03

CHAPTER

Django プロジェクトを始める

「バッテリー同梱」フレームワークで Hello, World! まで

本書のサンプル章 — 全文無料公開

AGENDA

この章で学ぶこと

所要時間： 約 1.5 時間

01
Django とは何か、なぜ選ばれるのか
Web フレームワークと「バッテリー同梱」

02
Django プロジェクトの作成
uv で初期化 → startproject まで

03
Django の構造理解（ MTV ）
Model — Template — View

04
Hello, World! を表示する
テンプレートを使った最初のページ

05
管理画面を見てみる
createsuperuser → /admin/

06
つまずきポイントと Git コミット
よくある 6 つのエラーと対処法

PART 1

Django とは

Python で最も人気の Web フレームワーク

WHY DJANGO

Django が選ばれる 5 つの理由

WEB FRAMEWORK

Web アプリの「よくある処理」 — URL ルーティング・ DB 操作・ HTML 生成・入力処理 — を
まとめてくれるライブラリ。毎回ゼロから作る必要がなくなる。

01

バッテリー同梱

必要な機能が最初から揃う

02

管理画面が自動生成

CRUD をブラウザで操作

03

セキュア既定

SQLi / XSS / CSRF 対策込み

04

強力な ORM

SQL を書かず Python で DB 操作

05

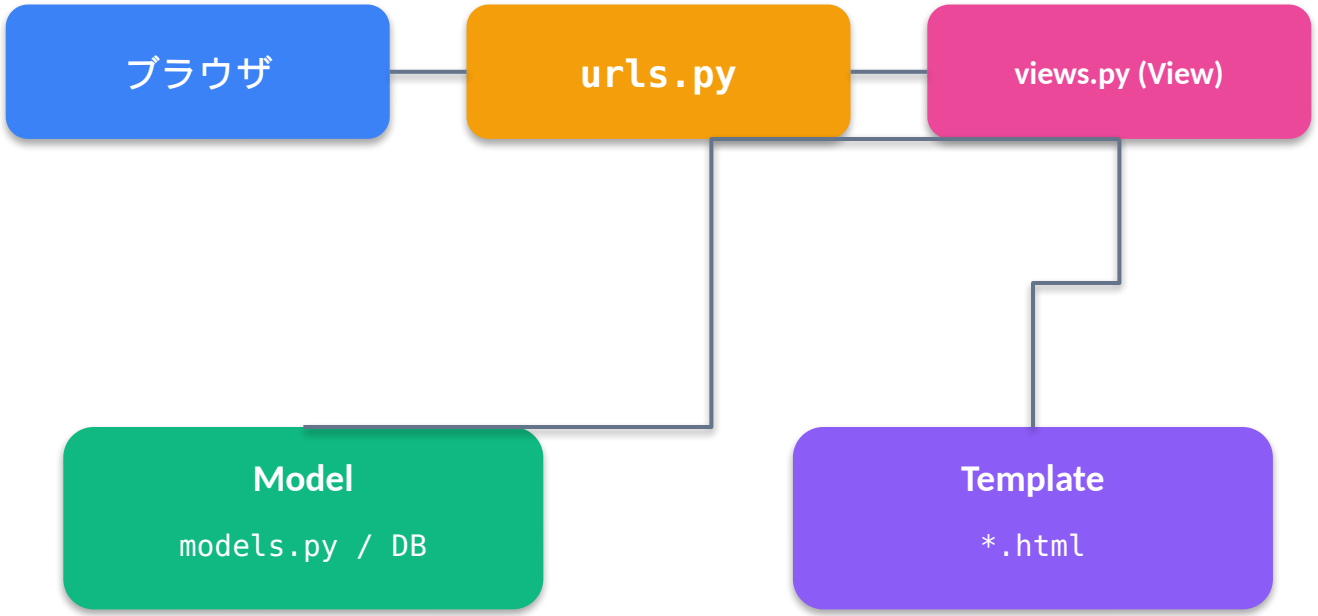
大規模実績

Instagram ・ Pinterest ・ NASA

ARCHITECTURE

MTV — Django の設計パターン

Model — Template — View（他の FW で言う MVC に相当）



それぞれの役割

Model

データの構造と保存

Template

見た目（HTML）

View

ロジック・橋渡し

他 FW (Rails 等) の MVC に相当 — Template = MVC の View, View = MVC の Controller

PART 2

プロジェクト作成

uv init から runserver まで一気に

PROJECT SETUP

プロジェクト作成の 4 ステップ

01 INIT

プロジェクトを初期化

```
$ uv init --python 3.13
```

pyproject.toml が生成される

02 ADD

Django をインストール

```
$ uv add django==5.2
```

.venv 作成 + uv.lock 生成

03 STARTPROJECT

プロジェクト作成

```
$ uv run django-admin startproject server .
```

manage.py と server/ が生成される。最後の . を忘れずに

04 RUN

開発サーバーを起動

```
$ uv run python manage.py runserver
```

→ http://127.0.0.1:8000/ で動作確認

STRUCTURE

生成されたファイル構造

```
adventure-game/
├── .venv/                  #  仮想環境
├── server/                 #  プロジェクト設定
│   ├── __init__.py
│   ├── settings.py         #  設定（重要）
│   ├── urls.py             #  URL ルーティング
│   ├── asgi.py
│   └── wsgi.py
├── manage.py               #  管理コマンド（重要）
├── pyproject.toml
├── uv.lock
└── README.md
```

覚えるべき 4 つのファイル

manage.py — Django コマンドの入口

runserver, migrate, startapp…

settings.py — プロジェクトの設定

DB / 言語 / アプリ登録

urls.py — URL とビューの対応付け

ルーティング

wsgi.py — 本番サーバー用エントリ

デプロイ時に使う（後章）

settings.py

プロジェクト設定の主要 6 項目

SECRET_KEY

セキュリティの要

本番では環境変数で管理

DEBUG

デバッグモード

True= 開発 / False= 本番

ALLOWED_HOSTS

受け付けるホスト

本番では必須設定

INSTALLED_APPS

使うアプリのリスト

自作アプリもここに登録

DATABASES

DB 接続情報

デフォルトは SQLite

LANGUAGE_CODE TIME_ZONE

言語・タイムゾーン

'ja' / 'Asia/Tokyo' に変更

MIGRATION

データベースを整える

DATABASE 101

DB は Excel に似ている

Excel のシート	→	テーブル
Excel の行	→	レコード
Excel の列	→	カラム

マイグレーション = DB 構造を作成・変更する仕組み

新しい Excel シートや列を追加する作業を、 Django が自動で管理してくれる。

実行コマンド

```
$ uv run python manage.py migrate
```

実行結果 (一部)

```
Operations to perform:
  Apply all migrations: admin, auth,
    contenttypes, sessions
Running migrations:
  Applying contenttypes.0001... OK
  Applying auth.0001...          OK
  Applying admin.0001...         OK
  Applying sessions.0001...      OK
```

→ *db.sqlite3 が生成される*

PART 3

Hello, World!

テンプレートを使って初めての自作ページ

7 STEPS

Hello, World! を表示するまで

01	アプリ作成	startapp game
02	アプリ登録	INSTALLED_APPS に 'game'
03	**templates** ディレクトリ作成	game/templates/game/
04	**home.html** を作成	テンプレート + {{ current_time }}
05	**views.py** を編集	render() で context を渡す
06	**urls.py** を設定	game 側 + server 側 (include)
07	ブラウザで確認	http://localhost:8000/

THE CODE

Hello, World! の 3 つのファイル

views.py

Python ロジック

```
from django.shortcuts import render
from datetime import datetime

def home(request):
    context = {
        'current_time':
        datetime.now()
            .strftime('%Y...')
    }
    return render(
        request,
        'game/home.html',
        context,
    )
```

home.html

テンプレート

```
<!DOCTYPE html>
<html lang="ja">
<head>
  <meta charset="UTF-8">
  <title>冒険ゲーム</title>
</head>
<body>
  <h1>廃墟研究所</h1>
  <p>現在の時刻：
    {{ current_time }}
  </p>
</body>
</html>
```

urls.py

URL ルーティング

```
# game/urls.py
from django.urls import path
from . import views

urlpatterns = [
    path('', views.home,
         name='home'),
]

# server/urls.py
urlpatterns = [
    path('admin/',
         admin.site.urls),
    path('',
         include(
           'game.urls')),
]
```

RESULT

ブラウザで動作を確認

アクセス URL

```
http://localhost:8000/
```

確認ポイント

- ✓ タイトル　「廃墟研究所からの脱出」
- ✓ サブタイトル　Django + LLM で作るアドベンチャー
- ✓ 動的データ　現在の時刻が表示される
- ✓ リロード　時刻が毎回更新される

PART 4

管理画面・つまずき・ Git

仕上げと、よくあるエラーの備え

ADMIN PANEL

管理画面 — Django の最強機能のひとつ

セットアップ

2 コマンドで完了

STEP 1 — スーパーユーザー作成

```
$ uv run python manage.py createsuperuser
```

→ *対話形式で username / email / password を入力*

STEP 2 — サーバー起動 & アクセス

```
$ uv run python manage.py runserver
```

```
http://localhost:8000/admin/
```

ログインすればダッシュボード表示。Users / Groups から始まる。

管理画面でできること

C — Create
データの追加（フォーム自動生成）

R — Read
一覧表示・検索・フィルタリング

U — Update
データ編集（バリデーション付き）

D — Delete
データ削除（確認ダイアログ付き）

TROUBLESHOOTING

つまずきポイント TOP 6

01 python が無い

```
command not found: python
```

→ uv run python を使う

02 django-admin が無い

```
django-admin: command not found
```

→ uv run django-admin / uv add django

03 ポートが既に使用中

```
Error: That port is already in use.
```

→ runserver 8001 で別ポート

04 ModuleNotFoundError

```
No module named 'game'
```

→ INSTALLED_APPS に追加 / 大文字確認

05 ImproperlyConfigured

```
SECRET_KEY setting must not be empty
```

→ get_random_secret_key() で再生成

06 404 Not Found

```
Page not found (404)
```

→ URL パターンを確認 / DEBUG ページを読む

READY?

Hello, World! 達成 + Git に保存。

.gitignore — 除外しないと危険

`__pycache__/`	—	Python の自動生成
`*.pyc`	—	Python コンパイル済み
`.venv/`	—	仮想環境（環境依存）
`db.sqlite3`	—	個人データを含む
`.env`	—	API キーなど機密情報

→ uv.lock は コミットする（依存の再現用）

コミットの流れ

```
$ git add .
$ git status                    # 確認
$ git commit -m \
    "feat: Django プロジェクト \
    とアプリの作成 "
```

Conventional Commits — feat: / fix: / docs: で始める

NEXT

第 4 章 — ゲームの企画とデータ設計

LLM とブレストして企画を固め、データモデルを設計する。

ここまで来れば、もう Web アプリ開発者。次へ進もう。