Dali Tooling for Dynamic and NoSQL Persistence

Neil Hauge, Dali Project Lead, Oracle

Background

- Agenda
  - Review of Dali, dynamic persistence, and NoSQL persistence
  - Demo
- People
  - Neil Hauge – Dali project lead / committer
  - John Bracken – EclipseLink committer and major contributor to the NoSQL technology presented in this demo
  - All Dali and EclipseLink committers

Dali Persistence Tools Review

- Started in late 2005!
- Contains rich (extensible) models for JPA and JAXB persistence
  - “Really complicated” with defaults/overrides and database variations
  - ~4000 classes, 500K+ LOC
- Dali’s rich UI, validation, and content assist are all a part of the tools offering
- Offer many other tools for automated mapping, entity generation, and EclipseLink specific mapping support

Where do we go from here?

- HTML5 and NoSQL have created new persistence challenges for application developers
- Our goal is to reduce complexity and improve efficiency during development

Dynamic Persistence

- What is dynamic persistence?

Regular Java Entity

```
@Entity
public class Employee
    @Id
    private int id;
    @Column(name="L_NAME")
    private String lastName;
...
```

Dynamic Entity (Declarative)

```
<entity access="VIRTUAL" class="Employee">
    <attributes>
        <id attribute-type="int" name="id">
        </id>
        <basic attribute-type="java.lang.String"
            name="description"
            column="L_NAME">
        </basic>
...
```

Dynamic Persistence

- Dynamic entities are defined in the EclipseLink ORM XML file
- Dali is able to generate a fully mapped model in metadata form (eclipselink-orm.xml) from an existing relational database
- Paired with EclipseLink JPA-RS* support, dynamic persistence provides a simple yet powerful way for web applications to access their data sources without the need for Java code

*RESTful interface for interacting with JPA Persistence Units

EclipseLink NoSQL Persistence

- EclipseLink was non-relational ready
- Possible to use familiar Java Persistence concepts to persist data to NoSQL databases
- Exploring this area in the run-time (EclipseLink) and design-time tooling (Dali entity generation)
- Dali would like to eventually provide a comprehensive tooling solution for what has been called “Polyglot Persistence”

EclipseLink NoSQL Persistence

- What does it look like?

NoSQL Java entity

```
@Entity
@NoSql(dataType="employees")
public class Employee
    @Id
    @Field(name="_id")
    private int id;
    private String lastName;
...
```

NoSQL dynamic entity (Declarative)

```
<entity access="VIRTUAL" class="Employee">
  <no-sql data-type="employees" />
  <attributes>
    <id attribute-type="java.lang.String" name="id">
      <field name="_id" />
    </id>
    <basic attribute type="java.lang.String"
      name="lastName">
    </basic>
  </attributes>
</entity>
```

Demo

- Part 1 – Demonstrate persistence tooling for dynamic entities and NoSQL using MongoDB
- Part 2 – Demonstrate persisting data from the web using a RESTful interface (JPA-RS) with a NoSQL database (MongoDB)
  - Using a “REST” Client
  - Using a simple JavaScript application