**Lightning component development Training**

**DEX601 – Salesforce Lightning component development using AURA components**

**Pre-requisites:**

1. **Attended or completed Salesforce Apex Training or Certification (Platform Developer 1)**
2. **Should be familiar with Web Technologies (HTML5, Java Script, CSS etc.…)**
3. **Should be working as developer in salesforce for at least 6 months.**

**Agenda**

1. **Lightning component Fundamentals** 
   1. **Preparing components and understanding framework**
   2. **Publishing of components into salesforce platform**

**(2.5 days)**

1. **Case Study (project simulation with client requirements)**

**(2.5 days)**

**Training Org – virtual org given by salesforce to practice case study and training features**

**40 days validity**

**Lightning component development**

**To build and manage custom salesforce projects into Lightning experience (Desktop) and Salesforce Mobile.**

**Projects are Salesforce Apps that are built for specific requirement and Salesforce Apps are made up of “Lightning Pages”**

**These lightning pages are built using “Lightning Components” for implementing functionality of UI for the end users.**

**“Lightning components are BUILDING BLOCKS of pages which are used in Applications”**

**How to build components into Salesforce Lightning**

**Salesforce has 2 frameworks for developing components for applications**

1. **Standard Lightning Component Framework ( AURA framework)**
2. **LWC Framework (Lightning Web Components)**

**Standard Lightning Component Framework is salesforce Proprietary framework for building Lightning components.**

1. **AURA Opensource Framework for binding elements.**
2. **HTML5 – is backend for creating UI to be published in Web Browsers or Mobile**
3. **Java Script – for event handling**
4. **CSS3 – for component styling or branding**
5. **JSON – for distributing data between components**
6. **Boot Strap – Responsive web design**
7. **XML – for creating component meta-data**
8. **SVG – for defining ICONS used by the components**

**Lightning Web Components Framework is Open-Source Web standards implementation of Salesforce for developing Lightning Components**

**Developers can create components using traditional Java Script frameworks for building lightning components and publishing in Salesforce platforms.**

* **HTML5 DOM**
* **Java Script classes**
* **CSS**
* **XML**
* **SVG**

**Component Development using Standard Lightning Framework (AURA)**

**Components are built in AURA framework using a special feature called “Component Bundle”**

**Component Bundle is a collection of files belonging to different technologies bounded by Framework elements to represent functionality of a single component.**

**There are 8 files in the component bundle**

1. **Component – (XML) -- .cmp ---- is used to prepare the UI of induvial functionality**
2. **Controller – (Java Script) .js – used to handle events of the component**
3. **Helper – (Java Script) .js – used to work with re-usable logic of controller functionality**
4. **Style – (CSS3) .css --- to provide client specific branding for individual components**
5. **Documentation – (HTML5) .auradoc – providing component help in salesforce docs**
6. **Renderer (Java Script) .js – has methods for handling rendering of pages in browser**
7. **Design (XML) .design – helps in design properties of component**
8. **SVG (SVG) .svg – svg file that provides unique Icon for the component.**

**How to develop Lightning Projects**

1. **Building a lightning Application for usage of component fundamentals**

* **Lightning App Manager**

1. **Preparing Lightning Page for working with custom Lightning components**

* **Lightning App Builder**

**3 types of Lightning pages**

1. **Application Page – for creating “single page application” to implement requirements**
2. **Home Page – for creating “custom HOME page” for application.**
3. **Record Page – for creating record pages on Salesforce objects.**

**After creation, page need to activated for showing into the particular application.**

**SLDS – Salesforce Lightning Design System**

**Inbuilt stylesheets of Salesforce to apply consistent branding across components in platform**

**Event Handling in Lightning components**

**Lightning component contain below entities**

1. **HTML tags – DIV, SPAN, A, P, AUDIO, VIDEO**
2. **Pre-built lightning components:**

**Component library: lightning namespace**

**Ex:**

**Lightning:card**

**Lightning:button**

**Lightning:input**

**Lightning:textarea**

**Lightning:select**

**150+**

**All these entities throw events when user interact with them**

**In Salesforce, we have 3 types of events**

1. **Standard Events (HTML events)**

**Ex: Click, MouseOver, MouseOut, Change, Init, Select etc..**

1. **Custom Events (developer events)**

**Ex: Component Event, Application Event**

1. **Platform Events (custom events pre-built by Salesforce)**

**Ex: showToast, navigateToURL, refreshView etc…**

**Lightning programming is called “Event-Driven programming” as all transactions of lightning components are triggered by events and should be handled to execute the business logic.**

**Standard Events Handling**

**all entities in the component will fire standard events and they will handled by Java Script functions defined in the “Controller” file of component bundle.**

**function(component,event,helper)**

**{**

**// business logic**

**}**

**component: represents backend component for referring attribute or element data into the java script.**

**To refer attributes, syntax is:**

**component.get(“v.attributename”) --- fetch the attribute data into javascript.**

**component.set(“v.attributename”,”value”) --- upload the data of attribute with value.**

**To refer elements into Java Script:**

**Add “aura:id” to the element to uniquely represent the element in the HTML DOM**

**Refer using below syntax:**

**component.find(“aura:id”)**

**.getElement() – to access the HTML tag into Java Script**

**.get() or .set() – to provide properties of the elements from Java Script.**

**event : this will act as super object to refer all fired events in the backend component**

**.getSource() – to refer which element in component firing the event**

**.destroy()**

**.getId()**

**helper: is used as a connector to link helper functions in the component bundle with the controller functions to enhance re-usability of the logic.**