1. About me
   1. Software Dev @ TSS-Yonder
   2. Main interest on Java technologies
   3. „Lazy” developer – caut cele mai simple, rapide (initial de implementat :P) si elegante solutii.
2. Motivatie
   1. trebuie sa iti implementezi propriul data access layer cu API-uri specifice tehnologiilor
   2. Multitudinea de modalitati de implementare a DAO-ului
   3. din cauza asta se ajunge la boiler plate
   4. foarte greu de mentinut
   5. voi prezenta o solutie care va usura mult munca pe repositories
3. Cunoasterea audientiei:
   1. Cine a mai auzit de Spring Data JPA
   2. Cine a lucrat cu Spring Data JPA
4. Solutii
   1. Spring Data project a fost creat pentru a rezolva problemele acestea
   2. 2010 - Rod Johnson (Spring Source) – Emil Eifrem (Neo Technologies) @ Hacking session
   3. obiectivul este sa ofere o cale mai usoara de a dezvolta aplicatii care utilizeaza ori RDB ori tehnologii mai noi
   4. sa reduca semnificativ cantitatea de boilerplate code necesar pentru a implementa layerul de data acces
   5. JPA, Mongo DB, Redis, Solr, REST
5. Take off
   1. Cum lucram cu JPA ?
      1. mecanismul de persistenta din EJB (heavyweight, complicated XML configurations, programming a lot of boilerplate)
      2. JDBC API – deployable in orice servlet container, dar foarte mult boilerplate este necesar – trasnformarea din domain model in queriuri si resultatele inapoi in domain model)
      3. ORM – Hibernate/JPA – e o alternativa buna pentru ca ne-a eliberat de scrierea de cod ne-necesar cu JDBC dar boilerplate-ul tot este prezent – JPQL, HQL, Criteria API)
   2. Cum vom lucra cu JPA ?
      1. queriuri by method naming convention
      2. Suporta HQL, native queries, named queries
      3. DSL (Domain specific language) queries
6. Demo time –
   1. Hibernate (already in place) vs Spring Data JPA (developed at that moment) with live examples and comparisons
7. Drawbacks
   1. Not suitable for some projects (example my project)
   2. etc