

Welcome to Flask's documentation. Get started with <u>Installation</u> and then get an overview with the <u>Quickstart</u>. There is also a more detailed <u>Tutorial</u> that shows how to create a small but complete application with Flask. Common patterns are described in the <u>Patterns for Flask</u> section. The rest of the docs describe each component of Flask in detail, with a full reference in the <u>API</u> section.

Flask depends on the <u>Jinja</u> template engine and the <u>Werkzeug</u> WSGI toolkit. The documentation for these libraries can be found at:

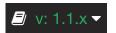
- Jinja documentation
- Werkzeug documentation

User's Guide

This part of the documentation, which is mostly prose, begins with some background information about Flask, then focuses on step-by-step instructions for web development with Flask.

- Foreword
 - What does "micro" mean?
 - Configuration and Conventions
 - Growing with Flask
- Foreword for Experienced Programmers
 - Thread-Locals in Flask
 - Develop for the Web with Caution
- Installation
 - Python Version
 - Dependencies
 - Virtual environments
 - Install Flask
 - Install virtualenv





- A Minimal Application
- What to do if the Server does not Start
- Debug Mode
- Routing
- Static Files
- Rendering Templates
- Accessing Request Data
- Redirects and Errors
- About Responses
- Sessions
- Message Flashing
- Logging
- Hooking in WSGI Middleware
- Using Flask Extensions
- Deploying to a Web Server

Tutorial

- Project Layout
- Application Setup
- Define and Access the Database
- Blueprints and Views
- Templates
- Static Files
- Blog Blueprint
- Make the Project Installable
- Test Coverage
- Deploy to Production
- Keep Developing!

• Templates

- Jinja Setup
- Standard Context
- Standard Filters
- Controlling Autoescaping
- Registering Filters
- Context Processors

• Testing Flask Applications

- The Application
- The Testing Skeleton
- The First Test
- Logging In and Out
- Test Adding Messages



- Other Testing Tricks
- Faking Resources and Context
- Keeping the Context Around
- Accessing and Modifying Sessions
- Testing JSON APIs
- Testing CLI Commands

Application Errors

- Error Logging Tools
- Error handlers
- Logging

• Debugging Application Errors

- When in Doubt, Run Manually
- Working with Debuggers

Logging

- Basic Configuration
- Email Errors to Admins
- Injecting Request Information
- Other Libraries

• Configuration Handling

- Configuration Basics
- Environment and Debug Features
- Builtin Configuration Values
- Configuring from Files
- Configuring from Environment Variables
- Configuration Best Practices
- Development / Production
- Instance Folders

Signals

- Subscribing to Signals
- Creating Signals
- Sending Signals
- Signals and Flask's Request Context
- Decorator Based Signal Subscriptions
- Core Signals

• Pluggable Views

- Basic Principle
- Method Hints



- Method Based Dispatching
- Decorating Views
- Method Views for APIs

The Application Context

- Purpose of the Context
- Lifetime of the Context
- Manually Push a Context
- Storing Data
- Events and Signals

• The Request Context

- Purpose of the Context
- Lifetime of the Context
- Manually Push a Context
- How the Context Works
- Callbacks and Errors
- Context Preservation on Error
- Notes On Proxies

Modular Applications with Blueprints

- Why Blueprints?
- The Concept of Blueprints
- My First Blueprint
- Registering Blueprints
- Blueprint Resources
- Building URLs
- Error Handlers

Extensions

- Finding Extensions
- Using Extensions
- Building Extensions

Command Line Interface

- Application Discovery
- Run the Development Server
- Open a Shell
- Environments
- Debug Mode
- Environment Variables From dotenv
- Environment Variables From virtualenv
- Custom Commands



- Plugins
- Custom Scripts
- PyCharm Integration

Development Server

- Command Line
- In Code

• Working with the Shell

- Command Line Interface
- Creating a Request Context
- Firing Before/After Request
- Further Improving the Shell Experience

· Patterns for Flask

- Larger Applications
- Application Factories
- Application Dispatching
- Implementing API Exceptions
- Using URL Processors
- Deploying with Setuptools
- Deploying with Fabric
- Using SQLite 3 with Flask
- SQLAlchemy in Flask
- Uploading Files
- Caching
- View Decorators
- Form Validation with WTForms
- Template Inheritance
- Message Flashing
- AJAX with jQuery
- Custom Error Pages
- Lazily Loading Views
- MongoDB with MongoEngine
- Adding a favicon
- Streaming Contents
- Deferred Request Callbacks
- Adding HTTP Method Overrides
- Request Content Checksums
- Celery Background Tasks
- Subclassing Flask
- Single-Page Applications



- Deployment Options
 - Hosted options
 - Self-hosted options
- Becoming Big
 - Read the Source.
 - Hook. Extend.
 - Subclass.
 - Wrap with middleware.
 - Fork.
 - Scale like a pro.
 - Discuss with the community.

API Reference

If you are looking for information on a specific function, class or method, this part of the documentation is for you.

- API
 - Application Object
 - Blueprint Objects
 - Incoming Request Data
 - Response Objects
 - Sessions
 - Session Interface
 - Test Client
 - Test CLI Runner
 - Application Globals
 - Useful Functions and Classes
 - Message Flashing
 - JSON Support
 - Template Rendering
 - Configuration
 - Stream Helpers
 - Useful Internals
 - Signals
 - Class-Based Views
 - URL Route Registrations
 - View Function Options
 - Command Line Interface



Additional Notes

Design notes, legal information and changelog are here for the interested.

- Design Decisions in Flask
 - The Explicit Application Object
 - The Routing System
 - One Template Engine
 - Micro with Dependencies
 - Thread Locals
 - What Flask is, What Flask is Not
- HTML/XHTML FAQ
 - History of XHTML
 - History of HTML5
 - HTML versus XHTML
 - What does "strict" mean?
 - New technologies in HTML5
 - What should be used?
- Security Considerations
 - Cross-Site Scripting (XSS)
 - Cross-Site Request Forgery (CSRF)
 - JSON Security
 - Security Headers
- Unicode in Flask
 - Automatic Conversion
 - The Golden Rule
 - Encoding and Decoding Yourself
 - Configuring Editors
- Flask Extension Development
 - Anatomy of an Extension
 - "Hello Flaskext!"
 - Initializing Extensions
 - The Extension Code
 - Using _app_ctx_stack
 - Learn from Others
 - Approved Extensions
- Pocoo Styleguide



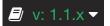
- General Layout
- Expressions and Statements
- Naming Conventions
- Docstrings
- Comments

• Upgrading to Newer Releases

- Version 0.12
- Version 0.11
- Version 0.10
- Version 0.9
- Version o.8
- Version 0.7
- Version o.6
- Version 0.5
- Version 0.4
- Version 0.3

Changelog

- Version 1.1.2
- Version 1.1.1
- Version 1.1.0
- Version 1.0.4
- <u>Version 1.0.3</u>
- Version 1.0.2
- Version 1.0.1
- Version 1.0
- Version 0.12.5
- Version 0.12.4
- Version 0.12.3
- Version 0.12.2
- Version 0.12.1
- Version 0.12
- Version 0.11.1
- Version 0.11
- Version 0.10.1
- Version 0.10
- Version 0.9
- Version 0.8.1
- Version o.8
- Version 0.7.2
- Version 0.7.1



- Version 0.7
- V<u>ersion 0.6.1</u>
- Version o.6
- Version 0.5.2
- Version 0.5.1
- Version 0.5
- Version 0.4
- Version 0.3.1
- Version 0.3
- Version 0.2
- Version 0.1

• License

- Source License
- Artwork License
- How to contribute to Flask
 - Support questions
 - Reporting issues
 - Submitting patches
 - Caution: zero-padded file modes

