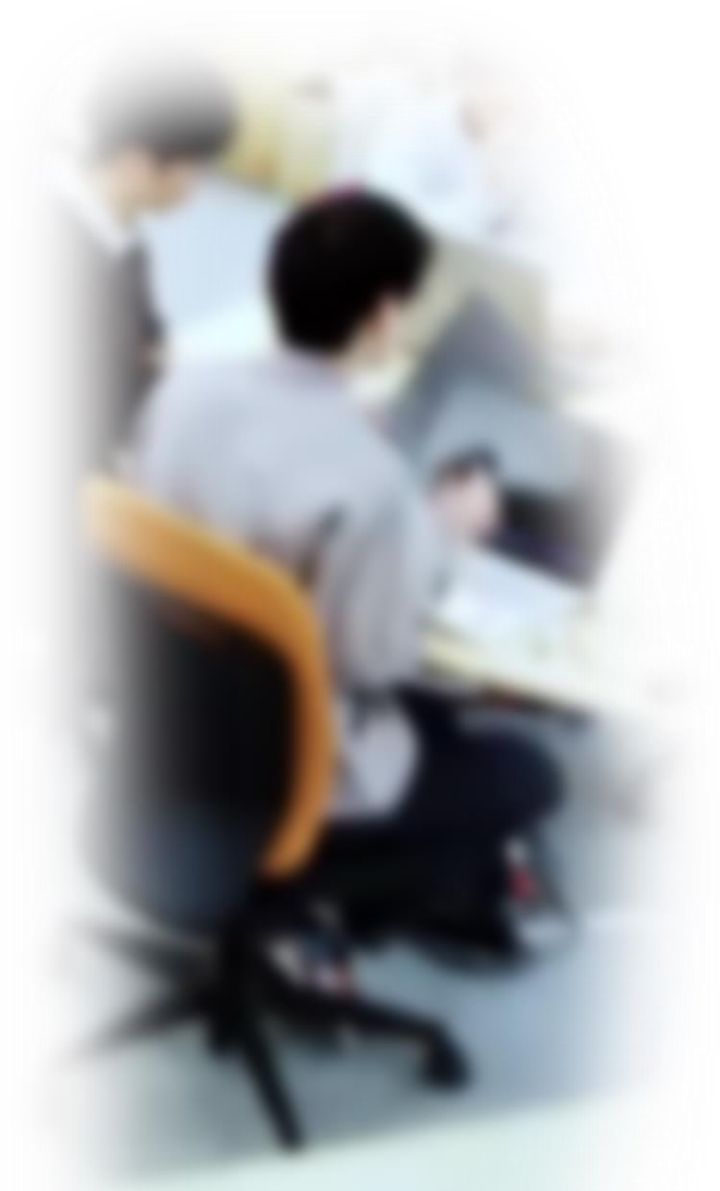
Web アプリケーション

URL: https://www.kkaneko.jp/cc/dbenshu/index.html

金子邦彦

CC BY NC SA



アウトライン

1. Webサービス
2. ルーティング，Flask
3. Dash

Web サービス

Webサービス

- **情報へのアクセス**

ニュース，天気予報，WikiPedia，地図

- **コミュニケーション**

ソーシャルネットワーク

- **視聴サービス**

映画，音楽など．ゲーム配信を受けることも.

- **オンラインサービス**

ショッピング，銀行

- **学習**

授業配信，プログラミング学習サイト，プログラミング開発（Repl.it, Google Colaboratory など）

Webサービスの特質

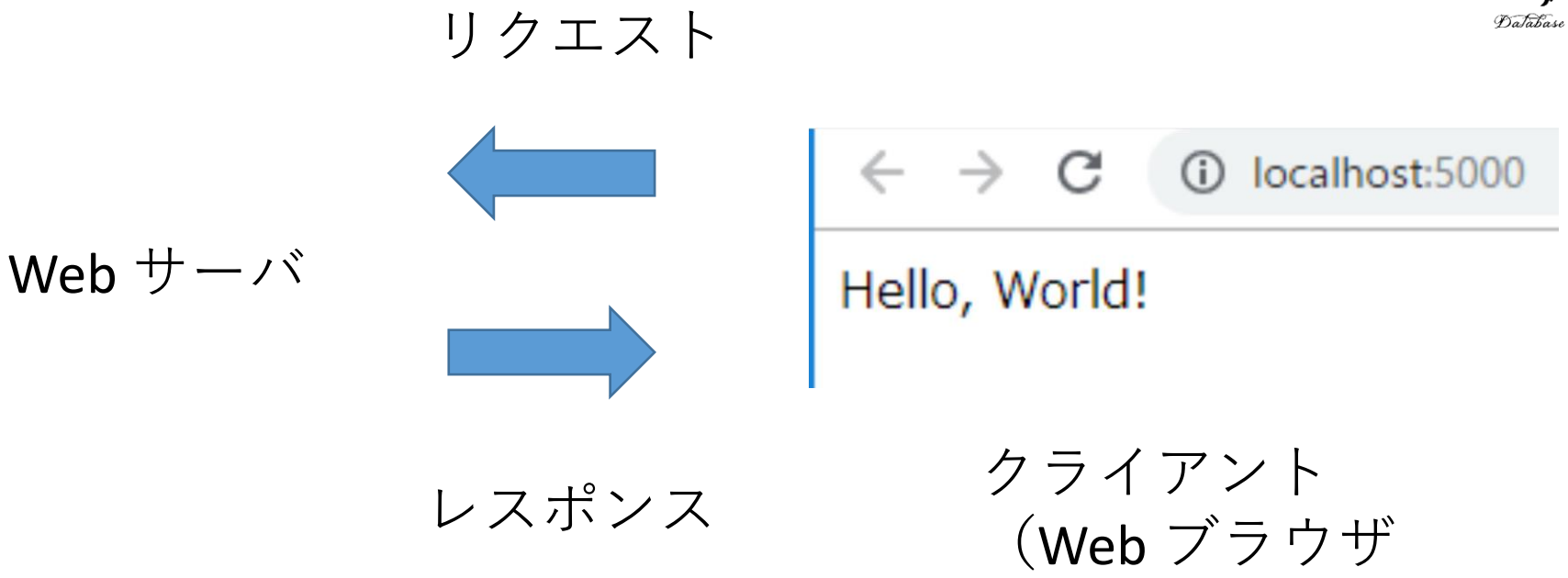
- **インターネットサービスの一種**
- **HTTP プロトコル**
- **Webブラウザ**を通じて，人間にサービスを提供するだけでなく，**コンピュータ間のコミュニケーションにも**使われることも多い

HTTPプロトコルの要点

- **クライアント／サーバモデル**
- 複数のメソッド：GET, POST, PUT, DELETE
- 各リクエストは独立

以前のリクエストに依存しない．**単独のリクエストの中で，リクエストの処理に必要なすべての情報が含まれている**こと

Webサービスのサーバとクライアント



- **Webサーバ**と**クライアント**との間でデータのやり取りが行われる
- リクエスト：クライアントは，Webブラウザを介して，データやサービスを Web サーバに要求
- レスポンス：Webサーバがリクエストを受けて，クライアントに返答

ルーティング，Flask

Webサーバのルーティング

- **ルーティング**は，**リクエストURL**を振り分ける仕組み

http://www.example.jp/ → **ようこそメッセージ**を表示

http://www.example.jp/about →**紹介メッセージ**を表示

http://www.example.jp/products →**製品メッセージ**を表示

http://www.example.jp/products/<product_id> → <product_id>は製品IDであるとする．**個別製品メッセージ**を表示

Flask

- **Flask** は，Pythonのための Web アプリケーションフレームワーク。
- **Web アプリケーションの設計と構築を簡単に行う**ための機能を提供
  - **Webサーバ**
  - **ルーティング**
  - テンプレートエンジン
  - エラーハンドリング
  - デバッグ

Flask におけるルーティング

- **リクエストURL**を，特定の**Python関数**（ビューと呼ぶ）に**関連付け**
- **@app.route()デコレータ**を使用して**リクエストURL**と**関数**を**関連付ける**

ルートURL（‘/’）を関数home()に関連付けするPython プログラム

```
from flask import Flask
app = Flask(__name__)
@app.route('/')
def home():
    return "This is the homepage."
```

Flask における動的ルーティング

動的ルーティングでは，**リクエストURLの一部**を**変数**として扱う。その**変数**を**ビュー関数の引数**として利用

ユーザー名をパラメータとして取る

```
from flask import Flask
app = Flask(__name__)
@app.route('/user/<username>')
def user(username):
    return 'User %s' % username
```

<username> はリクエストURLの一部

ルーティングを行う Python プログラム（Flask を使用）

```
from flask import Flask
app = Flask(__name__)

@app.route('/')
def home():
    return "This is the homepage."

@app.route('/about')
def about():
    return "This is the about page."

@app.route('/products')
def products():
    return "This is the products page."

@app.route('/products/<product_id>')
def product_detail(product_id):
    return "This is the page for product with id: " + product_id
```

ここまでのまとめ

- **Webサービス**は**多様なサービス**を提供：

情報へのアクセス，コミュニケーション，視聴サービス，オンラインサービス，学習

- **HTTPプロトコル**は，**Webブラウザ**だけでなく，人間だけでなく、**コンピュータ間のコミュニケーション**にも使用
- **Webサービス**での**クライアント／サーバモデル**は，リクエストの送信とレスポンスの受け取り
- **Flask**は，PythonのWebアプリケーションフレームワーク：

ルーティング機能を持つ．**特定のURLをPython関数に関連付け**る．動的なURL（**URLの一部を変数として扱う**）も可

Dash

Dash の概要

- **Dash** は，PythonのWebアプリケーションフレームワーク
- **データ可視化**，**ユーザーインターフェース**の作成に特化
- Plotly社により開発
- 描画エンジンは，Plotly.js (JavaScript) を基にしている
- ユーザインタフェースは React.js (JavaScript) を基にしている
- HTML, JavaScript, CSS を**使わなくても**アプリケーションを作成可能

Dash の主な機能

- **データ可視化**：
  - データ分析結果のリアルタイム可視化、共有に適する
- **ユーザーインターフェース**：
  - スライダーやドロップダウンなどのウィジェット
  - ユーザの入力に応じて表示が動的に更新可能
- **Webサーバの機能を含む**
- **複数のユーザ**が同時にアクセス可能：
  - サーバーサイドでのセッション管理
- **利用コスト**：
  - オープンソース版は無料で利用可能
  - 企業向けの有料版も提供されている

棒グラフのプログラム（Dash を使用）

```
import dash
import dash_core_components as dcc
import dash_html_components as html
app = dash.Dash(__name__)
app.layout = html.Div(children=[
    html.H1(children='Hello Dash'),
    html.Div(children='''
        Dash: A web application framework for Python.
    '''),
    dcc.Graph(
        id='example-graph',
        figure={
            'data': [
                {'x': [1, 2, 3], 'y': [4, 1, 2], 'type': 'bar', 'name': 'SF'},
                {'x': [1, 2, 3], 'y': [2, 4, 5], 'type': 'bar', 'name': 'Montreal'},
            ],
            'layout': {
                'title': 'Dash Data Visualization'
            }
        }
    )
])
if __name__ == '__main__':
    app.run_server(debug=True)
```

Dash のメリット

- **Python** による開発ができる
- **インタラクティブ**なグラフを作ることができる
- スライダーやドロップダウンなどの**ユーザインタフェース**を作成できる
- できたアプリケーションは，**Webアプリケーション**であり，簡単に**公開できる**

まとめ

- **Dash** は PythonのWebアプリケーションフレームワーク
- **データ可視化**と**ユーザーインターフェース**の作成に特化
- **リアルタイム可視化**や，**表示の動的な更新**が可能
- **複数のユーザ**が同時にアクセス可能
- オープンソースで無料で利用可能
- JavaScriptやCSSの知識が**不要**で**簡単に Web アプリケーションを作成**できる

関連するページ

- **Python のインストール**

次のページで，「Python 3.10 のインストール，pip と setuptools の更新（Windows 上）」を行う

https://www.kkaneko.jp/tools/win/python.html

- **Flask のインストール**

https://www.kkaneko.jp/pro/webui/flask.html

- **Dash のインストールと基本**

https://www.kkaneko.jp/pro/webui/dash.html

- **Webアプリケーション，表や散布図の Web での表示，Dash を使用.**

https://www.kkaneko.jp/pro/webui/dashtable.html