34

# Oracle JDBC Support for FAN Events

Starting from Oracle Database 12*c* Release 2 (12.2.0.1), Oracle JDBC driver supports Oracle RAC Fast Application Notification (FAN) events, for planned and unplanned outages. This facilitates third-party connection pools to leverage Oracle RAC features for high availability. Java applications not using Oracle Universal Connection Pool (UCP) or WebLogic Server can now leverage on this support. For example, scenarios like rolling upgrades at the Oracle RAC server-side do not cause JDBC errors within applications.

> **Note:**
>
> Although the Oracle JDBC drivers now support the FAN events, Oracle UCP provides more comprehensive support for all FAN events.

> **See Also:**
>
> *Oracle Universal Connection Pool Developer's Guide*



## 34.1 Overview of Oracle JDBC Support for FAN events

You must use an Oracle RAC Database or an Oracle Restart on a single instance database to use this feature.

This feature supports:

- Planned maintenance

  This case deals with planned maintenance on Oracle RAC servers, where an Oracle RAC service can be gracefully shutdown. In this case, borrowed or in-use connections from a connection pool are not interrupted, and are closed only until any safe-draining API is invoked. For example, when an application completes work on such a connection and returns it to the connection pool.

- Unplanned outages

  In this case, dead connections are rapidly detected and terminated, so that the network connections to the server do not become nonresponsive.. In this case, borrowed and in-use connections are interrupted during unplanned outages. Applications are expected to handle any exception on affected connections and perform necessary recovery, either on their own, or using Oracle high-availability solutions such as Application Continuity.

**Related Topics**

- Application Continuity for Java
  The outages of the underlying software, hardware, communications, and storage layers can cause application execution to fail. In the worst cases, the middle-tier servers may need to be restarted to deal with a logon storm, which is a sudden increase in the number of client connection requests.

**Related Topics**

- Safe Draining APIs for Planned Maintenance

> **See Also:**
>
> *Oracle Real Application Clusters Administration and Deployment Guide*for server-side configuration about using Oracle RAC. This chapter describes only the client-side configuration steps that an application must perform when using Oracle JDBC driver support for FAN events.

# 34.2 Safe Draining APIs for Planned Maintenance

For planned Oracle RAC maintenance, the JDBC driver supports a list of safe-draining APIs, which are required for additional handshake or integration work with a third-party Java connection pool. These APIs serve as the *draining-points*, where the driver can safely close any connection affected by a planned maintenance, without causing application-visible errors. Following is the list of safe-draining APIs that driver FAN supports:

- `java.sql.Connection.isValid(int timeout)`
- `oracle.jdbc.OracleConnection.pingDatabase()`
- `oracle.jdbc.OracleConnection.pingDatabase(int timeout)`
- `oracle.jdbc.OracleConnection.endRequest()`
- All standard JDBC and Oracle JDBC extension `EXECUTE***` calls on `Statement`, `PreparedStatement`, and `CallableStatement` interfaces

For the standard JDBC and Oracle JDBC extension `EXECUTE***` calls, the executed SQL command string must contain the following SQL hint as the first noncomment token within the SQL string:

```
/*+ CLIENT_CONNECTION_VALIDATION */
```

Qualified SQLs are treated as connection-validation SQLs. For example:

```
/*+ CLIENT_CONNECTION_VALIDATION */ SELECT 1 FROM DUAL
```

Typically, a third-party connection pool places calls to these APIs. It is expected that on detection of any bad connection with such invocations, a third-party connection pool closes and removes the related connection from the pool, so that no errors are visible to applications. When the application itself calls these APIs, then it is expected that application is actively validating the underlying connection and will close and remove any bad connection detected.

# 34.3 Installation and Configuration of Oracle JDBC Driver for FAN Events Support

Oracle JDBC driver automatically determines whether to enable Oracle JDBC support for FAN events, by checking the database server that it connects to, and whether the following necessary JAR files are available in the application environment, in addition to the Oracle JDBC driver or not:

- The `simplefan.jar` and `ons.jar` files

  You must install the `simplefan.jar` and `ons.jar` files from the JDBC and UCP Downloads page and include them in the `CLASSPATH`.

  If either one is missing, or the driver is unable to load it, then this feature is disabled. When used with a third-party connection pool, these JAR files must be placed in the same location, where the connection pool retrieves and loads the driver JAR files.

- Oracle JDBC data sources

  You can use the same typical Oracle JDBC data sources, such as `oracle.jdbc.pool.OracleDataSource` or `oracle.jdbc.OracleDriver` for obtaining JDBC connections. When used together with a third-party connection pool, your application must specify these classes as connection factories for the connection pool.

Applications that want to explicitly disable this feature, can set the `oracle.jdbc.fanEnabled` property to `FALSE`. This property is available as both a system property and a connection property. For applications using Universal Connection Pool (UCP) or WebLogic Server Active GridLink (AGL), this property is set to `FALSE` by default. Otherwise, the default value is `TRUE`.

> **Note:**
>
> - When the JDBC driver automatically enables support for FAN events, with both the `simplefan.jar` and the `ons.jar` files present on the `CLASSPATH`, then calling the `getConnection` method may throw an exception, such as, `java.lang.IllegalArgumentException`. To avoid this, you can perform either of the following:
>   - Remove either `simplefan.jar` or `ons.jar` from the `CLASSPATH`.
>   - Set the `oracle.jdbc.fanEnabled` property to `FALSE` to disable this feature explicitly.
> - Setting the `oracle.jdbc.fanEnabled` property to `TRUE` may not enable Oracle JDBC Support for FAN Events feature as the feature depends on other factors too.

The JDBC driver requires minimal configuration changes or code changes to a third-party connection pool for supporting Oracle FAN events. For a connection pool that does not need any configuration change or code change, it is assumed that it fulfills one of the following criteria:

- The pool has a configuration option for validating JDBC connections at pool checkout time.

- The pool uses `javax.sql.PooledConnection` and has a configuration option for plugging in a `javax.sql.ConnectionPoolDataSource` implementation. Such a connection pool is also assumed to be able to check for closed or bad physical connections at connection returns.

Following are a few connection validation options on some third-party Java connection pools. The majority of these options are based on SQL, and not on validation APIs:

| Java Connection Pool | Connection Validation Options |
|---|---|
| Oracle WebLogic Generic and MDS data sources | `TestConnectionsOnReserve`, `TestConnectionsOnRelease`, `TestConnectionsOnCreate` |
| IBM WebSphere | `PreTest` Connection |
| RedHat JBoss | `check-valid-connection-sql` |
| Apache TomCat | `TestonBorrow`, `TestonRelease` |

When you use Oracle JDBC support for FAN events feature with an Oracle RAC server Release 11*g*, then applications must explicitly set the remote ONS configuration string for Oracle RAC FAN through the `oracle.jdbc.fanONSConfig` system property. The value and the format of the property are the same as for UCP Fast Connection Failover (FCF).

> **See Also:**
>
> Universal Connection Pool Developer's Guide

# 34.4 Example of Oracle JDBC Driver FAN support for Planned Maintenance

The section discusses how you can typically enable and use JDBC Oracle FAN support with planned maintenance on Oracle RAC.

Applications should not receive any exception during a planned maintenance after following these instructions:

1. Upgrade Oracle JDBC driver to Release 23ai to use the `ojdbc11.jar` or `ojdbc17.jar` file.
2. Install and use the 23ai version of the `ons.jar` and `simplefan.jar` files.
3. Use the `oracle.jdbc.pool.OracleDataSource` class to obtain physical connections or configure this class as the connection factory on a third-party Java connection pool. In the latter case, you must set the specific pool property that enables connection validation.

   Optionally, when running against Oracle RAC Release 21c, specify the system property `oracle.jdbc.fanONSConfig` to configure remote ONS.
4. The application runs until you are ready to perform planned maintenance activities like a rolling upgrade on the Oracle RAC. During the planned maintenance, for each service based on usage patterns, a DBA will perform the following activities using the extended `srvctl` interface:
   - Relocate or stop the services on the next instance to upgrade, with no `-f` (force)
   - Wait until all connections to this service are drained by driver FAN

- When the timeout is reached, disconnect the sessions with the defined stop mode (transactional is recommended)
- When all services are relocated or stopped, shutdown the instance and apply the upgrade or patch
- Restart the instance and restart the services if they were stopped
- Iterate until all instances are upgraded/patched

# 34.5 Using Third-Party Connection Pools with Oracle JDBC

This section lists the requirements to configure third-party connection pools for supporting planned and unplanned outages.

For configuring the third-party connection pools, ensure that you:

- Enable High Availability (HA) support in the driver
- Enforce connection validation while borrowing the connection
- Tune the connection validation technique to limit the cost
- Enable statement caching in the driver

With the preceding settings, the connections in the third-party connection pools use the draining APIs during a planned down event, for example, when a service is stopped on an instance. For handling unplanned outages, you must also enable Transparent Application Continuity (TAC) on the database service and configure one of the following replay data sources:

- `oracle.jdbc.datasource.impl.OracleDataSource` with Oracle Databases 23ai Release or Oracle Database 21c Release
- `oracle.jdbc.replay.OracleDataSource` with Oracle Database 19c Release

**Example**

For example, if you want to support planned outages with HikariCP, then you must take care of the following settings:

- Ensure HA support:
  Include the `ons.jar` and `simplefan.jar` files in the classpath
- Ensure connection validation:

  ```
  com.zaxxer.hikari.aliveBypassWindowMs=-1
  ```

- Tune the connection validation technique:

  ```
  oracle.jdbc.defaultConnectionValidation=LOCAL
  ```

- Enable statement caching in the driver:

  ```
  oracle.jdbc.implicitStatementCacheSize=50
  ```