**A workflow rule does not does not respect Validation Rule. So, if there is a situation where workflow and validation rule are conflicting, workflow rule will win because of the order of execution prescribed by Salesforce.**

**Q:** **Can a workflow rule be triggered from a field update event happening in an approval process?**

**A:** You have to select the "Re-evaluate Workflow Rules after Field Change" box in order for the Approval process to trigger the workflow.

An approval process can specify a field update action that re-evaluates workflow rules for the updated object. If, however, the re-evaluated workflow rules include a cross-object field update, those cross-object field updates will be ignored

**Q:** **A process builder is set to re-evaluate the record, which make it iterate multiple times. Does the process builder iterate indefinitely or stop after a few iterations?**

A: The process will iterate for a stack depth of 5. Which means 5 times the process will iterate and then it will halt. So, the chain will continue for 5 times. If infinite loop will happen then the org will hit governor Limits. And in order to avoid such conditions, Salesforce has kept the stack depth of 5 and post that process will halt.

**Q: We have an approval process on a custom object, when the approver rejects the record, that should be automatically deleted. How to achieve this?**

**A:** Create a custom checkbox on that custom object and then Create a Approval Process, as soon as the approver rejects the record then do a field(checkbox) update. After that create a **Autolaunched (No trigger) flow.** Now use this flow to delete records based on that checkbox value.

So, in this way as soon as that record is rejected by approver, field update will happen. Once the field is updated, that field will be picked by Flow and then that record will be deleted automatically.

**Q:** **Why do we prefer custom object over custom settings?**

**A:** No triggers on Custom Settings but can have triggers on custom Objects.

Custom Settings are stored in the Cache Memory whereas Custom Objects are stored in Force.com Database.

Custom Settings will have only limited Data types, but custom Objects have large variety of data types.

Custom Settings cannot have Relationships like MD or Lookup but custom objects can have.

Custom Settings cannot have its own custom tab in app but a custom object can have its custom tab in app.

No formula fields or RollUp summary in Custom Settings. But in custom Objects we have.

**Q: can I call a future method from batch class?**

A: Future method and Batch apex both are Asynchronous apex. So, we cannot call a async method from another async methods. So, we cannot directly call a future method from batch class But there can be a indirect or substitute way.

1st: Call the future method from finish() of batch class. Since, finish() is used for post processing logics. We can call a future method from finish().

2nd: Call a webservice method from Batch class, which in return will call a future method.

**Q: write a test class for batch class?**

**A:** The only catch here is to use **@testSetup** to create test records and  **Test.startTest() and Test.stopTest()**. This is used to make all the async calls between this block of code as sync in test context.

@isTest

Public class batchTest {

@testSetup

Public static void setup(){

/\*Create Records here\*/

}

Private static testMethod void(){

/\*Some logic\*/

Create a user and use System.runAs

Test.startTest()

Call the Batch Apex class

Assert statements

Test.stopTest()

}

}

**Q: OOPs Concepts in Salesforce?**

**A:** Oops concepts that we have in Apex are:

1. Objects
2. Class
3. Inheritance
4. Polymorphism
5. Abstraction
6. Encapsulation

**Objects:** It is a real life entity. Like Car, Bike, book, Customer. It is an instance of class. A class can have multiple instances. Objects has methods and member variables or instance variables.

Eg: Account, Contact, Opportunity, Lead, etc.

**Class:** A class is just like a collection of real-life entity. Like a showroom of cars, a company having employees, etc. It is a collection of similar entities.

Eg: public class AccountHandler{

}

**Inheritance:** When the object acquires the behaviour/properties of its parent’s class then it is called Inheritance. Inheritance is possible when a child class extends the Parent class. And the Child class inherits the properties of parent or super class.

**Polymorphism:** Polymorphism is a concept in oops. Polymorphism means Many forms/shapes. Polymorphism can be of 2 types static or dynamic.

Static Polymorphism happens at compile time only. Meaning that the compiler knows at compile time only that which method will be called.

Suppose there are two methods in a class and they are having different signatures then it will be a static polymorphism. This is also called method Overloading.

Eg: public class CalculateArea {

Integer side;

Integer length;

Integer breadth;

Public void **Area**(Integer side) {

Double area = side\*side;

System.debug(area);

}

Public vid **Area**(Integer length, Integer breadth) {

Double area = length \* breadth;

System.debug(area);

}

}

In the above example the method area is Overoaded. And this method has different method signatures.

Dynamic Polymorphism happens at runtime. Meaning that at runtime it will be decided that which method will be called. Dynamic Polymorphism is also called Method Overriding.

Eg: Public class car {

Public void color() {

System.debug(‘Generic Red car’);

}

}

Public class TataCar extends Car {

Public void color() {

System.debug(‘Tata Red car’);

}

}

Now when this color() will be called/invoked, at runtime it will be decided that which class’s method will be invoked.

**Abstraction:** Hiding the data and showing/exposing only the relevant/necessary part is called Abstraction. Hiding internal details and showing functionality is abstraction. We use inheritance for implementing abstraction. Access Specifiers(public, private, protected) are also used to achieve abstraction. Hiding internal complexity and showing functionality is known as abstraction.

**Encapsulation:** Capsulating/binding the data into a single unit is called Encapsulation. We use classes and Interfaces to implement encapsulation. Encapsulation is used to provide a better security to data. Moreover, we can manipulate the data using encapsulation.

Global class BatchClass implements Database.Batchable {

}

**Q: Constructor Overloading in Apex?**

**A:** public class TestClass {

Public TestClass(){

//Default constructor or ParameterLess Constructor

}

Public TestClass(Boolean isFalse){

//Paramaterized Constructor

}

Public TestClass(Boolean isTrue, String Name) {

//Paramaterized Constructor

}

}

**Q: Assert statements in Apex**

**A:** System.assert()

This statement has 2 parameter. 1. Is the condition. 2nd(Optional) is the message.

System.assert() is used to assert that the specified condition is true. If the condition is false then a error is thrown.

System.AssertEquals()

This statement takes 3 parameters 1st The actual result, 2nd the expected result, 3rd is the message(optional).

System.AssertEquals() is used to compare whether the actual result and the expected result are equal or not. If not equal then it will throw an Error.

System.AssertNotEquals()

This statement takes 3 parameters 1st The actual result, 2nd the expected result, 3rd is the message(optional).

System.AssertNotEquals() is used to compare whether the actual result and the expected result are equal or not. If equal then it will throw an Error.

**Q: How can we maintain the chain of batch classes?**

**A:** In order to maintain the chain of batch classes. We first need to understand the 3 methods inside Database.batchable.

1. Start() this method is executed once to send the list of sobjects to execute().
2. Execute() this method is executed multiple times depending upon the batch size, taking the sObjects on chunks.
3. Finish() is executed once and is used to do post processing implementations like sending email alerts etc.

So, the batch apex chaining can happen through the finish(). We can call another batch class from finish() method. Important point to remember here is that we can have 5 jobs running concurrently. So the only 5 jobs can be chained. **Ex:**

Public void finish(Database.BatchableContext bc) {

Id firstBatchId = bc.getJobId();

Id job2Id = Database.executeBatch(new otherBatchJob(), 200);

}

**Q: How can I restrict access of RECORDS from higher hierarchy level?**

**A:**  To tackle this scenario we need to take help of OWD. Since, we need to prevent the record access so set the OWD to private for the object. Post that, uncheck the Grant Access Using Hierarchies checkbox. By unchecking this checkbox, the higher roles in the hierarchy won’t be able to access the records. **Important point here is that this checkbox is Enabled for Custom Objects only. For standard objects we cannot uncheck this checkbox. It is by default checked and we cannot edit it**.

**Q: How can I stop the execution of trigger, workflow and validation rules while data uploader?**

**A:** Custom settings are available from both Workflows and Triggers. If you create a "Hierarchy" custom setting object with a checkbox you can reference that checkbox to enable/disable the validation or code. The hierarchy design would allow you to set system wide defaults and then override individual users as needed. It’s been pretty effective in a number of occasions.

**Q: Different types of Workflow Actions?**

**A:** There are 5 types of Workflow actions available:

1. New Task: Create a new Task when record meets criteria
2. New Email Alert: Create a New Email Alert
3. Field Update: Performs a specified field update
4. New Outbound Message: Sends a new Outbound message
5. Select Existing Action: Performs an existing action

**Q: Different criteria of workflow?**

**A:** 2 types

1. Evaluation Criteria
2. Created
3. Created and every time edited
4. Created and any time it's edited to subsequently meet criteria
5. Rule Criteria
6. When criteria are met (field and values are present here)
7. When formula field is set to true (we can write our own formula here)



**Q: Write Validation Rule formula such that LeadSource\_\_c should be either Web or Email or Walk-In.**

**A:** NOT( OR(

ISPICKVAL(LeadSource\_\_c, ‘Web’),

ISPICKVAL(LeadSource\_\_c, ‘Email’),

ISPICKVAL(LeadSource\_\_c, ‘Walk-In’)

) )

**Q: Write a Validation Rule such that Ownership field in Account should not be blank and should be other than ‘Public’ or ‘Private’.**

**A:** OR(

ISPICKVAL(Ownership, ‘Public’),

ISPICKVAL(Ownership, ‘Private’),

ISPICKVAL(Ownership, ‘’)

)

**Q: What is a WSDL? Difference between Enterprise WSDL and Partner WSDL?**

**A:** WSDL(Web Service Description Language) file is provided by Salesforce. It is a XML document which contains a standardized description on how to communicate using a web service.

WSDL is used by developers to aid in the creation of Salesforce Integration.

Enterprise WSDL

1. It is strongly typed.
2. This file is bound/tied to a specific configuration of Salesforce. Meaning every org has its own Enterprise WSDL.
3. The Enterprise WSDL changes if modifications are made in the org’s Configuration.

Partner WSDL

1. It is loosely typed.
2. Partner WSDL can be used to reflect against any Salesforce’s org configuration.
3. It is static. Meaning it does not change if the modifications are made to an salesforce’s org configuration.

**Q: Can we perform DML operation in Constructor?**

**A:** DML operation is not allowed in the constructor of Apex class. A constructor is mainly used to initialization of variables. Salesforce has blocked this due to security issues. Whenever we go for a DML operation in the constructor it restricts us in Salesforce. Whenever we trying to perform any DML operations at the time of initial page load we will be ending up with an error called System.LimitException: DML currently not allowed.

First, using DML in constructor will slow down initialization of your object. Second, it may fail and object just doesn't create at all.

**Q: How restrict record access using record types?**

**A:** Create different record Types as per your requirement.

Now to control the access. First of all, make the object OWD to private.

After that, create Sharing rule and use the criteria based rule on Record type values and then finally add public groups, users, roles to whom you need to provide the Access.

Reference: <https://www.youtube.com/watch?v=n0sYCKhZ-e0>

**Q: Suppose I have 2 users A and B; both have same profile. Now I want A to have access to a field and B should not have access to that field. How to achieve this?**

**A:** The best way to tackle this scenario is to create a permission set which has the access to that specific field. And assign this permission set to User A so that it will have access to that field and User B won’t have access to that field.

**Q: Order of Execution in Salesforce**

**A:** When we save a record using Insert, update or upsert statement. Salesforce performs the following events in order.

1. Loads the original record from the database or initializes the record for an upsert statement.
2. Performs all the system validations like Mobile or email format etc.
3. Executes all the record triggered flows that are configured before the record is saved.
4. All the before triggers are executed.
5. Runs most system validations again, runs all Custom Validation rules.
6. Executes duplicate rules. If the duplicate rules identify the record as duplicate and uses the block action. Record is not saved and no further steps such as Workflow rule after triggers are taken.
7. Executes all after triggers.
8. Executes all assignment rules.
9. Executes auto response rules.
10. Executes workflow rules. If there is field update then
11. Updates the record again.
12. Runs system validations again
13. Custom Validation rules, duplicate rules, processes and escalations are not run again.
14. Runs all before and after triggers again regardless of the dml operation one more time and only one more time.
15. Executes escalation rules.
16. Executes the following automation processes but not in a guaranteed order: -
17. Processes (Process builder)
18. Flows launched by Processes
19. Flows launched by a workflow rule
20. Executes entitlement rules.
21. Executes record triggered flows that are configured to run after the record is saved.
22. Executes Roll Up summaries.
23. Executes criteria-based sharing evaluation.
24. Commits all DML operations to Database.
25. After changes are committed to database, executes post commit logic like sending email alerts, enqueued Async Apex Jobs, Future, Queueable methods.

**Q: What are assignment Rules?**

**A:** Assignment rules helps in automation of your org’s Lead generations and support processes.

1. Use Lead Assignment rules to specify how leads are assigned to users or queues.
2. Use Case Assignment rules to determine how cases are assigned to users or put to queues.

Lead assignment rules can assign leads regardless of whether leads are created manually, are generated from Web-to-Lead forms, or are imported using the Data Import Wizard.

Case assignment rules can assign cases regardless of how cases are created. Cases can be created manually or automatically using Web-to-Case, Email-to-Case, On-Demand Email-to-Case, the Self-Service portal, the Customer Portal, Outlook, or Lotus Notes.

**Q: What are auto response rules?**

**A:** Auto-response rules let you automatically send email responses to lead or case submissions based on the record’s attributes. For example, you can send an automatic reply to customers to let them know someone at your company received their inquiry.

**Q: What are escalation rules?**

**A:** An escalation rule automatically reroutes a case and can notify a user if the case remains open after a certain period of time has passed. With an escalation rule, you can:

1. Choose to escalate a case to a queue or to another user.
2. Configure the rule to automatically notify a user.
3. Configure rule entries to define the order, criteria, and escalation actions.

**Q: What are Entitlement processes?**

**A:** Entitlement processes are timelines that include all of the steps (milestones) that your support team must complete to resolve cases or work orders. Set up an entitlement process to apply to entitlements in your Salesforce org.

**Q: What happens if workflow rule has time trigger set to past?**

**A:** If a workflow rule has a time trigger set for a time in the past, Salesforce queues the associated time-dependent actions to start executing within one hour. For example, if a workflow rule on opportunities is configured to update a field 7 days before the close date, and you create an opportunity record with the close date set to today, Salesforce starts to process the field update within an hour after you create the opportunity.

**Q: What happens if we change the rule criteria of time based workflow and updated rule don’t match the criteria?**

**A:** Time-dependent actions remain in the workflow queue only as long as the workflow rule criteria are still valid. If a record no longer matches the rule criteria, Salesforce removes the time-dependent actions queued for that record.

**Q: Change set deployement deMerits?**

**A:** We cannot deploy profiles.

We cannot roll back destructive changes.

Destructive changes are those which are not required or which are causing some errors in SIT or UAT.

**Q: What is the use of Database.Stateful Interface?**

**A:** We know that in a Batch Apex each batch is considered as a discrete transaction. So, the class variables loose there state after each batch is processed. So, in order to maintain the state of the class Variables and to create a link b/w the batches we use Database.Stateful.

Ex: You want to count the number of records updated via Batch apex, use a instance(class) variable and increment its value in execute().

**Q: How to create LWC dynamically?**

**A:** We cannot create LWC Dynamically, but a workaround for it will be using render() lifecycle hook to create a template in that.

**Q: Advantage of Queueable over Future Method?**

**A:**

1. Chaining is not possible in Future but Queueable we do chaining.
2. We cannot monitor Future Methods, but we can monitor Queueable methods from Apex Jobs page in Ui. Also, we have a jobId in Queueable which is not present in Future.
3. We cannot pass sObjects in Future as parameters, but in Queueable we can do that.

**Q: How many callouts we can make from Batch Apex, and how many callouts we can make from simple apex?**

**A:** From batch apex, we can make 100 callouts for each batch. In simple apex, we can make overall 100 callouts.

**Q: How to differentiate whether an API is SOAP or REST?**

**A:** If the API is accepting the response in any format then it will REST API. And if the API is accepting only XML response then it will be a SOAP API.

**Q: What are Named Credentials in Salesforce?**

**A:** A named credential specifies the URL of a callout endpoint and its required authentication parameters in one definition. To simplify the setup of authenticated callouts, specify a named credential as the callout endpoint. If you instead specify a URL as the callout endpoint, you must register that URL in your org’s remote site settings and handle the authentication yourself. For example, for an Apex callout, your code handles authentication, which can be less secure and especially complicated for OAuth implementations.

Salesforce manages all authentication for callouts that specify a named credential as the callout endpoint so that you don’t have to. You can also skip remote site settings, which are otherwise required for callouts to external sites, for the site defined in the named credential.

Named Credentials also include an OutboundNetworkConnection field that you can use to route callouts through a private connection. By separating the endpoint URL and authentication from the callout definition, named credentials make callouts easier to maintain. For example, if an endpoint URL changes, you update only the named credential. All callouts that reference the named credential simply continue to work.

To reference a named credential from a callout definition, use the named credential URL. A named credential URL contains the scheme callout:, the name of the named credential, and an optional path. For example: callout:My\_Named\_Credential/some\_path.

If transmitting sensitive information such as healthcare data or credit card data, authenticated Named Credentials are required. Salesforce recommends that Customers consider providing their own Certificates for extra security of sensitive data transmissions.

**Apex Code with Named Credentials:**

HttpRequest req = new HttpRequest();

req.setEndpoint('callout:My\_Named\_Credential/some\_path');

req.setMethod('GET');

Http http = new Http();

HTTPResponse res = http.send(req);

System.debug(res.getBody());

**Apex Code without using Named Credentials:**

HttpRequest req = new HttpRequest();

req.setEndpoint('https://my\_endpoint.example.com/some\_path');

req.setMethod('GET');

// Because we didn't set the endpoint as a named credential,

// our code has to specify:

// - The required username and password to access the endpoint

// - The header and header information

**String username = 'myname';**

**String password = 'mypwd';**

**Blob headerValue = Blob.valueOf(username + ':' + password);**

**String authorizationHeader = 'BASIC ' +**

**EncodingUtil.base64Encode(headerValue);**

**req.setHeader('Authorization', authorizationHeader);**

// Create a new http object to send the request object

// A response object is generated as a result of the request

Http http = new Http();

HTTPResponse res = http.send(req);

System.debug(res.getBody());

**Q: What is the difference b/w component and application events in terms of security?**

**OR**

**Q: What are the security constraints in Pub/Sub Model while using LWC?**

**A:** So when we use Application events(in AURA) or use PubSub in LWC many components are Listening to what a component is firing. So, some other component can Listen whatever data we are passing via PubSub. So, if we are sending some sensitive information via PubSub or application Event then security can be compromised.

**Q: How to take the backup of your org?**

**A:** Your Salesforce org can generate backup files of your data on a weekly or monthly basis depending on your edition. You can export all your org’s data into a set of comma-separated values (CSV) files.

You can generate backup files manually once every 7 days (for weekly export) or 29 days (for monthly export). In Professional Edition and Developer Edition, you can generate backup files only every 29 days. You can schedule backup files to generate automatically at weekly or monthly intervals (only monthly intervals are available in Professional Edition and Developer Edition).

Heavy traffic can delay an export delivery. For example, assume that you schedule a weekly export to run until the end of the month, beginning April 1. The first export request enters the queue, but due to heavy traffic, the export isn’t delivered until April 8. On April 7, when your second export request is scheduled to be processed, the first request is still in the queue. So, the second request isn’t processed until April 14.

**Only active users can run export jobs. If an inactive user schedules an export, error emails are generated and the export doesn’t run.**

**Q: How to take backup of your org’s Metadata ?**

**A:** To back up your metadata, create an unmanaged package.

In Setup, enter Package Manager in the Quick Find box, and then select Package Manager.

Select New, name the package, and select Save.

In the Components tab, select Add.

In the Component Type dropdown list, select the types of metadata you want to include in your backup, and click Add To Package for each metadata type.

To finish creating this unmanaged package, click Upload.

It’s likely that your org’s metadata is changing daily, and that new metadata types are added. To protect these changes, periodically create a new version of your unmanaged package.

Create a new version of your unmanaged package.

In Setup, enter Package Manager in the Quick Find Box, and then select Package Manager.

Click the Package Name for the unmanaged package you created.

In the Components tab, select add.

In the Component Type dropdown list, select any new types of metadata you want to add to your backup, and click Add To Package for each metadata type.

**Q: What are Custom Labels?**

**A:** Custom labels enable developers to create multilingual applications by automatically presenting information (for example, help text or error messages) in a user’s native language. Custom labels are custom text values that can be accessed from Apex classes, Visualforce pages, Lightning pages, or Lightning components. The values can be translated into any language Salesforce supports.

You can create up to 5,000 custom labels for your organization, and they can be up to 1,000 characters in length. Custom labels from managed packages don’t count toward this limit.

**Q: Managed Package vs Unmanaged Package?**

**A:**

**Unmanaged packages**

Unmanaged packages are typically used to distribute open-source projects or application templates to provide developers with the basic building blocks for an application. Once the components are installed from an unmanaged package, the components can be edited in the organization they are installed in. The developer who created and uploaded the unmanaged package has no control over the installed components, and can't change or upgrade them. Unmanaged packages should not be used to migrate components from a sandbox to production organization. Instead, use Change Sets.

**Managed packages**

Managed packages are typically used by Salesforce partners to distribute and sell applications to customers. These packages must be created from a Developer Edition organization. Using the AppExchange and the License Management Application (LMA), developers can sell and manage user-based licenses to the app. Managed packages are also fully upgradeable. To ensure seamless upgrades, certain destructive changes, like removing objects or fields, can not be performed.

Managed packages also offer the following benefits:

* Intellectual property protection for Apex
* Built-in versioning support for API accessible components
* The ability to branch and patch a previous version
* The ability to seamlessly push patch updates to subscribers
* Unique naming of all components to ensure conflict-free installs

**Q: How to override the functionality of standard buttons?**

**A:** We can override this using VF pages or Aura Components. We cannot use LWC directly for overriding the buttons.

But there is a workaround for this. We can call our LWC from a Aura Component. So When we will click that button, first Aura Component will be called, which in turn will call the LWC component.

**Q: Custom Buttons, links and actions?**

**Q: What is lightning Message Service(LMS)?**

**Q: What is difference between Standard Controller and Custom Controller?**

**//**

**Q: What is Test.isRunningTest() ?**

**A:** 1. To ensure the trigger doesn't execute the batch if Test.IsRunningTest() is true, and then test the batch class with it's own test method.

2. Testing callouts - in your callout code you check to see if you're executing within a unit test context by checking Test.isRunningTest() and instead of getting your callout response from an HttpResponse.send() request, you return a pre-built test string instead.

3. System.purgeOldAsyncJobs if called within this context, can cause long executions, based on data to be purged. Here using Test.isRunningTest() can help.

**Q: How does Asynchronous Apex works?**

**A:** Asynchronous Processing in Multi-tenant Environment present some challenges: -

* Make sure no asynchronous Requests are lost due to equipment or Software failure.

The platform uses a **queue-based** asynchronous processing framework. This framework is used to manage asynchronous requests for multiple organizations within each instance. The request lifecycle is made up of three parts:

1. Enqueue
2. Persistence
3. Dequeue

## **Enqueue: -**

The request gets put into the queue. This could be an Apex batch request, future Apex request or one of many others. The platform will enqueue requests along with the appropriate data to process that request.

## **Persistence: -**

The enqueued request is persisted. Requests are stored in persistent storage for failure recovery and to provide transactional capabilities.

## **Dequeue: -**

The enqueued request is removed from the queue and processed. If the processing fails, transaction control ensures that requests are not lost.

Each request is processed by a handler. The handler is the code that performs functions for a specific request type. Handlers are executed by a finite number of worker threads on each of the application servers that make up an instance. The threads request work from the queuing framework and when received, start a specific handler to do the work.

**Q: Batch Apex Best Practices?**

**A:**

1. To ensure fast execution of batch jobs, minimize Web service callout times and tune queries used in your batch Apex code. The longer the batch job executes, the more likely other queued jobs are delayed when many jobs are in the queue.
2. Only use Batch Apex if you have more than one batch of records. If you don't have enough records to run more than one batch, you are probably better off using Queueable Apex.
3. Tune any SOQL query to gather the records to execute as quickly as possible.
4. Minimize the number of asynchronous requests created to minimize the chance of delays.
5. Use extreme care if you are planning to invoke a batch job from a trigger. You must be able to guarantee that the trigger won’t add more batch jobs than the limit.