Constructing queries

First of create a query object

```
SolutionQuery myQuery = session.getSolutionQuery("mynewsolution");
// or (as of version 11230)
SolutionQuery myQuery = session.query("mynewsolution");
```

A query is built much like an SQL query by appending different fields to be SELECT'ed

```
myQuery.addSelectField("MYNUMBER");
myQuery.addSelectField("MYOTHERNUMBER");
// or (as of version 9336)
myQuery.addSelectField("MYNUMBER", "MYOTHERNUMBER");
// or (as of version 11230)
myQuery.select("MYNUMBER", "MYOTHERNUMBER");
```

Additionally WHERE components are added

```
myQuery.addWhereCriterion("StatusID", "In progress");
myQuery.addWhereCriterion("INCOME", QueryPart.MORE, 22);
myQuery.addWhereInList("PARENTITEMS", arrayListOfDataID);
// or (as of version 7347)
myQuery
  .addWhereCriterion("StatusID", "In progress")
  .addWhereCriterion("INCOME", QueryPart.MORE, 22)
  .addWhereInList("PARENTITEMS", arrayListOfDataID);
// or (as of version 11218)
myQuery
  .where("StatusID", "In progress")
  .where("INCOME", QueryPart.MORE, 22)
  .whereIn("PARENTITEMS", arrayListOfDataID);
```

Likewise operators can be added

```
myQuery.addWhereOR();


myQuery.addWhereAND();
```

```
// or (as of version 11218)
myQuery.or();


myQuery.and();
```

Note that lookup values etc. will be translated silently.

```
myQuery.setSortOrder("INCOME", true);
// or (as of version 11218)
myQuery.sortBy("INCOME", true);
```

Order records by income in descending order (true).

By default the API limits solution queries to 100 rows. To increase this:

```
myQuery.setLimit(100000);
// or (as of version 11218)
myQuery.limit(100000);
```

To do pagination use this:

```
myQuery.setLimitAndOffset(100, 100);
// or (as of version 11218)
myQuery.limitAndOffset(100, 100);
```

Finally execute the query and return the results

```
SolutionQueryResultSet records = myQuery.executeQuery();
// or (as of version 11218)
SolutionQueryResultSet records = myQuery.execute();
```

SolutionQuery supports method chaining, so the following two samples are equivalent.

```
SolutionQuery sq = session.getSolutionQuery("cars");
sq.addSelectField("MANUFACTURER", "MODEL");
sq.addWhereCriterion("PRICE", QueryPart.MORE, 250000);
sq.addWhereOR();
sq.addWhereCriterion("ONSALE", true);
sq.setSortOrderAsc("PRICE");
SolutionQueryResultSet rs = sq.executeQuery();
// or (as of version 7347)
```

```
SolutionQueryResultSet rs = session.getSolutionQuery("cars")
  .addSelectField("MANUFACTURER", "MODEL")
  .addWhereCriterion("PRICE", QueryPart.MORE, 250000)
  .addWhereOR()
  .addWhereCriterion("ONSALE", true)
  .setSortOrderAsc("PRICE")
  .executeQuery();
// or (as of version 11218)
SolutionQueryResultSet rs = session.query("cars")
  .select("MANUFACTURER", "MODEL")
  .where("PRICE", QueryPart.MORE, 250000)
  .or()
  .where("ONSALE", true)
  .sortByAsc("PRICE")
  .execute();
```

Revision #4
Created 25 March 2025 14:56:58 by Theis Villumsen
Updated 11 November 2025 12:12:07 by Theis Villumsen