Salesforce Development

Visualforce Pages

Learning Objectives

After completing this unit, you'll be able to:

- Explain what a Visualforce page is and describe its key attributes.
- List and open existing Visualforce pages in your organization.
- Create and edit a Visualforce page using the Developer Console.
- Identify, add, and customize Visualforce tags and attributes in the editor.

Introduction to Creating Visualforce Pages

Visualforce pages are basic building blocks for application developers. A Visualforce page is similar to a standard Web page, but includes powerful features to access, display, and update your organization's data. Pages can be referenced and invoked via a unique URL, just as they would be on a traditional web server.

Visualforce uses a tag-based markup language that's similar to HTML. Each Visualforce tag corresponds to a coarse or fine-grained user interface component, such as a section of a page, a list view, or an individual field. Visualforce boasts nearly 150 built-in components, and provides a way for developers to create their own components. Visualforce markup can be freely mixed with HTML markup, CSS styles, and JavaScript libraries, giving you considerable flexibility in how you implement your app's user interface.

You can view, create, and edit Visualforce pages several different ways in Salesforce. Visualforce pages can also be created and modified using Salesforce APIs, enabling a variety of external tools.

Input Components

Use the lightning:inputField component in lightning:recordEditForm to display and edit the value of a record field on a Salesforce object. Use the fieldName attribute to specify the API field name.

For standard and custom objects, find the field names in Lightning Experience from Setup > Object Manager > (object-name) > Fields & Relationships. Standard object fields are documented in Standard Objects. See Supported Objects in lightning:recordEditForm for usage considerations about objects.

The component displays an editable field based on the data type for the field you specify. For example, fieldName="Birthdate" on the Contact object references a date value, so the component renders an input field with a date picker and the label Birthdate. On the Account object, fieldName="Type" references a picklist, so the component renders a dropdown menu with the label Type. The list shows the account types defined in the org.

In orgs that support multiple languages, lightning:inputField automatically shows the translated labels and picklist values.

This component inherits styling from form layout in the Lightning Design System.

Get Started with Apex

Learning Objectives

After completing this unit, you'll be able to:

- Describe the key features of the Apex programming language.
- Save an Apex class and call methods with Anonymous.Apex.
- Use the Developer Console to inspect debug logs.

Get Started with Apex

Apex is a programming language that uses Java-like syntax and acts like database stored procedures. Apex enables developers to add business logic to system events, such as button clicks, updates of related records, and Visualforce pages.

As a language, Apex is:

- Hosted—Apex is saved, compiled, and executed on the server—the Lightning Platform.
- Object oriented—Apex supports classes, interfaces, and inheritance.
- Strongly typed—Apex validates references to objects at compile time.
- Multitenant aware—Because Apex runs in a multitenant platform, it guards closely against runaway code by enforcing limits, which prevent code from monopolizing shared resources.
- Integrated with the database—It is straightforward to access and manipulate records. Apex provides direct access to records and their fields, and provides statements and query languages to manipulate those records.
- Data focused—Apex provides transactional access to the database, allowing you to roll back operations.
- Easy to use—Apex is based on familiar Java idioms.
- Easy to test—Apex provides built-in support for unit test creation, execution, and code coverage. Salesforce ensures that all custom Apex code works as expected by executing all unit tests prior to any platform upgrades.
- Versioned—Custom Apex code can be saved against different versions of the API.