Alert Configuration Developer Guide
Oracle Banking APIs
Patchset Release 22.2.1.0.0

Part No. F72988-01

May 2023

ORACLE

Alert Configuration Developer Guide

May 2023

Oracle Financial Services Software Limited

Oracle Park

Off Western Express Highway

Goregaon (East)

Mumbai, Maharashtra 400 063

India

Worldwide Inquiries:

Phone: +91 22 6718 3000

Fax:+91 22 6718 3001

www.oracle.com/financialservices/

Table of Contents

1. Preface

1.1 Intended Audience

This document is intended for the following audience:

- Customers
- Partners

1.2 Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

1.3 Access to Oracle Support

Oracle customers have access to electronic support through My Oracle Support. For information, visit

http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit

http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

1.4 Structure

This manual is organized into the following categories:

Preface gives information on the intended audience. It also describes the overall structure of the User Manual.

The subsequent chapters describes following details:

- Introduction
- Preferences & Database
- Configuration / Installation.

1.5 Related Information Sources

For more information on Oracle Banking APIs Patchset Release 22.2.1.0.0, refer to the following documents:

- Oracle Banking APIs Installation Manuals
- Oracle Banking APIs Licensing Guide

2. Context

This alert configuration contains step to configure alerts for any event in the system. Alerts are configured against the pre configured activity event .Any alert is identified by 3 properties as follows:

1. Activity Id: An identifier for the activity being performed .It is the combination of the fully qualified name for the class and the method name.

   E.g. - Request fund activity in the Wallet.

   The activity id would be "**com.ofss.digx.app.wallet.service.core. Wallet.requestFunds**"

2. Event Id: An identifier for the event occurred while performing the activity. An activity can have multiple events. It should start from the module name followed by the logical name for the event.

   E.g. – Request fund success is an event in the wallet module.

   The Event Id can be "**WA_REQUEST_FUNDS_SUCCESS**"

3. Action Id: An identifier for the action to be executed during event processing .The action can be of 3 types.

   a. Alerts: Raise a message alert for the specified destination type like EMAIL, SMS etc. This is the default action performed while alerts processing.

   b. Notifications: The notifications to be generated for the dashboard etc.

   c. Business Logic: Any business Logic to be performed while alerts processing.

3. Database Configurations

All the configurations are explained with respect to Wallets request fund activity.

1. The Activity entry is added in the **DIGX_EP_ACT_B** table.

```
Insert into digx_ep_act_b
  (COD_ACT_ID,
   TXT_ACT_NAME,
   TXT_ACT_DESC,
   MODULE_TYPE,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'Wallet.requestFunds',
   'Wallet Request Funds',
   'WA',
   'SYSTELLER',
   sysdate,
   'SYSTELLER',
   sysdate,
   1,
   'A');
```

COLUMN NAME	DESCRIPTION
COD_ACT_ID	Primary key of the table. An identifier for the activity raising the event. It is the combination of the fully qualified name for the class and the method name.
TXT_ACT_NAME	Name of the activity. As a convention it is **'.'** separated combination of class name and method name.
TXT_ACT_DESC	Description of the activity.
MODULE_TYPE	Module type of the activity. It maps to the **ModuleType** enumeration.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_ACT_B.sql

2. The event is added in the table **DIGX_PM_EVENT_ALL_B** table.

```
Insert into digx_pm_event_all_b
  (EVENT_CODE, EVENT_DESC, ALERTS_FLAG)
values
  ('WA_REQUEST_FUNDS_SUCCESS',
   'Wallet Request Funds Successful',
   'Y');
```

COLUMN NAME	DESCRIPTION
EVENT_CODE	Primary key of the table. An identifier for the event occurred. It should start from the module type followed by the logical name for the event.
EVENT_DESC	Description of the event.
ALERTS_FLAG	Identifies whether the alert is required for this event or not. Possible values : **'Y'** or **'N'**.

SVN Location for Seed Data Script :

http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_PM_EVENT_ALL_B.sql

3. The activity Event combination is added in **DIGX_EP_ACT_EVT_B** table. Separate entries are required for all the events of the activity i.e. Suppose activity **'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds'** has two events one for success and other for failure, 2 entries will go in the table for both of them.

```
insert into DIGX_EP_ACT_EVT_B
  (COD_ACT_ID,
   COD_EVENT_ID,
   TXT_ACT_EVT_DESC,
   TXT_EVT_TYP,
   TXT_ACT_EVT_TYP)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'WA_REQUEST_FUNDS_SUCCESS',
   'Wallet Request Funds Successful',
   'OTHER',
   'ONLINE');
```

```
insert into DIGX_EP_ACT_EVT_B
  (COD_ACT_ID,
   COD_EVENT_ID,
   TXT_ACT_EVT_DESC,
   TXT_EVT_TYP,
   TXT_ACT_EVT_TYP)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'WA_REQUEST_FUNDS_FAILURE',
   'Wallet Request Funds Failure',
   'OTHER',
   'ONLINE');
```

COLUMN NAME	DESCRIPTION
COD_ACT_ID	Activity Id. It must match to the **COD_ACT_ID** column of **DIGX_EP_ACT_B** table.
COD_EVENT_ID	Event Id. It must match to the **EVENT_CODE** column of **DIGX_PM_EVENT_ALL_B** table.
TXT_ACT_EVT_DESC	Description of the activity event combination.
TXT_EVT_TYP	Event type. It maps to **EventType** enumeration.
TXT_ACT_EVT_TYP	Activity Event type. It maps to **ActivityEventType** enumeration. Possible values : '**BULK'** or **'ONLINE'**.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_ACT_EVT_B.sql

4. Message templates are added based on the destination types to the table **DIGX_EP_MSG_TMPL_B** table.

```
insert into digx_ep_msg_tmpl_b
  (COD_TMPL_ID,
   DESTINATION_TYPE,
   MSG_TMPL_NAME,
   MSG_TMPL_DESC,
   TXT_MSG_TMPL,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS,
   TXT_SUBJECT_TMPL,
   DETERMINANT_VALUE)
values
  ('WA_REQUEST_FUNDS_EMAIL',
   'EMAIL',
   'Wallet Request Funds EMAIL',
   '',
   '<p>Dear #WalletId#,</p></br><p>Please fund my wallet by
#Amount#.</p></br></br><p>Regards</p></br>#SenderName#',
   'SYSTELLER',
   sysdate,   'SYSTELLER',
   sysdate,
   1,
   'A',
   'Fund Wallet Request',
   'OBDX_BU');
```

COLUMN NAME	DESCRIPTION
COD_TMPL_ID	Primary key of the table. Uniquely identifies a message template. It should start from the module type followed by the logical name for the template.
DESTINATION_TYPE	Destination type of the template. It maps to **DestinationType** enumeration.
MSG_TMPL_NAME	Logical name of the message template.
MSG_TMPL_DESC	Description of the message template.
TXT_MSG_TMPL	It contains the format for the message body. It is stored as CLOB in the table.
TXT_SUBJECT_TMPL	It contains the subject for the message. It is also stored as CLOB in the table.
DETERMINANT_VALUE	It determines the entity code for the template.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_MSG_TMPL_B.sql

As you can see in the above example, the data elements like wallet id, amount and sender name are defined in between **'#'**.The entry for those data elements(or attributes) is done in the following tables.

5. Message attributes are added in the table **DIGX_EP_MSG_ATTR_B** table.

```
insert into digx_ep_msg_attr_b
  (COD_MESS_TMPL_ID, COD_ATTR_ID, ATTR_MASK, DETERMINANT_VALUE)
values
  ('WA_REQUEST_FUNDS_EMAIL', 'Amount', 'D', 'OBDX_BU');
```

```
insert into digx_ep_msg_attr_b
  (COD_MESS_TMPL_ID, COD_ATTR_ID, ATTR_MASK, DETERMINANT_VALUE)
values
  ('WA_REQUEST_FUNDS_EMAIL', 'WalletId', 'D','OBDX_BU');
```

COLUMN NAME	DESCRIPTION
COD_MESS_TMPL_ID	Message template Id. It must match to the **COD_TMPL_ID** column of **DIGX_EP_MSG_TMPL_B** table.
COD_ATTR_ID	Name of the attribute. It must match to the one defined inside **TXT_MSG_TMPL** of **DIGX_EP_MSG_TMPL_B** table.
ATTR_MASK	Masking format for the attribute value. Characters given as **'X'** will be masked and the ones given as **'D'** will be displayed as it is.
DETERMINANT_VALUE	It determines the entity code for the template.

SVN Location for Seed Data Script :

http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/metadata/DIGX_MD_SERVICE_ATTR.sql

6. Service attributes are added in **DIGX_MD_SERVICE_ATTR** table.

```
insert into digx_md_service_attr
  (COD_SERVICE_ATTR_ID,
   TYP_DATA_AVAIL,
   TYP_DATA_SRC,
   COD_ATTR_ID,
   COD_SERVICE_ID,
   PARAMETER_NAME,
   CREATED_BY,   CREATION_DATE,
   LAST_UPDATED_BY,   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS,
   REF_FIELD_DEFN_ID)
values
 ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds.WalletId.INPUT',
   'INDIRECT',
   'INPUT',
   'WalletId',
   'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'requestFundDTO',
   'SETUP',   sysdate,
   'SETUP',   sysdate,
   1,
   'A',
   'com.ofss.digx.app.wallet.dto.transfer.RequestFundDTO.WalletId.Value');
```

```
insert into digx_md_service_attr
  (COD_SERVICE_ATTR_ID,
   TYP_DATA_AVAIL,
   TYP_DATA_SRC,
   COD_ATTR_ID,
   COD_SERVICE_ID,
   PARAMETER_NAME,
   CREATED_BY,   CREATION_DATE,
   LAST_UPDATED_BY,   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS,
   REF_FIELD_DEFN_ID)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds.Amount.INPUT',
   'INDIRECT',
   'INPUT',
   'Amount',
   'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'requestFundDTO',
   'SETUP',   sysdate,
   'SETUP',   sysdate,
   1,
   'A',
   'com.ofss.digx.app.wallet.dto.transfer.RequestFundDTO.Amount.Amount');
```

COLUMN NAME	DESCRIPTION
COD_SERVICE_ATTR_ID	Primary key of the table. As a convention, **‘.’** separated **c**ombination of **COD_SERVICE_ID**, **COD_ATTR_ID** and **TYP_DATA_SRC**.
TYP_DATA_AVAIL	Possible values : **‘DIRECT’** or **‘INDIRECT’**. **‘DIRECT’** only when **‘TYP_DATA_SRC’** is **‘INPUT’** and the attribute value is one of the arguments passed to the activity. Otherwise **‘INDIRECT’**.
TYP_DATA_SRC	Possible values : **‘INPUT’** or **‘DTO’**. **‘INPUT’** when the attribute value can be obtained from the arguments passed to the activity. **‘DTO’** when the attribute value cannot be obtained from the arguments and is genertated/fetched within the activity.
COD_ATTR_ID	Name of the attribute.
COD_SERVICE_ID	Activity Id. It must match to **COD_ACT_ID** column of **DIGX_EP_ACT_B** table.
PARAMETER_NAME	The name of the argument passed to the activity. Its value will be null when **TYP_DATA_SRC** is **‘DTO’**.
REF_FIELD_DEFN_ID	Fully qualified path from which the attribute value can be obtained.

Here , in case of TYP_DATA_SRC as ‘INPUT’, there can be 2 cases :

- The attribute value is passed directly to the activity i.e. the attribute value is one of the arguments passed to the activity. In this case, TYP_DATA_AVAIL will be ‘DIRECT’.
- The attribute value is not passed directly to the activity , but it can be obtained from one of the arguments passed to the activity i.e. it is part of one of the DTOs which is passed to the activity. In this case, TYP_DATA_AVAIL will be ‘INDIRECT’.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/metadata/DIGX_MD_SERVICE_ATTR.sql

7. Source of the message attributes are added in **DIGX_EP_MSG_SRC_B** table.

```
INSERT INTO digx_ep_msg_src_b
  (COD_MESS_TMPL_ID, COD_ATTR_ID, COD_ACT_ID, COD_SERVICE_ATTR_ID,
DETERMINANT_VALUE)
VALUES
 ('WA_REQUEST_FUNDS_EMAIL',
  'WalletId',
  'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
  'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds.WalletId.INPUT',

  'OBDX_BU');
```

```
INSERT INTO digx_ep_msg_src_b
  (COD_MESS_TMPL_ID, COD_ATTR_ID, COD_ACT_ID, COD_SERVICE_ATTR_ID,
DETERMINANT_VALUE)
VALUES
  ('WA_REQUEST_FUNDS_EMAIL',
   'Amount',
   'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds.Amount.INPUT',
   'OBDX_BU');
```

COLUMN NAME	DESCRIPTION
COD_MESS_TMPL_ID	Message template Id. It must match to the **COD_TMPL_ID** column of **DIGX_EP_MSG_TMPL_B** table.
COD_ATTR_ID	Name of the attribute. It must match to the one defined inside **TXT_MSG_TMPL** of **DIGX_EP_MSG_TMPL_B** table.
COD_ACT_ID	Activity Id. It must match to **COD_ACT_ID** column of **DIGX_EP_ACT_B** table.
COD_SERVICE_ATTR_ID	Service attribute id. It must match to **COD_SERVICE_ATTR_ID** of **DIGX_MD_SERVICE_ATTR** table.
DETERMINANT_VALUE	It determines the entity code for the template.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_MSG_SRC_B.sql

8. The attributes which are input to the activity are added in **DIGX_MD_SERVICE_INPUTS** table.

```
insert into digx_md_service_inputs
  (COD_SERVICE_ID,
   PARAMETER_NAME,
   PARAMETER_INDEX,
   DATA_TYPE,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'requestFundDTO',
   1,
   'com.ofss.digx.app.wallet.dto.transfer.RequestFundDTO',
   'SETUP',
   sysdate,
   'SETUP',
   sysdate,
   1,
   'A');
```

COLUMN NAME	DESCRIPTION
COD_SERVICE_ID	Activity Id. It must match to **COD_ACT_ID** column of **DIGX_EP_ACT_B** table.
PARAMETER_NAME	The name of the argument passed to the activity.
PARAMETER_INDEX	Unique index of the argument for an activity. It starts from 0 for a particular activity.
DATA_TYPE	Data type of the argument passed to the activity.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_MSG_SRC_B.sql

9. The generic attributes along with their datatypes are added in **DIGX_MD_GEN_ATTR_LEGACY_B** table.

```
insert into DIGX_MD_GEN_ATTR_LEGACY_B

  (cod_constraint_attr_id,
   txt_constraint_attr_name,
   data_type,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS)
values
  ('WALLET_ID',
   'UniqueWalletIdentifier',
   'java.lang.String',
   'SETUP',
   sysdate,
   'SETUP',
   sysdate,
   1,
   'A');
```

```
insert into DIGX_MD_GEN_ATTR_LEGACY_B
  (cod_constraint_attr_id,
   txt_constraint_attr_name,
   data_type,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   OBJECT_STATUS)
values
  ('AMOUNT',
   'TransactionAmount',
   'java.lang.String',
   'SETUP',
   sysdate,
   'SETUP',
   sysdate,
   1,
   'A');
```

COLUMN NAME	DESCRIPTION
COD_CONSTRAINT_ATTR_ID	Attribute Id.
TXT_CONSTRAINT_ATTR_NAME	Name or description of the attribute.
DATA_TYPE	Data type of the attribute to format the attribute value.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/metadata/DIGX_MD_GEN_ATTR_LEGACY_B.sql

10. Entry for alert is added in **DIGX_EP_ACT_EVT_ACN_B** table.

```
insert into digx_ep_act_evt_Acn_b
  (COD_ACT_ID,
   COD_EVENT_ID,
   COD_ACTION_ID,
   FLG_TRANSACTIONAL,
   COD_DEC_ID,
   FLG_CONDITIONAL,
   COD_ACN_TMPL_ID,
   ALERT_NAME,
   CREATED_BY,
   CREATION_DATE,
   LAST_UPDATED_BY,
   LAST_UPDATE_DATE,
   OBJECT_VERSION_NUMBER,
   EXPIRY_DATE,
   ALERT_TYPE,
   ALERT_DISPATCH_TYPE,
   OBJECT_STATUS)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'WA_REQUEST_FUNDS_SUCCESS',
   'A',
   'Y',
   '0',
   'N',
   '1',
   'Wallet Request Funds',
   'OFSSUser',
   sysdate,
   'OFSSUser',
   sysdate,
   1,
   to_date('23-02-2018', 'dd-mm-yyyy'),
   'S',
   'I',
   'A');
```

COLUMN NAME	DESCRIPTION
COD_ACT_ID	Activity Id.
COD_EVENT_ID	Event Id.
COD_ACTION_ID	Action Id. Possible value : **'A'** (means Alert)
FLG_TRANSACTIONAL	Possible values : 'Y' or 'N'. This flag indicates whether events under this event category are transactional events or not.A Transactional event is an event which get processed within the same session of manager API.
COD_DEC_ID	Possible Value : 0
FLG_CONDITIONAL	Possible value : 'N'.

COLUMN NAME	DESCRIPTION
COD_ACN_TMPL_ID	Possible values : **1** or **2** . 1 indicates the importance of alert is critical. 2 indicates the importance of alert is informational.
ALERT_NAME	Unique name for the alert.
EXPIRY_DATE	Expiry Date of the alert.
ALERT_TYPE	Alert Type. Possible values: **'M'** or **'S'**. 'M' indicates the alert is of mandatory type and cannot be subscribed/unsubscribed by the user. 'S' indicates the alert is of subscribed type which can be subscribed/unsubscribed by the user.
ALERT_DISPATCH_TYPE	Alert Dispatch Type. Possible values: **'I'** or **'D'**. 'I' indicates immediate i.e. the alert needs to be send immediately. 'D' indicates deffered i.e. the alert will be sent later.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_ACT_EVT_ACN_B.sql

11. Entry for recipient message templates is added in **DIGX_EP_EVT_REC_B** table. Separate entries are required for all the destination types of the alert i.e. Suppose activity **'com.ofss.digx.app.wallet.service.core.Wallet.requestFunds'** has two destination types, EMAIL and SMS, 2 entries will go in this table.

```
insert into digx_ep_evt_rec_b
  (COD_ACT_ID,
   COD_EVENT_ID,
   COD_ACTION_ID,
   COD_MSG_TMPL_ID,
   TXT_DEST_TYP,
   SUBSCRIBER_TYPE,
   SUBSCRIBER_VALUE,
   ALERT_TYPE,
   LOCALE)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'WA_REQUEST_FUNDS_SUCCESS',
   'A',
   'WA_RequestFunds_EMAIL',
   'EMAIL',
   'PARTY',
   'CUSTOMER',
   'S',
   'en');
```

```
insert into digx_ep_evt_rec_b
  (COD_ACT_ID,
   COD_EVENT_ID,
   COD_ACTION_ID,
   COD_MSG_TMPL_ID,
   TXT_DEST_TYP,
   SUBSCRIBER_TYPE,
   SUBSCRIBER_VALUE,
   ALERT_TYPE,
   LOCALE)
values
  ('com.ofss.digx.app.wallet.service.core.Wallet.requestFunds',
   'WA_REQUEST_FUNDS_SUCCESS',
   'A',
   'WA_RequestFunds_SMS',
   'SMS',
   'PARTY',
   'CUSTOMER',
   'S',
   'en');
```

COLUMN NAME	DESCRIPTION
COD_ACT_ID	Activity Id.
COD_EVENT_ID	Event Id.
COD_ACTION_ID	Action Id. Possible value : **'A'** (means Alert)
COD_MSG_TMPL_ID	Message Template Id. Foreign key to **COD_TMPL_ID** of **DIGX_EP_MSG_TMPL_B.**
TXT_DEST_TYP	Destination Type. Possible value : **'EMAIL'** , **'SMS', 'SECURE_MAIL_BOX' , 'PUSH_NOTIFICATION'**
'SUBSCRIBER_TYPE	Possible value : **'PARTY'.**
SUBSCRIBER_VALUE	Possible value : **'CUSTOMER'**.
ALERT_TYPE	Alert Type. Possible values: **'M**' or **'S'**. 'M' indicates the alert is of mandatory type and cannot be subscribed/unsubscribed by the user. 'S' indicates the alert is of subscribed type which can be subscribed/unsubscribed by the user.
LOCALE	Locale to pick the location/language specific template for.

SVN Location for Seed Data Script :
http://obcpsvn.oraclecorp.com:8080/svn/clip/trunk/core/seed/oracle/alerts/DIGX_EP_EVT_REC_B.sql

Note: Entries for most of the activities, events, corresponding activity events , message templates, message attributes are already added. Please check for the entries in the table to avoid repetition.

3.1 API for Raising an EVENT

For raising an event, registerActivityAndGenerateEvent API has been provided in the **AbstractApplication** class.

It takes 4 parameters:

- Session Context
- EventId
- ActivityLog

Alerts can be either Account based or Party based.

If it is Account based, populating 2 attributes(accountId and accountType) of ActivityLog is bare minimum requirement.

Similarily, if it is Party based,populating 1 attribute(customerId) of ActivityLog is bare minimum requirement.

For the other attributes, In case the attribute is already present in **com.ofss.digx.app.alerts.dto.eventgen.ActivityLog** class use the existing ActivityLog instance.

Else create a subclass of ActivityLog having your attribute. Set the attribute value in the ActivityLog child class and pass its instance as an argument to r**egisterActivityAndGenerateEvent** method.

```
ActivityLog activityLog = new ActivityLog();
// if it is a party based alert setCustomerId
      activityLog.setCustomerId(sessionContext.getTransactingPartyCode());
// if it is a account based alert setAccountId and setAccountType
      activityLog.setAccountId("<<AccountNumber>>");
      activityLog.setAccountType("<<AccountType>>");
//If required, set other attributes in activityLog

super.registerActivityAndGenerateEvent(sessionContext, <<EventId>>, activityLog);
```

3.2 Custom Fields For Push notifications

Following Keys can be used to customize Push Notifications.

KEY NAME	VALUE
SOUND_IOS	File name of custom sound file added to OBAPI IOS App
SOUND_ANDROID	File name of custom sound file added to OBAPI Android App
LARGE_ICON_ANDROID	URL of icon image to be displayed as large icon in Big Style Push Notification of OBAPI Android App.
LARGE_IMAGE_ANDROID	URL of image to be displayed in Big Style Push Notification of OBAPI Android App.

These custom keys are to be added to the value of "TXT_MSG_TMPL" column of **DIGX_EP_MSG_TMPL_B** table.

If alerts are being created through front end UI, add following keys to "Notification Message" section.

Syntax for adding custom keys to Push Notification alert messages

[customfield1Name~customfield1Value|customfield2Name~customfield2Value]

Example 1:

You have requested for #NoOfChequeBook# cheque book with #ChequeBookOption# leaves on Account #AccountNo#.

[SOUND_ANDROID~isntit|LARGE_IMAGE_ANDROID~http://static1.squarespace.com/static/54ac6f9ae4b0cf1d82a4b59e/t/587f9e52cd0f68e84c5548fd/1484758653422/?format=300w|SOUND_IOS~chime.m4a]

Example 2:

You have requested for #NoOfChequeBook# cheque book with #ChequeBookOption# leaves on Account #AccountNo#.

[SOUND_ANDROID~isntit|LARGE_ICON_ANDROID~http://static1.squarespace.com/static/54ac6f9ae4b0cf1d82a4b59e/t/587f9e52cd0f68e84c5548fd/1484758653422/?format=300w|SOUND_IOS~chime.m4a]

3.3 Multi-Entity Specific templates

Entity specific templates can be created by following ways :

1. Creation of a new alert and template before the entity creation.
   If a new alert has to be maintained before the creation of any new entity, the data for the same has to be inserted in the following tables twice.
   One for DETERMINANT_VALUE '*' and the other for DETERMINANT_VALUE 'OBDX_BU', which is the default entity.
   Tables :
   DIGX_EP_MSG_TMPL_B
   DIGX_EP_MSG_ATTR_B
   DIGX_EP_MSG_SRC_B

2. Creation of a new alert and template after the entity creation.
   If a new alert has to be maintained after the creation of entity/entities, the same can be replicated for the different entities using the below queries.

   First insert the templates for DETERMINANT_VALUE '*' and DETERMINAT_VALUE 'OBDX_BU' and then execute the below queries for the respective entities.

```
insert into DIGX_EP_MSG_TMPL_B (DETERMINANT_VALUE, CREATION_DATE,
LAST_UPDATED_DATE, OBJECT_VERSION_NUMBER, COD_TMPL_ID, DESTINATION_TYPE,
MSG_TMPL_NAME, MSG_TMPL_DESC, TXT_MSG_TMPL, CREATED_BY, LAST_UPDATED_BY,
OBJECT_STATUS, TXT_SUBJECT_TMPL, DOMAIN_OBJECT_EXTN)
    (select '<ENTITY_CODE_TO_BE_REPLICATED>', sysdate, sysdate, 1,
COD_TMPL_ID, DESTINATION_TYPE, MSG_TMPL_NAME, MSG_TMPL_DESC, TXT_MSG_TMPL,
CREATED_BY, LAST_UPDATED_BY, OBJECT_STATUS, TXT_SUBJECT_TMPL,
DOMAIN_OBJECT_EXTN  from DIGX_EP_MSG_TMPL_B where DETERMINANT_VALUE = '*');

insert into DIGX_EP_MSG_ATTR_B (DETERMINANT_VALUE, COD_MESS_TMPL_ID,
COD_ATTR_ID, ATTR_MASK, DATA_ATTR_ORDER, DOMAIN_OBJECT_EXTN)
    (select '<ENTITY_CODE_TO_BE_REPLICATED>', COD_MESS_TMPL_ID, COD_ATTR_ID,
ATTR_MASK, DATA_ATTR_ORDER, DOMAIN_OBJECT_EXTN  from DIGX_EP_MSG_ATTR_B where
DETERMINANT_VALUE = '*');

insert into DIGX_EP_MSG_SRC_B (DETERMINANT_VALUE, COD_MESS_TMPL_ID,
COD_ATTR_ID, COD_ACT_ID, COD_SERVICE_ATTR_ID)
    (select '<ENTITY_CODE_TO_BE_REPLICATED>', COD_MESS_TMPL_ID, COD_ATTR_ID,
COD_ACT_ID, COD_SERVICE_ATTR_ID  from DIGX_EP_MSG_SRC_B where
DETERMINANT_VALUE = '*');
```

3.4 Configuring business logic for an event

In OBDX, for a certain event, either alert can be triggered, or a business logic can be called. To configure a business logic following steps need to be performed.

- Create a class to write the required business logic and implement com.ofss.fc.domain.ep.entity.action.logic.ILogic interface
- Override execute (com.ofss.fc.domain.ep.entity.action.IActivityEventAction eventAction, com.ofss.fc.xface.ep.dto.IActivityLog request) method to write the required business logic. Return true if the logic is successfully completed.
- Make entry in digx_fw_config_all_b with prop_id as '<<activity_id>>#<<eventId>>#L' , category_id as 'NotificationTrigger' and prop_value as fully qualified name of above class.

3.5 Configuring custom activity log mapper class for approval service

Activity log is used in alert framework to pass dynamic values to message template of the alert. For OOTB alerts the activityLog is defined in the service and is not available for modification. However, for approval related events, activity log mapper class can be configured to have extra fields in activity log from the transaction dto. This mapper class can be configured based on the task id. Steps to write custom activity log mapper class.

- Create a mapper class and implement interface com.ofss.digx.app.approval.alert.mapper.ITransactionActivityLogMapper.
- Override com.ofss.digx.app.alerts.dto.eventgen.ActivityLog getActivityLogForTransaction(com.ofss.digx.app.approval.dto.transaction.TransactionDTO transactionDTO) method to provide mapping logic.

- Make an entry in digx_fw_config_all_b table with category_id ‘transaction_activity_log_mapper’, prop_id as task_code of the transaction going through approval and prop_value as fully qualified name of above custom class.

3.6 Configuring Do Not Disturb (DND) for Mandatory Alerts

DND alerts can be configured by the following steps:

1. **Creating Alert Categories in the DIGX_EP_CAT_B table**.

Alert categories can be created in the DIGX_EP_CAT_B table using the following script.

INSERT INTO digx_ep_cat_b (cat_id, cat_name) VALUES (categoryId,categoryName);

For example: The below script can be used to add an alert category for Approval events.

INSERT INTO digx_ep_cat_b (cat_id, cat_name) VALUES ('APR', 'Approval');

Column Name	Description
CAT_ID	Unique alert category Id
CAT_NAME	Name of the alert category

2. **Mapping alert categories to alert events**

To map the alert categories to the events in DIGX_EP_ACT_EVT_B table the following script can be used.

UPDATE digx_ep_act_evt_b SET cat_id = 'APR' where cod_act_id like ('%com.ofss.digx.app.approval.service.transaction%');

Note: This functionality is applicable only for mandatory alerts.

3.7 Event Id enrichment for approval related alerts

In case of approval related alerts, a configuration is provided to enrich the event id. This enrichment can help developers to provide specific logic in approval related events for specific task ids. The event can be enriched for alert activity id + event id + task id combination. To configure the same, following configuration needs to be used.

- table : digx_fw_config_all_b
- category Id : dayoneconfig
- prop_id : <<TASK_ID>> + "#" + <<activity_Id>> + "#" + <<event_Id>> + "#LOGIC"
- prop_value : Enriched Event Id

For this enriched event id respective entry needs to be done in following tables.

- DIGX_PM_EVENT_ALL_B
- DIGX_EP_ACT_EVT_B
- DIGX_EP_ACT_EVT_ACN_B

4. Actionable Alerts

The actionable alert framework allows you to define an action inside a message template. By using this action (which can be in form of a hyperlink or button), the OBAPI user can navigate to the desired screen directly from alert. The alert could be from any destination type like EMAIL, SMS, On-Screen Alert or push notification.

Following are the important tables regarding configuration of actionable alert. Both the tables are multi-entity specific tables.

1. DIGX_EP_MSG_ACN_B

This table contains the primary information about the action needs to be taken on the alert.

Important column description:

Name	Description
COD_ACT_ID	Activity id of the template
COD_MESS_TMPL_ID	Unique identifier of the template
COD_MESS_ACN_ID	Unique action identifier for the respective template
MESS_ACN_DESC	Description of the action. This description will be visible to the admin user in alert maintenance screen in read-only mode
MODULE_ID	The module in which the UI component lies
COMPONENT_ID	UI Component identifier
DISPLAY_TXT	The text will appear on the hyperlink or button in the message body of alert. Admin users can modify this text from the alert maintenance screen.
MSG_ACN_HLDR	ANCHOR
FLG_ENABLED	Y/N (Specifies whether the action is currently enabled or disabled. Admin user can enable/ disable an action from alert maintenance screen)

Name	Description
FLG_LOGIN_REQD	Y/N (specifies whether user login is required to redirect to the desired screen in OBAPI application)

2. DIGX_EP_MSG_ACN_PRM_B

This table is used if any parameters need to be passed along with the action. These parameters can be used in the destination screen for further processing.

Important columns description:

Name	Description
COD_ACT_ID	Activity id of the template
COD_MESS_TMPL_ID	Unique identifier of the template
COD_MESS_ACN_ID	Action identifier of the respective template for which parameters are required
COD_MESS_ACN_PRM_ID	Unique identifier of the parameter. This value will be available as key in the params element of the redirected page
MESS_ACN_PRM_VAL	The data attribute id of the template whose dynamic value should be passed in the URL along with the key mentioned in 'COD_MESS_ACN_PRM_ID'

Steps to include action in an alert:

1. Identify the template id for which actionable alert needs to be configured (refer entry in 'DIGX_EP_MSG_TMPL_B').
2. Identify the UI component of OBAPI application to which the link should be redirected. This information is needed to insert entry in ‘MODULE_ID’ and ‘COMPONENT_ID’ columns of 'DIGX_EP_MSG_ACN_B' table.
3. Insert an entry in 'DIGX_EP_MSG_ACN_B' for the action.
4. e.g. Insert into DIGX_EP_MSG_ACN_B (COD_ACT_ID,COD_MESS_TMPL_ID,COD_MESS_ACN_ID,DETERMINANT_VALUE,MESS_ACN_DESC,MODULE_ID,COMPONENT_ID,DISPLAY_TXT,MSG_ACN_HLDR,FLG_ENABLED,OBJECT_VERSION_NUMBER,CREATED_BY,CREATION_DATE,LAST_UPDATED_BY,LAST_UPDATED_DATE,DOMAIN_OBJECT_EXTN,FLG_LOGIN_REQD) values

('com.ofss.digx.app.approval.service.transaction.Transaction.checkApprovals.financial','Financial_Transaction_Pending_Approval_EMAIL','act1','*','Description','approvals','transaction-detail','click here','ANCHOR','Y',1,null,sysdate,null,sysdate,'CZ','Y');

5. If any parameters need to be passed in the URL, make an entry in 'DIGX_EP_MSG_ACN_PRM_B'
6. e.g. Insert into DIGX_EP_MSG_ACN_PRM_B (COD_ACT_ID,COD_MESS_TMPL_ID,COD_MESS_ACN_ID,COD_MESS_ACN_PRM_ID,DETERMINANT_VALUE,MESS_ACN_PRM_VAL,OBJECT_VERSION_NUMBER,CREATED_BY,CREATION_DATE,LAST_UPDATED_BY,LAST_UPDATED_DATE,DOMAIN_OBJECT_EXTN) values ('com.ofss.digx.app.approval.service.transaction.Transaction.checkApprovals.financial','Financial_Transaction_Pending_Approval_EMAIL','act1','transactionId','*','TxnReferenceNo',1,null,sysdate,null,sysdate,'CZ');
7. After the entries are seeded, the actions will be available in Alert maintenance transaction for the respective template. Using the Alert Maintenance transaction the message body can be updated to place action at the desired location.
8. The action id should be inserted as #<action_Id># in the message body. (In the same way, as data attributes are inserted currently).
9. After the maintenance is done, the alert will contain the action in the form of hyperlink in the message body. Taking the action on the alert will redirect the user to the desired page. If login is required for the corresponding action, the user will be prompted for the login screen. If the user is already logged in or if the action does not require login, user will be directly navigated to the desired page.

Notes:
1. For parameters passed in the URL, additional handling will be required in the UI files of the component. The parameters will be available in key value pair, where the key will be parameter id used in ‘DIGX_MSG_ACN_PRM_B’ table. To access the value of parameter following format should be used.

<value> = rootParams.rootModel.params.<key>

2. To use actionable alert feature, it is mandatory to map ’Fetch Alert Action’ and ‘Read Alert Action’ transaction to the corresponding role using ‘Role Transaction Mapping’ functionality.

The hierarchy is: Essentials -> Alerts -> Fetch alert Action / Read Alert Action

---

Configurations Related to Actionable Alert:

Following properties are present 'DIGX_FW_CONFIG_ALL_B' for actionable alert.

PROP_ID	Existing PROP_VALUE	CATEGORY_ID	Description
OBAPI_WEB_PUB_ACTN_COMP	index.html?homeModule=alerts&homeComponent=alerts-action	DispatchDetails	Url For Non-Login redirection
OBAPI_WEB_PVT_ACTN_COMP	home.html?homeModule=alerts&homeComponent=alerts-action	DispatchDetails	Url for Login redirection
OBAPI_WEB_HOST_PROP	https://${OBAPI.WEB.HOST}:${OBAPI.WEB.PORT}	DispatchDetails	Web server url
url.shortener	Not seeded	DispatchDetails	To be configured for URL shortener adapter