JDBC RowSet Maintenance Release 1.1

Description:
Maintenance review of JDBC RowSet 1.0

Maintenance Lead:
Lance Andersen, Oracle Corporation

Feedback:
Comments should be sent to jsr114-comments@jcp.org

Rationale for Changes:
The goal is to address a several specification issues and to provide a factory for RowSet initialization

Accepted Changes:

1. CachedRowSet.COMMIT_ON_ACCEPT_CHANGES is now deprecated. This field is final therefore its value could not be changed.

2. The following fields in javax.sql.rowset.spi.SyncFactory have been made final:
    1. ROWSET_SYNC_PROVIDER
    2. ROWSET_SYNC_PROVIDER_VERSION
    3. ROWSET_SYNC_VENDOR

3. Clarify that javax.sql.rowset.spi.SyncFactory.getInstance(String providerID) will throw a SyncFactoryException if the providerID is null.

4. The following javax.sql.rowset.spi.SyncFactory methods require a SQLPermission which grants the permission setSyncFactory in order to succeed:
    1. setJNDIContext(javax.naming.Context)
    2. setLogger(java.util.logging.Logger)
    3. setLogger(java.util.logging.Logger, java.util.logging.Level)

    If a SecurityManager exists and the checkPermission method denies calling these methods, a SecurityPermission Exception will be thrown.

5. The following fields in javax.sql.rowset.spi.SyncProvider have been made final:
    1. DATASOURCE_DB_LOCK

2. DATASOURCE_NO_LOCK
3. DATASOURCE_ROW_LOCK
4. DATASOURCE_TABLE_LOCK
5. GRADE_CHECK_ALL_AT_COMMIT
6. GRADE_CHECK_MODIFIED_AT_COMMIT
7. GRADE_LOCK_WHEN_LOADED
8. GRADE_LOCK_WHEN_MODIFIED
9. GRADE_NONE
10. NONUPDATEABLE_VIEW_SYNC
11. UPDATABLE_VIEW_SYNC

6. Add the RowSetFactory interface that defines the implementation of a factory that is used to obtain different types of RowSet implementations. The new methods are:

    1. javax.sql.rowset.CachedRowSet createCachedRowSet() throws SQLException

        Creates a new instance of a CachedRowSet
        Returns: A new CachedRowSet instance
        Throws: SQLException if a CachedRowSet cannot be created.

    2. javax.sql.rowset.FilteredRowSet createFilteredRowSet() throws SQLException

        Creates a new instance of a FilteredRowSet
        Returns: A new FilteredRowSet instance
        Throws: SQLException if a FilteredRowSet cannot be created.

    3. javax.sql.rowset.JdbcRowSet createJdbcRowSet() throws SQLException

        Creates a new instance of a JdbcRowSet
        Returns: A new JdbcRowSet instance
        Throws: SQLException if a CachedRowSet cannot be created.

    4. javax.sql.rowset.JoinRowSet createJoinRowSet() throws SQLException

        Creates a new instance of a CachedRowSet
        Returns: A new JoinRowSet instance
        Throws: SQLException if a JoinRowSet cannot be created.

5. javax.sql.rowset.WebRowSet createWebRowSet() throws SQLException

   Creates a new instance of a WebRowSet
   Returns: A new WebRowSet instance
   Throws: SQLException if a WebRowSet cannot be created.

7. Add the javax.sql.rowset.RowSetProvider class that is a factory API that enables applications to obtain a `RowSetFactory` implementation that can be used to create different types of `RowSet` implementations. The following methods are provided by RowSetProvider:

   1. javax.sql.rowset.RowSetFactory newFactory() throws SQLException:

      Creates a new instance of a `RowSetFactory` implementation. This method uses the following look up order to determine the `RowSetFactory` implementation class to load:

- The System property `javax.sql.rowset.RowsetFactory`. For example:
  - -Djavax.sql.rowset.RowsetFactory=com.sun.rowset.RowSetFactoryImpl
- The ServiceLocator API. The ServiceLocator API will look for a classname in the file `META-INF/services/javax.sql.rowset.RowSetFactory` in jars available to the runtime. For example, to have the the RowSetFactory implementation `com.sun.rowset.RowSetFactoryImpl` loaded, the entry in `META-INF/services/javax.sql.rowset.RowSetFactory` would be:
  - `com.sun.rowset.RowSetFactoryImpl`
- Platform default `RowSetFactory` instance.

      Once an application has obtained a reference to a `RowSetFactory`, it can use the factory to obtain RowSet instances.

      Returns: New instance of a `RowSetFactory`
      Throws: SQLException - if the default factory class cannot be loaded, instantiated. The cause will be set to actual Exception

   2. javax.sql.rowset.RowSetFactory newFactory(`String`

```
          factoryClassName,java.lang.ClassLoader cl)
throws SQLException:
```

Creates a new instance of a `RowSetFactory` from the specified factory class name. This function is useful when there are multiple providers in the classpath. It gives more control to the application as it can specify which provider should be loaded.

Once an application has obtained a reference to a `RowSetFactory` it can use the factory to obtain RowSet instances.

Parameters:
`factoryClassName` - fully qualified factory class name that provides an implementation of `javax.sql.rowset.RowSetFactory`.
`cl` - `ClassLoader` used to load the factory class. If `null` current `Thread`'s context classLoader is used to load the factory class.

Returns: New instance of a `RowSetFactory`
Throws: SQLException - if `factoryClassName` is `null`, or the factory class cannot be loaded, instantiated.