Documentation Language: Swift

Framework

HummingbirdWebSocket

Adds support for upgrading HTTP connections to WebSocket.

Overview

WebSockets is a protocol providing simultaneous two-way communication channels over a single TCP connection. Unlike HTTP where client requests are paired with a server response, WebSockets allow for communication in both directions asynchronously, for a prolonged period of time.

It is designed to work over the HTTP ports 80 and 443 via an upgrade process where an initial HTTP request is sent before the connection is upgraded to a WebSocket connection.

HummingbirdWebSocket allows you to implement an HTTP1 server with WebSocket upgrade. HummingbirdWebSocket passes all the tests in the Autobahn test suite, supporting both compression and TLS.

To add `HummingbirdWebSocket` to your project, run the following command in your Terminal:

```
# From the root directory of your project
# Where Package.swift is located

# Add the package to your dependencies
swift package add-dependency https://github.com/hummingbird-project/hummingb

# Add the target dependency to your target
swift package add-target-dependency HummingbirdWebSocket <MyApp> --package h
```

Make sure to replace <MyApp> with the name of your App’s target.

To integrate `HummingbirdWebSocket` into your project, you need to specify WebSocket support in your `Application`'s configuration:

```
import Hummingbird
import HummingbirdWebSocket

let app = Application(
    router: router,
    server: .http1WebSocketUpgrade { request, channel, logger in
        // upgrade if request URI is "/ws"
        guard request.uri == "/ws" else { return .dontUpgrade }
        // The upgrade response includes the headers to include in the respo
        // the WebSocket handler
        return .upgrade([:]) { inbound, outbound, context in
            // Send "Hello" to the client
            try await outbound.write(.text("Hello"))
            // Ending this function automatically closes the connection
        }
    }
)
```

Get started with the WebSockets here: WebSocket Server Upgrade

Topics

Server

`static func http1WebSocketUpgrade(configuration: WebSocketServerConfiguration, additionalChannelHandlers: @autoclosure () -> [any RemovableChannelHandler], shouldUpgrade: (HTTPRequest, Channel, Logger) async throws -> ShouldUpgradeResult<WebSocketDataHandler<HTTP1WebSocketUpgradeChannel.Context>>) -> HTTPServerBuilder`

HTTP1 channel builder supporting a websocket upgrade

```
static func http1WebSocketUpgrade(configuration: WebSocketServer
Configuration, additionalChannelHandlers: @autoclosure () -> [any
RemovableChannelHandler], shouldUpgrade: (HTTPRequest, Channel,
Logger) throws -> ShouldUpgradeResult<WebSocketDataHandler<HTTP1Web
SocketUpgradeChannel.Context>>) -> HTTPServerBuilder
```

HTTP1 channel builder supporting a websocket upgrade

```
static func http1WebSocketUpgrade<WSResponderBuilder>(webSocket
Router: WSResponderBuilder, configuration: WebSocketServer
Configuration, additionalChannelHandlers: @autoclosure () -> [any
RemovableChannelHandler]) -> HTTPServerBuilder
```

HTTP1 channel builder supporting a websocket upgrade

`struct HTTP1WebSocketUpgradeChannel`

Child channel supporting a web socket upgrade from HTTP1

`struct WebSocketServerConfiguration`

Configuration for a WebSocket server

`struct AutoPingSetup`

Automatic ping setup

`enum ShouldUpgradeResult`

Should HTTP channel upgrade to WebSocket

Handler

`typealias WebSocketDataHandler`

Function that handles websocket data and text blocks

`class WebSocketInboundStream`

Inbound WebSocket data frame AsyncSequence

`struct WebSocketOutboundWriter`

Outbound websocket writer

`struct WebSocketDataFrame`

WebSocket data frame.

`protocol WebSocketContext`

Protocol for WebSocket Data handling functions context parameter

Messages

`enum WebSocketMessage`

Enumeration holding WebSocket message

`struct WebSocketInboundMessageStream`

Inbound WebSocket messages AsyncSequence.

Router

`protocol WebSocketRequestContext`

Request context protocol requirement for routers that support WebSockets

`struct BasicWebSocketRequestContext`

Default implementation of a request context that supports WebSockets

`struct WebSocketRouterContext`

WebSocket Context for upgrades initiated via a router

`struct WebSocketHandlerReference`

Reference to a WebSocket handler

`struct WebSocketUpgradeMiddleware`

An alternative way to add a WebSocket upgrade to a router via Middleware

`enum RouterShouldUpgrade`

Enum indicating whether a router `shouldUpgrade` function expects a WebSocket upgrade or not

Extensions

`protocol WebSocketExtension`

Protocol for WebSocket extension

`protocol WebSocketExtensionBuilder`

Protocol for WebSocket extension builder

`struct WebSocketExtensionContext`

Basic context implementation of `WebSocketContext`.

`struct WebSocketExtensionHTTPParameters`

Parsed parameters from `Sec-WebSocket-Extensions` header

`struct WebSocketExtensionFactory`

Build WebSocket extension builder

See Also

Related Documentation

WSCompression

Compression support for WebSockets

HummingbirdWSTesting

Testing framework for WebSockets

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.

Jobs

Offload work your server would be doing to another server.

Mustache

Mustache template engine.

WSClient

Support for connecting to WebSocket server.