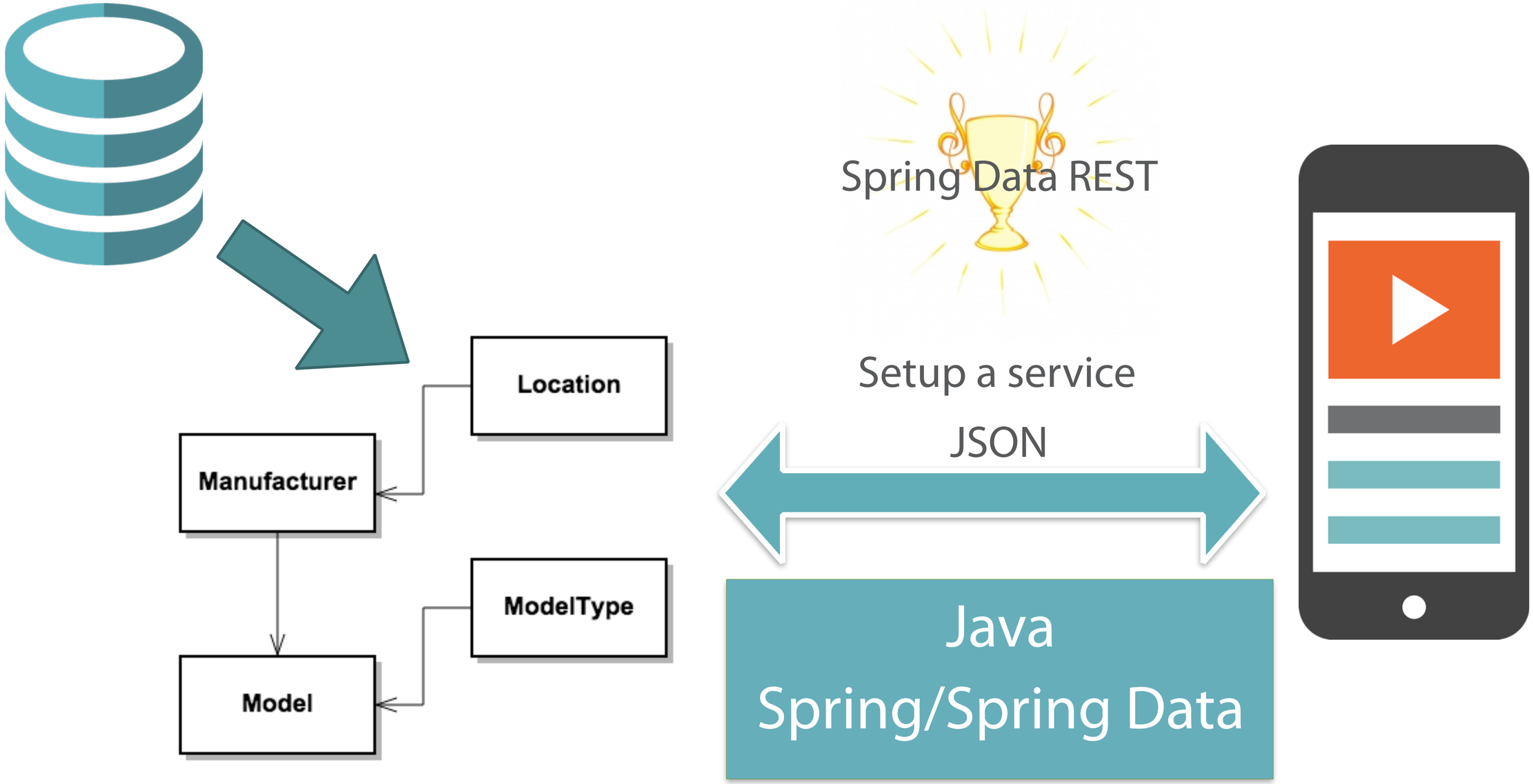
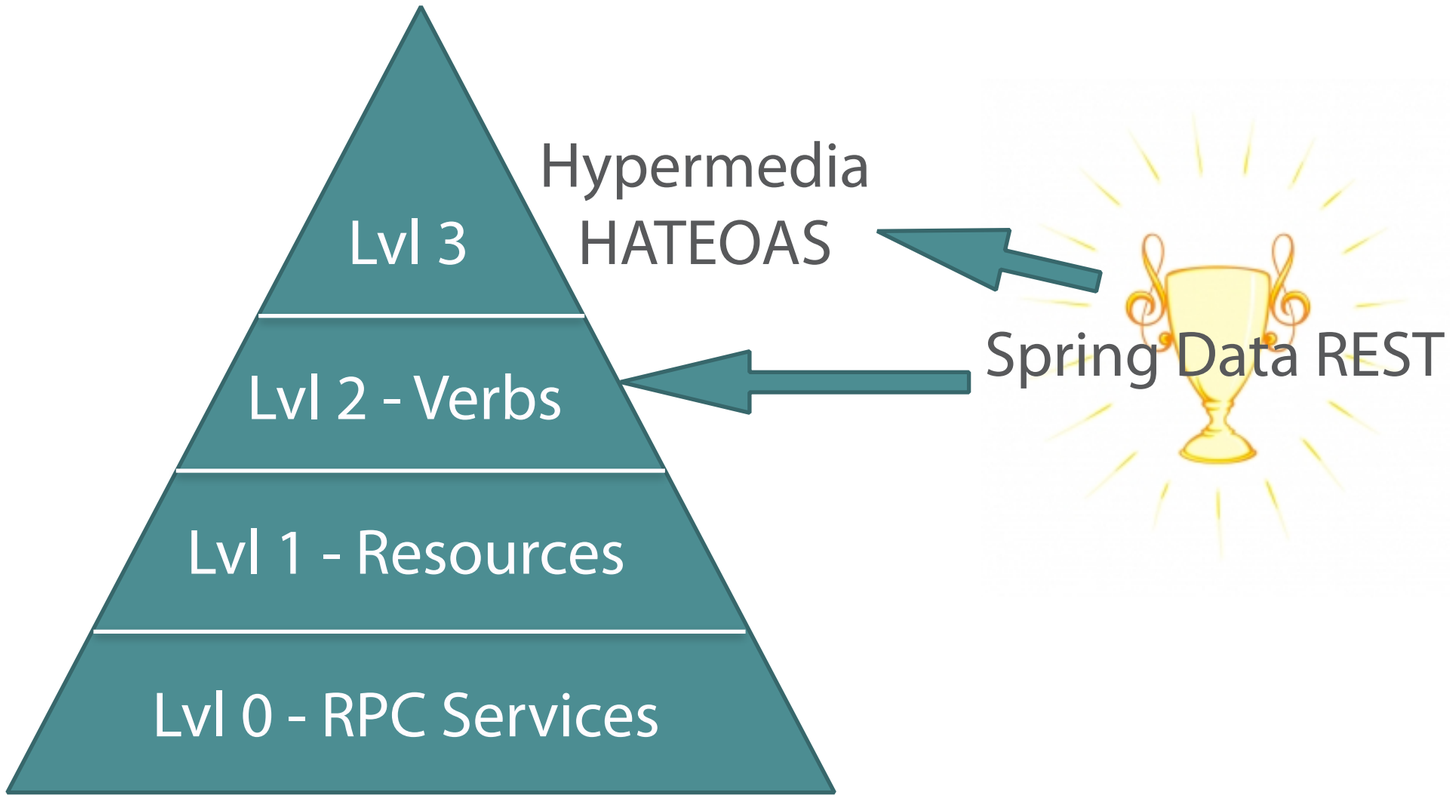
Getting Started with Spring Data REST

What Can Spring Data REST Do?

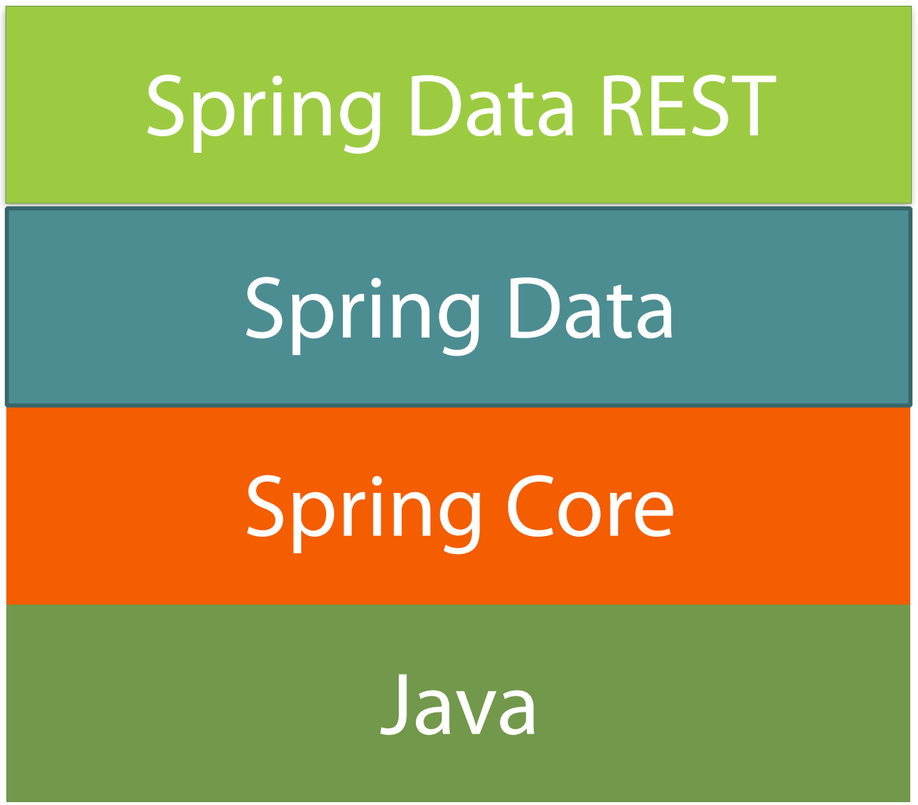


A Short Lecture on REST



Richardson Maturity Model

Learning Path



REST, Spring MVC

JPA/JDBC/etc.
Databases

Prerequisites

Prerequisites

- git
  - http://git-scm.com
- JDK 1.8+
  - http://www.oracle.com
- Maven 3+
  - http://maven.apache.org

```
git --version
```

```
javac -version
```

http://www.oracle.com/technetwork/java/javase/downloads/index.html

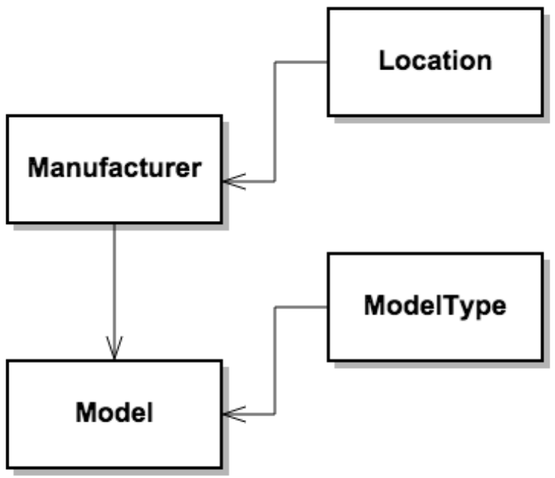
```
mvn --version
```

M2_HOME JAVA_HOME

Getting Started

Basic Spring Data JPA Project

- Standard JPA
- H2 embedded relational database
- JUnit tests



Integrating Spring Data REST

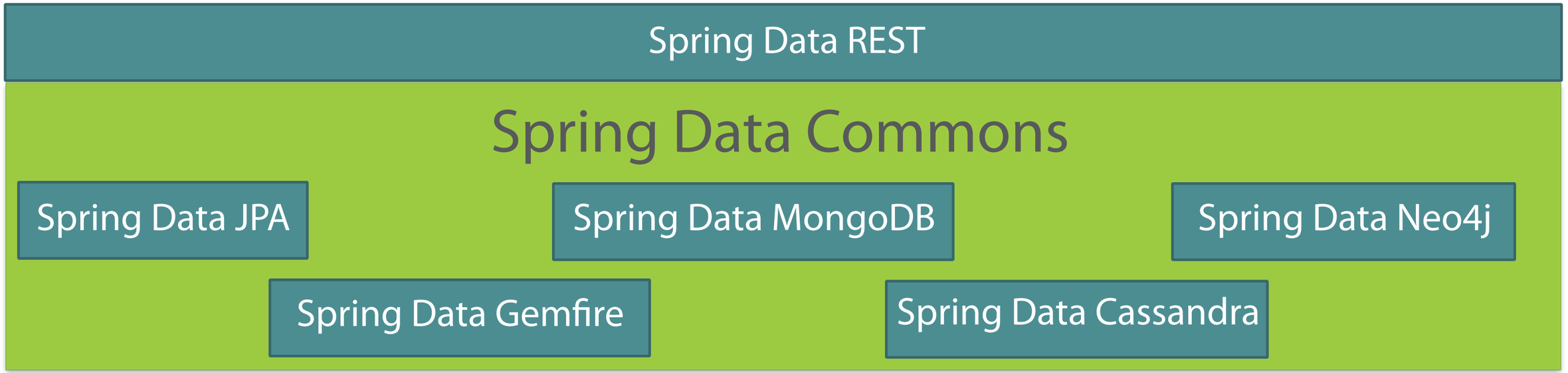
```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-rest</artif
</dependency>
```

```
<dependency>
  <groupId>org.springframework.data</groupId>
  <artifactId>spring-data-rest-webmvc</artifactId>
  <version>2.3.0.RELEASE</version>
</dependency>
```

Integrating Spring Data REST

```
@Configuration
public class CustomizedRestMvcConfiguration extends
RepositoryRestMvcConfiguration {
  @Override
  public RepositoryRestConfiguration config() {
    RepositoryRestConfiguration config = super.config();
    config.setBasePath("/api");
    return config;
  }
}
```

Supported Technologies



Summary

REST definition

Learning path and prerequisites

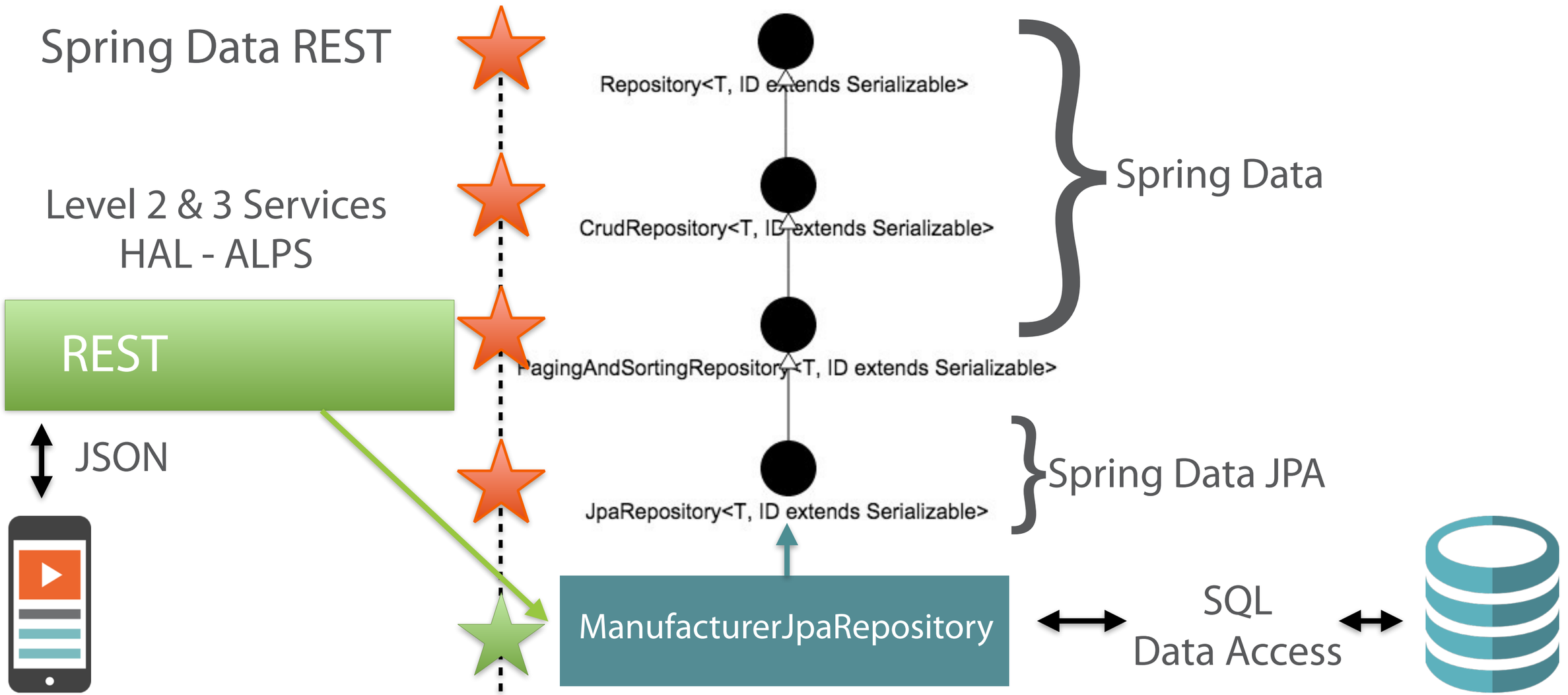
Installed our sample project

Integrated Spring Data REST

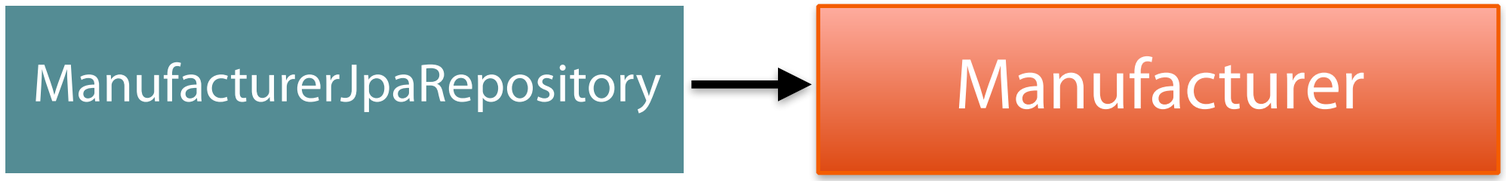
Spring Data REST's supported technologies

Making Repositories RESTful

Spring Data Repositories



Spring Data REST Resource Pattern



Pattern -> http(s)://server:port/restBaseUri/manufacturers

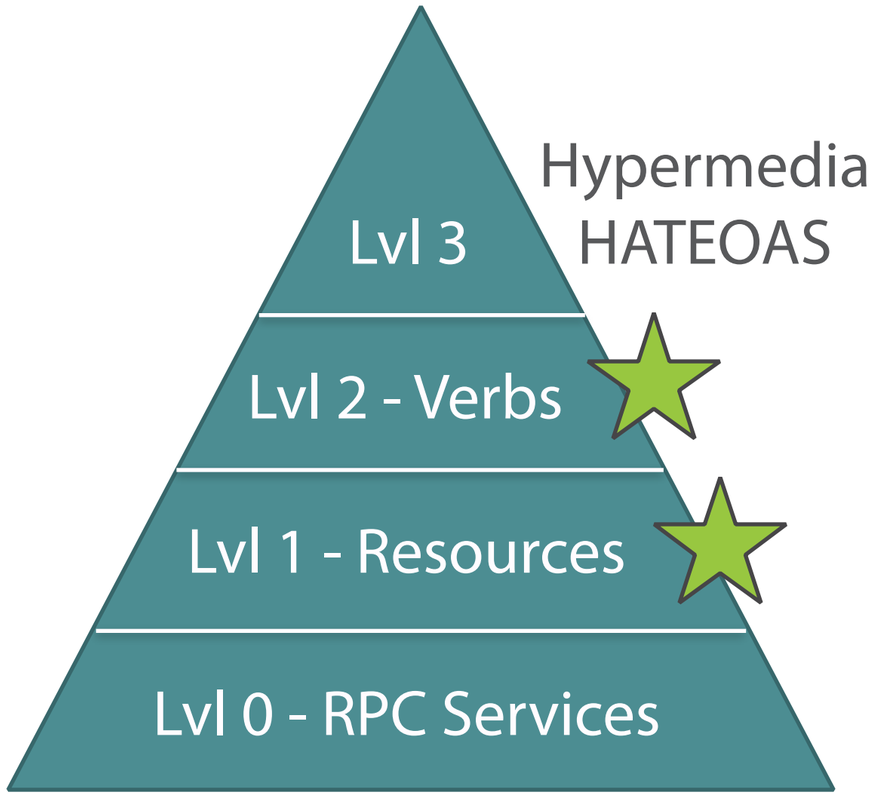
Concrete Example -> http://localhost:8080/api/manufacturers

```
@RepositoryRestResource(path="companies")
public interface ManufacturerJpaRepository extends JpaRepository<Manufacturer, Long> {
…
}
```

Concrete Example -> http://localhost:8080/api/companies

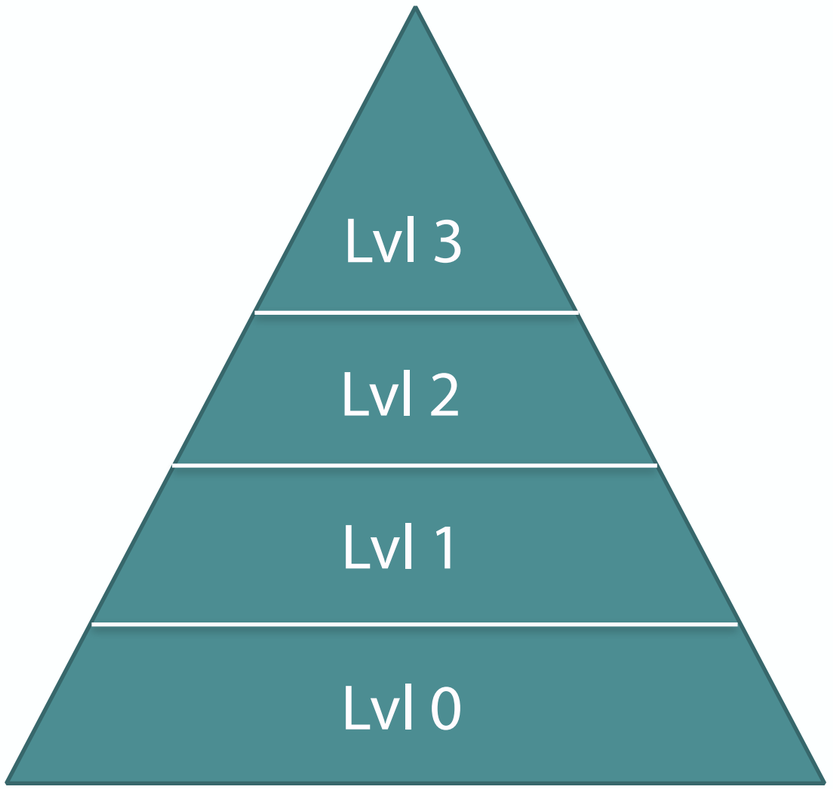
Spring Data REST Resource Pattern

http://localhost:8080/api/manufacturers



Richardson Maturity Model

HTTP Verb	URL	Backing Method
GET	/api/manufacturers	findAll(…)
POST	/api/manufacturers	save(…)
GET	/api/manufacturers/{id}	findOne(…)
PUT	/api/manufacturers/{id}	save(…)
PATCH	/api/manufacturers/{id}	save(…)
DELETE	/api/manufacturers/{id}	delete(…)



Richardson Maturity Model

REST and Spring Data repositories

Collection resources

Item resources

Customizing Spring Data REST services

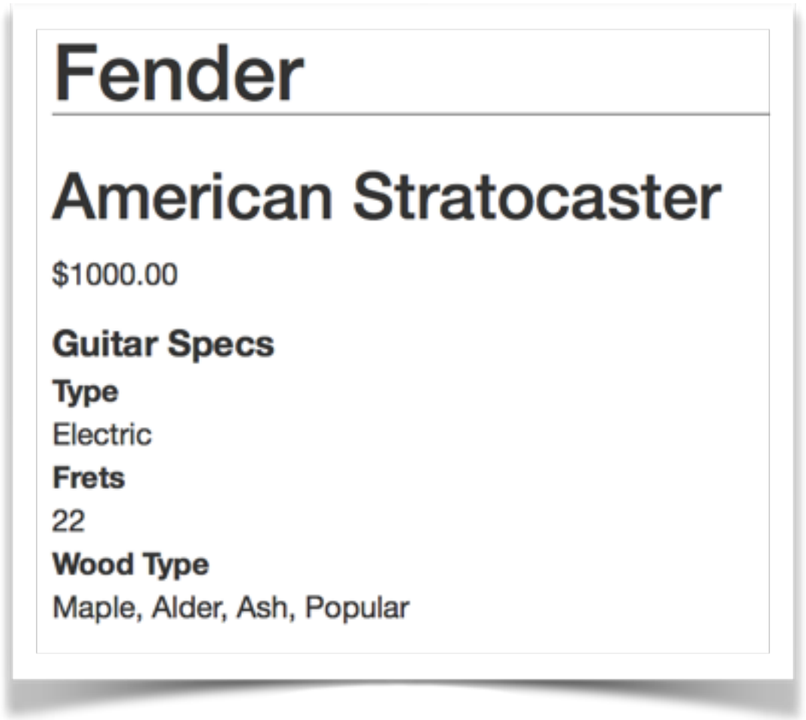
Association resources

Customizing REST Payloads

Spring Data REST Projections

```
public class Model {
	@Id
	@GeneratedValue(strategy=GenerationType.AUTO)
	private Long id;

	private String name;
	private BigDecimal price;
	private int frets;

	@Column(name="WOODTYPE")
	private String woodType;

	@Column(name="YEARFIRSTMADE")
	private Date yearFirstMade;

	@ManyToOne
	private Manufacturer manufacturer;

	@ManyToOne
	@JoinColumn(name="MODELTYPE_ID")
	private ModelType modelType;
}
```

Creating a Projection




```
@Projection(name="modelDetail", types={Model.class}
public interface ModelDetail {
    String getName();
    String getPrice();
    Manufacturer getManufacturer()
    …
}
```

Interfaces go in model or entity package or sub-package

Projections: Pros and Cons

Cons

- Read only
- Clients can ignore them
- Not a service layer

Pros

- Customizable payloads
- Buffers entity changes
- Minimal coding

Jackson Payload Options

Jackson JSON Parser

```
@Entity
public class User {
    private String username;
    @JsonIgnore
    private String password;
}
```

```
@Configuration
public class SpringDataRestConfiguration extends RepositoryRestMvcConfiguration {
    @Override
    protected void configureJacksonObjectMapper(ObjectMapper objectMapper) {
        super.configureJacksonObjectMapper(objectMapper);
        objectMapper.setPropertyNamingStrategy(new FirstLetterCapsStrategy());
        objectMapper.setDateFormat(new ISO8601DateFormat());
    }
}
```

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ],
  "children": [],
  "spouse": null
}
```


Projections

Virtual/View projections

Projection excerpts

Projection pros and cons

Jackson JSON parser

Authenticating with Spring Data REST

pom.xml

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-security</artifactId>
</dependency>
```

SecurityConfiguration.java

```
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {
    @Autowired
    public void configureGlobal(
            AuthenticationManagerBuilder auth) throws Exception {
        auth
            .inMemoryAuthentication()
                .withUser("jimi").password("hendrix").roles("USER")
                .and()
                .withUser("admin").password("admin").roles("USER","ADMIN");
    }
}
```

Client Request

```
GET http://localhost:8080/api/mfgs
Authorization: Basic YWRtaW46YWRtaW4=
```

Authorization with Spring Data REST

SecurityConfiguration.java

```
@EnableGlobalMethodSecurity(prePostEnabled = true)
```

```
@PreAuthorize("hasRole('ROLE_ADMIN')")
```

- Spring EL
- Custom Security expressions

```
@Secured("ROLE_ADMIN")
```

Spring Data REST Validation Options

JPA Constraints

```
@NotNull
@Min(10)
@Max(100)
@Size(min=2, max=20)
…
```

Validator event

BeforeCreateEvent	***AfterCreateEvent***
BeforeSaveEvent	***AfterSaveEvent***
BeforeLinkSaveEvent	***AfterLinkSaveEvent***
BeforeDeleteEvent	***AfterDeleteEvent***

- Security authorization
- Auditing

http://docs.oracle.com/javaee/6/tutorial/doc/gircz.html

REST Hypermedia in Depth

Hypermedia in a REST API

Hypermedia provides a common way to understand and self describe an API by providing metadata and allowable state transitions for the requested resource.

REST Hypermedia in Depth

HAL

Hypertext Application Language

links

- Self
- Manufacturer
- Model Type
- …

application/hal+json

ALPS

Application-Level Profile Semantics

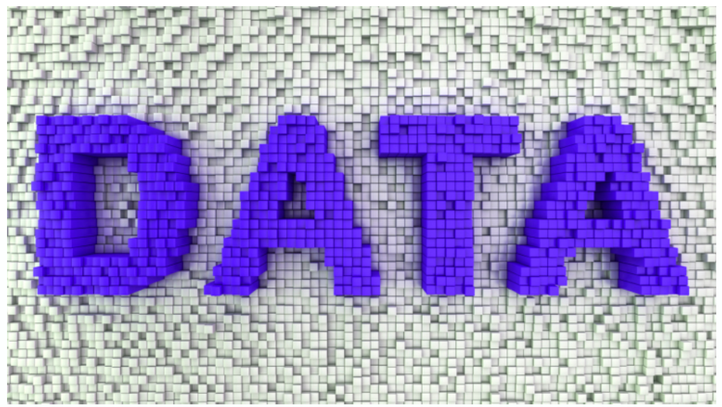
- Model representation
  - Descriptors
- RESTful transitions
  - Descriptors

application/alps+json

JSON Schema

- Model info
- Attributes
- Data types
- Links

application/schema+json



Security

Validation and events

Hypermedia