**Lightning Training – 22-Feb\_2022**

Apex Data Connectivity

Lightning Component Publishing

**Apex Data Connectivity**

Client – Server model is implemented for handling transactions in Apex data connectivity

Client – Web browser or Mobile App in which lightning pages and lightning components reside.

Server – Salesforce platform where Objects and Server-Side Controller (Apex Class) is available.

Steps for Client Server connectivity to handle data from lightning components

1. Define lightning component with “Controller” binding of “Apex Class – Server side” to establish connectivity for data.
2. In the Apex class, prepare business methods that expose data to the lightning component using @AuraEnabled annotation.
3. Define XMLHttpRequest (XHR) object into javascript as “action” for sending request from client to server in Async process.

Async process means – sending request to server from client as batches using Server Queue.

1. Handle response from server using XMLHttpResponse (response) object in the callback function defined in java script to get server response and bind with lightning component controls.

response object has 3 methods for handling server response

getState() – SUCCESS, INCOMPLETE, ERROR

getReturnValue() – result from the server in JSON string (name-value pair)

getError() – provide exception of transaction

Best Practices to followed in Apex Data connectivity

1. Always check Governor Limits of Salesforce to be referred by Transactions in Apex Classes
2. In Lightning components, use Helper methods to work with code-reusability within the component.
3. If code re usability need to applied for entire solution of project, prepare Utility Components as Parent component, and inherit into other components to access re-usability features.

Utility Component – PARENT

extensible=”true”

{!v.body}

Parent Component

* ATTRIBUTES
* CONTROLLER FUNCTIONS
* HELPER FUNCTIONS
* STYLE CLASSES
* CUSTOM EVENT

**Publishing of Lightning components**

Lightning components can be published to use in different platform implementations of Salesforce.

Framework provides “Marker” interfaces that helps in making lightning components published into various channels of salesforce.

1. Standalone Lightning TAB

Lightning component can be shown as Salesforce Lightning TAB to be added into lightning application for end users. If the project is build based on single component and designed using parent – child hierarchy of components in the implementation, Lightning Tabs will be created using single lightning component and extended the functionality from the component.

Inheritance is of 2 types:

IS-A relationship ------- child extends parent

HAS – A relationship ----- child is embedded into parent

Framework provides interface ---- force:appHostable

1. Composite Lightning Page

Developer can create “lightning page” with multiple lightning components arranged using different templates in a single page.

On a page, developer can assemble

* Standard components
* Custom components
* Custom – Managed components

The components in the page will be reloaded without effecting other components or the page in which they are embedded.

Page will be loaded into the web browser initially and components of the page will be refreshed as part of transactions without reloading the whole page. PAGE is STATIC and COMPONENTS will be DYNAMIC. This process is called “SINGLE PAGE ARCHITECTURE”

Framework provides interface : flexipage:availableForAllPageTypes

1. Lightning Record Page

Developer can create or override record pages for salesforce objects by preparing a custom lightning component and converting it into “Lightning record component”

This helps in customizing look of existing record pages using client specific requirements or branding.

Platform provides 2 interfaces to be implemented for preparing record component

* force:hasRecordId --- (provides salesforce id of the record for processing)
* flexipage:availableForRecordHome – (converts component into rec comp)