

Literal Contains Search

Contents of this Page

- [Literal Contains Implementation Details](#)
 - [Algorithm:](#)
 - [Example of use:](#)
 - [Associated JUnits:](#)

Literal Contains Implementation Details

Works the same as contains but uses the literal property value enabling searches on special characters.

Algorithm:

The Literal Contains search has the following characteristics:

- This search is case in-sensitive.
- It searches on the literal property.
- A trailing wild card is added to each token in the search text.
- The literal property part (without the wild cards) of the query is boosted by 50. This gives a literal match priority.
- Parsing is done with the following analyzer:
 - `literal_propertyValue` - Uses our custom literal analyzer. This literal analyzer uses Lucene's `WhitespaceTokenizer` with Lucene's `LowerCaseFilter`.

Example of use:

The following examples are based on the Automobiles coding scheme.

Example 1:

Search string: a^s

Lucene query: `+literal_propertyValue:a^s* literal_propertyValue:a^s^50.0`

Result: 1 result

- entity code: `SpecialCharactersConcept`
- entity description: Concept containing special characters

Example 2:

Search string: a^s sp*cial co{nce]pt

Lucene query: `+(+literal_propertyValue:a^s* +literal_propertyValue:sp*cial* +literal_propertyValue:co{nce]pt*) ((+literal_propertyValue:a^s +literal_propertyValue:sp*cial +literal_propertyValue:co{nce]pt)^50.0)`

Result: 1 result

- entity code: `SpecialCharactersConcept`
- entity description: Concept containing special characters

Associated JUnits:

Junit tests can be found here: <https://github.com/lexevs/lexevs/blob/master/lbTest/src/test/java/org/LexGrid/LexBIG/Impl/function/query/lucene/searchAlgorithms/TestLiteralContains.java>