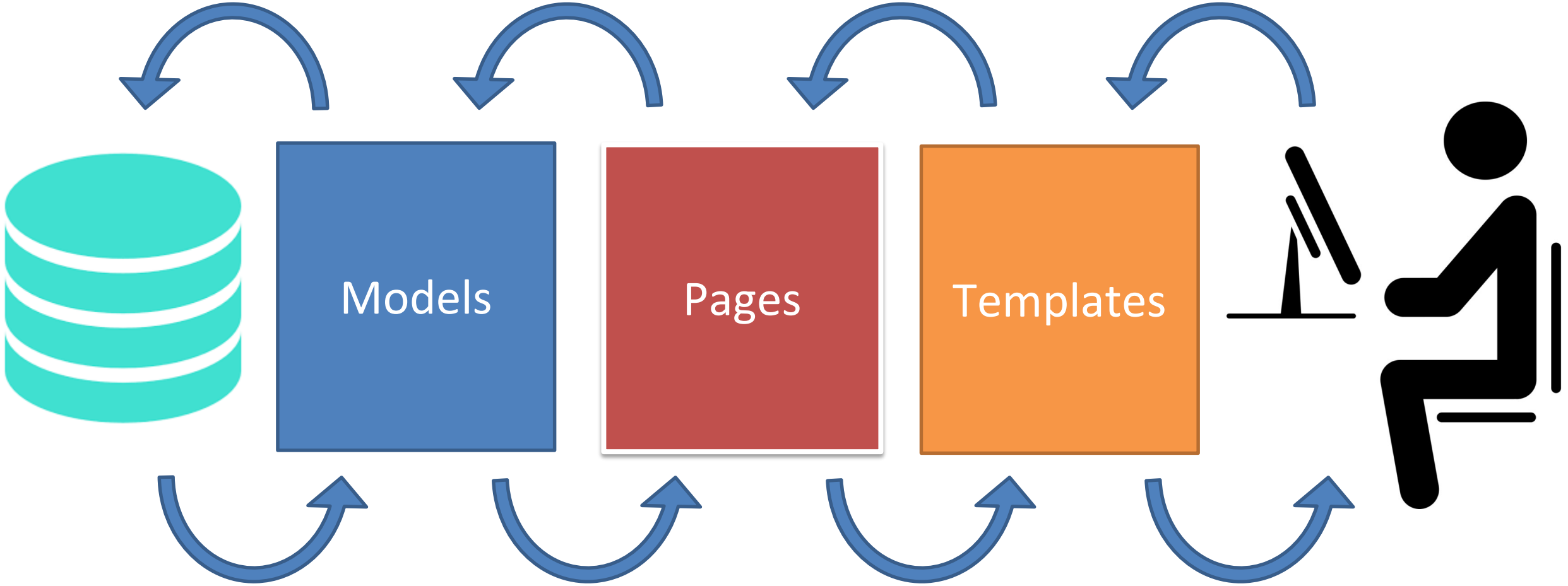


Data represented by objects, boils down to table format

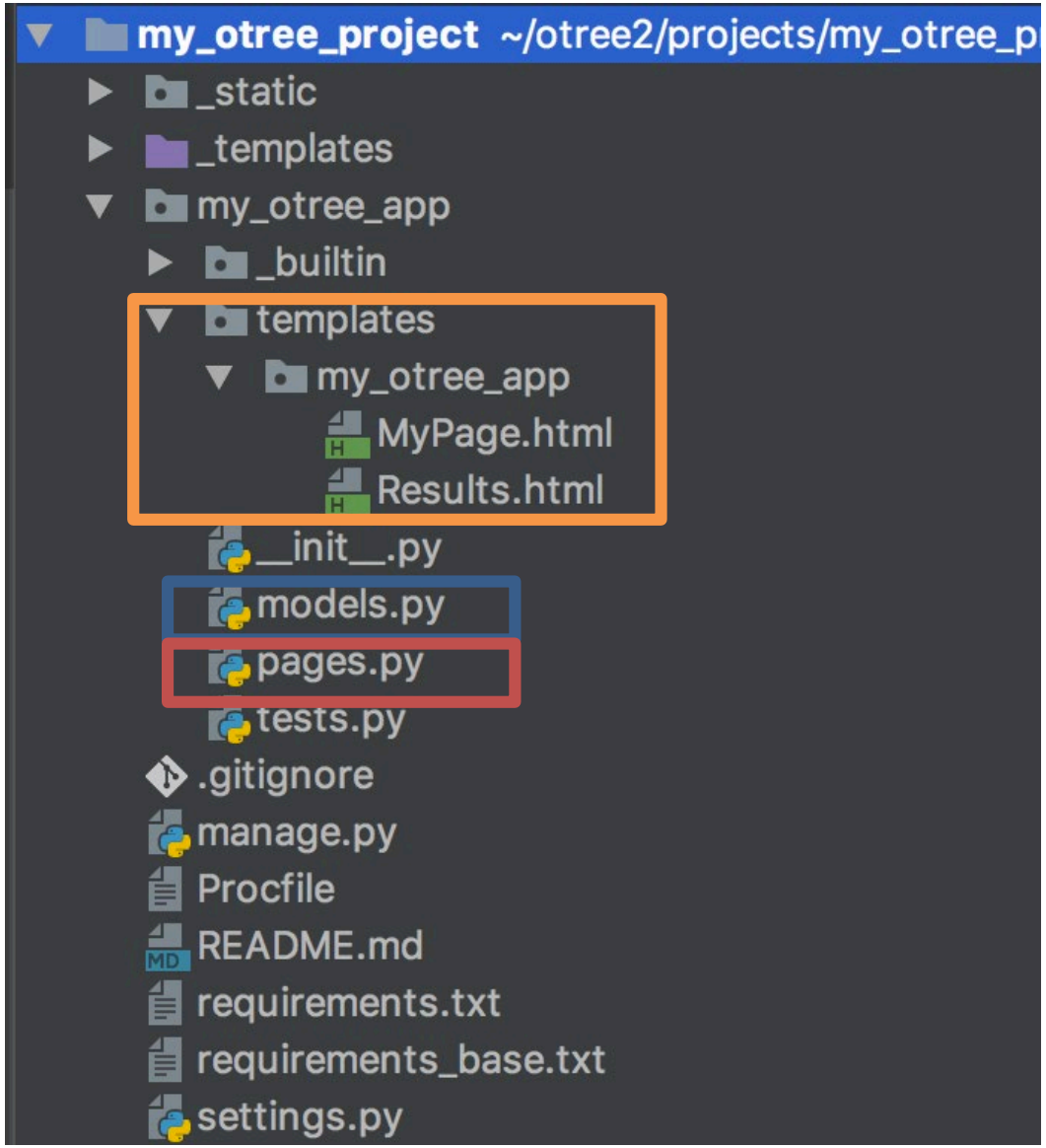
Programming any experiment

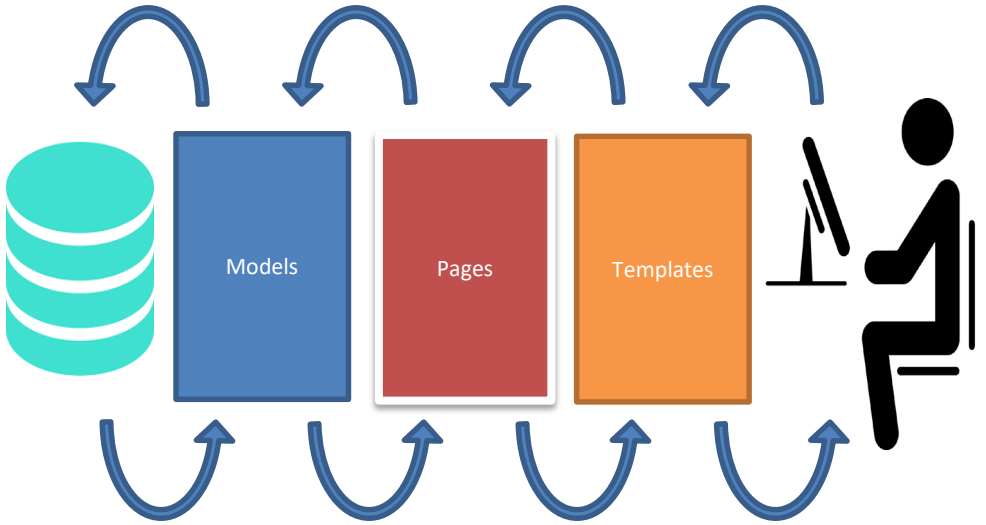
1. Show something to the participant.
2. Get his/her reaction on it

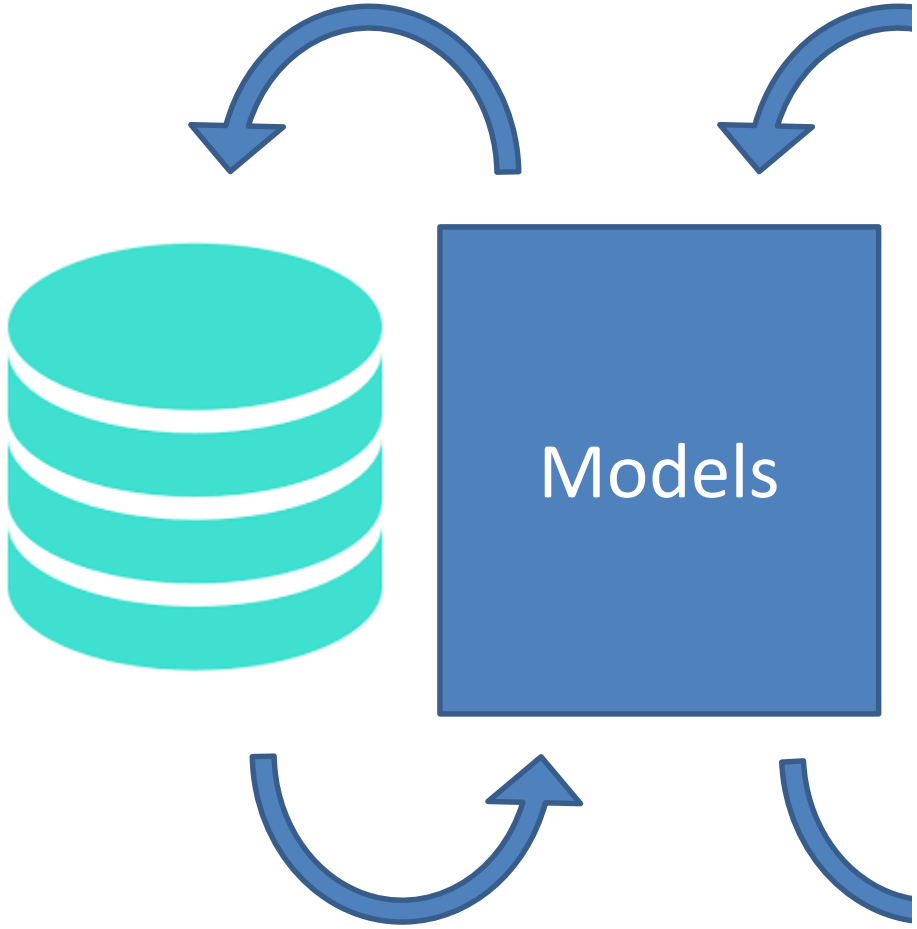
How to do this in oTree



Structure within the app folder







Models are responsible for storing and processing data in the database

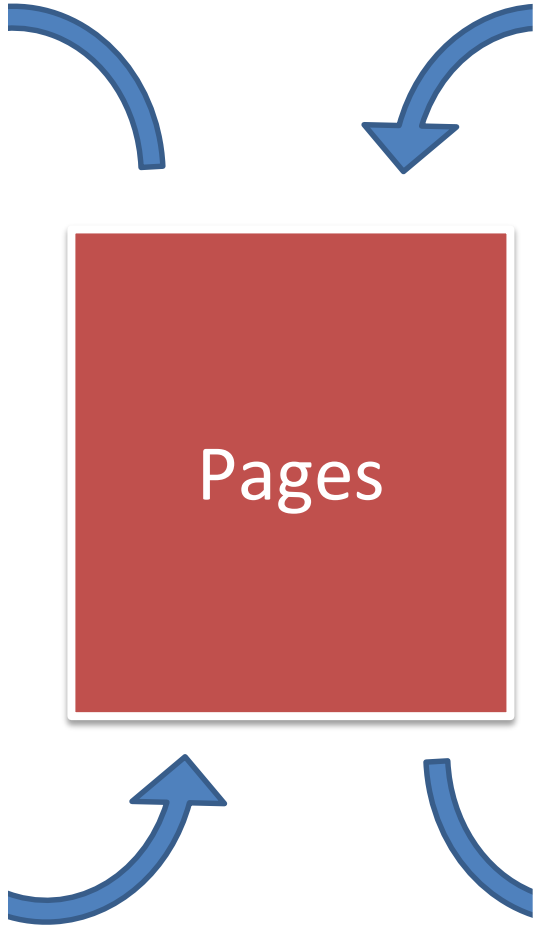
Anything that you need to be stored should be defined as a field in models.py file

Models.py example

```
from otree.api import (…)

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

class Subsession(BaseSubsession):
    pass

class Group(BaseGroup):
    pass

class Player(BasePlayer):
     age = models.IntegerField(label='What is your age?', min=13, max=125)
     gender = models.StringField(
        choices=['Male', 'Male', 'Other'],
        label='What is your gender?',
    )

    …
```

Pages are responsible for retrieving and passing back data from models to templates and vice versa.

If you need to show something to a participant or to get his/her input, you need to state this in pages.py

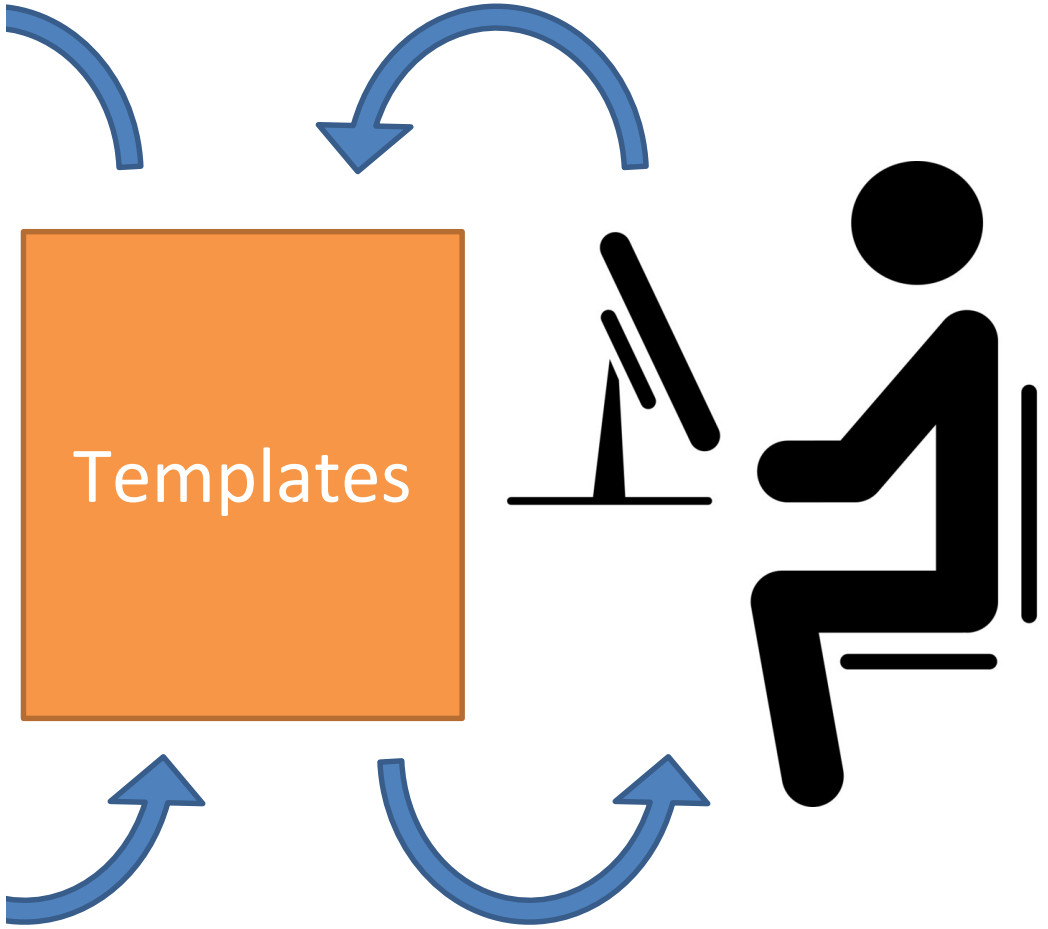


Pages.py example

```
from otree.api import Currency as c, currency_range
from ._builtin import Page, WaitPage
from .models import Constants

class Demographics(Page):
    form_model = 'player'
    form_fields = ['age', 'gender']

class CognitiveReflectionTest(Page):
    form_model = 'player'
    form_fields = ['crt_bat', 'crt_widget', 'crt_lake']

page_sequence = [Demographics, CognitiveReflectionTest]
```

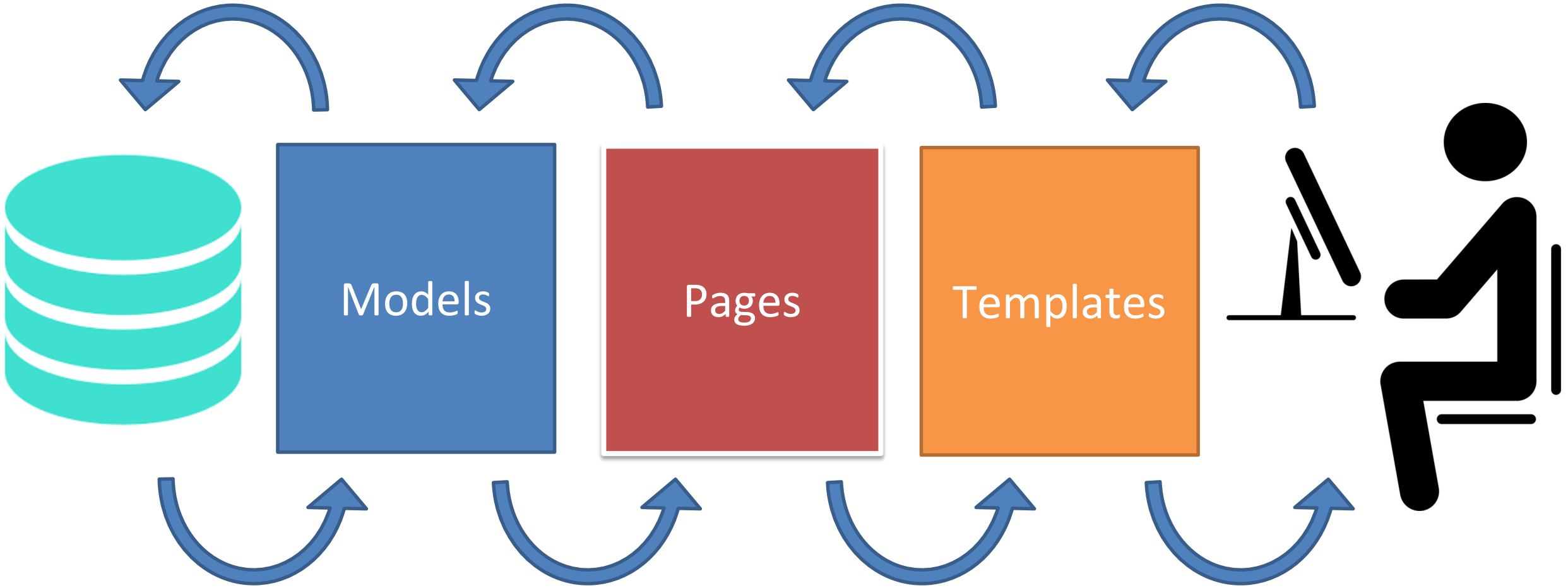
Templates are html files which get the info from pages and show it to a participant.

As soon as a participant clicks ‘Next’ the data he enters is passed back to pages



Template example (Demographics.html)

```
{% extends "global/Page.html" %}
{% load otree %}

{% block title %}
    Survey
{% endblock %}

{% block content %}

        <p>
        Please answer the following questions.
       </p>

        {% formfields %}

        {% next_button %}

    {% endblock %}
```


Models
Pages
Templates

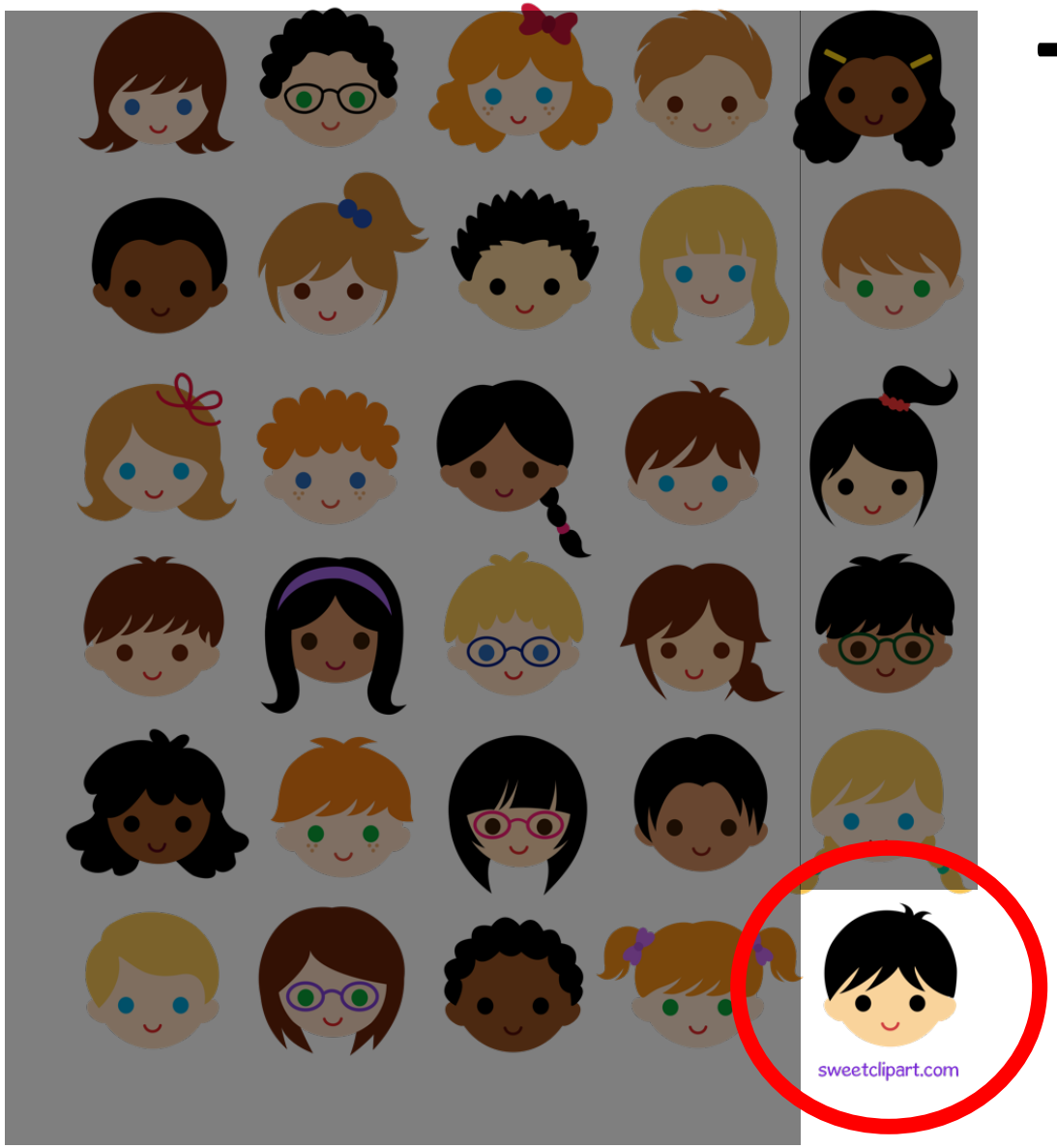
Concepts and data structure

- oTree relies on Class-Objects to structure the experiment and the data
- Each experiments comes with pre-defined structures that you would expect
  - Constants
  - Participants
  - Player
  - Group
  - (Sub)session
  - Pages

Conceptional overview

This is a session.

The entire set of all participants in your lab or online



This is a participant.

A session may consist of multiple rounds.

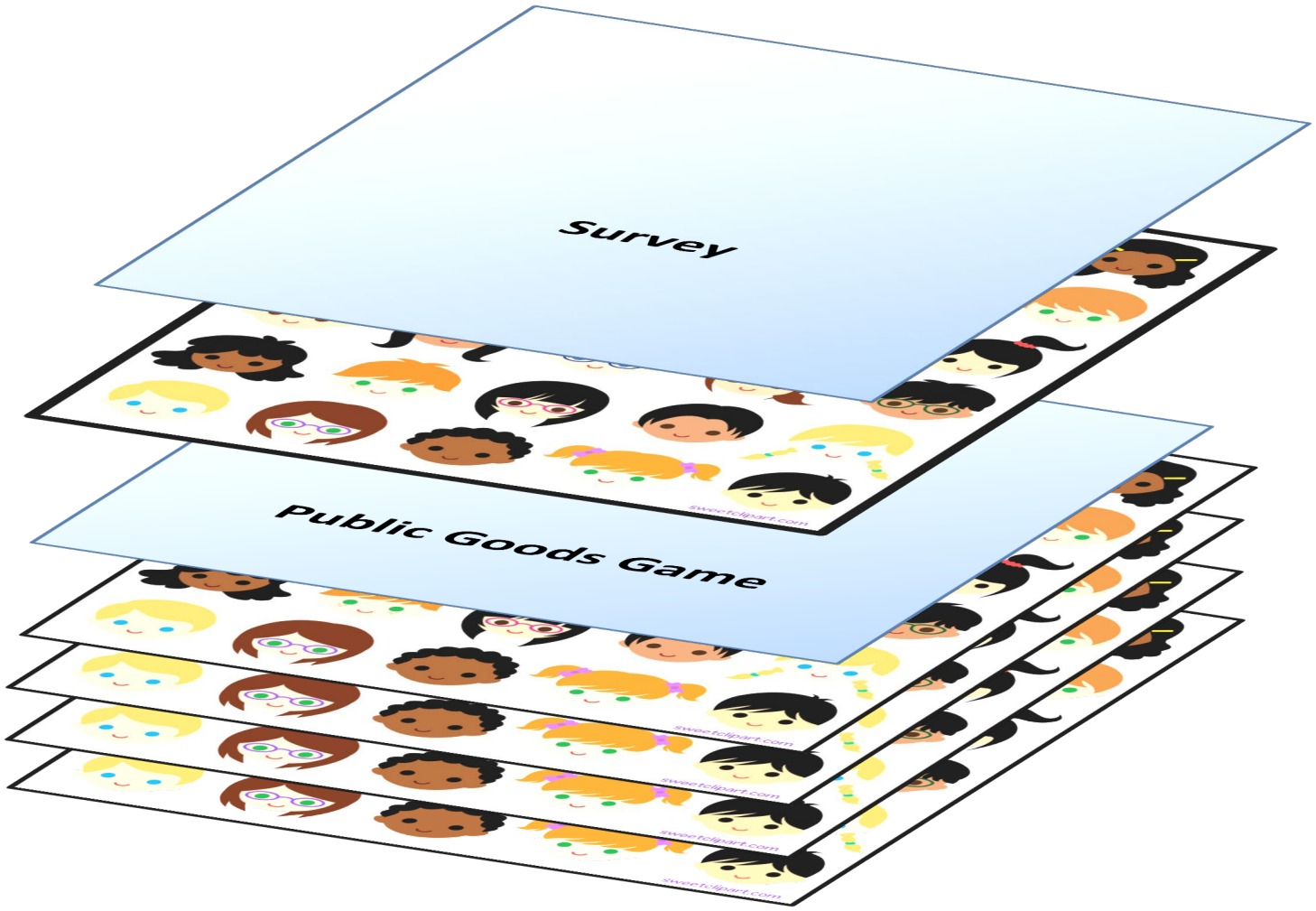
Subsession is a set of all players in one round

Player is an element of Subsession.

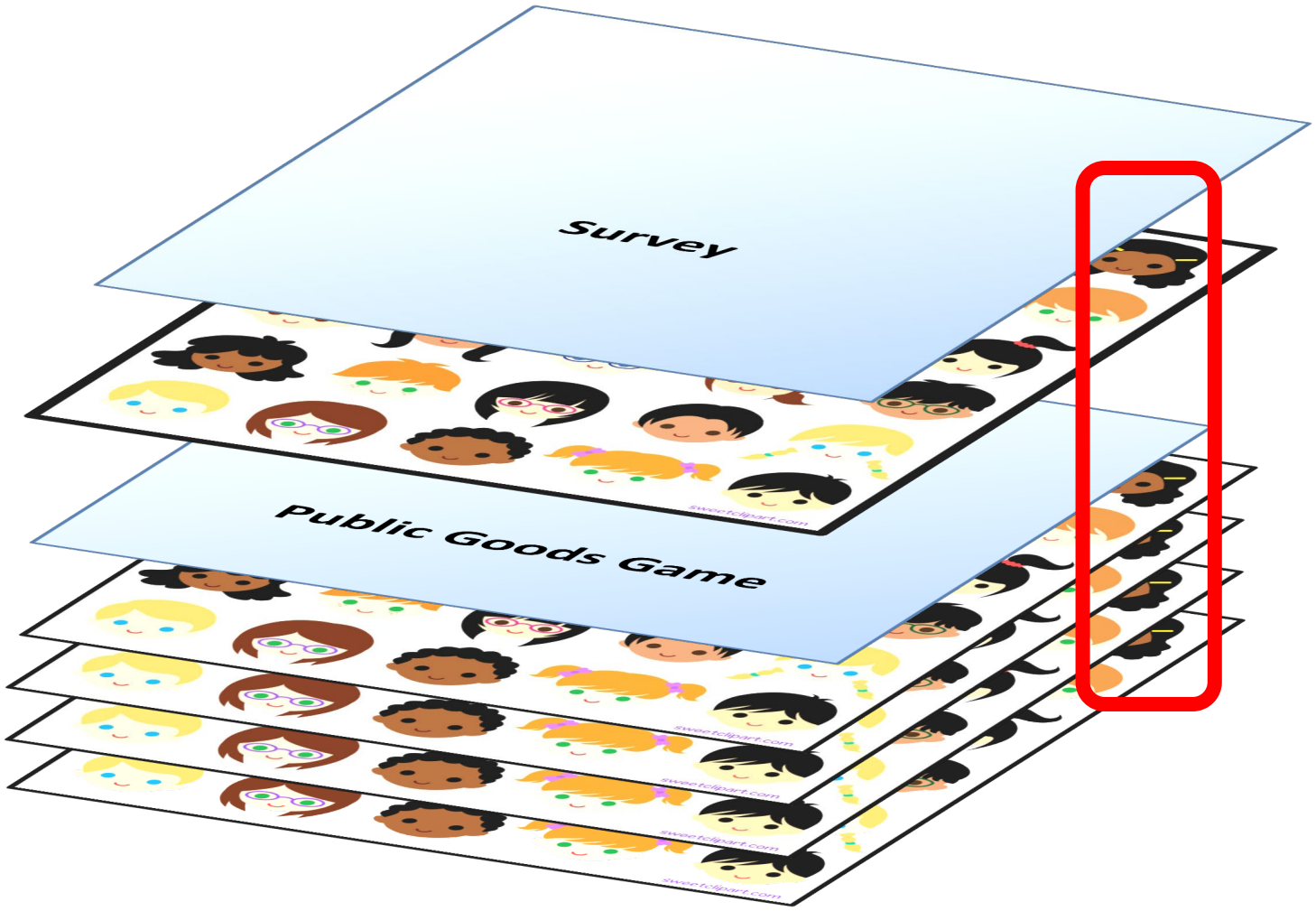
Note:
Participant ≠ Player

One participant contains the info about all players who he/she ‘owns’ across subsessions.

The group
is a set of players in one particular Subsession.

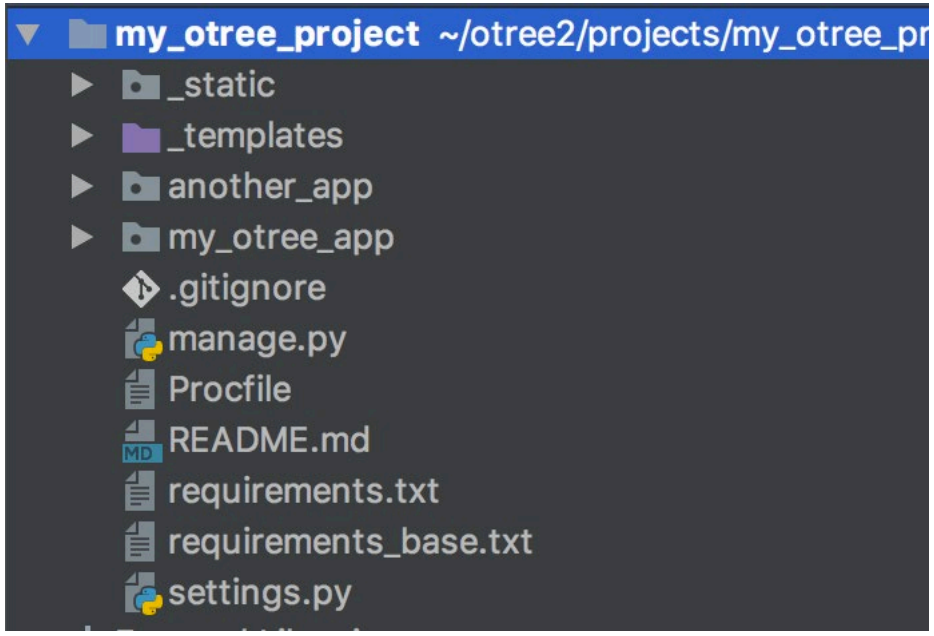


One session can contain **several** apps



Also if there are several apps, **participant** owns players across subsessions.

One session can contain **several** apps



in settings.py:

```
SESSION_CONFIGS = [
    {
        'name': 'some_apps',
        'display_name': "Demo apps",
        'num_demo_participants': 1,
        'app_sequence': ['my_otree_app', 'another_app'],
    },
]
```

Classes in Models.py

```
from otree.api import (...)

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

class Subsession(BaseSubsession):
    pass

class Group(BaseGroup):
    pass

class Player(BasePlayer):
    age = models.IntegerField(label='What is your age?', min=13, max=125)
    gender = models.StringField(
        choices=['Male', 'Male', 'Other'],
        label='What is your gender?',
    )
    crt_bat = models.IntegerField(
        label='''
        A bat and a ball cost 22 dollars in total. The bat costs 20 dollars more than the ball. How many dollars does the ball cost?'')
    crt_widget = models.IntegerField(
        label='''If it takes 5 machines 5 minutes to make 5 widgets, how many minutes would it take 100 machines to make 100 widgets?'')
    crt_lake = models.IntegerField(
        label='''In a lake, there is a patch of lily pads.
        Every day, the patch doubles in size.
        If it takes 48 days for the patch to cover the entire lake,
        how many days would it take for the patch to cover half of the lake?
        '''
    )
```

What is missing here?

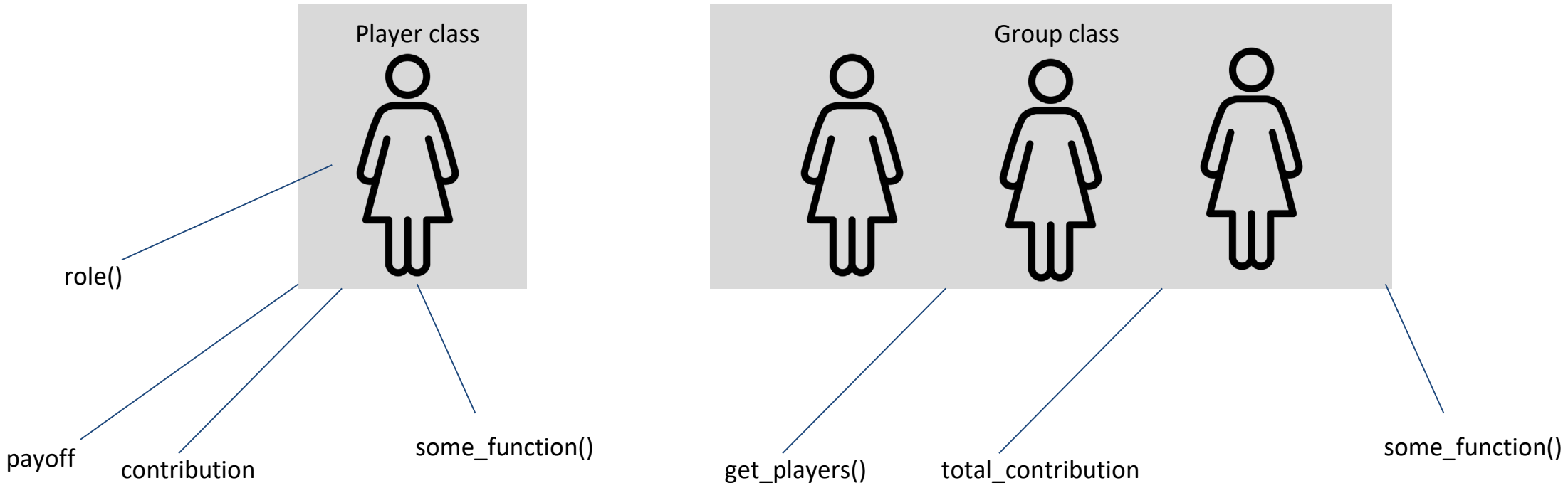
- Participants class usually used in practice to pass information between apps (participants.vars dict)
- Session class used to store information which is persisted across the entire experiment (session.vars dict)
- Both exist in every experiment in the background but you usually do not change them actively

Concepts and data structure

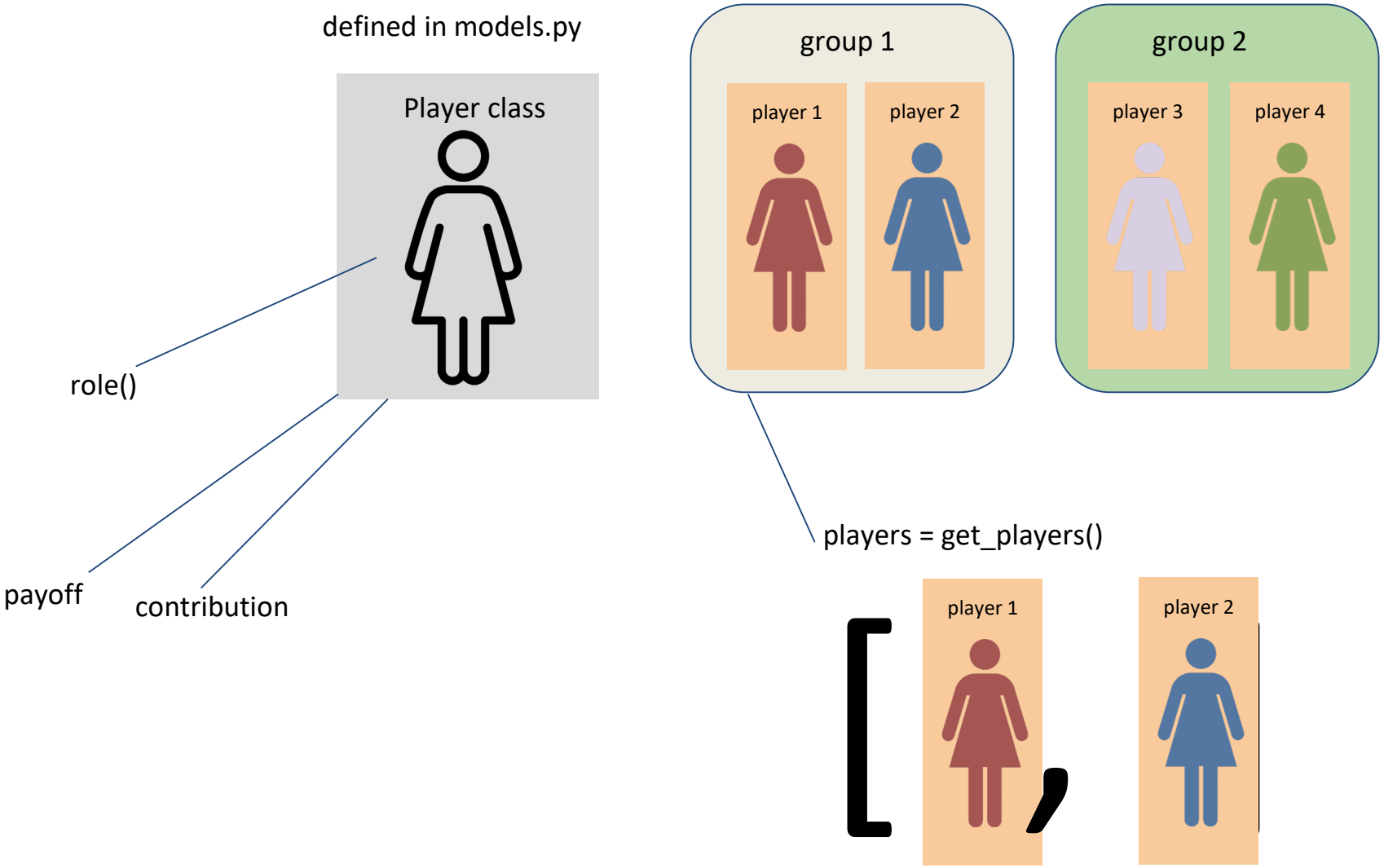
- Object hierarchy:
  - Session
    - Subsession
      - Group
        - Player
          - Page
- Keep this object hierarchy in mind when using self-statement in objects
  - https://otree.readthedocs.io/en/self/conceptual_overview.html#what-is-self

What is a model? Class and and instance?

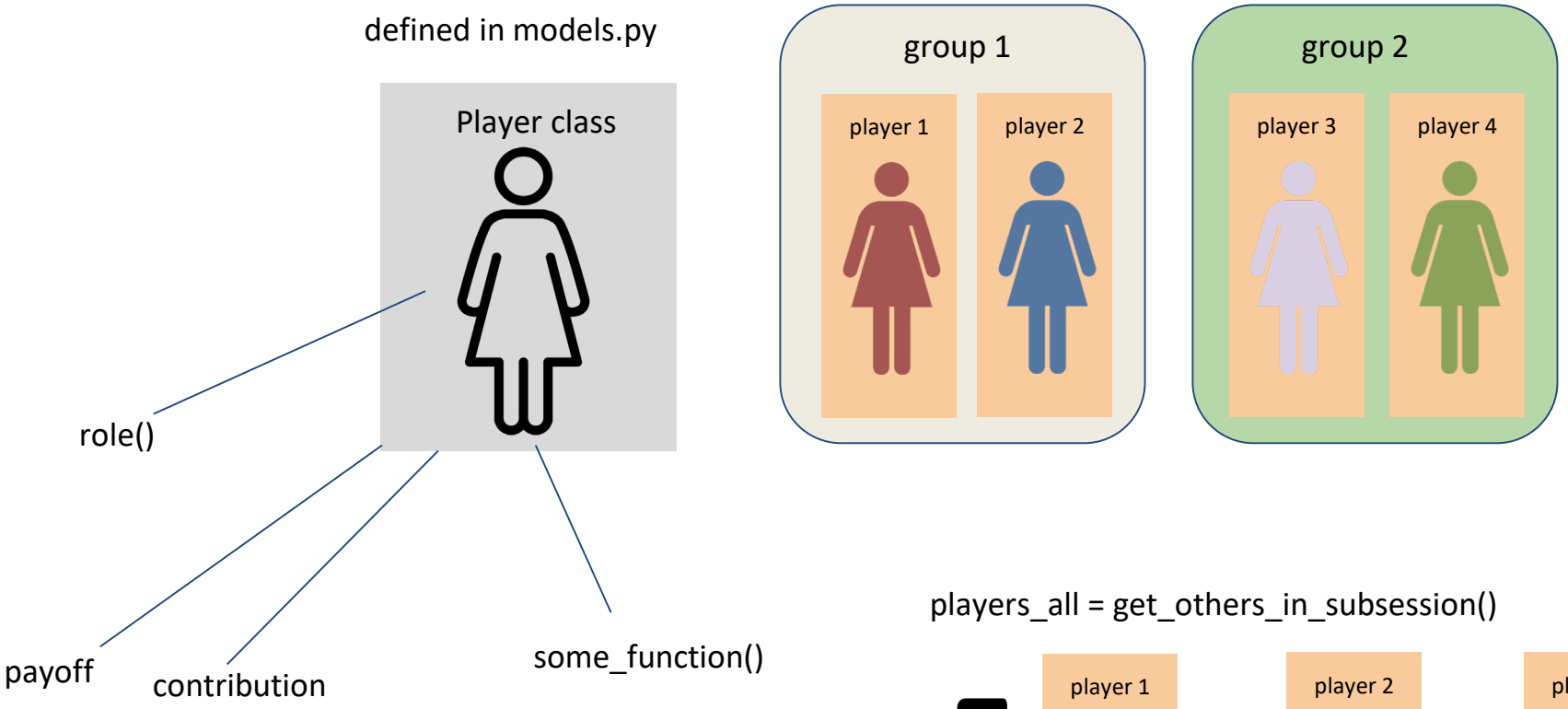
A **class** is a code blueprint for objects. **Objects** have variables (attributes) and functions (method)



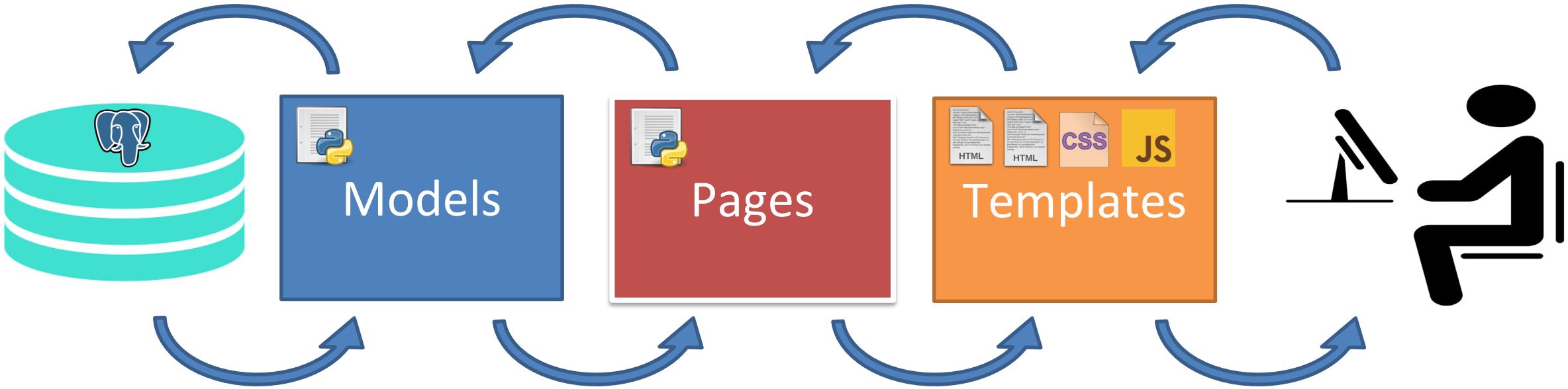
What is a model? Class and and instance?



What is a model? Class and and instance?



Summary



- Manages the DB
- Defines what to be stored
- Sets up the structure
- Functions on them

A model means a particular data structure on an object. Player, Group, Subsession …

A variable defined in a model is a field

- Defines pages to show
- Connecting templates and models
- Intermediate Calculations, Displaying rules
- Helper Functions

- Defines how to show things to user
- Connecting user with pages.py
- Partially HTML/, partially with some programmable chunks of code

Show oTree folder

Questions?