Salesforce GROW

Module : Apex Triggers

OurUniversity
EFMD CLIP ACCREDITED

Apex Triggers

Capgemini

Triggers

A trigger:

- Executes code within Apex when a DML event occurs on a specific sObject.
- Can contain its own logic or call methods in other classes.
- Can be created on any custom or standard object.
- Can be executed either before or after the DML event:
  - A `before` trigger can update or validate values before they are saved to the database.
  - An `after` trigger accesses field values that are set automatically or affect changes in other records.
- Executes regardless of how the triggering data has been saved.

Execution Order of Triggers

The following steps describe the execution order of triggers:

1. The original record is loaded from the database or initialized for insert.
2. The new record field values are loaded and old values are overwritten.
3. System validation rules are executed and required fields are verified.
4. All `before` triggers are executed.
5. Most system validation steps are run again and validation rules are checked.
6. The record is saved to the database, but not committed.
7. All `after` triggers are executed.
8. Assignment rules, auto-response rules, and workflow rules are executed.
9. Records are updated again if workflow fields are updated.
10. `before` and `after` triggers are executed again if fields are updated based on workflow rules.
11. Escalation rules are executed.
12. Rollup summary formula or cross-object formula fields are updated in the parent records and affected records are saved.
13. Repeat the same process for affected grand parent records.
14. Criteria Based Sharing evaluation is executed.
15. All DML operations are committed to the database.
16. Post-commit logic is executed.

Type of Trigger Context Variables

- Trigger Context Boolean Variables
- size
- new
- newMap
- old
- oldMap

Introduction

- Triggers contain implicit variables in the `System.Trigger` class.
- Developers can access runtime context by using the trigger context variables.

Trigger Context Variables

Considerations of Trigger Context Variables

- `Trigger.new` and `Trigger.old` cannot be used as a part of the argument of a DML operation in Apex.
- `Trigger.old` is always read-only.
- `Trigger.new` cannot be deleted.
- You can modify field values before they are written to the database by using `Trigger.new` in `before` triggers.
- When `Trigger.new` is used in `after` triggers, it is not saved and an exception is thrown.
- `Trigger.new[0]` processes only the first record in a multi-record batch.

Considerations of Coding Triggers

Triggers:

- Should have the capability to process multiple records at a time.
- Can also execute through API access.
- Run as `System` by default.
- Are invoked only for those DML operations that are handled by the application server.
- Cannot modify the fields that are set during the system save operation.
- Can prevent DML operations from occurring.

Trigger code:

- Cannot contain the `static` keyword.
- Can only contain keywords applicable to an inner class.

DML Considerations for Triggers

Developers should follow these DML considerations for triggers:

- Triggers can modify other records of the same sObject type or records for other sObjects.
- Cascading execution of triggers are part of the same execution context for governor limits.
- Upsert events cause `insert` and `update` triggers.
- Merge events execute:
  - `delete` triggers for the unsuccessful records.
  - `update` triggers for the successful records.
- The `after undelete` trigger works only with recovered records.
- Undelete events run only on top-level objects.

Check the *Force.com Apex Code Developer's Guide* for a complete list of trigger events and the operations that are allowed.

Trigger Limits

isInsert
isUpdate
isDelete

isBefore
isAfter

Triggers can execute a maximum of 200 records at a time

If the number of records exceeds 200, triggers are executed in chunks of 200

```
Trigger myTrigger on Account (before insert, before update
if(Trigger.isInsert){
//do something
}
```

Hands-On Exercise

Trailhead:

https://developer.salesforce.com/trailhead/module/apex_triggers

Apex: Implementing Triggers Exercise Guide

Exercise 5-1: Creating an Automated Chatter Subscription (Part 2)

Exercise 5-2: Managing Access to Position Objects via Sharing

Exercise 5-3: Applying Best Practices for Bulkification and Limits in Triggers

Exercise 5-4: Determining to Use Declarative or Programmatic Solutions

https://lms.cfs-api.com/v1/content/0a5b35ed-be5b-4ed7-bbbb-042c6b61b8f0/presentation_content/external_files/implementingtriggersexerciseguide.pdf