**What to do next?**

**The lessons fort the second session on Pages.py – finish the issues**

**and then JavaScript, Django: tricks likes changing buttons and other information?**

**Preliminary programme:**

**Day 1:**

**09:00-10:30 Welcome**

**11:00-12:30 Lesson: Basics: web**

**13:30-15:00 Lesson: Basics: python**

**15:30-17:00 Lesson: Basics: web & python together**

**Day 2:**

**09:00-10:30 Lesson: Bigger picture**

**11:00-12:30 Lesson: Advanced problems**

**13:30-15:00 Lesson: Interactive designs**

**15:30- Last session: online resources, getting started with own projects**

1. **Welcome:**

* Introductions: Who are you, what are your coding interests?
* Double checking everything works (Python interpreter, oTree, PyCharm)
* How to start up the server (working with the cmd/Terminal window)
* Server information: Data, monitor, links, payments, (admin, rooms?)
* Open the workshop folder/directory in PyCharm
* Overall structure of the course
* Zero is the first number in Python, One in the rest of the languages.

**Log for 90 minutes**

**0-10min Introductions and Goals**

* Hey, I’m…
* Who are you? Name, department, why you are here? How you coded before? Are you thinking of running an experiment

**10-20min Basic Starting**

* How far have you gotten with oTree? Everyone has installed it? Have you tried out to launch it? (2min)
* Download workshop folder, place it in an easy place (Your user folder!)
* Open PyCharm, command window (Terminal in Mac) – start the server, access the programme – collect the commands on a slide:
* **cd workshop** & **otree devserver**

**20-30min What can oTree offer**

* Demonstrate on your computer (maybe with a different set of applications, maybe the generic oTree page? or Maybe a slide show?):
* How to start a session
* How to see the data
* How to monitor what page each participant is on
* How to see the payments

**30-40min What do we do on the course**

* Plan of the course:

First day

1. Web site, html, JavaScript, CSS
2. Python: the basic structure
3. Putting them together

Second day

1. Bigger picture
2. Advanced coding
3. Real time interaction
4. Q&A

**40-70min Exercise on HTML5**

**HTML is made out of elements in <>. Some are long, and require an ending tag </>, some are short, that just appear. Each element you can then modify within the brackets, for example, change colour, or margins.**

**Django is this language made by web site developers that simplifies the HTML. Django code is within {% %} brackets.**

**The exercises will demonstrate with some examples how each of these languages work.**

* done
* w3 schools is the resource

**70-90min Recap**

* Go through the answers
* Any problems that occurred?
* Puzzles?
* What did we learn this lesson – short recap.
* Basic principles of html next
* w3 school is going to give answers…

1. **Lesson : Basic structure of oTree 1: web pages**

* Basic Django commands
* Basic HTML5, CSS, JavaScript
* Buttons, different content by Django ifs, JavaScript on clicks, inputs, forms
* pages.py: Manage (Add, Remove, Control) pages in the sequence
* “Inspect” html code on your Chrome Browser
* (Have a ready made models.py, tell the names in the html file.)

**Log for 90 minutes**

**0-10min To start**

* Recap, set goals for this session: we learned how to construct web pages last session, now we learn how to manage the sequence of pages + JavaScript
* Varying the static content, varying dynamic content with JavaScript
* What are the basic good practice coding principles with these coding languages? No repetition, if it can be avoided

**10-20min Topic**

* Pages.py file controls the sequence of pages.
* Defines each page as Python class and then defines the order of these pages
* The page classes inherit properties from generic oTree pages, set up already – easy to demonstrate for example with wait pages (you do not html code them but a lot of information appears anyways)

**20-40min Exercises**

* Do modifications to the page.py file
* Formfields (keep the models.py ready)
* Parameters for templates

**40-60min Recap + Topic extensions**

* **Go through the correct answers (10 mins) – crucial for the next exercises**
* What is self? What is a class?
* Useful functions for the wait page (maybe just mention as possibility):
* wait\_for\_all\_groups = True
* There are a few problems with the app: the amount of reciprocity is not limited to the maximum, like is the case with the trust decision. Code coding practice: do not allow for mistakes.
* One solution: inspect code, copy paste element, add a max by hand.

**60-80min JavaScript & Django exercises**

* Change the color of buttons
* Django if – else sentences

**80-90min Recap**

* Difference between dynamic and static code: when to use JS and when to use Django?

1. **Lesson: Basic structure of oTree 2: python**

* Look at the ready made aspects in the models.py file
* Adjust models
* Add a form/data point that you then collect on the page, fx. change answer scales
* Players, Groups, Subsession
* Get payoffs-fuctions

**2. Log for Lesson 2, 90 minutes, (After lunch lesson)**

**0-10min**

* What have we done so far: pages, html, what the users see and how that is constructed
* The experimenter’s side: what data to collect: models.py
* Python, Classes… some basic language understanding of python, ready built functions to use heritage (existed already with the pages and waiting pages
* **10-20min Basic models file-** Go through the model of the trust game, explain some features
* Defines the basic structure of the experiment: how many players, types, groups, rounds
* Defines what variables are for groups and players
* Defines payoff functions
* Can do a lot of other things, constrain variables etc.

**20-40min Exercises**

* Createa basic model (but have the html ready? but that creates errors)

**40-60 min Recap**

* Example: radio grid Big five: custom model, that defines everything basically in the models
* Subsession, reading session variables

**60-80 min Exercise**

* make a payoff function?

**80-90 min Recap**

* Models: defines your data structures
* Next we are going to modify models, pages, html templates all together

1. Lesson: Basic structure of oTree 3: putting web pages and python core together:

* Import commands
* Modify models.py, pages, and html templates all together
* Add forms, pages
* Make a payoff function
* Condition who sees pages
* Build from scratch by highlight the potential to copypaste from the earlier examples!!!
* Make a feedback questionnaire – send the code – and what did you learn today?

Day 2:

1. Lesson: Bigger picture

* settings.py
* html page, pages, models – need to correspond
* adding apps, adding projects
* Sequencing apps
* Revise previous topics

1. Lesson: Advanced problems

* Groups, ready made functions, etc.
* Waiting pages
* Interpreting the error codes

1. Lesson: Interactive applications

* Simple extensions through Channels: consumers.py and routing.py
* Make a chat box
* Inspect code -console, print on the command window, error feedback
* Modify an auction
* Django custom models

1. Last lesson:

* Online resources
* To get started with your project: what do you need to do?
* Split the projects in small goals, spread across groups, send codes in advance and then, we will investigate them in the follow-up session
* Feedback

Quiz, pictures, people give preferences and then they are allocated to roles by those preferences, strict quotas. Subgroup structure for, vote for the politician.

Twitter actions in the programme

Schedule:

Send lesson 1/Welcome on Monday 17th

Lesson 2/ 27th Friday.

Lesson 3/ 3rd January

Skype on the 8th.