Apex Interview Questions

* 1. **What are Apex Best practices?**
     1. Bulkify Code
     2. Avoid SOQL or DML inside Loop
     3. Avoid Querying Large Datasets. Use Limits or Add filters in Query
     4. Avoid Hardcoding Ids
     5. Handle Exceptions properly
     6. Have a trigger framework
     7. Use Async apex for long running operations
     8. Write Test classes

* 1. What access modifiers are available in Apex?
     1. Global
     2. Public
     3. Private
     4. Protected
     5. Abstract
  2. **What is asynchronous Apex? How can we perform operations asynchronously in Apex?**
     1. Asynchronous operation means that a process operating independently of other processes
     2. These options are provided to perform asynchronous operations in Apex
        1. Batch Class
           1. Mainly used to process bulk of data, for ex, data cleansing and archiving
           2. Each transaction within batch job starts with new set of governor limits
           3. Syntax:

|  |
| --- |
| public class BatchClassName implements Database.Batchable{        public Database.QueryLocator/Iterable<sObject> start(Database.BatchableContext bc){        }      public void execute(Database.BatchableContext bc, List<sobject> list){        }      public void finish(Database.BatchableContext bc){        }  } |

* 1. Future Methods
     1. Mainly used to perform long-running operations such as callouts to web services to avoid CPU time limits error
     2. As it runs in it's own thread, it can be used to avoid mixed DML errors
     3. Syntax:

|  |
| --- |
| @future  public void methodName(){  // code  } |

* 1. Apex Scheduler
     1. Mainly used to invoke the apex classes to run at specific time
     2. Apex class must implement schedulable interface
     3. Execution can be scheduled by System.schedule method
     4. Syntax:

|  |
| --- |
| public class SchedulableClassName implements Schedulable{      public void execute(SchedulableContext sc){      }  } |

* 1. Queueable interface
     1. It enables you to add jobs to the queueing mechanism
     2. Similar to future methods but with additional benefits
     3. Job can be added to the Queue using System.enqueueJob
     4. Syntax:

|  |
| --- |
| public class QueueableClassName implements Queueable{      public void execute(QueueableContext sc){      }  } |

* 1. **Difference between Future methods and Queueable Apex?**
     1. Future methods don't return the Job Id; Queueable interface does
     2. Future methods don't allow us to use non-premitive data-types such as sObjects or Cutom Apex type; Queueable classes do that
     3. We cannot call one future method from another future method, but we can do that using Queueable classes

* 1. **Why sObjects can’t be passed as arguments in the future method?**
     1. Because the sObject might change between the time you call the method and the time it executes.
     2. To work with sObjects that already exist in the database, pass the sObject ID instead (or collection of IDs) and use the ID to perform a query for the most up-to-date record.

* 1. **Why future method returns void?**
     1. As they run at a future time from when they are called, they have nowhere to return a response

* 1. **How many records can be retrieved via batch apex?**
     1. Total number of records retrieved by SOQL query: 50 million records
     2. If more than 50 million records are returned, the batch job is immediately terminated and marked as Failed.

* 1. **What is maximum batch size of batch apex?**
     1. Default is 200 but can be set upto 2000.

* 1. **What if I want to process more than 50 million records in batch apex?**
     1. We need to set one flag on object and recall the batch class from finish method based on flag check

* 1. **What happens if a batch is having 200 records and 1 record fails?**
     1. whole batch will be rolled back, but other batches will not be affected

* 1. **Can we get records from a batch job?**
     1. No, the execute and finish method returns void

* 1. **What Collections are available in Apex?**
     1. List
     2. Set
     3. Map

* 1. **What context variables are available in apex triggers?**
     1. New
     2. newMap
     3. Old
     4. oldMap
     5. isInsert
     6. isUpdate
     7. isDelete
     8. isBefore
     9. isAfter
     10. isUndelete
     11. Size

* 1. **What trigger Events are available in salesforce?**
     1. Before Insert
     2. Before Update
     3. Before Delete
     4. After Insert
     5. After Update
     6. After Delete
     7. After Undelete

* 1. **What is Apex Managed Sharing?**
     1. Apex managed sharing is the process of granting access to records data programmatically using Apex
     2. To grant access, we use Share object associated with standard and custom objects that you want to share
        1. The share object looks like this: <<ObjectName>>\_\_Share
        2. It has following properties:
           1. AccessLevel: Set the level of access

Read

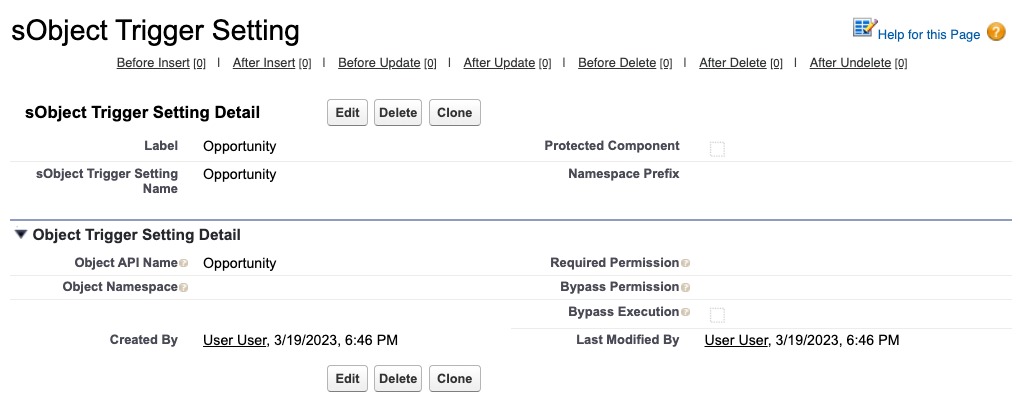
Edit

All

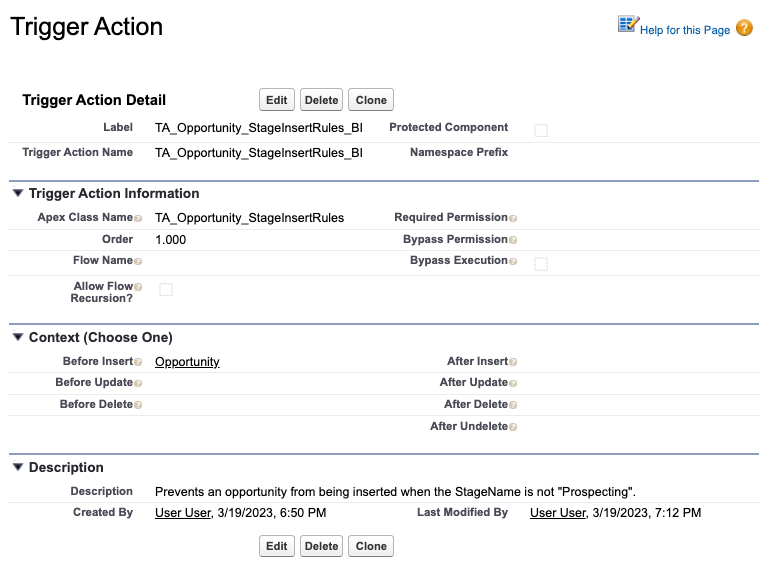
* + - * 1. ParentId: Id of the record being shared
        2. RowCause: Reason for sharing the record. Default is Manual.
        3. UserOrGroupId: user or group the record being shared to

* 1. **What is inherited sharing?**
     1. It inherits the sharing mechanism of the source it is called from
     2. It runs with 'With Sharing' context by default

* 1. **What trigger pattern do you know?**
     1. With the Trigger Actions Framework, we use custom metadata to configure our trigger logic
     2. The custom metadata defines:
        1. The sObject and context for which an action is supposed to execute
        2. The order to take those actions within a given context
     3. We use two metadata types:
        1. SObject\_Trigger\_Setting\_\_mdt: To define SObject



* 1. Trigger\_Action\_\_mdt: To define Apex class and trigger context



* 1. To define a specific action, we write an individual class which implements the applicable interface(s):

|  |
| --- |
| public class TA\_Opportunity\_StageInsertRules implements TriggerAction.BeforeInsert {  @TestVisible  private static final String PROSPECTING = 'Prospecting';    public void beforeInsert(List<Opportunity> newList){  }  } |

* 1. **What are the best practices of writing test classes?**
     1. Test for single as well as bulk records(at least 200 records)
     2. Cover positive and negative scenarios
     3. Test if restricted user can access to objects
     4. Focus 90+ test code coverage
     5. Add Assert statements
     6. Create test setup method using @testSetup
     7. Create TestFactory Class
     8. No seeAllData = true
     9. Test for Governer limits for Limits method
     10. Test for Exceptions

* 1. **What are the ways to apply field and Object level security in Apex?**
     1. To check if specific field has Read, create or update access for current user, we use Schema.DescribeFieldResult

if (Schema.sObjectType.Account.fields.PersonEmail.isAccessible()) {  
 Contact c = [SELECT PersonEmail FROM Account WHERE Id= :Id];  
}

* 1. To check if fields are accessible while querying through SOQL queries, we use WITH SECURITY\_ENFORCED clause

try{  
 List<Account> acts = [SELECT Id, Name, Email, (SELECT LastName FROM Contacts)  
 FROM Account WHERE Name like 'Universal' WITH SECURITY\_ENFORCED];  
} catch(System.QueryException){  
 //TODO: Handle Errors  
}

* 1. We use stripInaccessible method to check the access type of fields from sObject list
     1. stripInaccessible(System.AccessType accessCheckType, List<SObject> sourceRecords, [Boolean enforceRootObjectCRUD])
  2. <https://salesforcecodex.com/salesforce/enforce-object-level-and-field-level-permissions-in-apex/>

* 1. **How to loop over 10000 records efficiently?**
     1. By using SOQL for loops

* 1. **Difference between SOQL for loop vs looping on a list?**
     1. List will take Heap memory is the biggest advantage of using SOQL for loops as it will save memory
     2. 'Heap Size' issue can cause with list
     3. CPU time limit exception can also cause with list
     4. Otherwise in some use cases Adding records to a list is advantageous such as when you want to access the records outside of for loop

* 1. **We are inserting 300 records and we have a trigger on that object. How many times the trigger will trigger?**