Directives Breakdown

Goal

Familiarize with all the most popular directives

Understand filtering, chaining, extraction and transformation

What a Route Is

```
val myRoute: Route =
  path("home") {
    complete(StatusCodes.OK)
  }
```

```
type Route = RequestContext => Future[RouteResult]
```

A RequestContext contains

- the HttpRequest being handled
- the actor system
- the actor materializer
- the logging adapter
- routing settings
- etc

This is the data structure handled by a Route

You'll almost never need to build a RequestContext by yourself

What a Route Is [2]

Directives create Routes; composing routes creates a *routing tree*

- filtering and nesting
- chaining with ~
- *extracting data*

```
path("home") {
  get {
    complete(StatusCodes.OK)
  } ~
  post {
    complete(StatusCodes.Forbidden)
  }
}
```

→

```
Route 1, filtering on path "/home"  {
  Route 2, filtering on the GET verb {
    Route 3, complete with 200 OK
  } ~
  Route 4, filtering on the POST verb {
    Route 5, complete with 403 Forbidden
  }
}
```

What a Route can do with a RequestContext:

- complete it synchronously with a response
- complete it *asynchronously* with a Future(response)
- handle it asynchronously by returning a Source (advanced)
- reject it and pass it on to the next Route
- fail it

Akka rocks