Lightning component development using LWC\_Day 2

Agenda

1. Event handling in LWC components
2. Data Connectivity from Salesforce objects with LWC components

**Event Handling:**

Custom Lightning component is comprised of below entities

1. HTML tags ex: DIV, SPAN, P(para), A (anchor), AUDIO, VIDEO
2. Pre-built lightning components ex: lightning-card, lightning-button, lightning-input

All the above entities support “events” for user interaction.

“event” in UI framework is used for interactions as elements fire the event and while handling the event, developer can define what need to be done while working with that interaction.

In Lightning Framework of salesforce, there are 3 types of events

1. Standard Events: HTML events fired by HTML tags and pre-built lightning components for different interactions defined on those tags. Developer must create JS methods as handlers for those events to work with business logic.

Ex: click, mouseover, mouseout, change, select, init etc.…

To handle these events, use “on” keyword as prefix before the event on the respective tag

Ex: onclick, onmouseover, onmouseout, onchange, onselect….

1. Custom Events: Developer defined events, where developer can create event, associate data with the event and fire the event from specific component and need to handle the event in destination component to process the data.

Custom events are used for communication of data between components in lightning framework for unique requirements.

1. Platform Events: are salesforce defined custom events which are used for unique salesforce platform requirements to be implemented in the application functionality.

Developer must instantiate the event and fire the event whereas platform itself handles the event to provide functionality in browser or mobile app.

Ex: PlatformShowToastEvent, PlatformNavigateToUrl etc...

Salesforce programming in lightning components is about working with events and handling business logic through event handlers for different controls in the component and hence component programming is called “Event Driven Programming”

Activities for Standard Events in component

Activity 1:

Create a custom component to demonstrate “click” event in pre-defined components of salesforce

Components support “click” event”

Lightning-button

Lightning-buttonIcon

Lightning-menuButton

Activity 2:

Create a component to handle event from multiple controls (button) on the UI to apply the logic.

Activity 3:

Create a component for demonstrating “change” event in the lightning controls

Form component - Lightning-input

Helps in loading of data and change event fires whenever value is inputted (key input)

Activity 4:

Create a component to accept values from input controls without using “change” event.

**Data Handling in lightning components from Salesforce objects**

Salesforce follows MVC web design pattern for handling data in lightning components

Model : salesforce objects where data is persisted

View : lightning components residing in lightning page or mobile app

Controller : Apex classes with @AuraEnabled methods for transaction handling

LWC components uses “Async transactions” for processing of data in salesforce using salesforce objects and there are 2 approaches in data handling for LWC framework

1. Using WIRE service of LWC framework for handling data
2. Using “Method calling” in LWC

**WIRE service:**

This is Async service module loaded from “lwc” core module into JS class that runs in the background in separate thread which execute transactions and store the result of transaction into browser cache to provide the output into the component.

Because of result stored in the browser cache and component loads the data from the cache, the data will be immediately linked with component controls and hence the user experience is enhanced for end-users.

Whenever there is a change request sent from variables in the component which are reactive variables, then only the Async transaction will re-execute to refresh the browser cache or else the same static data from the cache will be reflecting in the component controls.

Pro – data will be loaded into cache when the Wire service launches in the component and it can be bound to controls using events and due to the data already available, binding is faster and UI can display information quickly without any latency in component.

Con – since the Wire service executes in its own thread asynchronously, developer can’t control when the data need to fetched for handling specific requirements like “onload” events in components.

Activity:

Create component for displaying records from salesforce object (Account) using WIRE service.

**Calling methods “imperatively” for executing data binding with components**

This is regular process of binding salesforce data using method execution from lightning components. The methods will be called explicitly using events in lightning component and data from the transaction is bound to controls using JS “promise” objects of framework.

This process is completely under control of developer and can process salesforce data using different event handlers of pre-built components in programming.

This way of getting data might cause some latency in fetching data and hence developer can use components like “spinner” to cover the data loading latency in the controls.

Activity:

Create a lightning component to display records from “Opportunity” object into component