Documentation

Hummingbird Documentation

Documentation for Hummingbird the lightweight, flexible, modern server framework.

Hummingbird

Hummingbird is a lightweight and flexible web application framework. It provides a router for directing different endpoints to their handlers, middleware for processing requests before they reach your handlers and processing the responses returned, custom encoding/decoding of requests and responses, TLS and HTTP2.

If you're new to Hummingbird, start here: Build a Todos application.

```
import Hummingbird
// create router and add a single GET /hello route
let router = Router()
    .get("hello") { request, _ -> String in
        return "Hello"
    }
// create application using router
let app = Application(router: router)
// run hummingbird application
try await app.runService()
```

Below is a list of guides and tutorials to help you get started with building your own Hummingbird based web application.

Topics

Getting Started

- Getting Started with Hummingbird

  Create a new project on GitHub or an app locally from a starter template.

- Build a Todos application.

  A tutorial showing you how to build a simple Todos application that allows you to store, access, edit and delete Todos in a database, using Hummingbird and PostgresNIO.

Hummingbird Server

- Router

  The router directs requests to their handlers based on the contents of their path.

- Request Decoding

  Decoding of Requests with JSON content and other formats.

- Response Encoding

  Writing Responses using JSON and other formats.

- Request Contexts

  Controlling contextual data provided to middleware and route handlers

- Middleware

  Processing requests and responses outside of request handlers.

- Error Handling

  How to build errors for the server to return.

- Logging, Metrics and Tracing

  Considered the three pillars of observability, logging, metrics and tracing provide different ways of viewing how your application is working.

- Result Builder Router

  Building your router using a result builder.

- Server protocol
  Support for TLS and HTTP2 upgrades
- Service Lifecycle
  Integration with Swift Service Lifecycle
- Testing
  Using the HummingbirdTesting framework to test your application
- Persistent data
  How to persist data between requests to your server.
- Migrating to Hummingbird v2
  Migration guide for converting Hummingbird v1 applications to Hummingbird v2

Authentication

- Authenticator Middleware
  Request authentication middleware
- Sessions
  Session based authentication
- One Time Passwords
  A one time password (OTP) valid for only one login session.

WebSockets

- WebSocket Server Upgrade
  Support for upgrading HTTP1 connections to WebSocket.
- WebSocket Client
  Connecting to WebSocket servers.

Database Integration

- Postgres Migrations

  Managing database structure changes.

- Store Data with Fluent

  A tutorial that shows you how to set up Hummingbird 2 with Fluent to create and access your Galaxies.

- Store Data with MongoKitten

  A tutorial that shows you how to set up Hummingbird 2 with MongoKitten to create and share your kittens.

Offloading work

- Jobs

  Offload work your server would be doing to another server.

Mustache

- Mustache Syntax

  Overview of Mustache Syntax

- Mustache Features

  An overview of the features of swift-mustache.

Reference Documentation

- Hummingbird

  Lightweight, modern, flexible server framework written in Swift.

- HummingbirdCore

  Swift NIO based HTTP server.

- HummingbirdAuth

  Authentication framework and extensions for Hummingbird.

- HummingbirdCompression

Middleware for decompressing requests and compressing responses

- HummingbirdFluent

  Integration with Vapor's Fluent ORM framework.

- HummingbirdLambda

  Run Hummingbird inside an AWS Lambda.

- HummingbirdPostgres

  Working with Postgres databases.

- HummingbirdRedis

  Add Redis support to Hummingbird server with RediStack.

- HummingbirdWebSocket

  Adds support for upgrading HTTP connections to WebSocket.

- Jobs

  Offload work your server would be doing to another server.

- Mustache

  Mustache template engine.

- WSClient

  Support for connecting to WebSocket server.