19-MAY-2026

APEX in the Green Zone: Zero-Downtime Deployments for Oracle APEX applications

Blue-Green deployments for instant, more seamless APEX rollouts

Maurice Wilhelm

code of change

Hyand

About me

Maurice Wilhelm

Oracle APEX Consultant @Hyand



8+ years of experience with Oracle technologies

APEX enthusiast since the beginning

Focus on DevOps, CI/CD and automation

Our key facts

Hyand





Germany

- Brunswick
- Ratingen
- Hamburg
- Dortmund
- Frankfurt
- Munich
- Berlin

Poland

- Wrocław

Lithuania

- Vilnius
- Kaunas

Romania

- Cluj-Napoca

India

- Pune

25+
Customers with > 1 billion € revenue

110+
million € turnover (2024)

750+
Employees

40
APEX devs

6
Oracle ACEs

What this talk will uncover...

- Why are sophisticated deployment strategies important?
- What are Zero-Downtime Deployments?
- Basic concepts of Blue-Green Deployments
- How does this work in a real-life app landscape?
- What are the advantages and disadvantages of this approach?
- How can the Oracle APEX PL/SQL APIs be used to your advantage?
- Looking ahead: when it is useful? Alternatives?

Blue-Green Deployments

Hyand

Simplify Rollouts - Minimize Downtime:
Blue-Green Deployments as a Zero-Downtime Strategy

"Blue-Green deployments offer a robust way to ensure **continuity** and **reliability** by providing an easy transition path between **old and new versions**, making it ideal for applications that require high availability."

Source: Sharma et al., *International Journal of Computer Applications*, Vol. 186, No. 50, 2024, p. 2

"Blue-green deployment is most effective when used for a shift in a **major version** because it creates a **testing phase** in a production environment before going live."

Source: *International Journal of Multidisciplinary on Science and Management*, Vol. 1(2), 2023, p. 22

Blue-Green Deployment approach uses two different environments (I)

- Two different environments: „blue“ (active), „green“ (new)
- Several goals:
  - minimizing downtime for users
  - simple deployments and rollbacks
  - suitable for automation
- Deployment and „switch“ separated
  - check if deployed changes are applied correctly
  - run (non-) functional tests (e.g., via Web UI, utPLSQL)
  - no users affected until version „switched“

Blue-Green Deployment approach uses two different environments (II)

- Two environments: two database schemas
- Failed deployment: no rollback necessary
  - Drop database schema, tablespace, …
  - Delete APEX app
  - Remove ORDS modules
- DDL: deployments from scratch - no custom install script necessary
- DML: exported from DEV, or table data synced in 2 ways:
  - per default: schemaA.tableA.columnA = schemaB.tableA.columnA, …
  - custom: ship your own SQL or PL/SQL code for a specific table
  - defined in 3 „deployment sync tables“

Store table deployment metadata in app schema

- deployment metadata table
- Filled by developer on DEV, exported to git

TABLE_NAME	DML_ENABLED	COPY_FROM_OLD_SCHEMA_ENABLED	PK_SEQUENCE_NAME	SEQUENCE_NO
TABLE_A	Y	N	-	1
TABLE_B	N	Y	TABLE_A_SEQ	3
TABLE_C	N	Y	TABLE_B_SEQ	2
...	...	...	...	...

* table constraints need to be disabled during copy process

Store custom sync scripts in a separate table in app schema

- custom sync scripts table
- Filled by developer on DEV, exported to git

SCRIPT_ID	CUSTOM_SCRIPT
1	INSERT /*+ APPEND */ INTO :tgt.TABLE_A(COL_A) SELECT COL_A FROM :src.TABLE_A

2	BEGIN
	  IF ... THEN ...
	END;

...	...

- SQL or PL/SQL
- :src and :tgt as placeholders for old and new schema
- Executed scripts are stored in a separate table by its SCRIPT_ID

Store release metadata and scripts inside deployment schema

- use custom deployment schema („data dictionary“)
- user should have necessary DBA grants, roles APEX_ADMINISTRATOR_ROLE and ORDS_ADMINISTRATOR_ROLE
- deployment logics (e.g., packages)
- metadata tables for release information (APEX app id, schema name, release date, …)

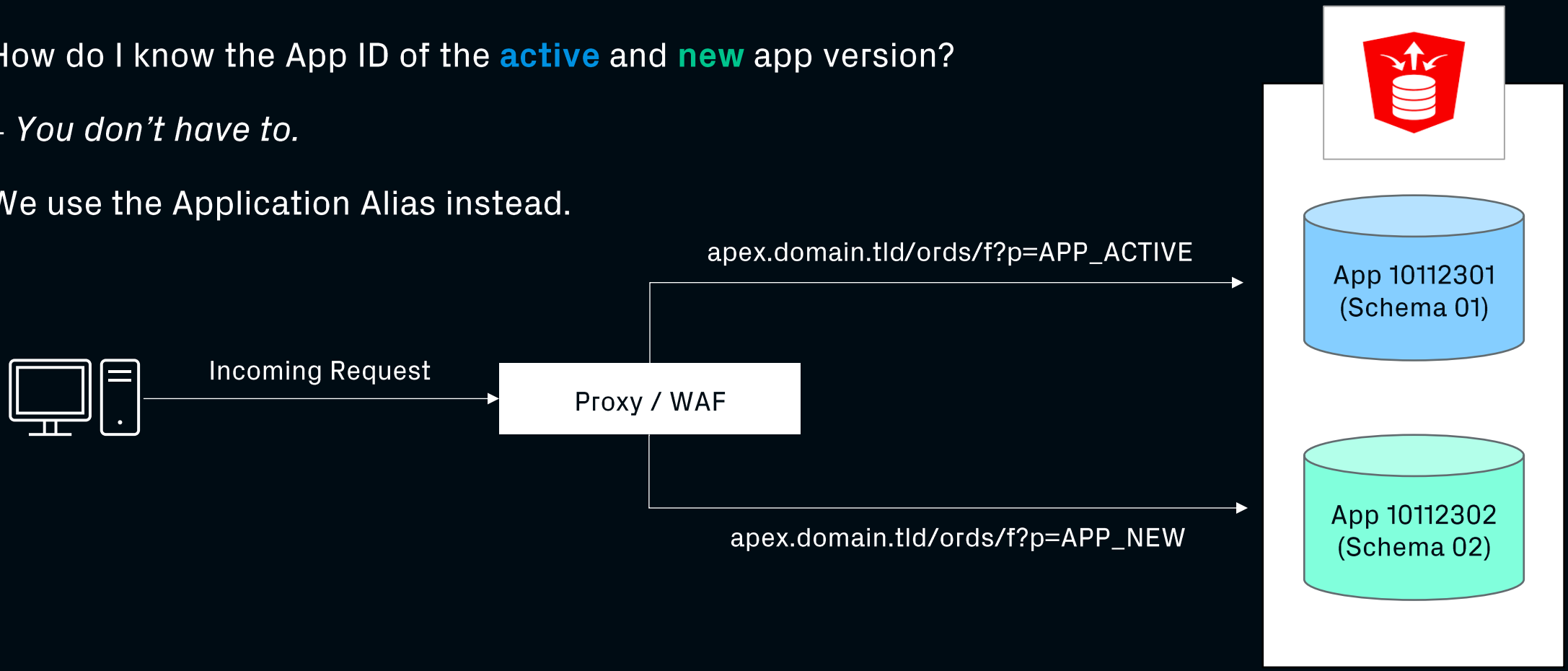
APEX_APP_ID	SCHEMA_NAME	APP_ID	STATUS	RELEASE_DATE	…
10112301	APP_SCHEMA_01	1	ACTIVE	2026/03/11 11:25	…
10112302	APP_SCHEMA_02	1	NEW	2026/05/08 13:25	…
…	…	…	…	…	…

Enabling reliable app routing through alias-based identification

How do I know the App ID of the **active** and **new** app version?

- *You don't have to.*

We use the Application Alias instead.



Process steps of deployments are separated in job stages (I)

	Declarative: Checkout SCM	init	pull deployment scripts	sync files	Export environment config	data enrichment	check existence	register customer in CCDM	create tablespace and user	grants and synonyms	Load java libs	Create java classes	Create APEX workspace	Create APEX workspace user (DEV)	Import APEX static workspace files	APEX admin configure	Adjust APEX UI attributes	app
Average stage times: (Average full run time: ~2min 58s)	2s	173ms	1s	3s	1s	1s	1s	268ms	2s	2s	1s	1s	1s	77ms	1min 17s	1s	1s	
#341 Apr 28 16:52 No Changes	2s	172ms	1s	2s	1s	1s	1s	1s	2s	4s	2s	3s	2s		1min 28s	1s	2s	
#340 Apr 14 14:28 No Changes	2s	163ms	1s	3s	1s	1s	1s	121ms	2s	2s	2s	1s	1s		1min 28s	1s	2s	
#339 Apr 14 14:25 No Changes	2s	185ms	1s	3s	1s	1s	1s	125ms	2s failed	73ms failed	78ms failed	81ms failed	69ms failed	77ms failed	81ms failed	73ms failed	73ms failed	
#338 Apr 14 13:44 No Changes	2s	167ms	1s	3s	1s	1s	1s	119ms	2s	2s	2s	1s	1s		1min 27s	1s	2s	
#337 Apr 14 09:20 No Changes	2s	191ms	1s	3s	1s	1s	1s	123ms	2s	2s	2s	1s	1s		1min 24s	1s	2s	
#336 Apr 14 No Changes	2s	169ms	1s	3s	1s	1s	1s	123ms	2s	2s	2s	1s	1s		1min 24s	1s	2s	

Process steps of deployments are separated in job stages (II)

customizing	acl	post installation tasks	Import APEX web credentials	Create credentials import script	Import module credentials	determine active schema name to sync	sync existing orders and data	module features	email whitelist	risk ruleset application	Create basic stuff	Create func users	Create portal groups	gather statistic	UT	mark custom sync scripts	register active version	Declarative: Post Actions
2s	2s	2s	2s	5s	1s	1s	5s	1s	1s	1s	1s	121ms	121ms	3s	121ms	1s	1s	582ms
2s	2s	4s	2s	5s	1s	1s		1s	1s	1s	2s	140ms	133ms	3s	124ms	1s	1s	606ms
2s	2s	4s	2s	5s	1s	1s	7s	1s	1s	1s	2s	121ms	131ms	4s	120ms	1s	1s	603ms
79ms failed	65ms failed	73ms failed	66ms failed	66ms failed	69ms failed	70ms failed	69ms failed	77ms failed	67ms failed	67ms failed	66ms failed	65ms failed	75ms failed	75ms failed	78ms failed	76ms failed	77ms failed	368ms
2s	2s	2s	3s	5s	1s	1s	19s	1s	1s	1s	1s	121ms	123ms	5s	127ms	1s	1s	613ms

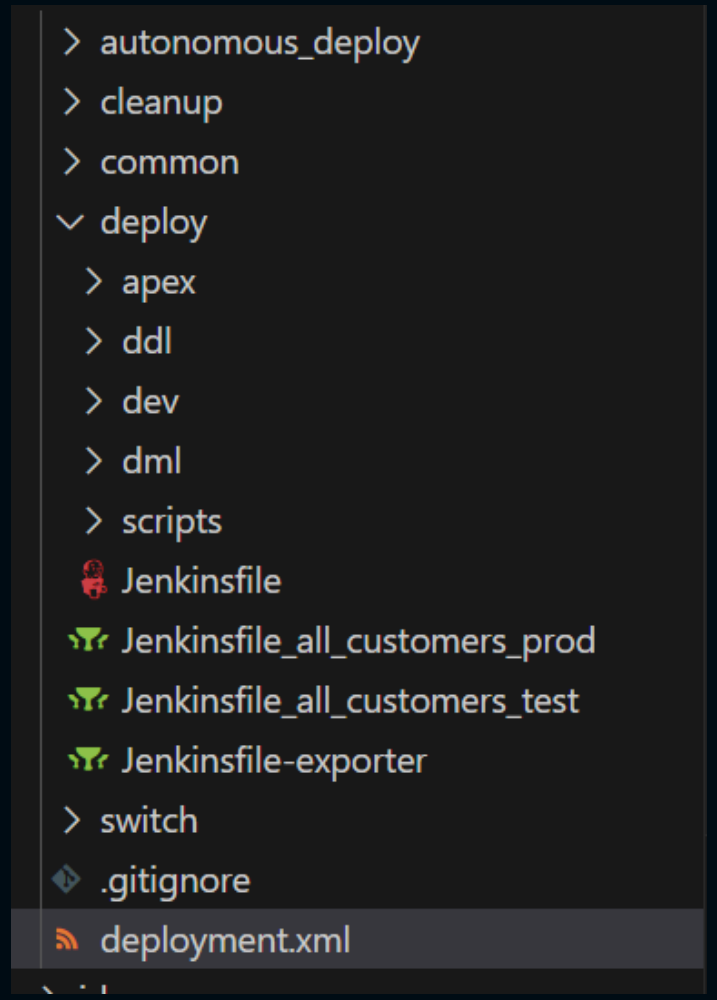
Central deployment logic sources for CI/CD based metadata

- define static values (e.g., app name) inside a config file (git repository)
- determine dynamic information and data at beginning of the pipeline run
  e.g.:

Function Name (PL/SQL)	Example Value
`get_active_schema_name(<app_id>, <customer_id>, '<env>')`	APP_CUSTOMER_PROD_01
`get_active_apex_app_id(<app_id, <customer_id>)`	10112301
`get_new_schema_name(<app_id>, <customer_id>, '<env>')`	APP_CUSTOMER_PROD_02
`get_new_apex_app_id(<app_id>, <customer_id>)`	10112302
`get_workspace_id(<app_id>, <customer_id>)`	5003102665918791
`get_new_apex_app_alias(<app_id>)`	APP_NEW
`get_new_apex_app_name(<app_id>)`	My App New

- values are transmitted to Job sub-stages (e.g., via env variables)

Store deployment configuration values in file (I)



Store deployment configuration values in file (II)

```
You, 3 months ago | 2 authors (You and one other)
<?xml version="1.0" standalone='yes'?>
<resources>
    <modul id="MYAPP">
        <deployment>
            <BASE_DIR>modules/myapp</BASE_DIR>
            <SQLSCRIPTS>
                <EXPORT_ENVIRONMENT_CONFIG replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/scripts/export_environment_config.sq
                <CREATEUSER replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/scripts/create_tablespace_and_schema.sql</CREATEUSE
                <APEXADMINCONFIG replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/apex/apex_admin_config.sql</APEXADMINCONFIG>
                <APEXAPPCREATE replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/apex/apex_app_import.sql</APEXAPPCREATE>
                <GETCSSFILENAMES replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/scripts/get_css_file_names.sql</GETCSSFILENAME
                <GRANTSANDSYNONYMS replacevars="true" runuser="ORADEPLOYMENTUSER">deploy/scripts/object_grants_and_synonyms.sql</GRAN
                <CREATETESTOBJECTS replacevars="true" runuser="ORANEWMODULSCHEMA">deploy/ddl/Build_All_Test.sql</CREATETESTOBJECTS>
                <CREATEOBJECTS replacevars="true" runuser="ORANEWMODULSCHEMA">deploy/ddl/Build_All.sql</CREATEOBJECTS>
```

Jenkinsfile executes framework script to run SQL CI/CD scripts

```
stage('create tablespace and user') {
    steps {
        script {
          def output = sh (script: "ssh deploy.sh -e '${env.G_ENV}' -c
           '${env.G_CUSTOMER}' -m '${env.G_MODULE}' -n EMPTY -o EMPTY -a 0
           -s CREATEUSER", returnStdout: true)
          echo output
          NEW_SCHEMA_NAME = (output =~ /(?<=NEW_SCHEMA_NAME: ).*/)[0]
          echo 'NEW_SCHEMA_NAME = ' + NEW_SCHEMA_NAME
          env.NEW_SCHEMA_NAME = NEW_SCHEMA_NAME
        }
    }
 }

...
```

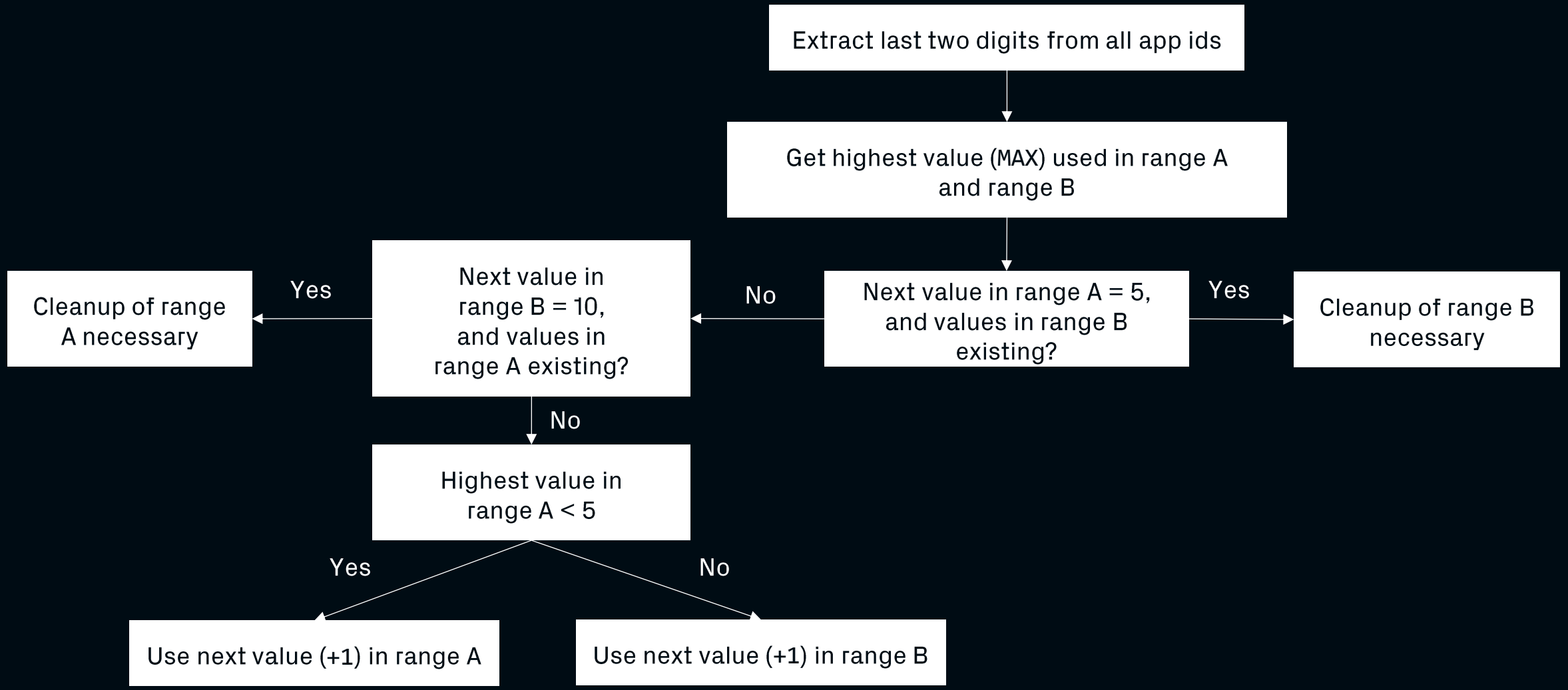
Version Numbers are determined by number ranges (I)

How to define the next version number?

– *We are using two **version ranges**: 1-5 (range A), 6-10 (range B)*

- Predictable Number Allocation
- Gap-Free Numbering
- Collision Prevention
- Overflow Handling

Version Numbers are determined by number ranges (II)



APEX Workspaces are automatically created

- In a multi-application landscape: create workspace during deployment of **base app**
- check on APEX metadata tables whether existing or not
- Approach: each app has its own schema
- first schema provisioned (FSP) should be **independent** from app schemas and **deterministic**

Initial deployment (workspace creation):

```
APEX_INSTANCE_ADMIN.ADD_WORKSPACE(
  p_workspace => '<workspace_name>',
  p_primary_schema => get_workspace_fsp(<customer_id>, '<env>'),
  p_additional_schemas => '<new_schema_name>'
);
```

All following deployments (workspace-schema assignments):

```
APEX_INSTANCE_ADMIN.ADD_SCHEMA(
   p_workspace => '<workspace_name>',
   p_schema => '<new_schema_name>'
);
```

Save application version when deployment is done

- after the deployment is done: save version info in metadata table as last step
  - set „active“ for inital deployment
  - set „new“ for all others

Switch process will be delayed in case active sessions are found

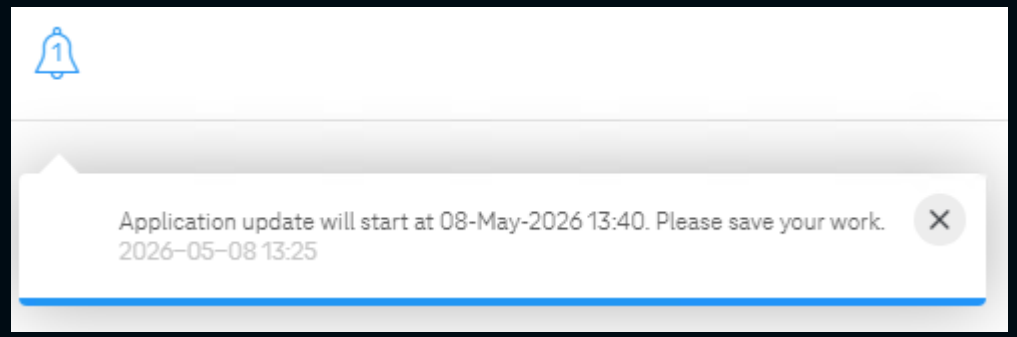
Declarative: Checkout SCM	init	pull deployment scripts	sync files	Export environment config	disable healthchecks	data enrichment	determine active schema name	determine new schema name	determine active application id	check active application users	wait 15 minutes	set active application unavailable
2s	112ms	1s	1s	1s	1s	1s	1s	1s	1s	1s	4min 25s	1s
2s	120ms	1s	1s	1s	1s	1s	1s	1s	1s	2s		2s
4s	120ms	1s	1s	1s	1s	1s	1s	1s	1s	2s		2s
2s	121ms	1s	1s	1s	1s	1s	1s	1s	1s	1s	4min 25s aborted	52ms aborted
2s	107ms	1s	1s	1s	1s	1s	1s	1s	1s	2s		1s

Application related metadata is updated and old app deactivated

lock modules	sync data	module features	email whitelist	risk ruleset application	gather statistic	switch	set old application unavailable with url	add release message	Declarative: Post Actions
1s	9s	1s	1s	1s	3s	2s	1s	53ms	1s
1s	23s	1s	1s	1s	6s	5s	1s		1s
1s	23s	1s	1s	1s	5s	4s failed	53ms failed	55ms failed	1s
56ms aborted	53ms aborted	79ms aborted	54ms aborted	53ms aborted	55ms aborted	58ms aborted	52ms aborted	53ms aborted	2s

Wait 15 minutes and inform the users about a release before switching

- check for active application sessions
- pause the job process to let users finish their work
- inform users about the upcoming release



Check active users (5 minutes):

```
SELECT DECODE(COUNT(*),0,0,1)
FROM APEX_WORKSPACE_SESSIONS ses
JOIN (
      SELECT DISTINCT apex_session_id
      FROM APEX_WORKSPACE_ACTIVITY_LOG
      WHERE application_id = <app_id>
) app_to_sess
ON ses.apex_session_id = app_to_sess.apex_session_id
WHERE user_name NOT IN ('APEX_PUBLIC_USER','nobody')
AND ROUND(
   (sysdate
    - (session_idle_timeout_on
       - (session_max_idle_sec/24/60/60)
      )
   ) * 24 * 60
) <= 5
AND ROWNUM = 1;
```

Switch the APEX and ORDS metadata

- switch process: should contain all steps to make app state consistent

  e.g.,

  - copy grants and synonyms (old –> new schema)
  - disable external reference constraints (owner != r_owner) for old schema
  - disable all scheduler jobs for old schema
  - switch APEX app metadata
  - switch ORDS metadata
  - set new version as „active“

Switch APEX app metadata:

```
APEX_APPLICATION_ADMIN.SET_APPLICATION_ALIAS(…)
APEX_APPLICATION_ADMIN.SET_APPLICATION_NAME(…)
```

Switch ORDS metadata:

```
ORDS_METADATA.ORDS.SET_URL_MAPPING(…)
```

Set old app version unavailable

- Set old application version

  „Unavailable (Redirect to URL)“
- Use the static APEX app alias you have defined
- Users will be redirected to the new active app

```
APEX_UTIL.SET_APPLICATION_STATUS(
  p_application_id => <apex_app_id>,
  p_application_status => 'UNAVAILABLE_URL',
  p_unavailable_value => 'f?p=APP_ACTIVE'
);
```

Using absolute URLs for reliable external linking

- common log-in URL:

  APEX_MAIL.GET_INSTANCE_URL || 'f?p=' || <apex_app_id>

- for custom links with page item values:

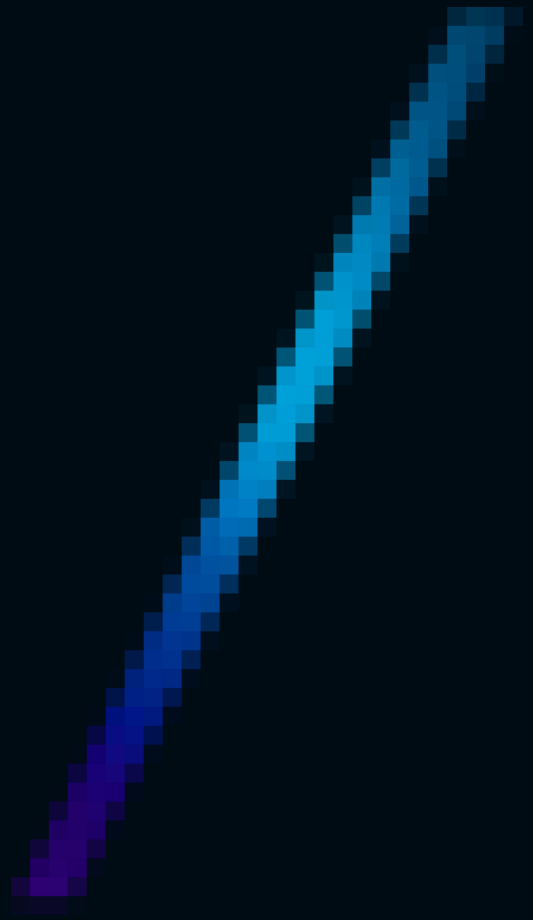
  APEX_UTIL.PREPARE_URL(p_url => …)

- APEX_UTIL.PREPARE_URL depends on the friendly URL configuration and HTTP context
  - since APEX 24.2: new parameter `p_absolute_url` for `APEX_PAGE.GET_URL`
  - For older APEX versions: https://blog.maurice-wilhelm.de/2025/10/08/optimizing-url-generation-with-apex_util-prepare_url-a-reliable-url-strategy-for-oracle-apex/

Blue-green deployment enables automated CI/CD workflow

Advantages

- Outages for users can be reduced
- No custom install file(s) needed, deployments from scratch
- Reduces manual effort for CI/CD, time can be used for other tasks
- Universally applicable
- Deployment process is more stable



Disadvantages

- High effort for implementing the CI/CD workflow (use of framework recommended)
- Physical storage on database
- Cross-dependencies between different applications
- Devs or DevOps Engineers need to know how to fix errors/failed pipelines

Alternative solutions?
SQLcl Projects can simplify In-Place Deployments

- SQLcl Projects for In-Place Deployments
- Red Stack Magazin No. 2/2026 (available May 8th)

  „Mit SQLcl und GitLab-Pipelines den CI/CD-Prozess optimieren“ (de)

  https://my.doag.org/magazines/id.231.ab-sofort-verfuegbar-red-stack-magazin-nr-2-2026/
- ESQUPRO - A CI/CD Utility Framework for SQL Deployments with SQLcl Projects

  Blog series (en): https://blog.maurice-wilhelm.de/2026/05/14/setup-gitlab-runners-and-container-registry-for-ci-cd-with-sqlcl-projects-part-1/

ESQUPRO

Streamlined deployments
for Oracle database applications

```
279 19:59:42 2026-04-16 19:59:41 UTC 191 Caused by: liquibase.exception.DatabaseException: Error occurred and failonerror set to true, stopping execution.
280 19:59:42 2026-04-16 19:59:41 UTC 192   at oracle.dbtools.raptor.liquibase.executor.jvm.SqlClExecutor.execute(SqlClExecutor.java:193)
281 19:59:42 2026-04-16 19:59:41 UTC 193   at oracle.dbtools.raptor.liquibase.executor.jvm.SqlClExecutor.execute(SqlClExecutor.java:333)
282 19:59:42 2026-04-16 19:59:41 UTC 194   at liquibase.database.AbstractJdbcDatabase.executeStatements(AbstractJdbcDatabase.java:1198)
283 19:59:42 2026-04-16 19:59:41 UTC 195   at liquibase.changelog.ChangeSet.execute(ChangeSet.java:816)
284 19:59:42 2026-04-16 19:59:41 UTC 196   ... 59 more
285 19:59:42 2026-04-16 19:59:41 UTC 197
286 19:59:42 2026-04-16 19:59:41 UTC 198 Error occurred and failonerror set to true, stopping execution.
287 19:59:42 2026-04-16 19:59:41 UTC 199
288 19:59:42 2026-04-16 19:59:41 UTC 200 Log file(s) location: /builds/dev/apex/cicd
289 19:59:42 2026-04-16 19:59:41 UTC 201 Removing the decompressed artifact: /tmp/7add8e54-d6fa-4fc0-acb5-4f5f60ecbdbb16276033718027512692...
290 19:59:42 2026-04-16 19:59:41 UTC 202 Disconnected from Oracle Database 23ai Free Release 23.0.0.0.0 - Develop, Learn, and Run for Free
291 19:59:42 2026-04-16 19:59:41 UTC 203 Version 23.4.0.24.05
292 19:59:42 2026-04-16 19:59:42 UTC .. > Search logfile for error patterns and codes (regex-based)
293 19:59:42 2026-04-16 19:59:42 UTC .. >> [ERROR] Found "ORA-[0-9]{5}:" on line 91. See output above.
294 19:59:42 2026-04-16 19:59:42 UTC .. >> [ERROR] Found "ORA-[0-9]{5}:" on line 92. See output above.
295 19:59:42 2026-04-16 19:59:42 UTC .. >> [ERROR] Found "ORA-[0-9]{5}:" on line 93. See output above.
```

Sources

International Journal of Multidisciplinary on Science and Management, Vol. 1(2), 2023, p. 22

Sharma et al., International Journal of Computer Applications, Vol. 186, No. 50, 2024, p. 2

Image sources

https://www.oracle.com/de/database/technologies/appdev/rest.html (last accessed on 11th May 2026)

Thank you very much for your attention!

Questions?



Feel free to reach out to me:

maurice.wilhelm@hyand.com

https://www.linkedin.com/in/maurice-wilhelm-511570272
https://blog.maurice-wilhelm.de



© 2026 – The thoughts and ideas developed are the intellectual property of Hyand and are subject to copyright. Reproduction, disclosure to third parties or use – even in part – is only permitted with the express consent of Hyand.