```
val addType = new Filter[Request, Response, Request, Response] {
  override def apply(request: Request, service: Service[Request,
Response]) = {
    service(request).map(resp => {
      resp.headerMap("Content-type") = "application/json"
      resp
    })
  }
}
```

Concept: Filter

“Provides pre and post processing on an HTTP operation”

ie. it’s a function!

```
Filter[ReqIn, RespIn, ReqOut, RespOut] =
        [ReqIn, Service[ReqOut, RespOut]] => Future[RespIn]
```

```
val addType = new Filter[Request, Response, Request, Response] {
  override def apply(request: Request, service: Service[Request,
Response]) = {
    service(request).map(resp => {
      resp.headerMap("Content-type") = "application/json"
      resp
    })
  }
}
```

Filters compose with other Filters/Services using andThen()

Let’s get started!

- clone: https://github.com/daviddenton/finagle-dojo.git
- project is using (sbt 1.0.1) - adjust before import!
- use the readmes at the root & in each step folder
- finagle docs @ https://twitter.github.io/finagle/guide/