

Salesforce Integration – The Complete Guide

Master Connecting Salesforce with External Systems

Integrating Salesforce with external systems is crucial for businesses to synchronize data, automate workflows, and improve efficiency. This series will cover everything about Salesforce Integration, from fundamental concepts to advanced architectures, with real-world examples.

◆ **What You'll Learn:**

- ✓ Types of Integration
- ✓ Salesforce APIs (REST, SOAP, Bulk, Streaming, GraphQL)
- ✓ Authentication & Authorization (OAuth, Named Credentials, Connected Apps)
- ✓ Middleware & Event-Driven Integration
- ✓ Advanced integration patterns
- ✓ Best Practices & Security Considerations

Part 1: Understanding Integration in Salesforce

Before diving into technical implementation, let's understand what integration is and why it matters.

What is Integration?

Integration is the process of connecting Salesforce with external applications (databases, ERPs, payment gateways, CRMs) to enable seamless data exchange.

Why Do We Need Integration?

-  Sync Salesforce with an external database (e.g., MySQL, PostgreSQL)
-  Connect Salesforce with marketing tools (e.g., HubSpot, Marketo)
-  Automate order processing between Salesforce and an ERP (e.g., SAP)
-  Enable real-time communication using webhooks or event-driven APIs

Types of Integration in Salesforce

- 1 Data Integration** – Synchronize data between Salesforce and external systems. Example: Syncing customer details from an external database into Salesforce.
- 2 Process Integration** – Automate workflows across platforms. Example: When an order is placed in Salesforce, automatically create an invoice in an ERP.
- 3 Security Integration** – Manage authentication and access control between systems. Example: Using Single Sign-On (SSO) to log in to Salesforce with Google credentials.

Part 2: Salesforce APIs – The Building Blocks of Integration

Salesforce provides multiple APIs (Application Programming Interfaces) to interact with external systems. Let's explore them with examples.

1 REST API – The Most Common Integration Method

- ✓ Best for web and mobile applications
- ✓ Uses JSON & XML format
- ✓ Works with HTTP methods (GET, POST, PUT, DELETE)

◆ Example: **Fetching Accounts from Salesforce using REST API**

Using Postman or a Web Client:

GET

`https://yourInstance.salesforce.com/services/data/v57.0/subjects/Account`

Authorization: Bearer access_token

Apex Callout (Consuming an External API):

```
Http http = new Http();  
HttpRequest request = new HttpRequest();  
request.setEndpoint('https://api.example.com/data');  
request.setMethod('GET');  
HttpResponse response = http.send(request);  
System.debug(response.getBody());
```

2 SOAP API – Enterprise-Grade Integration

- ✓ Uses XML format for structured messaging
- ✓ Best for legacy systems like SAP, Oracle, and banking applications
- ✓ Uses WSDL (Web Service Definition Language)

◆ Example: Creating an Account via SOAP Request

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <create xmlns="urn:partner.soap.sforce.com">
      <sObjects>
        <type>Account</type>
        <Name>Test Account</Name>
      </sObjects>
    </create>
  </soapenv:Body>
</soapenv:Envelope>
```

3 Bulk API – Handling Large Data Volumes

- ✓ Used for data migration and batch processing
- ✓ Processes records asynchronously

◆ Example: Bulk API JSON Request

```
{
  "operation": "insert",
  "object": "Account",
  "records": [
    {"Name": "Account 1"},
    {"Name": "Account 2"}
  ]
}
```

4 Streaming API – Real-Time Updates

- ✓ Push-based API for real-time notifications
- ✓ Uses PushTopics & Change Data Capture (CDC)

◆ Example: **Subscribing to Account Updates**

```
PushTopic pt = new PushTopic();  
pt.Name = 'AccountUpdates';  
pt.Query = 'SELECT Id, Name FROM Account';  
pt.ApiVersion = 57.0;  
pt.NotifyForOperations = 'All';  
insert pt;
```

5 GraphQL API – Fetching Data Efficiently

- ✓ Fetch multiple objects in a single query
- ✓ Uses GraphQL syntax for flexible queries

◆ Example: **Fetch Account and Contacts in One Query**

```
{  
  "query": "query { Account(id: \"001XX000003DHPX\") { Name Contacts { Name Email } } }"  
}
```

Part 3: Authentication & Authorization

To securely integrate Salesforce with external systems, we use OAuth 2.0 authentication and Named Credentials.

1 Connected Apps – Secure Access for External Applications

Connected Apps allow external applications to securely connect to Salesforce.

Steps to Create a Connected App:

1. Go to Setup → App Manager → New Connected App
2. Enable OAuth Settings and provide a callback URL
3. Select OAuth Scopes (permissions)
4. Generate Client ID & Client Secret

2 OAuth 2.0 Authentication Methods

- 1 Username-Password Flow – Quick but not secure for production
- 2 JWT Flow – Best for server-to-server integration
- 3 Web Server Flow – Best for user authentication

◆ Example: OAuth 2.0 Token Request

POST

<https://login.salesforce.com/services/oauth2/token>

Content-Type: application/x-www-form-urlencoded

```
grant_type=authorization_code
&client_id=YOUR_CLIENT_ID
&client_secret=YOUR_CLIENT_SECRET
&redirect_uri=YOUR_REDIRECT_URI
&code=AUTHORIZATION_CODE
```

3 Named Credentials – Secure Callouts Without Hardcoding

Named Credentials store authentication details for external APIs securely.

◆ Example: Calling External API using Named Credentials

```
HttpRequest req = new HttpRequest();
req.setEndpoint('callout:MyExternalService');
req.setMethod('GET');
HttpResponse res = new Http().send(req);
System.debug(res.getBody());
```

Part 4: Middleware & Advanced Integration

Middleware like MuleSoft, Dell Boomi, and Jitterbit helps connect complex systems.

- ◆ Use Case: Sync Salesforce with an ERP (SAP, Oracle).
- ◆ Example: Middleware processes data before sending it to Salesforce.

Part 5: Advanced Integration Patterns

1 Platform Events & Event-Driven Architecture

- ✓ Used for real-time communication
- ✓ Helps decouple systems

Example: Publishing a Platform Event

```
MyCustomEvent__e event = new MyCustomEvent__e(Field__c = 'Value');  
EventBus.publish(event);
```

2 External Objects & Salesforce Connect

- ✓ Access external databases in real-time without storing data in Salesforce.
- ✓ Uses OData for data connectivity.

Part 6: Best Practices & Security Considerations

- ✓ Use OAuth 2.0 & Named Credentials for authentication
- ✓ Optimize API calls to avoid hitting limits
- ✓ Implement Retry Mechanisms & Error Handling
- ✓ Use Event Monitoring & Debug Logs

Conclusion: Become a Salesforce Integration Expert

Salesforce integration empowers businesses by automating processes and ensuring real-time data sync. By mastering these concepts,