



Introduction to Flask

What is web framework ?



What is web framework ?

- ❑ A web framework is like a toolkit for building websites and web applications.
- ❑ It provides ready-made components and tools that handle common web development tasks, allowing developers to focus on creating unique features for their websites instead of rewriting common functionalities such as handling webpage requests, displaying content, and interacting with databases.

Definition:

A web framework is a toolkit for developing websites and web applications.

Purpose:

It helps automate common tasks to speed up development and reduce errors.

Key Components of Web Frameworks

Routing:

Directs users to different pages on a website.

Templates: Used to display information dynamically.

Database Interaction:

Simplifies data handling and storage.

Session Management: Keeps track of user activity across pages.

Types of Web Frameworks

Full-Stack Frameworks:

Provide comprehensive support for all web development needs (e.g., Django, Ruby on Rails).

Microframeworks:

Offer minimalistic tools for smaller applications (e.g., Flask, Sinatra).

Asynchronous Frameworks: Specialized for handling many tasks at once, useful for real-time applications (e.g., Node.js).

Benefits of Using a Web Framework

Efficiency: Speeds up the development process by providing pre-built components.

Organization: Helps maintain a clean and manageable codebase.

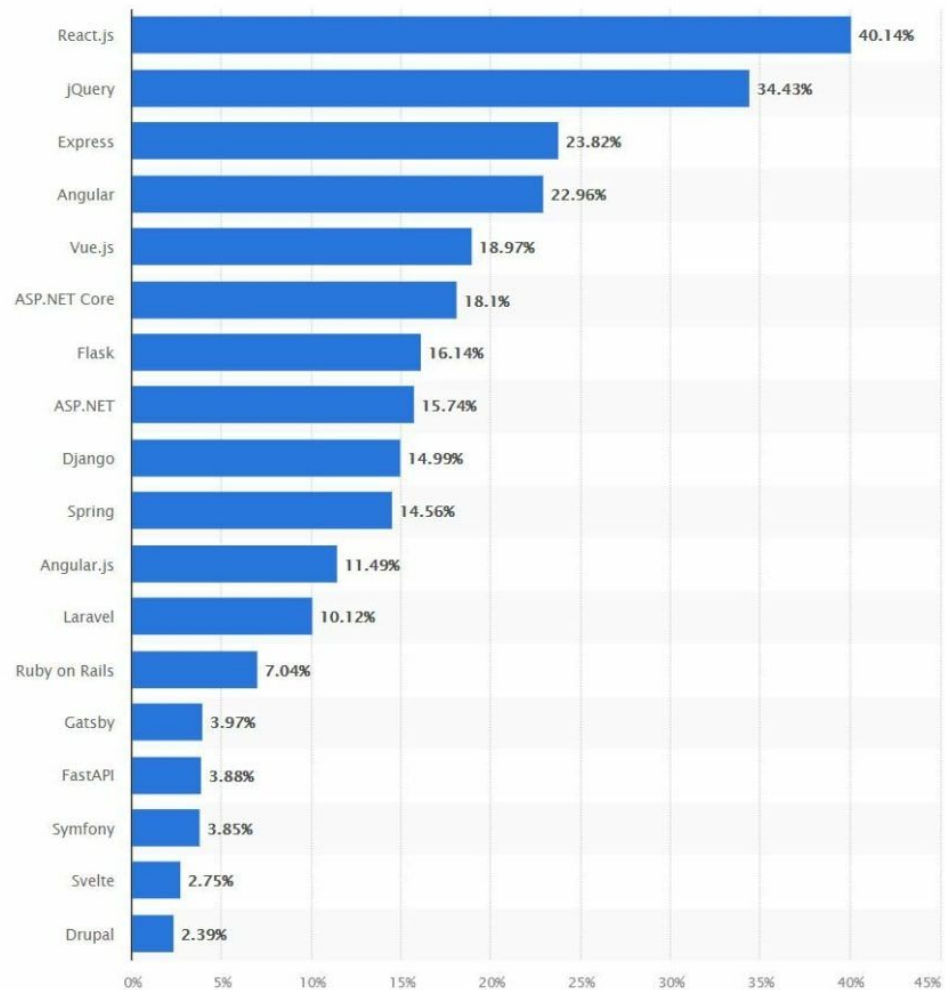
Security: Includes features to protect against common vulnerabilities.

Scalability: Equips developers with tools to handle growing amounts of traffic and data.

Conclusion

Summary: Web frameworks simplify the process of web development, making it faster and more secure.

Final Thought: Choosing the right framework depends on your project needs, size, and the specific functionalities you require.



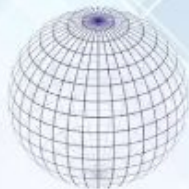


Introduction to Flask



What is Flask?

Flask is a web application framework written in Python.



Large community for
Learners and Collaborators



Let's get started then!

Armin Ronacher



Open Source



What is a Web Framework?



Libraries



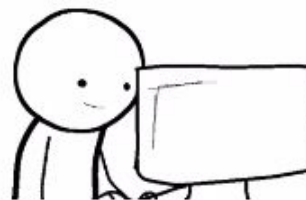
Modules



Web Developer

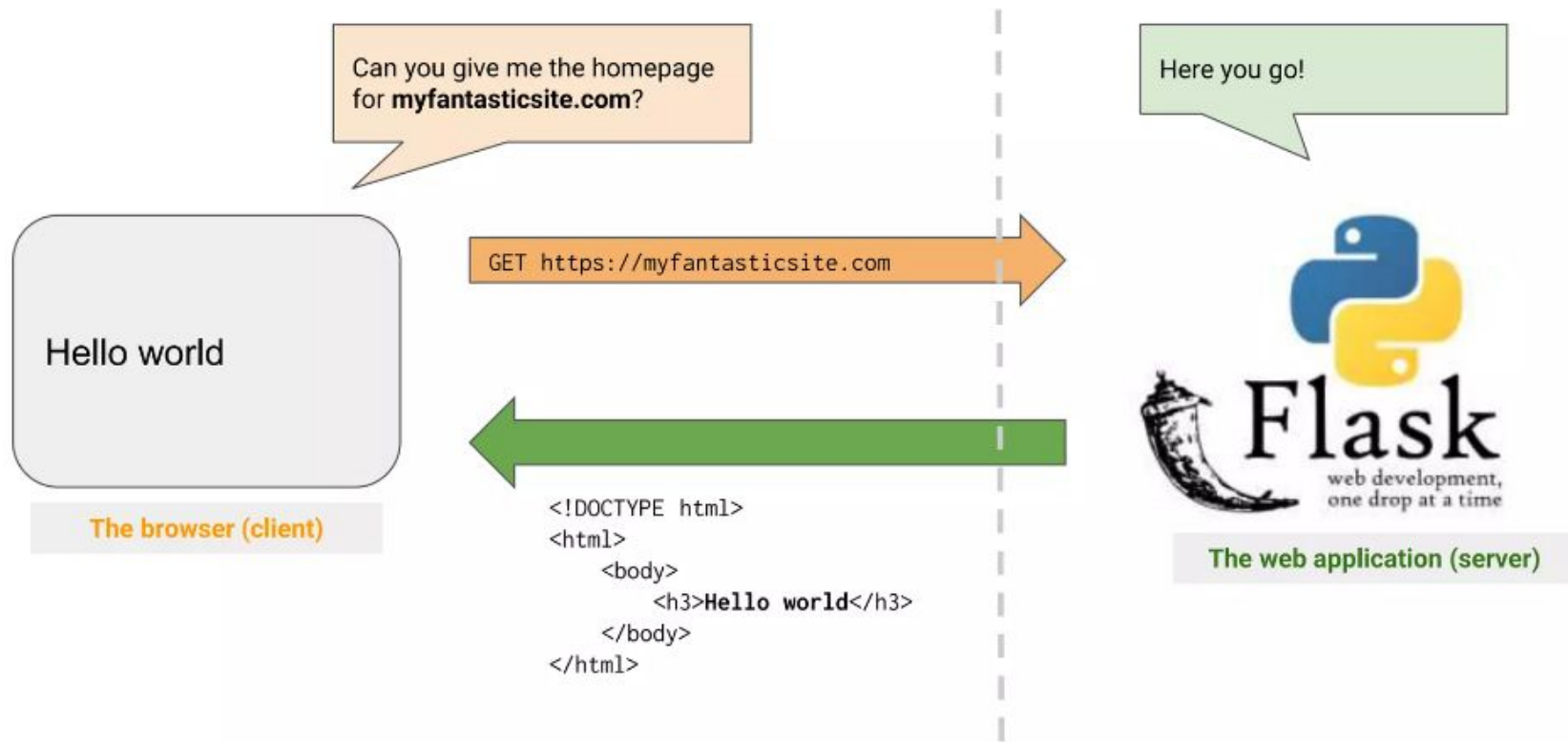


Life without Flask!



Using Flask!

Framework for building web applications in Python



Introduction to Flask

Definition: Flask is a lightweight and flexible web framework for Python.

Characteristics: Known for its simplicity and fine control over its components.

Features of Flask

Minimalist Design: Starts simple but can scale up to complex applications.

Extensible: Use extensions to add features like database integration, form validation, and more.

Built-in Development Tools: Includes a development server and debugger for easy testing.

Flask Components

Routing: Maps URLs to Python functions; easy URL parameter handling.

Templates: Uses Jinja2 for dynamic content generation.

HTTP Methods: Supports methods like GET and POST for data retrieval and submission.

Why Use Flask?

Microframework: Lightweight core, ideal for small to medium applications.

Flexibility: Add only the components you need.

Control: More direct control over your application compared to full-stack frameworks.

Flask vs. Other Frameworks

Compared to Django (a full-stack framework): Flask provides fewer features out-of-the-box but is lighter and offers more flexibility.

Suitability: Best for projects where a lightweight and modular approach is preferred.

Common Uses of Flask

Web Applications: From simple web pages to complex web services.

APIs: Developing backends and microservices.

Prototyping: Quick development cycles make it ideal for prototypes.

Conclusion

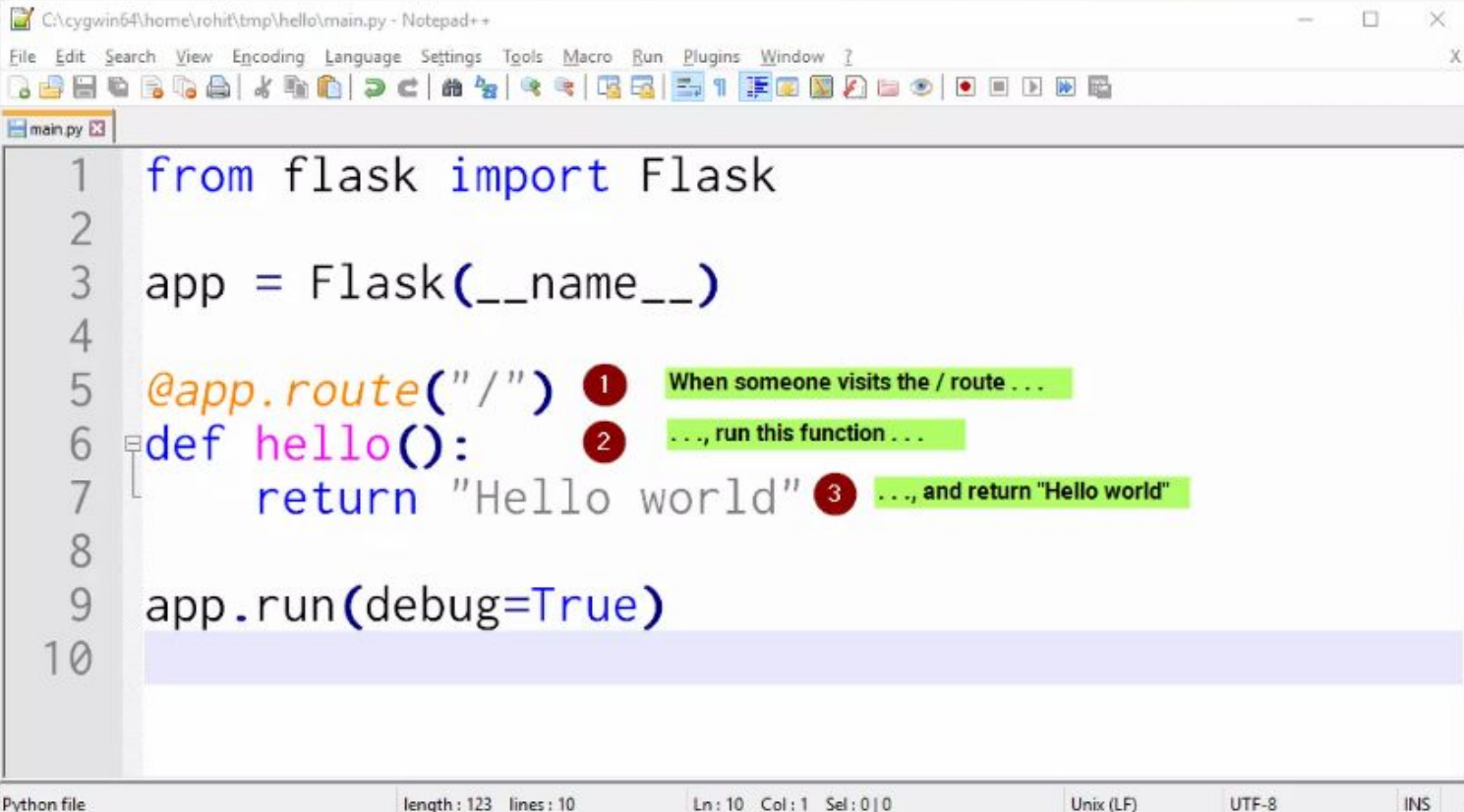
Summary: Flask is a powerful yet simple tool for web development.

Final Thought: Perfect for developers looking for a minimalist, flexible framework that they can extend as needed.

Explanation:

- ``from flask import Flask``: This line imports the Flask class from the Flask library.
- ``app = Flask(__name__)``: This line creates an instance of the Flask class.
- ``@app.route('/')``: This is a decorator that tells Flask what URL should trigger the function that follows.
- ``def hello_world()``: This defines the function that will run when the URL is accessed.
- ``return 'Hello, World!``: This line is what the function returns to the browser when the URL is accessed.
- ``app.run(debug=True)``: This line runs the application on the local development server. The ``debug=True`` argument allows possible Python errors to appear on the web page. This can help you trace the errors.

Defining routes



```
C:\cygwin64\home\rohit\tmp\hello\main.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
main.py
1 from flask import Flask
2
3 app = Flask(__name__)
4
5 @app.route("/") 1 When someone visits the / route ...
6 def hello():      2 ..., run this function ...
7     return "Hello world" 3 ..., and return "Hello world"
8
9 app.run(debug=True)
10
```

Python file length : 123 lines : 10 Ln : 10 Col : 1 Sel : 0 | 0 Unix (LF) UTF-8 INS

Defining routes

- The web server is running at `http://localhost:5000`, but you didn't tell it what to return when someone visits the `/` **route**
- To do that, you have to add a function for the `/` route
- Flask uses Python **decorators** to define routes
- There can be any number of routes in your application