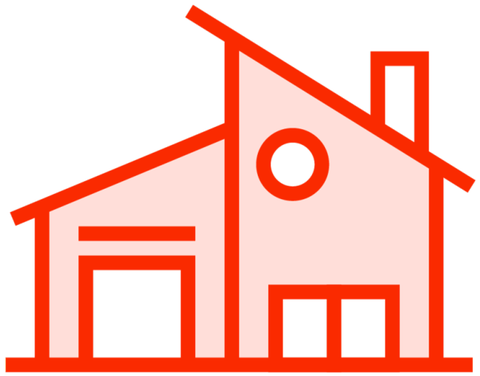
Understanding the Spring Data Query DSL

Dan Bunker
CONSULTANT | SAAS BUILDER | MENTOR
linkedin.com/in/bunkerdan

Advantages of Using a Query DSL

Spend Time on your Data Model



Reduced Codebase

Query Validity

DSL - Domain Specific Language

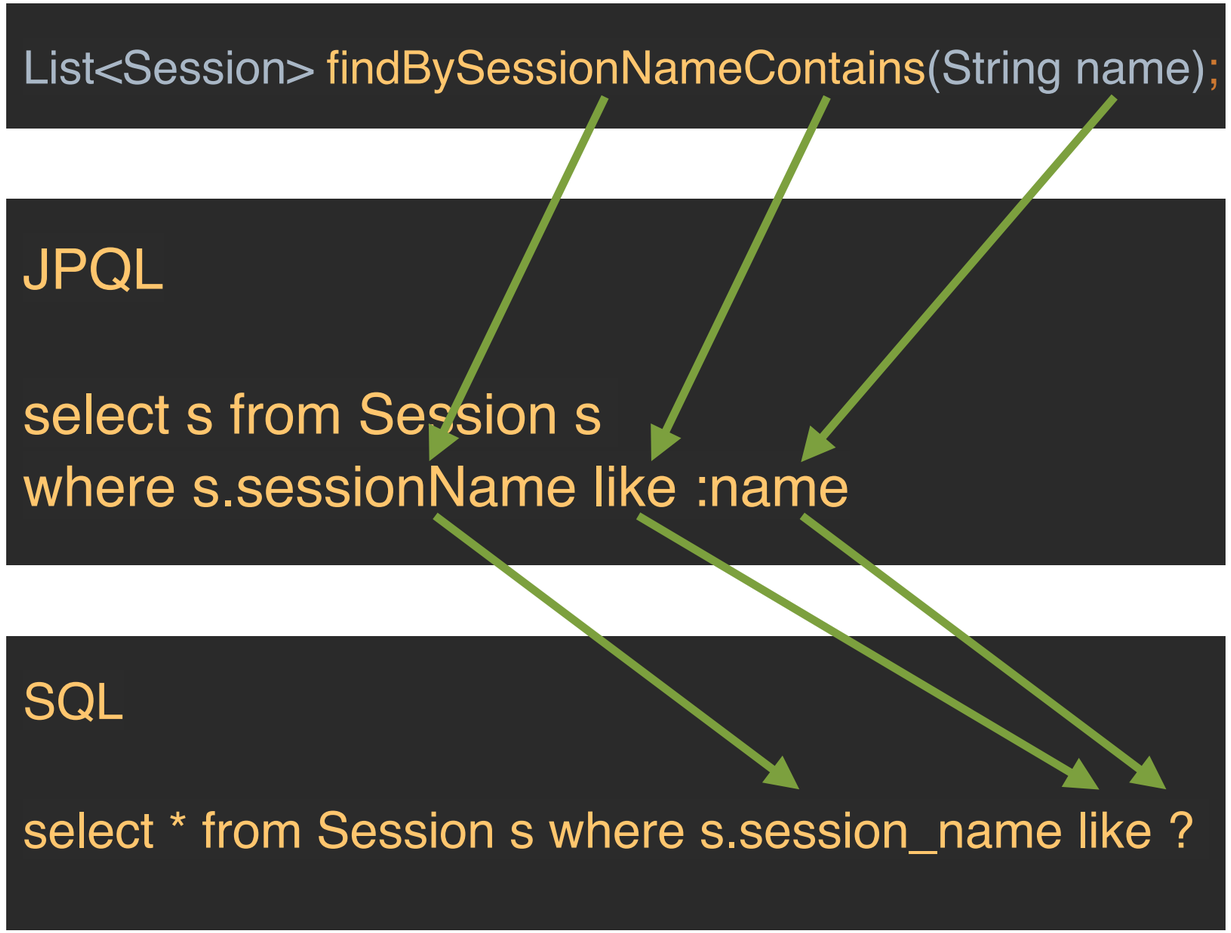
A domain specific language (DSL) is a customized extension of a software programming language that addresses a specific business or domain.

Clip 3 - Overview Placeholder

Demo: Query DSL

Query DSL Basics

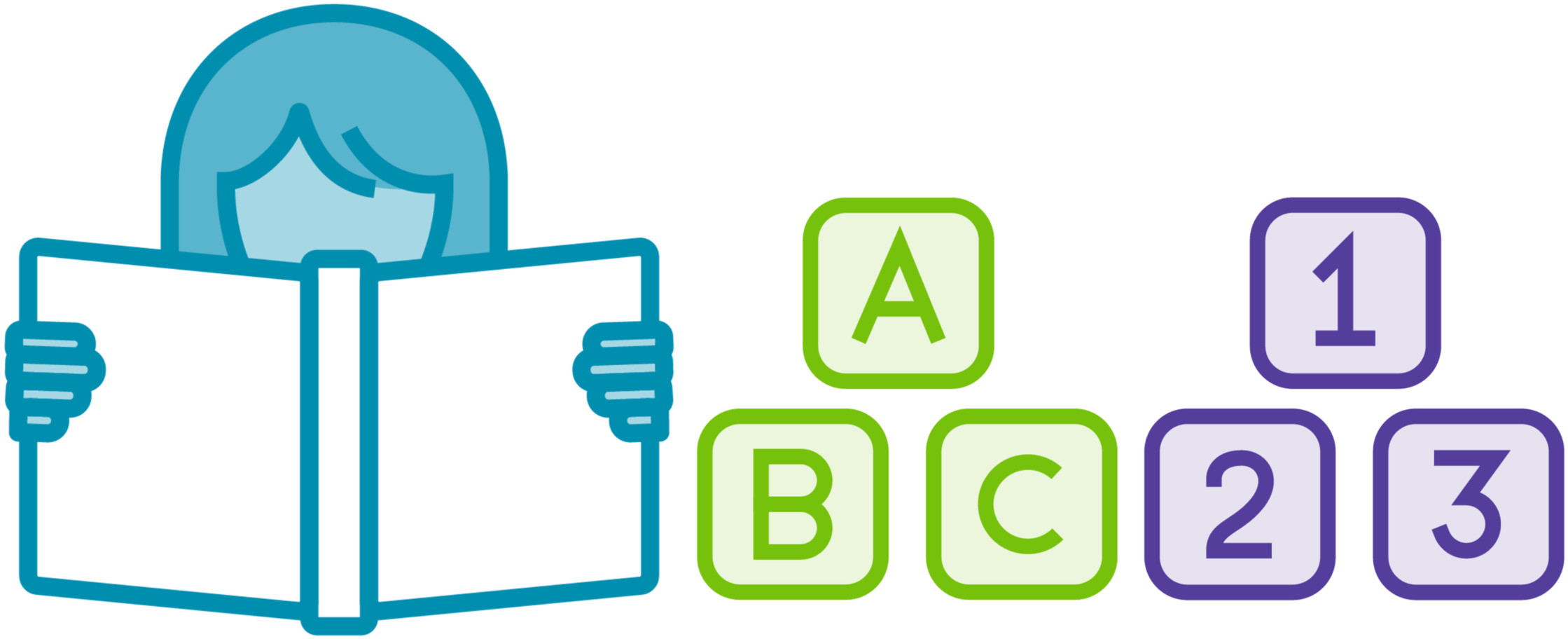
- Query DSL = Method Contracts
- Query DSL can begin with:
  - findBy, queryBy, readBy, countBy, getBy
- Query DSL uses JPA attribute names for criteria
- Multiple criteria combined with [“And”, “Or”]



Query DSL Method Return Types

```
public interface SpeakerJpaRepository extends JpaRepository<Speaker, Long> {
    List<Session> findBySessionNameContains(String name);
    Session findFirstBySessionNameContains(String name);
    Long countBySessionNameContains(String name);
}
```

Query DSL Learning Instructions



AND - OR

```
findByFirstNameAndLastName
findByFirstNameOrLastName
```

```
… where a.firstName = ?1
and a.lastName = ?2
```

```
… where a.firstName = ?1 or
a.lastName = ?2
```

Usage:

Combines multiple criteria query filters together using a conditional And or Or

◀ **Query DSL Example**

◀ **JPQL Example**

EQUALS - IS - NOT

```
findBySessionLength
findBySessionLengthIs
findBySessionLengthEquals
findBySessionLengthNot
```

```
… where a.sessionLength = ?1
… where a.sessionLength = ?1
… where a.sessionLength = ?1
… where a.sessionLength != ?1
```

Usage:

The default ‘=‘ when comparing the criteria with the filter value. Use Not when wanting to compare not equals

◀ **Query DSL Example**

◀ **JPQL Example**

LIKE - NOT LIKE

```
findBySessionNameLike("Java%")
findBySessionNameNotLike("Python%)
```

```
… where a.sessionName like ?1
… where a.sessionName not like ?1
```

Usage:

Useful when trying to match, or not match, a portion of the criteria filter value

◀ **Query DSL Example**

◀ **JPQL Example**

STARTING/ ENDING WITH - CONTAINING

```
findBySessionNameStartingWith("j"); //j%
findBySessionNameEndingWith("j"); //%j
findBySessionNameContaining("j"); //%j
```

```
… where a.sessionName like ?1
… where a.sessionName like ?1
… where a.sessionName like ?1
```

Usage:

Similar to the “Like” keyword except the % is automatically added to the filter value

- **Query DSL Example**
  Other variations: StartsWith, EndsWith, Contains, IsStartingWith, IsEndingWith, IsContaining, NotContaining, NotContains
- **JPQL Example**
  SQL is all the same, only the literal and the % placement changes

LESS THAN - GREATER THAN

```
findBySessionLengthLessThan(30);
findBySessionLengthLessThanEqual(30);
findBySessionLengthGreaterThan(30);
findBySessionLengthGreaterThanEqual(30);
```

```
… where a.sessionLength < ?1
… where a.sessionLength <= ?1
… where a.sessionLength > ?1
… where a.sessionLength >= ?1
```

Usage:

When you need to perform a <, <=, >, or >= comparison with number data types

◀ **Query DSL Example**

◀ **JPQL Example**

BEFORE - AFTER - BETWEEN

```
findByStartDateBefore(startDate);
findByStartDateAfter(startDate);
findByStartDateBetween(startDate, endDate);
```

```
… where a.startDate < ?1
… where a.startDate > ?1
… where a.startDate between ?1 and ?2
```

Usage:

When you need to perform a less than, greater than or range comparison with date/time data types

◀ **Query DSL Example**

◀ **JPQL Example**

TRUE - FALSE

```
findByIncludesWorkshopTrue
findByIncludesWorkshopFalse
```

```
… where a.includesWorkshop = true
… where a.includesWorkshop = false
```

Usage:

Useful when comparing boolean values with true or false.

◀ **Query DSL Example**

◀ **JPQL Example**

NULL - NOT NULL

```
findBySpeakerPhotoNull();
findBySpeakerPhotoIsNull();
findBySpeakerPhotoNotNull();
findBySpeakerPhotoIsNotNull();
```

```
… where a.speakerPhoto is null
… where a.speakerPhoto is null
… where a.speakerPhoto not null
… where a.speakerPhoto not null
```

Usage:

Used to check whether a criteria value is null or not null

◀ **Query DSL Example**

◀ **JPQL Example**

IN - NOT IN

```
findByCompanyIn(companies)
findByCompanyNotIn(companies)
```

```
… where a.company in ?1
… where a.company not in ?1
```

Usage:

When you need to test if a column value is part of a collection or set of values or not

◀ **Query DSL Example**

◀ **JPQL Example**

IGNORE CASE

```
findByCompanyIgnoreCase(cmpy)
findByCompanyContainsIgnoreCase(cmpy)
```

```
… where UPPER(a.company) = UPPER(?1)
… where UPPER(a.company) = UPPER(%?1%)
```

Usage:

When you need to perform a case insensitive comparison

◀ **Query DSL Example**

◀ **JPQL Example**

ORDER BY

```
findByLastNameOrderByFirstNameAsc(name)
findByLastNameOrderByFirstNameDesc(name)
```

```
… where a.lastName = ?1
  order by a.firstName asc

… where a.lastName = ?1
  order by a.firstName desc
```

Usage:

Used to setup an order by clause on your query

◀ Query DSL Example

◀ JPQL Example

FIRST - TOP - DISTINCT

```
findFirstByFirstName(name);
findTop5ByFirstName(name);
findDistinctByFirstName(name);
```

```
… where a.firstName = ?1 limit 1
… where a.firstName = ?1 limit 5
Select distinct where a.firstName = ?1
```

Usage:

Used to limit the results returned by the query

◀ Query DSL Example

◀ JPQL Example

Summary

Query DSL advantages

Basic Syntax

Keywords, keywords and keywords