Spring Professional Exam Tutorial v5.0

Question 04

Question 04 - What is a callback? What are the three JdbcTemplate callback interfaces that can be used with queries? What is each used for?

A callback is a code or reference to the code that can be passed as an argument to the method. This method will execute passed callback during execution.

On Java level callback can be:

- Class that implements interface
- Anonymous class
- Lambda expression – JDK 8
- Reference Method – JDK 8

Question 04 - What is a callback? What are the three JdbcTemplate callback interfaces that can be used with queries? What is each used for?

Jdbc Template Callbacks that can be used with queries:

- `RowMapper` - interface for processing `ResultSet` data on per-row basis, implementation should call `ResultSet.get*(..)` methods, but should not call `ResultSet.next()`, it should only extract values from current row and based on those values should create object, which will be returned from `mapRow` method, implementation is usually stateless
- `RowCallbackHandler` - interface for processing `ResultSet` data on a per-row basis, implementation should call `ResultSet.get*(..)` methods, but should not call `ResultSet.next()`, it should only extract values from current row, implementation is usually stateful, it keeps accumulated data in some object, `processRow` method from this class does not return any value, instead method saves results into for example object field that will keep state
- `ResultSetExtractor` - interface for processing entire `ResultSet` data, all rows needs to be processed and implementation should call `ResultSet.next()` method to move between rows, implementation is usually stateless, implementation should not close `ResultSet`, it will be closed by Jdbc Template

Question 04 - What is a callback? What are the three JdbcTemplate callback interfaces that can be used with queries? What is each used for?

Jdbc Template other Callbacks:

- `PreparedStatementCreator` - should create `PreparedStatement` based on `Connection` provided by `JdbcTemplate`, implementation should provide SQL and parameters
- `PreparedStatementSetter` - should set values on `PreparedStatement` provided by `JdbcTemplate`, implementation should only set parameters, SQL will be set by `JdbcTemplate`
- `CallableStatementCreator` - should create `CallableStatement` based on `Connection` provided by `JdbcTemplate`, implementation should provide SQL and parameters
- `PreparedStatementCallback` - used `internally` by `JdbcTemplate` - generic interface allowing number of operations on single `PreparedStatement`
- `CallableStatementCallback` - used `internally` by `JdbcTemplate` - generic interface allowing number of operations on single `CallableStatement`