Autoscout24 Tech Task Log

Monday 18:30

The task arrived: build a RESTful web-service in Scala. It sounds resonable, interesting and hopefully feasible!

Some thoughts:

- designing just one entity should be easy
- the structure and verbs for the REST calls are self-evident
- returning json is standard
- I have no experience with Dynamo DB – so I will probably use a simple relational database like SQLite, Derby or H2
- Using an ORM - for instance JPA - looks a little bit like „overkill“ - plain SQL/JDBC should do it for one entity.

Main question: which web-framework should I choose?

- NodeJS / Angular is build on the wrong language
- Spring and JEE are basically possible but don‘t fit well into the scala/sbt environment.
- Play is of course an option – but I know it only rudimentary
- Googling provides other options. For example Skinny – also a Scala Framework and easier?!

The first step, is to have a deeper look at the Play framework. Can I quickly learn enough to build a webapp in 3 days?

Monday 22:00
First impression on Play: Powerful and not trivial but many similarities to other web-frameworks.
I’ ll give it a try!

Projekt techTask created with ‘sbt new playframework/play-scala-seed.g8’

conf/routes: modified for RESTful services
app/CarAdController: created with dummy routines für CRUD services
Manually testing the services with Postman => ok!

Is there a suitable framework for automatically testing REST services?

Tuesday 10:00
Lets create the CarAd-Class, a database trait and a dummy implementation for tests.

Tuesday 13:00
CarAdDummyDB with first tests completed. Test should be generalised (perhaps with Dependency Injection) to work also for “production”-db.

Tuesday 16:00
Next step: Transfering JSON to class CarAd and vice versa.
Trying automated mapping – have to define Mappers for LocalDate and Fuel manually

Tuesday 19:00
After some changes automated mapping and connection to the dummyDB seems to work.
First manual tests with Postman look promising!

What's left to do:

- implementing automated tests (Ok: normally they should be build before starting to code, but I first had to get used to testing in Play!)
- using a "real" database
- POST and PUT should get the id via the JSON-object and not via path-variable ?!
- implement and test full validation
- write readme.md
- enabling CORS requests – they are prohibited by default ?!
- Implement sorting

Wednesday 10:30
Let's start with the validation and test cases for validation.
Next step is to include H2 as real Database – but only the inMemoryDatabase

Wednesday 14:00
Struggling with the usual difficulties:

- strange behaviour of "IDE" VS Code concerning imports – perhaps not really mature for scala development?
- where in the large Play documentation is accessing a JDBC Database described

Wednesday 19:00
Some Problems with injecting default db in Play: "No implementation for play.db.Database was bound."
Finally DB-connection works with H2 in-memory-DB

Wednesday 22:00
Finally found out, how to deal with multiple implementations of one trait with DI.
DB related work finished – for now.

Wednesday 24:00
Sorting finished, CORS finished, POST/PUT paths modified, H2 DB tested
Lets call it a day!

Thursday 10:00
Restructuring DAOs
Implementing end-to-end tests

Thrusday 16:00
Programming finished – after (as usual) some problems. Especially when implementing tests with the FakeRequest class with JsonBody.

Thursday 19:00
Finished documentation.