

Jinja – templating language built for python – generating dynamic content

Jinja is a modern and designer-friendly templating language for Python, modelled after Django’s templates. It is fast, widely used and secure with the optional sandboxed template execution environment:

Jinja template designer

<https://jinja.palletsprojects.com/en/2.11.x/templates/>

Blog website

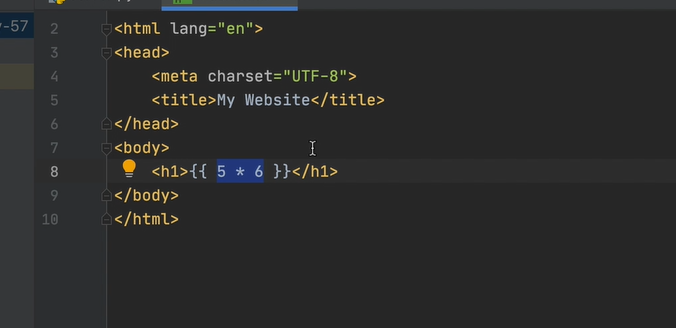
<https://updateyourfooter.com/>

Flask

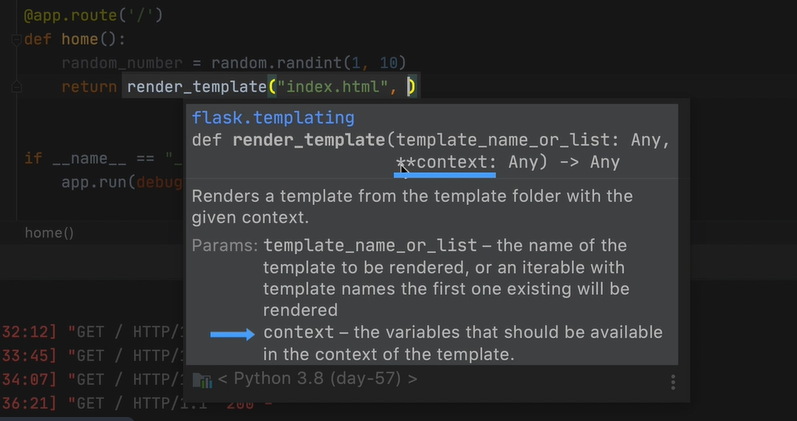
<https://flask.palletsprojects.com/en/1.1.x/quickstart/#rendering-templates>

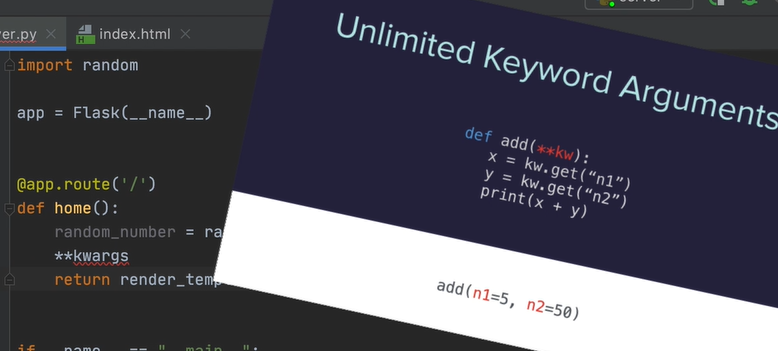


Jinja the stuff in the double {{}} is interpreted, run as python code



Random number in Jinja



\*\*contexts is like \*\*kw

 you can add as many keyword arguments as you want and each of these keyword arguments need to have a name for the variable and a value for the variable.

The reason why they need have a name and value is because we can refer to

that variable by its name inside the templated HTML file

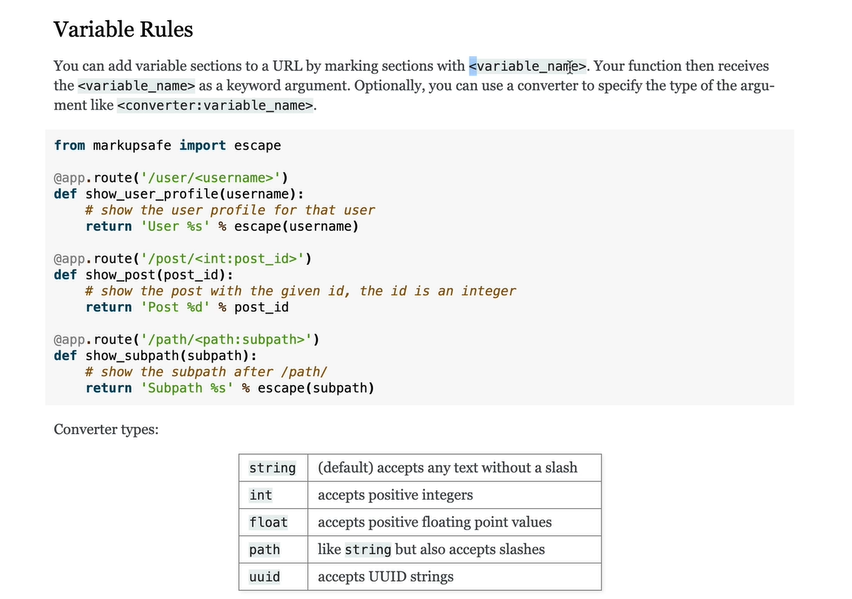
import random  
# boiler plate to spin up the server  
from flask import Flask**,** render\_template  
app = Flask(\_\_name\_\_)  
@app.route("/")  
def home\_route():  
 #puting the random number generator in the home route  
 random\_number = random.randint(**1,10**)  
 #send random\_number as keword arg , the arg gets passed to HTML in route  
 return render\_template("index.html"**,** num=random\_number)  
if \_\_name\_\_ =="\_\_main\_\_":  
 app.run(debug=True)

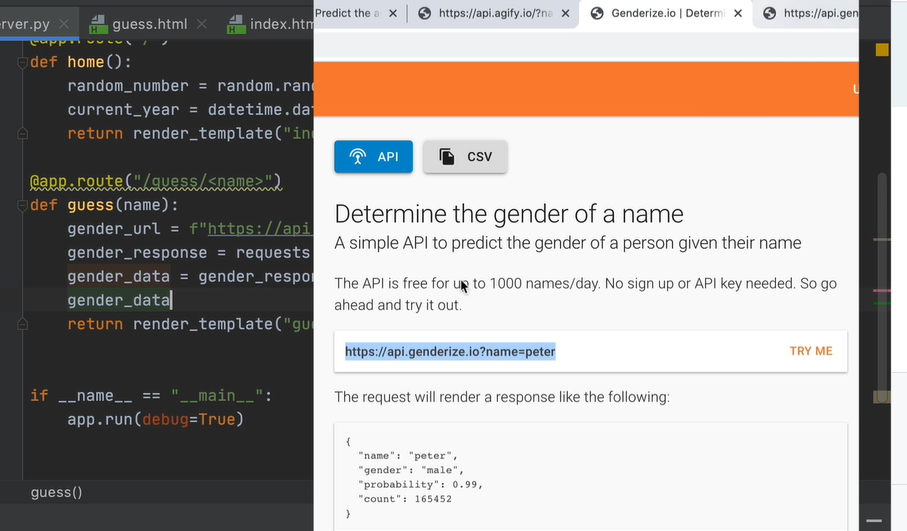
Working with APIs and jinga

<https://genderize.io/>

<https://agify.io/>

put angle brackets around the part of the URL that will be taken in as a variable





Multiline Statements with Jinja –

If statements and for lopps

Fake Blog Data

Add a {% for each of the lines %}

<https://www.npoint.io/>

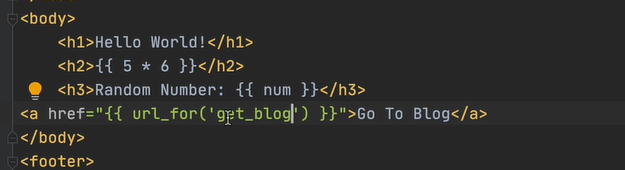
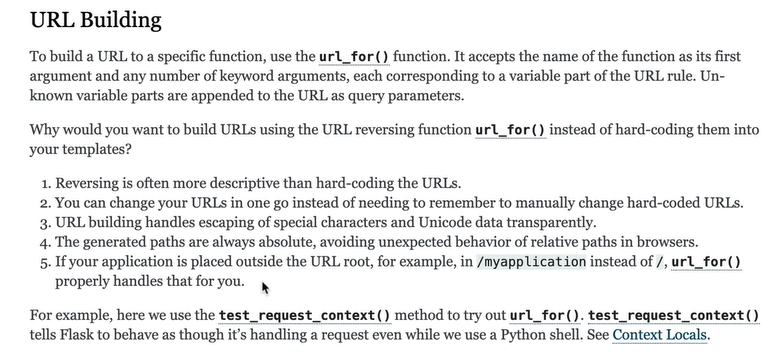
npoint JSON BIN

<https://www.npoint.io/docs/5abcca6f4e39b4955965>

Flask Docs Routing

<https://flask.palletsprojects.com/en/1.1.x/quickstart/#routing>

Building a URL with Flask



Blog Capstone Project Part 1 - Templating

1. Head over to the course resources and download the starting files for this project.

2.  Run the **main.py** file and you should see the following styling and website rendered:

3. Using the API for our [blog posts we created on n:Point](https://www.npoint.io/docs/5abcca6f4e39b4955965), render all the blogs' title and subtitles on the home page. e.g

4. Make a "**Read**" anchor tag at the end of each blog post preview link to a page with the entire blog - title, subtitle and body. The individual blog posts should live at the path: URL/post/blog\_id

e.g.