

[illegible][illegible][illegible]

com.novetta.clavin.gazetteer.query

The diagram illustrates the architecture of the LuceneGazetteer system, organized into several main sections:

- LuceneGazetteer**: Contains components like `executeQuery`, `getGeoName`, `resolveParents`, `getClosestLocations`, `buildFilters`, and `sanitizeQueryText`. These components interact with Lucene `Index` and `API` objects, and use various data sources like `logging`, `full-text`, `sif4j`, and `indexing`.
- LuceneGazetteerTest**: A collection of test classes such as `setUp`, `testBorderCases`, `testFilterDups`, `testFuzzyMode`, `testGetGeoName`, `testGetFullGeoName`, `testLoadAncestry`, `testResolveAncestry`, `testResolveLocations`, `testSanitizedInput`, and `testSanitizedOutput`. These tests use `JUnit` and `Lucene` components.
- QueryBuilder**: Contains the `build` method and various filters like `filterDups`, `location`, `maxResults`, `parents`, and `toString`. It also includes `addClause` and `getTopLevelBuilder`.
- UniqueFuzzyScoringRewrite**: Contains the `build` method and various filters like `indexing`, `index`, `lookup`, `full-text`, and `apache`.
- Gazetteer**: Contains the `Gazetteer` class and its `getGeoName` method.

The image displays 12 UML class diagrams for the `com.novetta.claavin.resolver.multipart` package, organized into four groups of three.

- MultipartLocationResolverTest** (Orange):
 - `setUpClass`: Associates `MultipartLocationResolverTest` with `MultipartLocationResolver`.
 - `parameters`: Associates `MultipartLocationResolverTest` with `Parameters`.
 - `verifyCity`: Associates `MultipartLocationResolverTest` with `City`.
 - `verifyLocation`: Associates `MultipartLocationResolverTest` with `Location`.
- MatchedLocation** (Green):
 - `getMatch`: Associates `MatchedLocation` with `Location`.
 - `getMatched`: Associates `MatchedLocation` with `MatchedLocation`.
 - `getScore`: Associates `MatchedLocation` with `Score`.
 - `toString`: Associates `MatchedLocation` with `String`.
- AdaptNlpExtractor** (Orange):
 - `apache`: Associates `AdaptNlpExtractor` with `ApacheExtractor`.
 - `client`: Associates `AdaptNlpExtractor` with `Client`.
 - `logging`: Associates `AdaptNlpExtractor` with `Logging`.
 - `json`: Associates `AdaptNlpExtractor` with `Json`.
 - `http`: Associates `AdaptNlpExtractor` with `Http`.
 - `slf4j`: Associates `AdaptNlpExtractor` with `Slf4j`.
- MultipartLocationResolver** (Orange):
 - `logging`: Associates `MultipartLocationResolver` with `Logging`.
 - `api`: Associates `MultipartLocationResolver` with `Api`.
 - `findCandidates`: Associates `MultipartLocationResolver` with `Candidates`.
 - `resolveLocation`: Associates `MultipartLocationResolver` with `Location`.
- MultipartLocationName** (Green):
 - `getCity`: Associates `MultipartLocationName` with `City`.
 - `getState`: Associates `MultipartLocationName` with `State`.
 - `getCountry`: Associates `MultipartLocationName` with `Country`.
 - `hashCode`: Associates `MultipartLocationName` with `Integer`.
- ResolvedMultipartLocation** (Green):
 - `getCity`: Associates `ResolvedMultipartLocation` with `City`.
 - `getState`: Associates `ResolvedMultipartLocation` with `State`.
 - `getCountry`: Associates `ResolvedMultipartLocation` with `Country`.
 - `hashCode`: Associates `ResolvedMultipartLocation` with `Integer`.
- MultiLevelMultipartLocationResolverTest** (Orange):
 - `setUpClass`: Associates `MultiLevelMultipartLocationResolverTest` with `MultiLevelMultipartLocationResolver`.
 - `parameters`: Associates `MultiLevelMultipartLocationResolverTest` with `Parameters`.
 - `verifyCity`: Associates `MultiLevelMultipartLocationResolverTest` with `City`.
 - `verifyLocation`: Associates `MultiLevelMultipartLocationResolverTest` with `Location`.
- DefaultScorer** (Green):
 - `score`: Associates `DefaultScorer` with `Score`.
- SearchResult** (Green):
 - `getLocation`: Associates `SearchResult` with `Location`.
 - `getScore`: Associates `SearchResult` with `Score`.
- ApacheExtractor** (Orange):
 - `testNull`: Associates `ApacheExtractor` with `TestNull`.

```

graph TD
    com[com] --> novetta[novetta]
    novetta --> clavin[clavin]
    clavin --> extractor[extractor]
    extractor --> main[.]
    extractor --> Entity[Entity]
    extractor --> LocationOccurrence[LocationOccurrence]
    extractor --> TextBody[TextBody]
    extractor --> ApacheExtractor[ApacheExtractor]
    main --> apache[apache]
    main --> client[client]
    main --> api[api]
    main --> logging[logging]
    main --> tools[tools]
    main --> json[json]
    main --> http[http]
    main --> nlp[nlp]
    main --> slf4j[slf4j]
    Entity --> entity[.]
    Entity --> entityType[getType]
    Entity --> entityLocation[getLocation]
    Entity --> entityText[getText]
    Entity --> entityLocationOccurrence[getLocationOccurrences]
    Entity --> entityTextOccurrences[getTextOccurrences]
    Entity --> entitySetText[setText]
    Entity --> entitySetType[setType]
    LocationOccurrence --> locationOccurrence[.]
    LocationOccurrence --> locationOccurrenceTest[testEquals]
    LocationOccurrence --> locationOccurrenceTest[testHashCode]
    TextBody --> textBody[.]
    TextBody --> textBodyTest[testEquals]
    TextBody --> textBodyTest[testHashCode]
    ApacheExtractor --> apacheExtractor[.]
    ApacheExtractor --> apacheExtractorTest[ApacheExtractorTest]
    ApacheExtractor --> apacheExtractorTest[ApacheExtractorTest]
  
```

com.novetta.clavin.extractor

AdaptNlpExtractor

extractLocationNames

Entity

ApacheExtractor

LocationOccurrenceTest

TextBody

ApacheExtractorTest

AdaptNlpExtractorTest

LocationOccurrences

```
graph TD
    subgraph com_novetta_clavin_util [com.novetta.clavin.util]
        DamerauLevenshteinTest
        ListUtilsTest
    end
    subgraph com_novetta_clavin [com.novetta.clavin]
        DamerauLevenshtein
        InfiniteCharArray
        Null
        TestUtilsTest
        ListUtils
        TextUtils
    end
```

The diagram illustrates the structure of the `com.novetta.clavin.util` module. It is organized into two main sections: `com.novetta.clavin.util` and `com.novetta.clavin`.

com.novetta.clavin.util contains the following components:

- DamerauLevenshteinTest**: Includes `testDistance` and `testEditDistance`.
- ListUtilsTest**: Includes `testChunkifyList` and `testZeroChunk`.

com.novetta.clavin contains the following components:

- DamerauLevenshtein**: Includes `testDistance`, `testEditDistance`, `testLevenshteinDistance`, and `testLevenshteinDistanceWithWeights`.
- InfiniteCharArray**: Includes `testGet`, `testSet`, and `testToString`.
- Null**: Includes `testIsNull`, `testIsNotNull`, `testIsNullOrEmpty`, and `testIsNotNullOrEmpty`.
- TextUtilsTest**: Includes `testTrimming` and `testTrimmingWithLength`.
- ListUtils**: Includes `testGet` and `testSet`.
- TextUtils**: Includes `testTrimming` and `testTrimmingWithLength`.