Apex Code Overview

Apex is a strongly typed, object-oriented programming language that allows developers to execute flow and transaction control statements on the Lightning platform server in conjunction with calls to the Lightning Platform API. Using syntax that looks like Java and acts like database stored procedures, Apex enables developers to add business logic to most system events, including button clicks, related record updates, and Visualforce pages. Apex code can be initiated by Web service requests and from triggers on objects.

REQUIRED EDITIONS

Available in: Salesforce Classic (not available in all orgs) and Lightning Experience
Available in: **Enterprise**, **Performance**, **Unlimited**, **Developer**, and **Database.com** Editions

Apex can be stored on the platform in two different forms:

- A ***class*** is a template or blueprint from which Apex objects are created. Classes consist of other classes, user-defined methods, variables, exception types, and static initialization code. From Setup, enter Apex Classes in the Quick Find box, then select **Apex Classes**. See Manage Apex Classes.
- A ***trigger*** is Apex code that executes before or after specific data manipulation language (DML) events occur, such as before object records are inserted into the database, or after records have been deleted. Triggers are stored as metadata in Salesforce. A list of all triggers in your organization is located on the Apex Triggers page in Setup. See Manage Apex Triggers.

Apex generally runs in system context; that is, the current user's permissions and field-level security aren’t taken into account during code execution. Sharing rules, however, are not always bypassed: the class must be declared with the without sharing keyword in order to ensure that sharing rules are not enforced.

You must have at least 75% of your Apex covered by unit tests before you can deploy your code to production environments. In addition, all triggers must have some test coverage. See Apex Unit Tests.

After creating your classes and triggers, as well as your tests, replay the execution using the Developer Console.

You can add, edit, or delete Apex using the Salesforce user interface only in a Developer Edition organization, a Salesforce Enterprise Edition trial organization, or sandbox organization. In a Salesforce production organization, you can change Apex only by using the Metadata API deploy call, the Salesforce Extensions for Visual Studio Code, or the Ant Migration Tool. The Salesforce Extensions for Visual Studio Code and Ant Migration Tool are free resources provided by Salesforce to

support its users and partners, but are not considered part of our Services for purposes of the Salesforce Main Services Agreement.
For more information on the syntax and use of Apex, see the Apex Code Developer's Guide.

- **Apex Developer Guide and Developer Tools**
  The Apex Developer Guide and Apex Reference Guide provide the complete reference for the Apex programming language. The Apex Developer Guide explains how to invoke Apex, how to work with limits, how to write tests, and more. The Apex Reference Guide provides reference information on Apex classes, interfaces, exceptions and so on. To write Apex code, you can choose from several Salesforce and third-party tools.
- **Define Apex Classes**
  Salesforce stores Apex classes as metadata.
- **Define Apex Triggers**
  Apex code can be invoked by using triggers. Apex triggers can be configured to perform custom actions before or after changes to Salesforce records, such as insertions, updates, or deletions.
- **Executing Anonymous Apex Code**
  The Developer Console allows you to execute Apex code as another way to generate debug logs that cover specific application logic.
- **What Happens When an Apex Exception Occurs?**
  When an exception occurs, code execution halts. Any DML operations that were processed before the exception are rolled back and aren’t committed to the database. Exceptions get logged in debug logs. For unhandled exceptions, which are exceptions that the code doesn’t catch, Salesforce sends an email that includes the exception information. The end user sees an error message in the Salesforce user interface.
- **Handling Apex Exceptions in Managed Packages**
  When you create a managed package for AppExchange, you can specify a user to receive an email notification when an exception occurs that Apex doesn’t catch.
- **Manage Apex Classes**
  An Apex class is a template or blueprint from which Apex objects are created. Classes consist of other classes, user-defined methods, variables, exception types, and static initialization code.
- **Manage Apex Triggers**
  A trigger is Apex code that executes before or after specific data manipulation language (DML) events occur, such as before object records are inserted into the database, or after records have been deleted.
- **Manage Version Settings for Apex**
  To aid backwards-compatibility, classes are stored with the version settings for a specified version of Apex and the API.
- **View Apex Classes**

After you have created a class, you can view the code contained in the class, as well as the API against which the class was saved, and whether the class is valid or active.

- **View Apex Trigger Details**
  Apex triggers are stored as metadata in the application under the object with which they are associated.
- **Create an Apex Class from a WSDL**
  An Apex class can be automatically generated from a WSDL document that is stored on a local hard drive or network.
- **Monitor the Apex Job Queue**
  The Apex Jobs Setup page has information about Apex jobs, including the percentage of async Apex usage and the number of Apex operations that have been used out of the 24-hour org limit. Monitor the status of Apex jobs to mitigate potential limit problems before they happen.
- **Monitoring the Apex Flex Queue**
  Use the Apex Flex Queue page to view and reorder all batch jobs that have a status of Holding. Or reorder your batch jobs programmatically using Apex code.
- **Schedule Apex Jobs**
  Use the Apex scheduler and the Schedulable interface if you have specific Apex classes that you want to run on a regular basis, or to run a batch Apex job using the Salesforce user interface.
- **Apex Hammer Test Results**
  Salesforce runs your org's Apex tests in both the current and new release and compares the results to identify issues for you.
- **Apex FAQ**
  Frequently asked questions about external Web services, supported WSDL Schema types, and differences between a Apex classes and triggers.

VISIT - https://help.salesforce.com/s/articleView?id=sf.code_about.htm&type=5