



Flask

web development,
one drop at a time

Kuba Sycz
Paweł Rus

Czym jest Flask?

MICRO-FRAMEWORK PYTHONA UMOŻLIWIJĄCY TWORZENIE APLIKACJI WEBOWYCH, KTÓREGO TWÓRCĄ JEST ARMIN RONACHER. FLASK JEST SKLASYFIKOWANY JAKO MICRO-FRAMEWORK, PONIEWAŻ NIE WYMAGA OKREŚLONYCH NARZĘDZI ANI BIBLIOTEK. NIE MA WARSTWY ABSTRAKCJI BAZY DANYCH, SPRAWDZANIA POPRAWNOŚCI FORMULARZY ANI ŻADNYCH INNYCH KOMPONENTÓW, W KTÓRYCH ISTNIEJĄCE BIBLIOTEKI STRON TRZECICH ZAPEWNIĄ WSPÓLNE FUNKCJE. FLASK STAWIA NA PROSTY „RDZEŃ” APLIKACJI I JEJ ROZSZERZALNOŚĆ. JEST NA LICENCJI BSD ZGODNĄ Z ZASADAMI OPEN SOURCE.



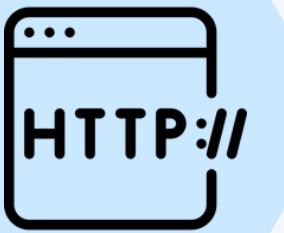
Główne Cechy



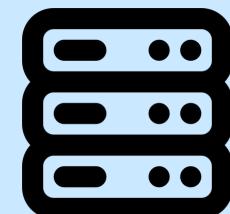
Lekki i
rozszerzalny



Blueprints & Jinja2
Template Engine



Obsługa
requestów



Wbudowany serwer
i debugger



Rozległa
Społeczność i Dobra
Dokumentacja



Komponenty Flask



Werkzeug



IT'S DANGEROUS
... so better sign this



Jinja2 - Silnik Szablonów

- PEŁNE WSPARCIE DLA UNICODE.
- AUTOMATYCZNE PRZECIWZIAŁANIE XSS (CROSS-SITE SCRIPTING) W ZNACZNIKACH MODYFIKACJI HTML.
- MOŻLIWOŚĆ WYKONYWANIA KODU W PIASKOWNICY.
- MOŻLIWOŚĆ TWORZENIA WŁASNYCH ZNACZNIKÓW, FILTRÓW, TESTÓW I ZMIENNYCH GLOBALNYCH.
- DZIEDZICZENIE SZABLOŃSKIE.
- ŁATWE DO DEBUGOWANIA.



```
run.py > ...
1  from flask import Flask, render_template
2  app = Flask(__name__)
3
4  @app.route('/')
5  def main():
6      return render_template('index.html', framework='Flask')
7
```

{{framework}}

A screenshot of a web browser window displaying the Flask application's homepage. The URL bar shows `127.0.0.1:5000`. The page content includes:

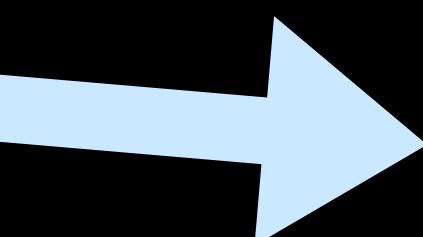
- # Witaj na stronie o Flask!
- Flask to mikroframework do języka Python, który umożliwia szybkie tworzenie aplikacji internetowych. Dzięki swojej prostocie i elastyczności, Flask stał się popularnym narzędziem wśród programistów.
- Możesz rozpocząć naukę Flaska, czytając oficjalną dokumentację na [stronie projektu](#). W niej znajdziesz wiele przykładów, poradników i informacji na temat budowania aplikacji webowych.
- Pamiętaj, że Flask jest świetnym narzędziem do nauki i tworzenia aplikacji webowych, więc nie wahaj się zacząć swojej przygody z nim już teraz!

```
templates > index.html > html
1  <!DOCTYPE html>
2  <html lang="en">
3
4  > <head>...
35 </head>
36
37 <body>
38 > <div class="container">...
44 </div>
45 </body>
46
47 </html>
```

```
run.py > page
1  from flask import Flask, render_template
2  app = Flask(__name__)
3
4  @app.route('/')
5  def main():
6      return 'Hello World!'
7
8  @app.route('/secondpage')
9  def secondpage():
10     return 'Second page'
11
12 @app.route('/page/<name>')
13 def page(name):
14     return 'Page for ' + name
15
16 @app.route('/sum/<number1>+<number2>')
17 def sum(number1, number2):
18     sum = int(number1) + int(number2)
19     return f'Sum of numbers: {sum}'
20
21
22 if __name__ == '__main__':
23     app.run(debug=True)
```

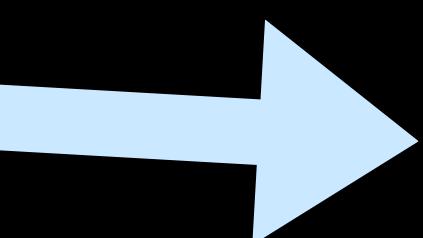
← → C ① 127.0.0.1:5000

Hello World!



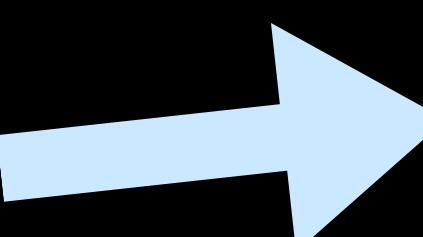
← → C ① 127.0.0.1:5000/secondpage

Second page



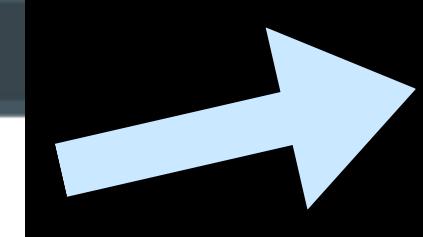
← → C ① 127.0.0.1:5000/page/agh

Page for agh



← → C ① 127.0.0.1:5000/sum/100+1839

Sum of numbers: 1939



Blueprints

```
app > views > ✎ index.py
  1  from flask import Blueprint, request, current_app, g, render_template,
  2  from flask_security import current_user, login_required
  3
  4  bp = Blueprint('bp_index', __name__)
  5
  6  @bp.route('/')
  7  def index_get():
  8      return render_template('index.html')

app > views > ✎ guestbook.py
  1  from flask import Blueprint, request, current_app, g, render_template
  2  from flask_security import current_user, login_required
  3  from sqlalchemy.exc import OperationalError
  4
  5  from flask_wtf import FlaskForm
  6  from wtforms import TextAreaField
  7  from wtforms.validators import DataRequired, Length
  8
  9  from ..app import db
 10 from ..models import GuestBook
 11
 12 bp = Blueprint('bp_guestbook', __name__)
 13
 14
 15 @bp.route('/guestbook', methods=['GET'])
 16 @bp.route('/guestbook/page/<int:page>', methods=['GET'])
 17 > def guestbook_get(page=1): ...
 18
 19
 20
 21
 22
 23
 24
 25
 26 @bp.route('/guestbook/add', methods=['GET', 'POST'])
 27 @login_required
 28 > def guestbook_add_get_post(): ...
 29
 30
 31
 32
 33
 34
 35
 36
 37
 38
 39
 40
 41
 42
 43
 44
 45
 46
 47
 48
 49
 50
 51
 52 @bp.route('/guestbook/delete/<int:entry_id>', methods=['GET'])
 53 @login_required
 54 > def guestbook_delete_get(entry_id=0): ...
```

```
app > views > ✎ index.py
  1  from flask import Blueprint, request, current_app, g, render_template,
  2  from flask_security import current_user, login_required
  3
  4  bp = Blueprint('bp_index', __name__)
  5
  6  @bp.route('/')
  7  def index_get():
  8      return render_template('index.html')
```

```
✓ app
  > templates
  > views
    ✎ __init__.py
    ✎ blog.py
    ✎ guestbook.py
    ✎ index.py
    ✎ __init__.py
    ✎ .gitignore
    ✎ app.py
    ✎ config.py
    ✎ models.py
```

```
from .views.index import bp as bp_index
app.register_blueprint(bp_index)
```

```
from .views.guestbook import bp as bp_guestbook
app.register_blueprint(bp_guestbook)
```

```
from .views.blog import bp as bp_blog
app.register_blueprint(bp_blog)
```

Rozszerzenia do Flaska

- **Flask SQLAlchemy** - obsługa ORM
- **Flask Babel** - wsparcie i18n i i10n
(wprowadzenie obsługi różnych języków)
- **Flask Security**
- **Flask Login** - zarządzanie sesją uwierzytelnianie użytkowników
- **Flask Admin** - interfejs admina
- **Flask-Session** - zarządzanie sesją
- **FLASK-WTF** - formsy

SQLA



Flask:Babel



Flask-Security



Flask Session



Flask WTF

Kto używa Flaska?

NETFLIX



 **zalando**



SAMSUNG

Uber



LinkedIn

Zalety

- Prostota i elastyczność
- Rozszerzenia
- Społeczność
- Szybkość rozwoju
- Obsługa żądań zgodnie z zasadami REST
- Dopasowywalny do nowych technologii
- Umożliwia eksperymentowanie z architekturą i biblioteką
- Świety dla mniejszych projektów
- Szybkie i łatwe w budowie prototypy
- Łatwe routowanie URLi za pomocą Werkzeug
- Łatwa integracja baz danych
- NoSQL Support

Wady

- Performance Overhead
- Brak konwencji
- Zależność od rozszerzeń
- Utrzymanie może być trudne
- No built-in admin site
- Brak funkcji zabezpieczeń i autoryzacji
- Brak mapowania obiektowo-relacyjnego (ORM)

Bezpieczeństwo Flaska

„Web applications usually face all kinds of security problems and it's very hard to get everything right. Flask tries to solve a few of these things for you, but there are a couple more you have to take care of yourself.”

CSRF Protection

```
from flask_wtf.csrf import CSRFProtect

csrf = CSRFProtect(app)

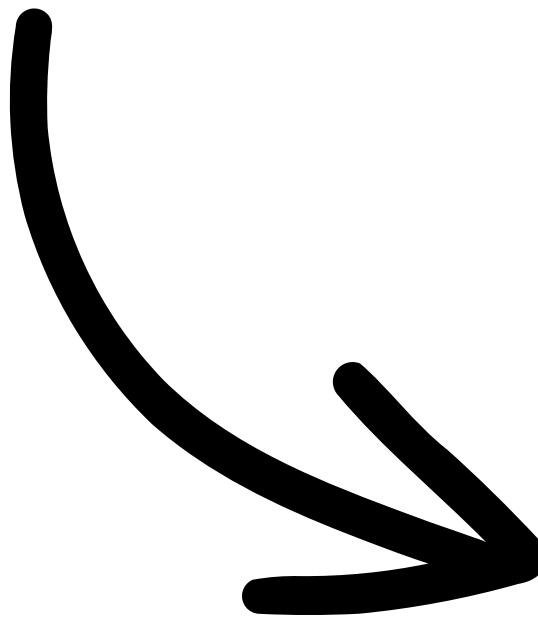
def create_app():
    app = Flask(__name__)
    csrf.init_app(app)
```

```
<form method="post">
    {{ form.csrf_token }}
</form>

<form method="post">
    <input type="hidden" name="csrf_token" value="{{ csrf_token() }}"/>
</form>
```

Cross-site Scripting (XSS)

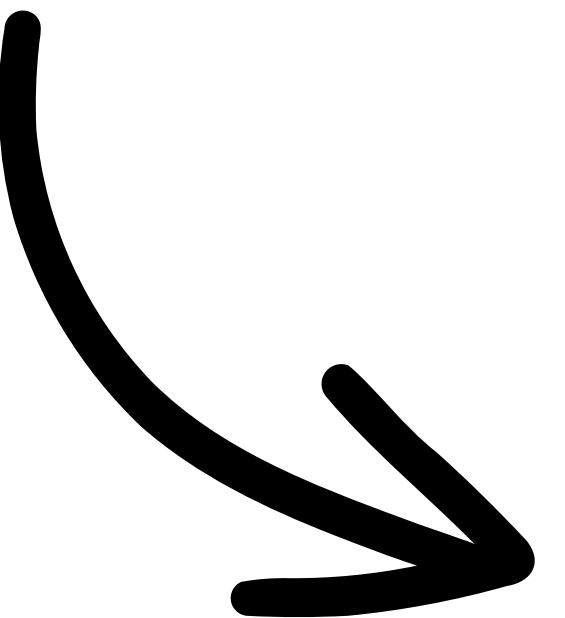
```
1. from flask import Flask, request, render_template_string
2.
3. app = Flask(__name__)
4.
5. @app.route('/post', methods=['POST'])
6. def post():
7.     content = request.form['content']
8.     return render_template_string('<p>{{ content }}</p>', content=content)
9.
10. if __name__ == '__main__':
11.     app.run()
```



```
1. from flask import Flask, request, render_template_string, escape
2.
3. app = Flask(__name__)
4.
5. @app.route('/post', methods=['POST'])
6. def post():
7.     content = request.form['content']
8.     return render_template_string('<p>{{ content }}</p>',
9.                                 content=escape(content)))
10. if __name__ == '__main__':
11.     app.run()
```

Session Hijacking

```
1. from flask import Flask, session, request
2.
3. app = Flask(__name__)
4. app.secret_key = 'your_secret_key'
5.
6. @app.route('/')
7. def index():
8.     session['username'] = 'admin'
9.     return 'Hello, {}'.format(session['username'])
10.
11. @app.route('/profile')
12. def profile():
13.     return 'Welcome to your profile, {}'.format(session['username'])
14.
15. if __name__ == '__main__':
16.     app.run()
```



```
1. from flask import Flask, session, request
2.
3. app = Flask(__name__)
4. app.secret_key = 'your_secret_key'
5. app.config['SESSION_COOKIE_SECURE'] = True
6. app.config['SESSION_COOKIE_HTTPONLY'] = True
7.
8. @app.route('/')
9. def index():
10.     session['username'] = 'admin'
11.     return 'Hello, {}'.format(session['username'])
12.
13. @app.route('/profile')
14. def profile():
15.     return 'Welcome to your profile, {}'.format(session['username'])
16.
17. if __name__ == '__main__':
18.     app.run()
```

Brute Force Attack

```
1. from flask import Flask, request, session
2.
3. app = Flask(__name__)
4. app.secret_key = 'your_secret_key'
5.
6. @app.route('/login', methods=['POST'])
7. def login():
8.     username = request.form['username']
9.     password = request.form['password']
10.
11.    if username == 'admin' and password == 'password':
12.        session['username'] = 'admin'
13.        return 'Login successful'
14.    else:
15.        return 'Invalid credentials'
16.
17. if __name__ == '__main__':
18.     app.run()
```



```
1. from flask import Flask, request, session
2. from flask_limiter import Limiter
3. from flask_limiter.util import get_remote_address
4.
5. app = Flask(__name__)
6. app.secret_key = 'your_secret_key'
7. limiter = Limiter(app, key_func=get_remote_address)
8.
9. @app.route('/login', methods=['POST'])
10. @limiter.limit("5/minute")
11. def login():
12.     username = request.form['username']
13.     password = request.form['password']
14.
15.     if username == 'admin' and password == 'password':
16.         session['username'] = 'admin'
17.         return 'Login successful'
18.     else:
19.         return 'Invalid credentials'
20.
21. if __name__ == '__main__':
22.     app.run()
```



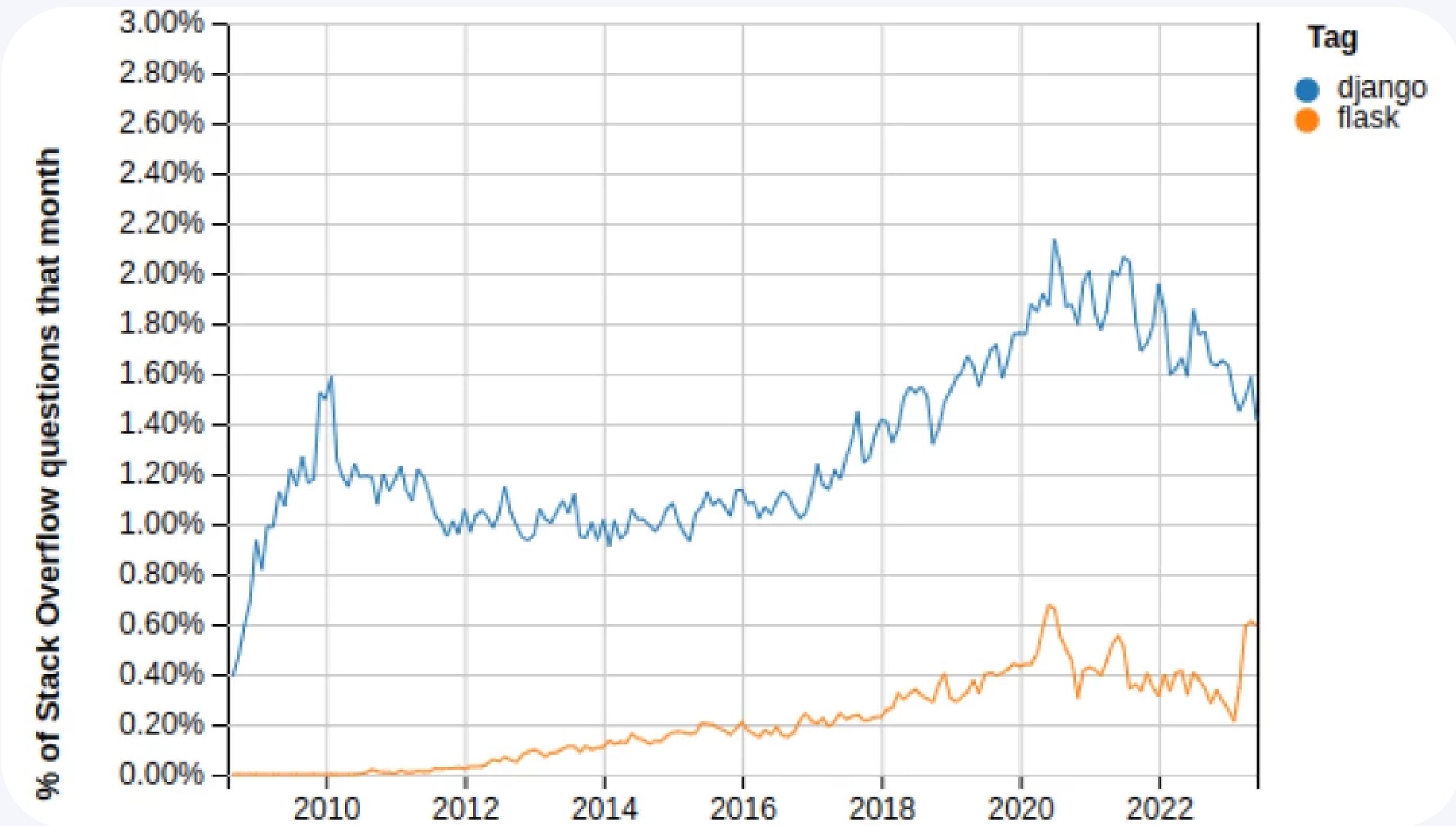
Flask

web development,
one drop at a time

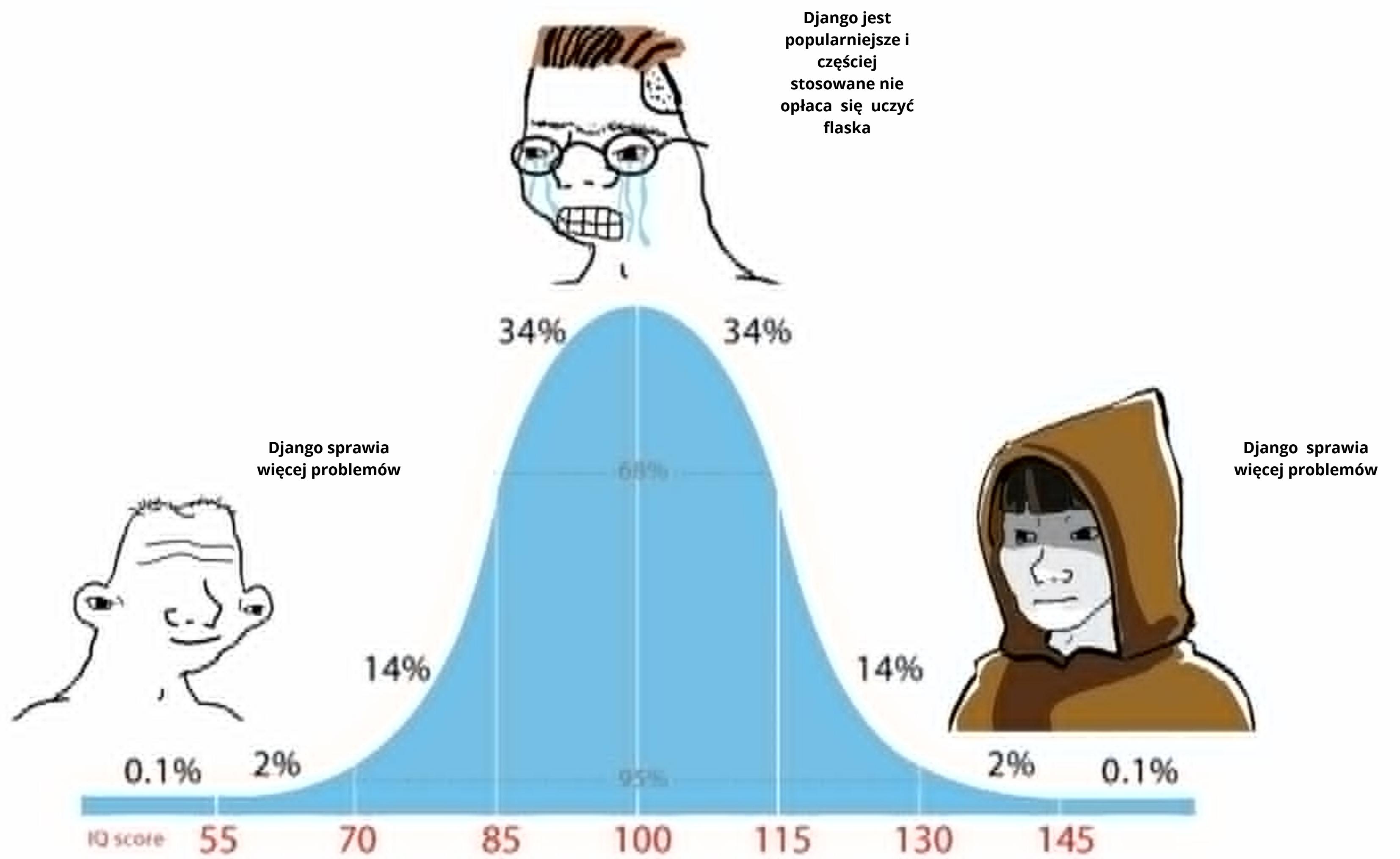
vs

django

Flask and Django tags on StackOverflow



source: <https://hackr.io/blog/flask-vs-django>





Microframework

Łatwiejszy - minimalizm

**Rozszerzenia typu
SQLAlchemy**

Większa

Dostępne rozszerzenia

Typ

Nauka

ORM

Elastyczność

Bezpieczeństwo

„Pełnoprawny” framework

Trudniejszy

Wbudowane ORM

Mniejsza

Wbudowane rozwiązania

A man with a shaved head and a goatee, wearing dark sunglasses and a black t-shirt, stands in the foreground. He is pointing his right index finger towards the camera. In the background, a large white airplane is parked on a tarmac under a clear blue sky.

Any questions?