Engineer

Django

D ウェブアプリを作るためのPythonフレームワーク

どのようなものか

Djangoは、Pythonで書かれたウェブフレームワークです。リクエストが届くと、URLの設定が、ビューと呼ぶ関数やクラスへ振り分けます。ビューは、通常、テンプレートから生成したHTMLか、JSONを返します。その中心の周りに、データベースを使うサイトの多くが必要とする機能を備えます。オブジェクト関係マッパー、スキーマのマイグレーション、フォーム、認証と権限管理、自動生成する管理画面などです。Djangoの文書でも、管理画面は特に人気のある機能の一つとされています。プロジェクト名は大文字のDで始めますが、インストールし、インポートするパッケージは、小文字の `django` です。

出発点は、新聞社の編集部でした。2003年の秋、カンザス州ローレンスの新聞社のウェブ部門、World Onlineの開発者、Adrian HolovatyとSimon Willisonが、報道の締め切りに合わせて作るサイトで、PHPからPythonへ移りました。Lawrence.comなどを作る中で、汎用フレームワークを取り出し、二年間、改良しました。2005年の夏、World Onlineが、オープンソースとして公開しました。名前は、ジャズ・マヌーシュのギタリスト、Django Reinhardtに由来します。Dは発音せず、「ジャンゴ」（JANG-oh）と読み、3条項BSDライセンスで配布しています。

A.Bという番号の方式では、機能リリースは約八か月ごとで、長期サポートに指定したリリースだけが、三年間のサポートを受けました。5.2 LTSは、2028年4月までサポートされます。2027年4月に予定されるLTSリリース、Django 6.2が、この方針での最後です。Django 2028以降は、機能リリースに年の名前を付け、毎年一月に公開し、どのリリースも、同じ三年間のサポートを受けます。

プロジェクトのFAQは、model、template、viewを表す、MTVフレームワークと呼ぶことを勧めています。モデルはデータを定義し、ビューは利用者に見せるデータを決め、テンプレートは見た目を決めます。この解釈では、各リクエストを正しいビューへ送る仕組み、つまりフレームワーク自体が、コントローラーで

す。モデルはPythonのクラスで、クエリーはQuerySetとして表します。QuerySetは遅延評価されるので、構築し、絞り込み、受け渡しても、データベースには触れません。ループなどが評価するときに、初めて問い合わせます。モデルの変更は、マイグレーションになります。`makemigrations` は、各変更をファイルへまとめます。文書は、それを、スキーマのバージョン管理システムにおけるコミットに例えます。`migrate` が、その変更を適用します。関連オブジェクトは、QuerySetで先に取得するよう指定しない限り、必要になったときに、その都度クエリーを実行して読み込みます。

```
# BookはAuthorへのForeignKeyを持つ
# 本の取得に1回、各本の著者の取得にそれぞれ1回のクエリー
for book in Book.objects.all():
    print(book.author.name)

# 1回のクエリー：SQL結合で本と著者を一緒に取得する
for book in Book.objects.select_related("author"):
    print(book.author.name)

# 2回のクエリー：著者を取得し、次にその全作品を一括取得する
for author in Author.objects.prefetch_related("book_set"):
    print(author.name, [b.title for b in author.book_set.all()])
```

概念を判断に生かす

「Djangoとは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 データの保存先、処理のルール、他のシステムとの接続方法など、利用者の目に見えない部分をつくります。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 ETL/ELTパイプライン6本を設計・運用 — 6 3種類のDBエンジンにまたがるETLパイプライン

ETL/ELTパイプライン6本を設計・運用 お問い合わせ →

なぜ重要なのか

Djangoの価値は、何が既定で行われるかにあります。QuerySetは、クエリーのテキストとパラメーターを分けてSQLを作るため、利用者が渡す値は、データベースのドライバーがエスケープします。テンプレートは、HTMLで特に危険な文字をエスケープします。CSRF対策は、すべてのPOSTでシークレットを確かめます。パスワードは、SHA-256を使うPBKDF2で保存します。文書は、この鍵のストレッチングの方式を、NISTが推奨していると記します。通常のDjangoコードを書くチームは、自分で要求しなくても、こうした保護を得られます。

同じ既定の動作が、リスクの形も決めます。直接SQLを実行するクエリー、`mark_safe`、`csrf_exempt`、本番サーバーに残した`DEBUG = True`など、一行で、それぞれを無効にできます。どの行も、レビューでは無害に見えます。ORMでは、性能にも同じようなトレードオフがあります。ページの長さとともにクエリー数が増えるビューを、簡単に書けます。Pythonのコードは、どちらの場合も、同じように読めます。

推奨するのは、データがアプリの形を決める場合に、Djangoを選ぶことです。人が作成・編集するレコード、権限のあるアカウント、それら全体の管理画面を必要とする担当者がいる場合です。各プロジェクトを、サポート中のリリースに保ち、A.Bのリリースが続く間はLTSを選び、バージョンの横にサポート終了日を書きます。パイプラインでは、本番設定に対して`manage.py check --deploy`を実行します。シークレットキーとデータベースのパスワードは、リポジトリへ置かず、重要なページごとに、ORMが書くSQLを読みます。顧客が主にページを編集して運用するサイトなら、フレームワークよりも、コンテンツ管理システムや静的ページが向きます。

実際の活用例

2015年から2022年のウェブ開発の仕事には、IPTVプラットフォームを取り巻くDjangoのウェブアプリが含まれました。常時稼働するサービスのうち、人が実際にクリックする部分です。チーム全体で、新機能、バグ修正、性能と安定性の継続的な改善に取り組みました。ページが述べるように、そのようなサービスでは、機能が守るべき制約は安定性です。アプリは、社内チームや顧客に対する信頼性を保ちながら、機能を増やし続けました。

それ以前の、2003年から2014年のフルスタックウェブ開発では、異なる要望を持つ顧客のために、百のウェブアプリを作成しました。独自のものが必要なら、HTML/HTML5、CSS/SCSS、Djangoを使いました。より標準的で、自分たちでも管理できるものが必要なら、WordPressやJoomlaの方が、速く、合理的な答えでした。ページが挙げる技能は、その違いを見極めることです。不要な独自開発を思いとどまらせることも、必要な独自開発へ進むよう勧めることも含みます。先の推奨も、同じ線を引いています。

技術構成のページは、その両面を保っています。ウェブ層は静的構成を優先し、アプリを提供する必要があるところに、DjangoとPHPを使います。言語の一覧は、SQLを、ORMが生成するものとしてではなく、ほかの開発言語と同等に扱います。依頼される性能改善の多くが、そこにあるためです。Djangoのアプリなら、ORMがクエリーを書いても、人が読むという意味です。

問題が起きるところ

- **ループが、一行につき一回問い合わせます。** 既定の取得モードでは、QuerySetが読み込まなかった外部キーを読むと、その一つのオブジェクトのために、クエリーを実行します。百冊の本と著者の一覧なら、101回です。テンプレートは、`all` などを括弧なしで呼ぶため、さらに見つけにくくします。外部キーと一対一のフィールドには `select_related()` 、多対多と逆向きの外部キーには `prefetch_related()` を使います。Django 6.1以降の `FETCH_PEERS` モードは、それらの取得を、追加の一回のクエリーへまとめます。 `FETCH_RAISE` は、速さが必要なコードで、予定外のクエリーを例外にします。
- **デバッグモードのまま、本番で稼働させます。** デプロイのチェックリストは、本番で `DEBUG` を決して有効にしないよう求めています。エラーページが、ソースコードの一部、ローカル変数、設定、使用するライブラリーを漏らすためです。 `DEBUG = False` では、 `ALLOWED_HOSTS` へ適切な値を入れるまで、Djangoはリクエストをまったく処理しません。ワイルドカードを使うなら、Hostヘッダーを自分で検証します。本番設定のファイルに対して `manage.py check --deploy` を実行すると、チェックリストの一部を自動化できます。
- **シークレットキーが、リポジトリにあります。** `SECRET_KEY` は、大半のセッションバックエンドでのセッション、すべてのパスワード再設定トークン、別の鍵を指定しないすべての暗号署名を保護します。設定リファレンスは、鍵が知られると、権限昇格やリモートコード実行につながり得ると警告します。環境かファイルから読み、バージョン管理へ入れず、 `SECRET_KEY_FALLBACKS` でローテーションし、古い鍵は速やかに一覧から取り除きます。
- **一つのビューで、安全な既定の動作を無効にします。** QuerySetはクエリーをパラメーター化しますが、直接SQLを使うクエリー、 `extra()` 、 `RawSQL` では、開発者がエスケープします。テンプレートはHTMLをエスケープしますが、 `mark_safe` 、 `safe` フィルター、自動エスケープの無効化でも、その仕事が開発者へ戻ります。 `csrf_exempt` は、ビューの検査を完全に取り除きます。どれもレビューで、理由の明記が必要です。利用者が保存するHTMLは、出力時だけでなく、入力時にもサニタイズします。
- **マイグレーションが、途中で止まります。** PostgreSQLとSQLiteでは、既定で、マイグレーションのすべての操作が、一つのトランザクション内で動きます。MySQLでは、スキーマ変更を囲むトランザクションがないため、途中で失敗すると、手で元に戻す変更が残ります。 `makemigrations` が、何を変更と判断したかを読み、本番へ届く前に `sqlmigrate` でSQLを確かめます。各マイグレーションは、モデルの変更と一緒にコミットします。

参考資料

- Djangoの文書 — 現在の安定版の文書です。チュートリアルからリファレンスまでを含みます。

- Django：デプロイのチェックリスト — 本番前に変える設定です。一部を `check --deploy` で自動化します。
- Django：データベースアクセスの最適化 — QuerySetの評価、関連オブジェクトの取得、クエリー数を一定に保つ習慣です。
- Django：マイグレーション — マイグレーションのワークフロー、`sqlmigrate`、各データベースエンジンのスキーマ変更の扱いです。
- Djangoのセキュリティー — フレームワークが既定で防ぐものと、保護の限界です。
- Djangoのダウンロード — サポート中のリリース系列と、通常サポート、延長サポートの終了日です。

用語集

実際の制約に合わせて選ぶ

業務の流れ、既存のインターフェース、変更を必要とする利用者をお知らせください。検証可能な納品内容と、お客様のチームが保守を続けられる引き継ぎ内容を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「Djangoとは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/django/