Lightning Web Component (Q&A)

LWC
Interview
Questions & Answers

Sanjay Gupta Tech School

https://StudySalesforce.com

Q : What is LWC?

A :

- LWC stands for Lightning Web Component.
- LWC is an implementation of the W3C's web component standards.
- It supports the parts of web components that works in browser and adds parts supported by Salesforce as well.
- Quick component development because developer has to use only HTML, CSS and JavaScript.

Q : Lightning Web Component Files?

A :

- test.html
- test.js
- test.css
- test.js-meta.xml

Q : What is LWC Module?

A :

- LWC uses modules to bundle core functionality and make it accessible to the JavaScript in your component file.
- The core module for lightning web component is lwc.
- Begin the module with the import statement and specify the functionality of the module that your component uses.
- Example : import {LightningElement} from ‘lwc’;
- The import statement indicates the JavaScript uses the LightningElement functionality from the lwc module.

Q : What is use of XML File?

A :

- This file defines component's configuration like where developer can use the lightning web component.
- <isExposed> true </isExposed>
- <targets>
  <target>lightning__HomePage</target>
  <target>lightning__RecordPage</target>
  </targets>

Q : Can Aura Component contain Lightning Web Component?

A :

- Yes

Q : Can Lightning Web Component contain Aura Component?

A :

- No

Q : Can a Lightning Web Component call another Lightning Web Component?

A :

- Yes

Q : Camel Case Vs Kebab Case?

A :

- LWC match web standards wherever possible. The HTML standard requires that custom element names contain a hyphen.Since all LWC have a namespace that's separated from the folder name by a hyphen, component names meet the HTML standard.
- For example, the markup for the Lightning web component with the folder name widget in the default namespace c is <c-widget>.

Q : Camel Case Vs Kebab Case?

A :

- However, the Salesforce platform doesn’t allow hyphens in the component folder or file names. What if a component’s name has more than one word, like “mycomponent”? You can’t name the folder and file my-component, but we do have a handy solution.
- Use camel case to name your component like myComponent. Camel case component folder names map to kebab-case in markup. In markup, to reference a component with the folder name myComponent, use <c-my-component>.

Q : What is decorator ?

A :

- Decorators are often used in JavaScript to modify the behavior of a property or function.
- Examples:
  - @api
  - @track
  - @wire

Q : What is the use of @api decorator?

A :

- Marks a field/property as public.
- HTML markup can access the component's public properties.
- All public properties are reactive. Reactive means the framework observe the property for change. When property changes value then the framework reacts and renders the component.

Q : What is the use of @track decorator?

A :

- Observe changes to the properties of an object or to the elements of an array.
- Framework render the component when changes occurs.

Q : What is the use of @wire decorator?

A :

- It provides a way to get and bind data from a Salesforce Org.

Q : Conditional Rendering in HTML?

A :

```
<template if:true={areDetailsVisible}>
        These is true block!
</template>
<template if:false={areDetailsVisible}>
        These is false block!
</template>
```

Q : Rendering List in HTML?

A :

```
<template for:each={contacts} for:item="contact">
        <li key={contact.Id}>
            {contact.Name}, {contact.Title}
        </li>
</template>
```

Q : How to write expression in component?

A :

```
<!--todoItem.html-->
<template>
      {itemName}
</template>

// todoItem.js
import { LightningElement, api } from 'lwc';
export default class TodoItem extends LightningElement {
    @api itemName;
}
```

Q : How to call Controller function in component?

A :

HTML

```
<lightning-input type="checkbox" label="Show details"
onchange={handleChange}></lightning-input>
```

JS

```
handleChange(event) {
    this.areDetailsVisible = event.target.checked;
}
```

Q : @AuraEnabled(cacheable=true)

A :

- AuraEnabled annotation exposes the method to lightning components and caches the returned list on the client.

Q : What is Lifecycle Hooks?

A :

- Lightning Web Components provides methods that allow you to “hook” your code up to critical events in a component’s lifecycle.
- These events include when a component is:
  - Created - constructor( )
  - Added to the DOM - connectedCallback( )
  - Rendered in the browser - renderedCallback( )
  - Encountering errors - errorCallback( )
  - Removed from the DOM - disconnectedCallback( )

Q : What is Lifecycle Flow?

A :

- Constructor Called on Parent
- Public Property value of Parent updated
- Parent Inserted into the DOM
- connectedCallback( ) called on Parent
- Parent rendered
  - Constructor called on child
  - Public Property value of child updated
  - Child inserted into the DOM
  - connectedCallback( ) called on child
  - Child rendered
  - renderedCallback( ) called on Child
- renderedCallback( ) called on Parent

Q : What are three ways to work with Salesforce Data?

A :

- Use Base Lightning Components Built on Lightning Data Service
  - lightning-record-form
  - lightning-record-view-form
  - lightning-record-edit-form
- Use Lightning Data Service Wire Adapters and Functions
  - lightning/ui*Api module
- Use Apex

Q : Lightning Data Service?

A :

- Lightning Data Service manages data for, changes to a record are reflected in all the technologies built on it. Whereas, data from Apex is not managed, you must refresh the data.
- Lightning Data Service does a lot of work to make code perform well:
  - Loads record data progressively.
  - Caches results on the client.
  - Invalidates cache entries when dependent Salesforce data and metadata changes.
  - Optimizes server calls by bulkifying tha deduping requests.

Q : Base Component Comparison?

A :

-

Feature	lightning-record-form	lightning-record-view-form	lightning-record-edit-form
Create Records	Yes		Yes
Edit Records	Yes		Yes
View Records	Yes	Yes	
Read-Only Mode	Yes	Yes	
Layout Types	Yes		
Multi Column Layout	Yes	Yes	Yes
Custom Layout for Fields		Yes	Yes
Custom Rendering of Record Data		Yes	Yes

Q : Wire Service?

A :

- The wire service provisions an immutable stream of data to the component.
- Each value in the stream is a newer version of the value that precedes it.
- Objects passed to a component are read-only.
- To mutate the data, a component should make a shallow copy of the objects it wants to mutate.

Q : Wire Service Syntax?

A :

```
import { adapterId } from 'adapterModule';
@wire(adapterId, adapterConfig)
propertyOrFunction;
```

- **adapterId (Identifier)—** The identifier of the wire adapter.
- **adapterModule (String)—** The identifier of the module that contains the wire adapter function, in the format namespace/moduleName.
- **adapterConfig (Object)—** A configuration object specific to the wire adapter. Configuration object property values can be either strings or references to objects and fields imported from @salesforce/schema.
- **propertyOrFunction—** A private property or function that receives the stream of data from the wire service. If a property is decorated with @wire, the results are returned to the property's data property or error property. If a function is decorated with @wire, the results are returned in an object with a data property and an error property.

Q : Why to import References to Salesforce Objects and Fields?

A :

- When you use a wire adapter in a lightning/ui*Api module, we strongly recommend importing references to objects and fields.
- Salesforce verifies that the objects and fields exist, prevents objects and fields from being deleted, and cascades any renamed objects and fields into your component's source code.
- It also ensures that dependent objects and fields are included in change sets and packages.
- If a component isn’t aware of which object it’s using, use strings instead of imported references. Use getObjectInfo to return the object’s fields.
- All wire adapters in the lightning/ui*Api modules respect object CRUD rules, field-level security, and sharing.
- If a user doesn’t have access to a field, it isn’t included in the response.

Q : How to import References to Salesforce Objects and Fields?

A :

- import POSITION_OBJECT from '@salesforce/schema/Position__c';
- import ACCOUNT_OBJECT from '@salesforce/schema/Account';
- import POSITION_LEVEL_FIELD from '@salesforce/schema/Position__c.Level__c';
- import ACCOUNT_NAME_FIELD from '@salesforce/schema/Account.Name';
- import POSITION_HIRINGMANAGER_NAME_FIELD from '@salesforce/schema/Position__c.HiringManager__r.Name__c';
- import ACCOUNT_OWNER_NAME_FIELD from '@salesforce/schema/Account.Owner.Name';

Q : How to get current record id in lightning web component?

A :

- Create a property named as recordId and decorate it with @api decorator.

Q : How can we deploy lightning web components?

A :

- Lightning components can be deployed like any other component using change set, ANT migration tool, Gearset, Copado or other migration tool.

Q : Communicate with Events in LWC?

A :

- Lightning web components dispatch standard DOM events.
- Components can also create and dispatch custom events.
- You can use events to communicate up the component containment hierarchy.
- Create and dispatch events in a component's JavaScript file.
- To create an event, use the CustomEvent( ) constructor. To dispatch an event, call the EventTarget.dispatchEvent( ) method.
- To listen an event use component's HTML template. to handle events, define methods in the component's JavaScript class.

Q : Use of Lightning Message Service?

A :

- To communicate between components within a single lightning page or across multiple pages, use Lightning message service to communicate over a Lightning message channel.
- The advantage over pubsub module is that message channels aren’t restricted to a single page.
- Any component in a Lightning Experience application that listens for events on a message channel updates when it receives a message.
- It works between Lightning web components, Aura components, and VF Pages in any tab or in any pop-out window in Lightning Experience.

Q : Use of pubsub Module?

A :

- In containers that don’t support Lightning Messaging Service, use the pubsub module.
- In a publish-subscribe pattern, one component publishes an event. Other Component subscribe to receive and handle the event.
- Every component that subscribes to the event receives the event.
- The pubsub module restricts events to a single page.

Q : Where we can use Lightning Web Components?

A :

- Distribute Components on AppExchange
- Lightning App Builder
- Flows
- Experience Builder
- Utility Bar
- Create Components for Outlook and Gmail Integrations
- Quick Actions
- Standalone Aura Apps
- Visualforce Pages
- Custom Tabs