Django Tutorial | Learn Django Framework

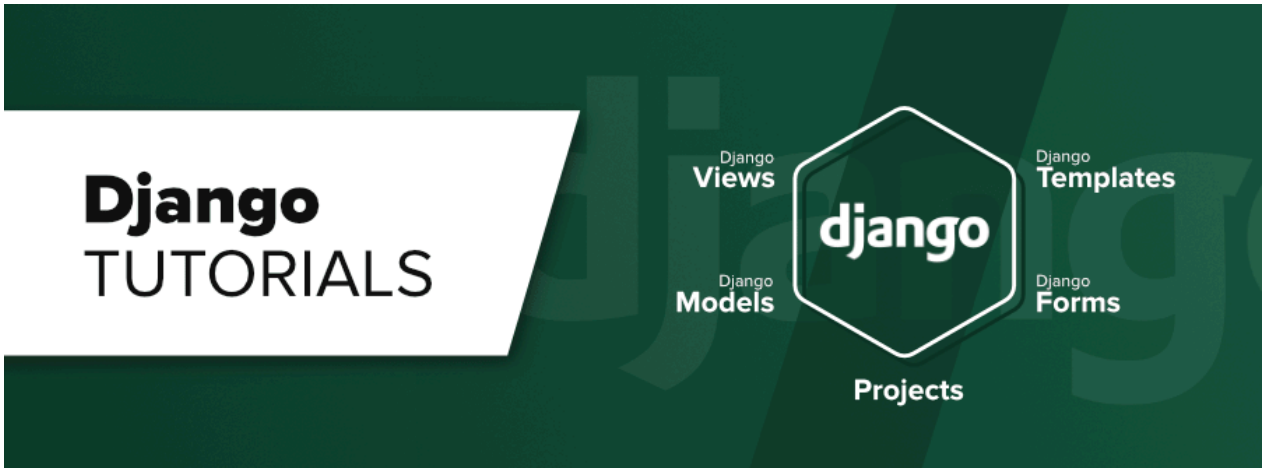
Last Updated : 03 Oct, 2024

Django, built with Python, is designed to help developers build secure, scalable, and feature-rich web applications quickly and efficiently. Whether you're a beginner looking to create your first dynamic website or an experienced developer aiming to enhance your skills, this tutorial will guide you through Django's core concepts.

This also tutorial provides you with **Django projects** to help you apply your knowledge and build some cool web applications. These projects not only provide you with experience in building with the Django framework but will also add value to your resume. This Python Django tutorial teaches basic to advanced Django concepts for **backend development**. Learn topics like **forms**, **templates**, **views, ORM, etc.**

Python Django

Python-based web framework Django allows you to create efficient **web applications** quickly. It is also called **batteries included web framework Django** because It provides built-in features for everything including Django Admin Interface, default database – SQLlite3, etc.



When you're building a website, you always need a similar set of components: a way to handle user authentication (signing up, signing in, signing out), a

management panel for your website, forms, a way to upload files, etc. Django Python gives you ready-made components to use for rapid development. There any many more benefits of using the Django framework. Let's look at some other reasons why you should learn Python Frameworks in Django.

Table of Content

Why Use Django Framework?

- Excellent documentation and high scalability.
- Used by Top MNCs and Companies, such as Instagram, Disqus, Spotify, Youtube, Bitbucket, Dropbox, etc. and the list is never-ending.
- Web framework Django is easy to learn, rapid development, and Batteries fully included.
- The last but not least reason to learn Django in **Python**, It has a huge library and features such as Web Scraping, Machine Learning, Image Processing, Scientific Computing, etc. One can integrate all this with web applications and do lots and lots of advanced stuff.

and Drupal can be chosen and launched quickly. The 4 active plans with prices ranging from **₹499/mo** to **₹1829/mo** :

- KVM1
- KVM2
- KVM4
- KVM8

These are designed to satisfy various needs. The **KVM2** plan is the most affordable and well-liked by users of lightweight apps.

> *Full root access, support for many operating systems, excellent performance, improved security and backup, one-click application, and scalability are features of Hostinger's VPS service .*

Prerequisites to Learn Django

Web framework Django based on Python. You have good knowledge about Python. Some other concepts you should be familiar with are:

- Understanding of **Syntax of Python** .
- Understanding of **importing and exporting modules** is required in the project development phase.
- Understanding **Python path concepts t** o access the data, images or any kind of data.
- Knowledge of **Object Oriented concepts** as it reduces the code repetition with classes and objects.
- Knowledge about **HTML** , **CSS** , **JavaScript** are very important. As they are the building block of **Web development** .
- Knowledge about **Data Structures** like **Tuple** and **List** are important.

Getting Started with Django

In this getting started with Django section, you will learn how to get up and running with Django, a powerful web framework for building dynamic websites and applications. Django makes it easy to organize your project into different apps, each handling specific functionality. You'll begin by creating a project,

which will automatically set up everything you need, from a folder structure to basic settings.

- Django Introduction and Installation
- When to Use Django? Comparison with other Development Stacks
- Django Project MVT Structure
- Django Basics
- Create an App in Django
- Create a Basic Project using MVT in Django

Once you've mastered the basics from this tutorial, the **Complete Django Web Development Course – Basics to Advance** is a perfect next step to dive deeper into both beginner and advanced Django concepts.

Django Views

In Django views are the backbone of handling user requests and rendering responses. There are two primary paradigms for implementing views: Function Based Views (FBVs) and Class Based Views (CBVs). Function Based Views offer simplicity and directness, allowing developers to define views as Python functions. Within this paradigm, common functionalities like creating, listing, displaying details, updating, and deleting objects are implemented as separate functions.

While both paradigms have their merits, the choice between FBVs and CBVs ultimately depends on factors such as project requirements, development preferences, and scalability concerns.

- Function Based Views
    - Create View
    - List View
    - Detail View
    - Update View
    - Delete View
- Class Based Generic Views Django
    - Createview
    - ListView
    - DetailView

    - UpdateView
    - DeleteView
    - FormView
- Class Based vs Function Based Views

Django URLs

In Django URL patterns serve as a crucial mechanism for directing incoming requests to the appropriate views within your web application. With the flexibility of regular expressions, Django's URL dispatcher allows you to define patterns that match specific URL patterns and route them to corresponding views. When dealing with GET parameters passed through URLs in Django, accessing these parameters within views is straightforward, enabling efficient handling of user inputs and customization of responses.

- Django URL patterns
- Get parameters passed by urls in Django
- URL Validator in Django
- URL Shortener with Django
- Django URLResolver error

Django Templates

In Django, URLs play a crucial role in navigating through different views and templates within your web application. When working with Django templates, several key concepts enhance the flexibility and functionality of your frontend. Template filters allow you to manipulate variables displayed in your templates, enabling transformations like date formatting or string manipulation.

- Template Filters
- Template Tags
- variables
- Boolean Operators
- for loop
- if – Django Templates
- Template Inheritance

Django Models

Django Models serve as the backbone of database operations, facilitating seamless management of data. This guide delves into various aspects of Django Models, starting from the fundamental operations of inserting, updating, and deleting data using the Object-Relational Mapping (ORM) provided by Django.

Here, You'll learn how to create a basic app model, initialize migrations, and execute them to synchronize your database schema. Moreover, we delve into built-in field validations, ensuring data integrity and consistency, while also delving into the customization of these validations to suit specific application requirements.

- ORM – Inserting, Updating & Deleting Data
- Basic App Model – Makemigrations and Migrate
- model data types and fields list
- Add the slug field inside Django Model
- Intermediate fields in Django
- Uploading images in Django
- Render Model in Django Admin Interface
- Change Object Display Name using __str__ function – Django Models
- Built-in Field Validations – Django Models
- Custom Field Validations in Django Models
- How to use Django Field Choices ?
- Overriding the save method – Django

Django Forms

To start, you can create a form using Django Forms by defining a class that inherits from Django's forms.Form class. Within this class, you can specify the fields you want to include in your form using various field types provided by Django, such as CharField, IntegerField, EmailField, etc. Once you've defined your form, you can render HTML forms in Django using both GET and POST methods. Django's built-in template tags and filters make it easy to render forms in your HTML templates while ensuring security and CSRF protection.

Django Forms offer a wide range of field types to cater to different data types and validation requirements. Additionally, you can customize the appearance

and behavior of form fields by using form field custom widgets, allowing you to enhance user experience and tailor forms to your specific needs.

- How to create a form using Django Forms ?
- Render HTML Forms (GET & POST) in Django
- Django Form Fields
- form field custom widgets
- Initial form data – Django Forms
- ModelForm – Create form from Models
- Render Form Fields Manually
- Django Formsets
- Django ModelFormSets

Misc

- Handling Ajax request in Django
- User groups with Custom permissions in Django
- Django Admin Interface
- Extending and customizing django-allauth
- Django – Dealing with warnings
- Sessions framework using django
- Django Sign Up and login with confirmation Email

After completing the tutorial and building some projects you might be interested in starting your career in Django development. We have provided a guide, that will help you in building your career as **Django developer** .

Django Projects

In this section, we'll explore how to structure and manage **Django projects**, which are the core framework for building any web application. These projects act as containers for multiple apps that handle specific functionalities, such as user authentication, blog management, or e-commerce operations.

- Google authentication and Fetching mails from scratch
- ToDo webapp using Django
- Django News App
- Weather app using Django
- College Management System Using Django

- E-Commerce Website Using Django
- Creating Word Counter App Using Django
- Youtube Video Downloader Using Django
- Voting System Project Using Django

Online Django Quiz

Test your Django knowledge by answering quiz questions. The quiz questions are meant to test your understanding of Django concepts.

Take Django Quiz

Python Django Interview Question

Interviews are most important aspect of job recruitment and you need to prepare for interviews if you want to get job sooner. We have compiled some of the most asked interview questions for Django Developers.

Visit the page Top 50 Django Interview Questions and Answers to check for interview questions.

Features of Django

- **Rapid Development**: Django's DRY principle accelerates development by reducing code repetition.
- **Admin Interface**: Comes with a ready-to-use, customizable admin panel for easy backend management.
- **Scalable**: Built to handle high traffic and complex applications, ideal for projects of any size.
- **Security**: Offers built-in protections against common security threats like XSS, CSRF, and SQL injection.
- **ORM**: Simplifies database interaction using Python, eliminating the need for raw SQL.
- **URL Routing**: Clean, readable URLs with easy mapping to views.
- **Template Engine**: Separates logic from presentation for dynamic, reusable web pages.

- **Extensive Documentation**: Well-organized resources for troubleshooting and learning.
- **Active Community**: Large community support with abundant third-party plugins and tools.

Applications of Django

Django is a versatile web framework used in a wide range of industries and projects. Here are some common applications:

- **Content Management Systems (CMS)**: Django is ideal for building custom CMS platforms due to its modularity and flexibility.
- **E-commerce Sites**: Platforms like e-commerce websites benefit from Django's scalability and robust security features.
- **Social Networking Platforms**: Django's ability to handle high traffic makes it perfect for social media apps and community-based websites.
- **Data-Driven Applications**: With its powerful ORM and database management capabilities, Django is great for building applications that rely on large datasets.
- **API Development**: Django, coupled with Django REST Framework (DRF), makes it simple to develop robust and scalable APIs.
- **Scientific Computing Platforms**: Django is used in platforms that require complex data analysis and visualization.
- **News & Publishing Platforms**: Its ability to manage large volumes of content efficiently makes it a go-to for news websites and online publications.
- **Educational Platforms**: Many e-learning websites and educational tools are built with Django for its scalability and security.

Django vs. Other Web Frameworks

Feature	Django	Java Spring Boot	Express.js
Language	Python	Java	JavaScript (Node.js)

Feature	Django	Java Spring Boot	Express.js
Architecture	Full-stack (MVT)	Full-stack (MVC)	Minimalist
Admin Interface	Built-in admin interface	No built-in admin interface	No built-in admin interface
Development Speed	Fast (due to built-in features)	Medium (more setup required)	Fast (simple routing)
Scalability	Highly scalable	Highly scalable	Highly scalable
Security	Excellent (with built-in protections)	Excellent (strong security features)	Needs additional security measures
ORM	Powerful ORM (built-in)	Powerful ORM (Hibernate)	No built-in ORM
Flexibility	Moderate (convention over configuration)	High (highly configurable)	High (minimalist with custom flexibility)
Documentation	Extensive and detailed	Extensive and detailed	Good but less detailed
Best for	Full-scale apps, CMS, e-commerce	Enterprise-level applications, APIs	Real-time apps, APIs, microservices
Learning Curve	Moderate (due to its many features)	Steeper (requires understanding of Java ecosystem)	Moderate
Use Cases	Social networks, CMS, e-commerce	Large-scale enterprise applications	Real-time apps, microservices

Careers with Django

Here's a table showcasing some common **Django career roles** along with their approximate salary ranges in both **INR** and **USD**:

Career Role	Salary (INR/year)	Salary (USD/year)
Django Developer	₹4,00,000 – ₹10,00,000	$50,000 – $90,000
Full-Stack Developer	₹6,00,000 – ₹15,00,000	$60,000 – $120,000
Software Engineer	₹5,00,000 – ₹12,00,000	$70,000 – $110,000
DevOps Engineer	₹8,00,000 – ₹18,00,000	$80,000 – $140,000
Technical Lead/Architect	₹12,00,000 – ₹25,00,000	$100,000 – $180,000

FAQs on Django Tutorial

Whst is Django?

Django is a high-level Python web framework that allows developers to build robust, scalable, and secure web applications quickly and efficiently. It follows the Model-View-Controller (MVC) architecture pattern, though it is often referred to as Model-View-Template (MVT) in Django terminology.

Is Django easy to learn

Web framework Django is relatively easy to learn, if you have prior experience with Python.

Is Django for frontend or backend?

Django is a open-source framework that is used for backend development of web application.

What are the Features of Django ?

- *Versatile which allows us to develop any kind of web page.*
- *It is scalable*
- *It is extremely fast.*
- *Secure thereby helping developers.*
- *It comes along with content administrations, authentications.*

What is Django Architecture ?

Django is based on MVT(Model View Template architecture) which is based on the MVC(Model View Controller architecture). The common difference between them is that Django take care of controller part.

What is the difference between Flask and Django ?

Django	*Flask*
Supports large projects.	*Supports smaller projects.*
Templates, Admin and ORM is built-in.	*Templates, Admin and ORM requires to be installed.*
Not easy to as compare to Flask.	*It is easy to learn.*

Complete Web development no need any third party tools.	*User can choose any third party tools according to their needs.*
Does not support visual debugging.	*Supports visual debugging.*
Inbuilt bootstrapping tool .	*Bootstrapping tools are not available.*

Name some companies that uses Django ?

Some companies that uses Django are: Instagram, DISCUS, Mozilla Firefox, Youtube, Instagram, Reddit etc. are using web framework Django.

Are you ready to elevate your web development skills from foundational knowledge to advanced expertise? Explore our Mastering Django Framework - Beginner to Advanced Course on GeeksforGeeks, designed for aspiring developers and experienced programmers. This comprehensive course covers everything you need to know about Django, from the basics to advanced features. Gain practical experience through **hands-on projects** and real-world applications, mastering essential Django principles and techniques. Whether you're just starting or looking to refine your skills, this course will empower you to build sophisticated web applications efficiently. Ready to enhance your web development journey? Enroll now and unlock your potential with Django!

Next Article

Django Basics

Similar Reads

Integrating Django with Reactjs using Django REST Framework

In this article, we will learn the process of communicating between the Django Backend and React js frontend using the Django REST Framework. For the sake...

15+ min read

Django REST Framework Installation

Django REST Framework can be installed via pip package similar to Django installation. Since Django REST Framework is a wrapper over default Django...

1 min read

How to Create a basic API using Django Rest Framework ?

Django REST Framework is a wrapper over the default Django Framework, basically used to create APIs of various kinds. There are three stages before...

4 min read

Creating and Using Serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Serializers allow complex data suc...

3 min read

URL fields in serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

5 min read

File upload Fields in Serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

5 min read

ListField in serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

4 min read

IPAddressField in serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

4 min read

Numeric fields in serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

5 min read

Date and time fields in serializers - Django REST Framework

In Django REST Framework the very concept of Serializing is to convert DB data to a datatype that can be used by javascript. Every serializer comes with some...

7 min read

Article Tags : Django Python Django Spotlight Tutorials

System Design Bootcamp
Interview Questions

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects