ProTech
protechtraining.com

"Charting the Course ...

... to Your Success!"

Django Bootcamp

Course Summary

Description

This dense course teaches existing Python developers how to develop full-stack Web applications using the Django framework. It starts with the basics of creating simple models, views and controllers, then moves on to more advanced topics such as administration, session management, authentication and form processing.

This is a hands-on programming class. All concepts are reinforced by informal practice during the lecture followed by graduated lab exercises.

Django Boot Camp is a comprehensive introduction to this popular Web framework. Students will immediately be able to use Django to create real-world Web applications.

Note: This course can be taught for either Python 2.x or Python 3.x. Version-specific course materials and exercises will be provided.

Objectives
At the end of this course, students will be able to:

- Develop full-stack web sites based on content stored in an RDMS
- Define data models
- Understand the architecture of a Django-based web site
- Create Django templates for easy-to-modify views
- Map views to URLs
- Take advantage of the builtin Admin interface
- Provide HTML form processing

Topics

- What is Django?
- Why use a framework?
- Configuring a project
- Creating an app
- Defining and querying models
- Developing templates
- Implementing controllers
- Session management
- Access control and authentication
- Form processing

Audience

This course is appropriate for existing Python developers who want to easily create and deploy database-oriented web applications

Prerequisites

Students should already have a working knowledge of Python, HTML 5 and CSS.

Duration

Five days

ProTech
protechtraining.com

"Charting the Course ...

... to Your Success!"

Django Bootcamp

Course Outline

I. What is Django?
- A. Django is a framework
- B. Frameworks vs Packages
- C. What can it do?
- D. What are the alternatives?

II. Django Architecture
- A. Sites and apps
- B. Shared configuration
- C. Minimal Django layout
- D. Builtin flexibility

III. Configuring a project
- A. Executing manage.py
- B. Starting the project
- C. Database setup
- D. The development server

IV. Adding an application
- A. Generate the application files
- B. Defining models
- C. Related objects
- D. SQL Migration
- E. App configuration
- F. Accessing models

V. Login for nothing and Admin for free
- A. Setting up the admin user
- B. Running the admin site
- C. Tweaking the admin interface
- D. Changing the admin index page

VI. Basic Views (AKA Controllers)
- A. What is a view
- B. HttpResponse
- C. URL route configuration
- D. Shortcut: get_object_or_404()

VII. Basic Views (AKA Templates)
- A. About templates
- B. Django template syntax
- C. Static files
- D. Loading templates
- E. The url tag
- F. Shortcut: template.render()

VIII. Querying the models
- A. QuerySets
- B. Field lookups
- C. Chaining filters
- D. Slicing QuerySets
- E. Related fields
- F. Q objects

IX. Enhancing models
- A. Custom methods
- B. Complex relationships
- C. Overriding standard methods

X. Working with templates
- A. Variable lookups
- B. Comments
- C. Inheritance
- D. Filters
- E. Escaping HTML
- F. Custom filters

XI. Forms
- A. Forms overview
- B. GET and POST
- C. The Form class
- D. Processing the form
- E. Widgets
- F. Validation
- G. Forms in templates
- H. Beyond the basics

XII. Static file managment
- A. Types of static files
- B. Configuring access
- C. Namespacing
- D. Templates
- E. Deploying from outside Django

XIII. Generic views
- A. About generic views
- B. Types of generic views
- C. Default generic views
- D. Class-based generic views
- E. List and detail views

XIV. User Authentication
- A. Authentication vs Authorization
- B. Configuring users
- C. Permissions
- D. Groups

XV. Session management
- A. Enabling sessions
- B. Types of session backends
- C. Session cookies
- D. Accessing sessions from views

XVI. Automated testing
- A. Why create tests?
- B. When to create tests
- C. Using Django's test framework
- D. Using the test client
- E. Running tests
- F. Checking code coverage

Django Bootcamp

Course Outline (cont'd)

XVII. Creating Reusable apps
- A. Packaging
- B. Choosing an appropriate name
- C. Deploying
- D. Using setuptools

XVIII. Serving data with a RESTful interface
- A. The Django REST framework
- B. Serialization
- C. Requests and Responses
- D. Function-based views
- E. Class-based views
- F. What about django-tastypie?

XIX. Building responsive web sites
- A. Choosing a JS library
- B. Creating API endpoints
- C. Fixing template delimiter conflicts

XX. Using the cache
- A. Types of caches
- B. Setting up the cache
- C. Per-site and per-view caching
- D. Low-level API
- E. Cache security
- F.