Placeholders in jinja2 Template – Python

Last Updated : 05 Feb, 2023

Web pages use HTML for the things that users see or interact with. But how do we show things from an external source or a controlling programming language like Python? To achieve this templating engine like Jinja2 is used. Jinja2 is a templating engine in which placeholders in the template allow writing code similar to Python syntax which after passing data renders the final document. In this article we will cover some of the points as mentioned below:

- Template Variables
- Template if Statements
- Template for Loops
- Template Inheritance

Let's start by creating a virtual environment. It's always a good idea to work in a virtual environment as it will not cause changes to the global system environment. For using Jinja2 in Python, we need to install the Jinja2 library.

```
pip install Jinja2
```

Template Variables in Jinja2

Jinja2 is a Python library that allows us to build expressive and extensible templates. It has special placeholders to serve dynamic data. A Jinja template file is a text file that does not have a particular extension.

Syntax of Template Variables in Jinja2

For a placeholder, we have the following syntax in Jinja2.

```
{{variable_name}}
```

Example

In an HTML file called **index_template.html**, write the following code.

HTML

```
<!-- index_template.html -->

Hello {{pl_name}}! Your email is: {{pl_email}}
```

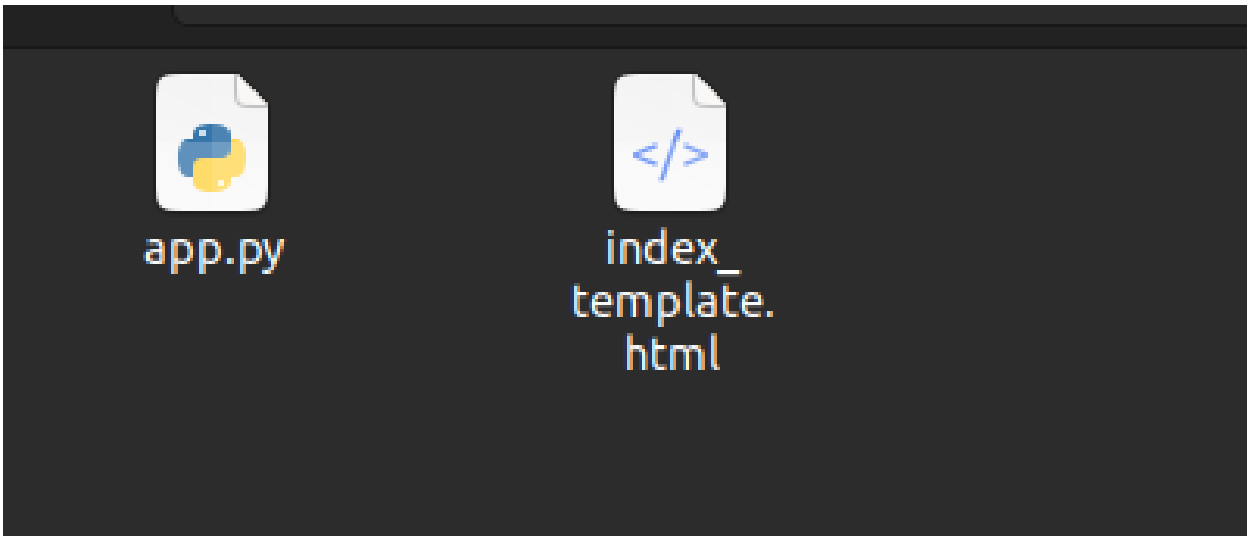
app.py

We open this HTML file in Python and read its content to a variable called content. Pass the content to *Template,* and store it in the *template* variable. Now, we will pass the name and email to render and replace the placeholders {{pl_name}} and {{pl_email}} respectively, by using template.render; and store this in rendered_form.

Python3

```
# app.py

# import Template from jinja2 for passing the content
from jinja2 import Template

# variables that contain placeholder data
name = 'John'
email = 'you@example.co'


# Create one external form_template html page and read it
File = open('index_template.html', 'r')
content = File.read()
File.close()

# Render the template and pass the variables
template = Template(content)
rendered_form = template.render(pl_name=name, pl_email=email)


# save the txt file in the form.html
output = open('index.html', 'w')
output.write(rendered_form)
output.close()
```

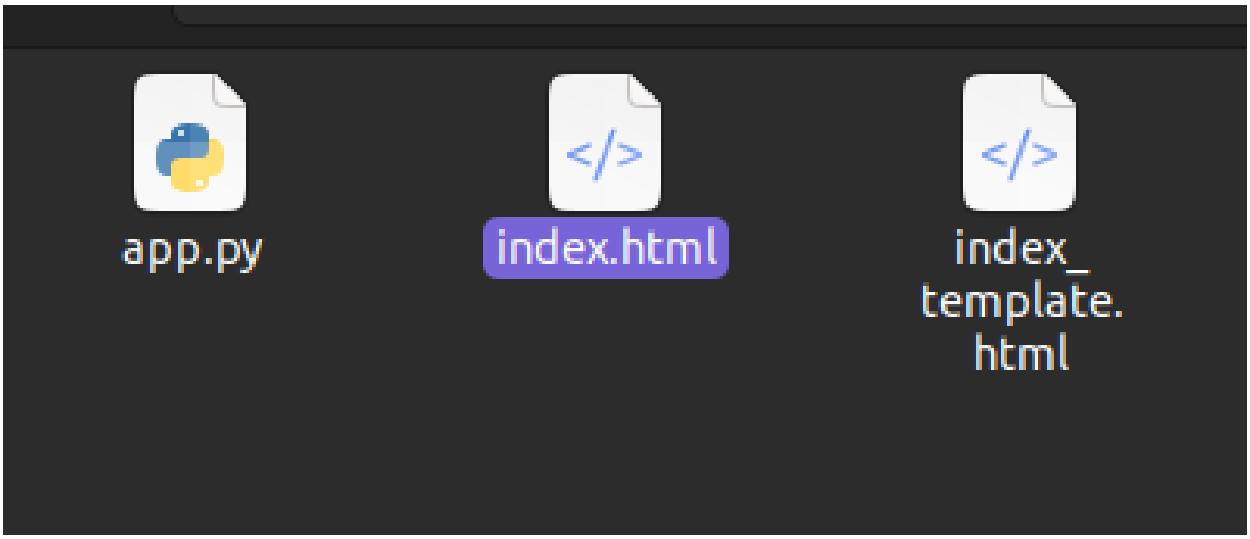
The index.html file is created in the variable output. Write the content to this HTML file using output.write(rendered_form). Below are the two files before running the Python program.

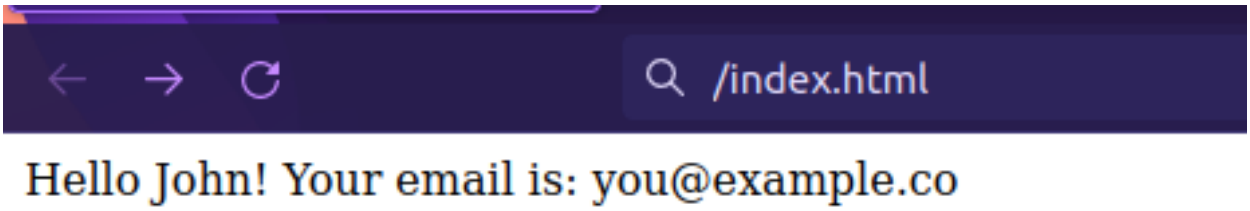


Now, run app.py using the following command:

```
python app.py
```

A new file is created named *index.html. Open it and see the code. The placeholder text is changed to the values that we passed.*





Conditionals and Looping in Jinja2

Jinja in-line conditionals are started with a curly brace and a % symbol, like {% if condition %} and closed with {% endif %}. You can optionally include both {% elif %} and {% else %} tags, and for loop, we use **{% for index in numbers %}** and end with **{% endfor %}.**

Syntax of Conditionals and Looping

For conditions, we have the following syntax in Jinja2.

For loop	If condition
{% for i in numbers %} {% endfor %}	{% if i % 2== 0 %} {% endif %}

Example

A list can also be passed using Jinja. To iterate through the list and for using conditions, similar to Python we use loop and if-condition. Let's pass a list of numbers as well:

Python3

```
# app.py

# import Template from jinja2 for passing the content
from jinja2 import Template
```

```
# variables that contain placeholder data
name = 'John'
email = 'you@example.co'
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9]

# Create one external form_template html page and read it
File = open('index_template.html', 'r')
content = File.read()
File.close()

# Render the template and pass the variables
template = Template(content)
rendered_form = template.render(pl_name=name,
                                pl_email=email, numbers=numbers)


# save the txt file in the form.html
output = open('index.html', 'w')
output.write(rendered_form)
output.close()
```

index_template.html

Here we will iterate the number and print the even number from the list.

HTML

```
<!-- index_template.html -->

Hello {{pl_name}}! Your email is: {{pl_email}}
<br>
Even numbers:
{% for i in numbers %}
    {% if i%2==0 %}
        {{i}}
    {% endif %}
{% endfor %}
```

Output:

Hello John! Your email is: you@example.co
Even numbers: 2 4 6 8

Template Inheritance in Jinja2

Template inheritance is a very good feature of Jinja templating. All that is needed is to add a {% extend %} tag. The home page {% extends “base. html” %} inherits everything from the base template.

Syntax of Jinja extend block

For Inherit the base page, we have the following syntax in Jinja2.

```
{% block content %}
<Code>
{% endblock %}


{% extends "base.html" %}
{% block content %}
   <Code>
{% endblock %}
```

Example

Here, we want to use the same HTML content across pages like a Website name or a search bar without repeating the HTML code. For this Jinja2 has a feature called template inheritance. Suppose we need this heading and search bar on every page without repeating the code:

base.html: This is the code of the website name and search bar.

HTML

```
<!-- base.html -->

<h1>My Blog</h1>
<input type="search">
<button>Search</button>

<!-- Child page code goes between this -->
{% block content %}{% endblock %}

<!-- You can continue base.html code after this if you want -->
<br><br>
```

Let's include this in our **index_template.html.** In the child template or the page, you want to include the website name and search bar.

HTML

```
<!-- index_template.html -->

<!-- include base.html -->
{% extends "base.html" %}

<!-- Write any code only in this block -->
{% block content %}

Hello {{pl_name}}! Your email is: {{pl_email}}
<br>
Even numbers:
{% for i in numbers %}
    {% if i%2==0 %}
        {{i}}
    {% endif %}
{% endfor %}

<!-- end the block -->
{% endblock %}
```

Output:

P prasa... 61

Previous Article

Template Inheritance in Flask

Next Article

How to serve static files in Flask

Similar Reads

Python Falcon - Jinja2 Template

Python Falcon is a lightweight and minimalist web framework designed for building web APIs, with a particular emphasis on simplicity, speed, and efficienc...

6 min read

Placeholders in Tensorflow

A placeholder is a variable in Tensorflow to which data will be assigned sometime later on. It enables us to create processes or operations without the...

2 min read

Templating With Jinja2 in Flask

Flask is a lightweight WSGI framework that is built on Python programming. WSGI simply means Web Server Gateway Interface. Flask is widely used as a...

6 min read

Python | Document field detection using Template Matching

Template matching is an image processing technique which is used to find the location of small-parts/template of a large image. This technique is widely used...

2 min read

Python | Set Background Template in kivy

Kivy is a platform-independent GUI tool in Python. As it can be run on Android, IOS, Linux and Windows, etc. It is basically used to develop the Android...

2 min read

Template Method - Python Design Patterns

The Template method is a Behavioral Design Pattern that defines the skeleton of the operation and leaves the details to be implemented by the child class. Its...

4 min read

Implementing News Parser using Template Method Design Pattern in Python

While defining algorithms, programmers often neglect the importance of grouping the same methods of different algorithms. Normally, they define...

4 min read

Python Pyramid - HTML Form Template

Pyramid is an open-source web application development framework written in Python. It is a flexible and modular framework that is used to build web...

4 min read

String Template Class in Python

In the String module, Template Class allows us to create simplified syntax for output specification. The format uses placeholder names formed by $ with valid...

4 min read

Template matching using OpenCV in Python

Template matching is a technique for finding areas of an image that are similar to a patch (template). A patch is a small image with certain features. The goal of...

6 min read

Article Tags : Python Technical Scripter Python Flask Technical Scripter 2022

Practice Tags : python

GeeksforGeeks

Corporate & Communications Address:- A-143, 9th Floor, Sovereign Corporate Tower, Sector- 136, Noida, Uttar Pradesh (201305) | Registered Address:- K 061, Tower K, Gulshan Vivante Apartment, Sector 137, Noida, Gautam Buddh Nagar, Uttar Pradesh, 201305

GET IT ON Google Play Download on the App Store

Company Languages

About Us
Legal
In Media
Contact Us
Advertise with us
GFG Corporate Solution
Placement Training Program
GeeksforGeeks Community

Python
Java
C++
PHP
GoLang
SQL
R Language
Android Tutorial
Tutorials Archive

DSA

Data Structures
Algorithms
DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Data Science & ML

Data Science With Python
Data Science For Beginner
Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD

Inteview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

System Design Bootcamp
Interview Questions

School Subjects
Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

GeeksforGeeks Videos
DSA
Python
Java
C++
Web Development
Data Science
CS Subjects

@GeeksforGeeks, Sanchhaya Education Private Limited, All rights reserved