Django Tutorial: The Local Library website

Overview: Django

The first article in our practical tutorial series explains what you'll learn, and provides an overview of the "local library" example website we'll be working through and evolving in subsequent articles.

Prerequisites:	Read the Django Introduction. For the following articles you'll also need to have set up a Django development environment.
Objective:	To introduce the example application used in this tutorial, and allow readers to understand what topics will be covered.

Overview

Welcome to the MDN "Local Library" Django tutorial, in which we develop a website that might be used to manage the catalog for a local library.

In this series of tutorial articles you will:

- Use Django's tools to create a skeleton website and application.
- Start and stop the development server.
- Create models to represent your application's data.
- Use the Django admin site to populate your site's data.

- Create views to retrieve specific data in response to different requests, and templates to render the data as HTML to be displayed in the browser.
- Create mappers to associate different URL patterns with specific views.
- Add user authorization and sessions to control site behaviour and access.
- Work with forms.
- Write test code for your app.
- Use Django's security effectively.
- Deploy your application to production.

You have learned about some of these topics already, and touched briefly on others. By the end of the tutorial series you should know enough to develop simple Django apps by yourself.

The LocalLibrary website

LocalLibrary is the name of the website that we'll create and evolve over the course of this series of tutorials. As you'd expect, the purpose of the website is to provide an online catalog for a small local library, where users can browse available books and manage their accounts.

This example has been carefully chosen because it can scale to show as much or as little detail as we need, and can be used to show off almost any Django feature. More importantly, it allows us to provide a *guided* path through the most important functionality in the Django web framework:

- In the first few tutorial articles we will define a simple *browse-only* library that library members can use to find out what books are available. This allows us to explore the operations that are common to almost every website: reading and displaying content from a database.
- As we progress, the library example naturally extends to demonstrate more advanced Django features. For example we can extend the library to allow users to reserve books, and use this to demonstrate how to use forms, and support user authentication.

Even though this is a very extensible example, it's called ***Local**Library* for a reason — we're hoping to show the minimum information that will help you get up and running with Django quickly. As a result we'll store information about books, copies of books, authors and other key

information. We won't however be storing information about other items a library might store, or provide the infrastructure needed to support multiple library sites or other "big library" features.

I'm stuck, where can I get the source?

As you work through the tutorial we'll provide the appropriate code snippets for you to copy and paste at each point, and there will be other code that we hope you'll extend yourself (with some guidance).

If you get stuck, you can find the fully developed version of the website on Github here.

Summary

Now that you know a bit more about the *LocalLibrary* website and what you're going to learn, it's time to start creating a skeleton project to contain our example.

Overview: Django

In this module

- Django Tutorial Part 4: Django admin site
- Django Tutorial Part 5: Creating our home page
- Django Tutorial Part 6: Generic list and detail views
- Django Tutorial Part 7: Sessions framework
- Django Tutorial Part 8: User authentication and permissions
- Django Tutorial Part 9: Working with forms
- Django Tutorial Part 10: Testing a Django web application
- Django Tutorial Part 11: Deploying Django to production
- Django web application security
- DIY Django mini blog

Last modified: Mar 18, 2019, by MDN contributors

Related Topics

Complete beginners start here!

- Getting started with the Web

HTML — Structuring the Web

- Introduction to HTML
- Multimedia and embedding
- HTML tables

CSS — Styling the Web

- CSS first steps
- CSS building blocks
- Styling text
- CSS layout

JavaScript — Dynamic client-side scripting

- JavaScript first steps
- JavaScript building blocks
- Introducing JavaScript objects
- Asynchronous JavaScript
- Client-side web APIs

Web forms — Working with user data

- Core forms learning pathway
- Advanced forms articles

Accessibility — Make the web usable by everyone

- Accessibility guides
- Accessibility assessment

Tools and testing

- Client-side web development tools
- Introduction to client-side frameworks
- React
- Ember
- Vue
- Git and GitHub
- Cross browser testing

Server-side website programming