

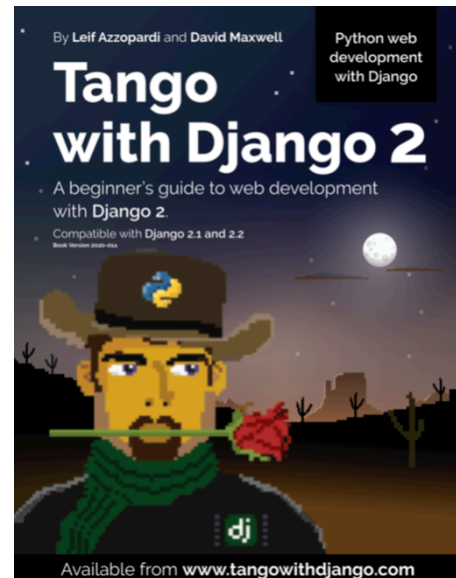
Web Application Development 2

Practical work & lab briefing sheet: weeks 1-5

Introduction

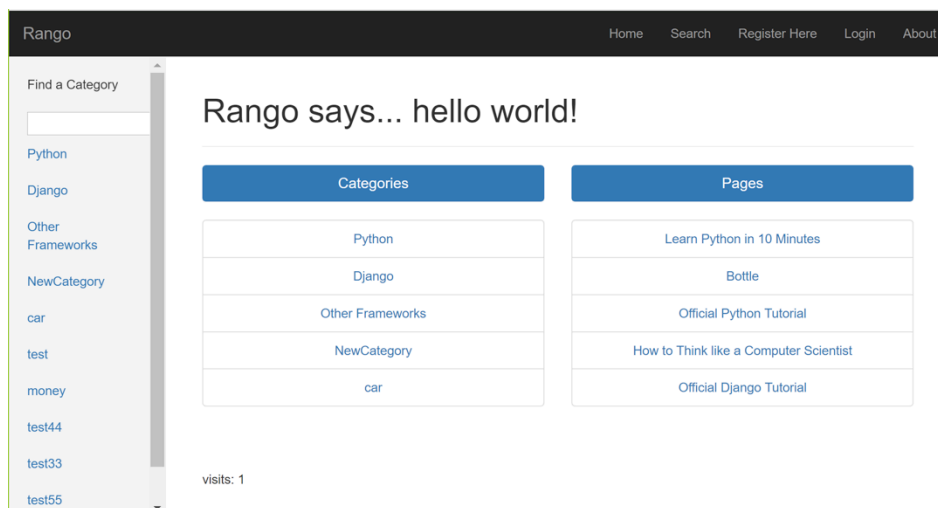
The course is based on “*Tango with Django 2: A beginners guide to web development*”, by Leif Azzopardi and David Maxwell, Lean Publishing (henceforth “TWD”). We will be using the version that has been updated for Django 2 – specifically the version that says “**2020-01a**” on the cover. You will be using this book primarily during the lab sessions, especially in the first half of the course.

TWD contains a step-by-step tutorial that guides the reader through the development of a web application called *Rango*, to be built using Python and Django. Rango lets users browse through user-defined categories to access various web pages. During the first five lab sessions in particular you will be working through the chapters of TWD to develop your own version of Rango.



We recommend that you work through all 20 chapters in TWD, however only your work on the first 10 chapters will be assessed. The development of your Rango application will account for 10% of your overall mark for the course. Moreover, successful completion of Rango will give you the skills that you need in order to work on the WAD2 team project later, which accounts for 40% of your overall mark.

Here is a screenshot of the final Rango application:



You can try out a completed version of Rango at <https://rangodemo2020.pythonanywhere.com>

Obtaining the Tango With Django ebook

The pdf of the TWD ebook can be downloaded from the course Moodle page.

Setup for first weeks

Ideally you should do this setup in week 1, in preparation for your first lab in week 2.

To begin we will setup 3 things: a virtual environment for your Rango exercise, a Django project, and a repository on GitHub. We'll then make a first commit of your work to GitHub.

Read Chapter 1 of TWD but skip Chapter 2. Instead, follow the instructions below to setup your working environment. For further help you can look at the instructions in the appendix entitled "Setting up your System" in TWD, or ask your lab demonstrators for assistance.

Note that this course is based on Python 3.9 and it is expected that you will develop Rango using this version of Python. Minor version differences are unlikely to matter but be sure to use Python 3 rather than Python 2, which has some syntax differences.

1. Setting up a virtual environment

The instructions in TWD chapter 2 and appendix describe a workflow using *mkvirtualenv* and associated commands. You are free to use these methods and then skip to stage (5), or you can ignore Chapter 2 in the book and follow the instructions (1) – (4) below which describe the use of Anaconda, which you might find easier if you are already familiar with it from University lab machines or work in other courses.

(0) You can install Anaconda on your own PC or laptop via

<https://www.anaconda.com/download/>

(1) Instructions vary per platform. e.g. on Windows click the magnifying glass symbol at the bottom left-hand corner of your screen and enter "anaconda" into the search box, then click on "Anaconda Prompt". This will start up the Anaconda command prompt with the "base" environment as shown in Figure 1. On a Mac, launch the Terminal app.

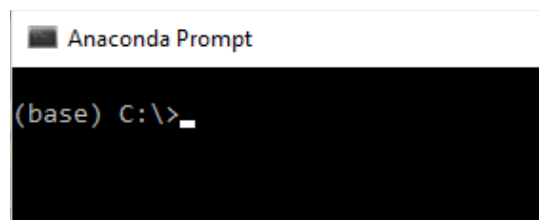


Figure 1. Anaconda Prompt

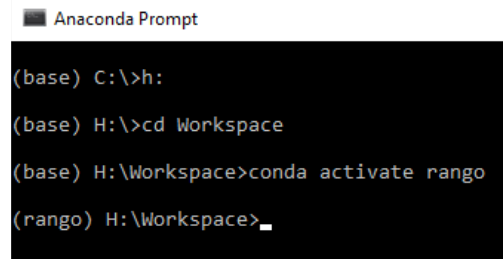
(2) You can list files in the current directory by entering the command **"dir"** (**"ls"** on Mac/Linux) and change directories with **"cd"**.

(3) Switch to your home directory by entering H:¹ at the command prompt. Move to the folder Workspace using **cd Workspace**. If this folder doesn't exist already, create it using **mkdir Workspace** and then move to the folder using **cd Workspace**. This document will assume the existence of an H:\Workspace directory. If you choose to work elsewhere in your file system, substitute in your preferred path for H:\Workspace when it is mentioned.

(4) Create a virtual environment for your development of the Rango application using the following command and follow the instructions: **conda create -n rango python=3.9**. You can replace "rango" with another virtual environment name if you prefer. The command prompt will not activate your new environment automatically, so you need to run the following command to activate it: **conda activate rango**.

¹ If your home directory is not mapped to H:, click on "This PC" to see what drive your home directory is mapped to.

At this point, you are now inside an Anaconda virtual environment – you can see every prompt line will begin “(rango)”. See Figure 2.



```
Anaconda Prompt
(base) C:\>h:
(base) H:\>cd Workspace
(base) H:\Workspace>conda activate rango
(rango) H:\Workspace>_
```

Figure 2. Rango environment activated.

(5) Install Django using the following command: **pip install django==2.2.26**

We will be using Django 2.2.26 and Python 3.9 in WAD2. Django 2.2 is a Long Term Support (LTS) version of Django, and supported by the TWD book. We will mark your coursework with Django 2.2.26 and Python 3.9.7. You can use a newer version of Django in future projects, but it is good software engineering practice to use Long Term Support (LTS) releases to minimise software development efforts in any project. You can look at the following link to find out which Django versions are LTS releases for Django: <https://www.djangoproject.com/download/>

(6) Install Pillow using the following command: **pip install pillow**

Now that you have your virtual environment created with Django and Pillow installed, you are ready to create a project. However, there is another step it is sometimes best to take first.

2. Setting up Git

You are going to create a project and use git and the external GitHub server for version control. It is often an easier workflow to start the project on GitHub before you do any work on your local machine.

A Git crash course

Before completing the steps below, read through the appendix in TWD entitled “A Git Crash Course” to familiarise yourself with Git. Git can be quite challenging for those who have never used it before, so make sure to follow this chapter carefully and do not skip this part!

(1) If you don’t already have a GitHub account, create one now on www.github.com.

See Appendix to this doc about making personal access token

(2) You will need a repository for your project, so on GitHub click the button to create a new repository

- Call it ‘tango_with_django_project’
- Be sure to make this repo **public** so that automated tests can access it when we mark it
- Add a README, and importantly, **choose a .gitignore file for Python**. This will prevent unwanted files being committed

You are now going to use the git commands from the Anaconda prompt.

(3) Your machine might already have Git installed. You could type ‘git’ at the prompt to check. If it’s there, you can go on to step (4). Otherwise, you can install Git via the following link: <https://git-scm.com/download/> (You might have to close and re-open the Anaconda prompt after installing Git before it is available)

(4) `cd` into your Workspace directory if you are not there already

(5) Set your user name: **`git config --global user.name "Firstname Surname"`** (replace "Firstname Surname" by your first name and surname, but make sure to retain the quotes). If you receive a message saying you are not in a git directory, first type: **`git init`** and then rerun the `user.name` command

(6) Set your user email: **`git config --global user.email "2099999z@student.gla.ac.uk"`** (replace 2099999z by your GUID, but make sure to retain the quotes).

(7) Enter the following command, to setup a local directory linked to the repository you just created on GitHub:

`git clone https://github.com/<username>/tango_with_django_project.git`

Replace <username> in the above command by your GitHub username. You have now set up a remote repository for Rango on GitHub, and have linked it to your local repository in your workspace.

Now you can use `add`, `commit`, `pull`, `push`, etc on your repository. But you haven't written anything yet to commit

3. Creating the Django project & commit it to GitHub

(0) For the steps below to work you need to be in your virtual environment. Remember, you can check this by looking for (rango) before the prompt, like in Figure 2. To activate it, you can always type in **`conda activate rango`**, and to deactivate it type **`conda deactivate`**.²

(1) The clone command above should have created a new directory within your Workspace called `tango_with_django_project`. Enter this directory (i.e, **`cd tango_with_django_project`**)

(2) To create a new Django project, issue the following command from within this directory (**note** the dot (full stop) after a space at the end of the command. If you omit this you will get an extra layer of subdirectories, which isn't a huge problem but you might find it annoying) :

`django-admin startproject tango_with_django_project .`

(3) You've created a web app! If you look inside the folder you're working in, you should see that Django has created a number of files and directories for you. We will talk through these in lectures. Run the app by issuing the command

`python manage.py runserver`

(4) Open a web browser and visit the link <http://127.0.0.1:8000/>.

You should see a congratulations message. Well done! Now press control-C to terminate the execution of `manage.py`, so you can continue working on the Anaconda prompt.

(5) We now have the basic shell of a Django app, and we can commit this to our GitHub repo. Make sure you are in directory `H:\Workspace\tango_with_django_project` and issue the following git commands:

`git add *`
`git commit -m "first commit"`
`git push`

² Information about other commands relating to virtual environments that you can run is available from the following URL: <https://conda.io/docs/user-guide/tasks/manage-environments.html>.

This sequence of commands is how you write the local files you're creating while developing your web app to a repository on GitHub's external servers.

If you refresh the page at GitHub.com, you should see your new files have been pushed to the server.

You might note that an initial database file has been created in your directory called `db.sqlite3`, and yet the **add** command has not added it and it does not appear on GitHub. This is the behaviour we want to see – your `.gitignore` file has filtered this out of your additions. You generally do not want to add databases to version control. Future lectures will cover more details on this.

It is very important to ensure that your git repository is set up correctly. Inside `H:\Workspace` you should have a folder `tango_with_django_project`. Inside that folder, you should have the file `manage.py` (and `db.sqlite3` if you have already executed the web application) and another folder `tango_with_django_project`. Inside the latter folder, you should initially have four files: `__init__.py`, `settings.py`, `urls.py` and `wsgi.py` (and the folder `__pycache__` if you have already executed the web application). Your file structure on GitHub should be similar, except without the `db` or `pycache` if you have a `.gitignore`. To verify things are working, you can compare your own remote repository with an example one at https://github.com/wad2/tango_with_django_project

If your git repository structure is not analogous to the structure described in the previous paragraph, **do not go any further**. Instead, repeat the above steps again as necessary. It is very important that your directory structure is set up correctly, otherwise the automated tests that will be run on your Rango app later on in the course will be likely to fail. It is much better to take the time now to complete this step correctly rather than to persist with an incorrect directory structure as this could make life very difficult for you later on (as many students have found in the past).

Lab sessions for weeks 2-5

During the lab sessions for weeks 2-5, you should continue working on developing Rango, following Chapter 3 onwards of TWD. As mentioned above, the minimum that you should complete is up to the end of Chapter 10 of TWD, but it is recommended that you develop Rango up to the end of Chapter 20 of TWD. Use any IDE or text editor for Python, e.g. the IDLE desktop application; another possibility is to develop Rango within PythonAnywhere. A recommended development schedule is as follows, although work is only assessed at the final deadline.

Week no.	Week ending	TWD chapters
1	14/01/2022	1, appendix on Git
2	21/01/2022	3(*), 4
3	28/01/2022	5, 6
4	04/02/2022	7, 8
5	11/02/2022	9, 10

(*) You will find that you have already completed the tasks in Sections 3.1 and 3.2 of TWD as a consequence of working through this lab sheet, although can still read through the sections for more details on these stages.

The deadline for completing Rango up to the end of Chapter 10 is **Friday 11 February at 6.30pm**. For more information, see the separate assessed exercise document on the Rango exercise.

Appendix: Personal Access Token for GitHub

If you have never used GitHub before, you will need to create a new account at GitHub.com. This is very straightforward and requires no instruction. However, there is a step that might need some more explanation

GitHub disabled simple username & password authentication for command line operations. Authentication is now controlled via a **Personal Access Token**. This is a sequence of characters that looks like a password.

It is a simple process to generate and use a PAT, but requires a few steps of setup. The process is described at the following link. Follow steps 1 to 9, then 'Using a token on the command line':

<https://docs.github.com/en/authentication/keeping-your-account-and-data-secure/creating-a-personal-access-token>

Instructions at the link should be straightforward. For step 8, select "repo"