The Low–Level Server API

Goal

Set up our first Akka HTTP server
Understand the basic principles of Akka HTTP

Assumed as known:

- Akka, Akka Streams and advanced Scala
- HTTP basics: requests, responses, statuses, headers
- JSON: syntax, structure

(if you need an intro on HTTP/JSON, let me know in Q/A)

Akka HTTP basics

Akka HTTP is

- a suite of libraries
- focused on HTTP integration of an application
- designed for both servers and clients
- based on Akka actors and Akka Streams

Akka HTTP is NOT:

- a framework

Akka HTTP strengths

- stream-based, with backpressure for free
- multiple API levels for control vs ease of use

Core concepts

- HttpRequest, HttpResponse
- HttpEntity
- marshalling

Akka HTTP server

Goal: receive HTTP requests, send HTTP responses

- synchronously via a function HttpRequest => HttpResponse
- async via a function HttpRequest => Future[HttpResponse]
- async via streams, with a Flow[HttpRequest, HttpResponse, _]

(all of the above turn into flows sooner or later)

Under the hood:

- the server receives HttpRequests (transparently)
- $\boxed{\text{the requests go through the flow we write}}$
- the resulting responses are served back (transparently)

Recap

Method 1a: handle connections manually

```
val connectionSource = Http().bind(interface = "localhost", port = 8080)
val requestHandler = (request: HttpRequest) => ... // an HttpResponse
connectionSource.runForeach(connection => connection.handleWithSyncHandler(requestHandler))
```

Method 1b: shorthand

```
Http().bindAndHandleSync(requestHandler, interface = "localhost", port = 8080)
```

Method 2b: async with functions returning futures

```
val asyncRequestHandler = (request: HttpRequest) => ... // a Future[HttpResponse]
Http().bindAndHandleAsync(asyncRequestHandler, interface = "localhost", port = 8080)
```

Method 3b: async with streams

```
val streamsRequestHandler = Flow[HttpRequest].map { ... /* return an HttpResponse */ }
Http().bindAndHandle(streamsRequestHandler, interface = "localhost", port = 8080)
```

Akka rocks