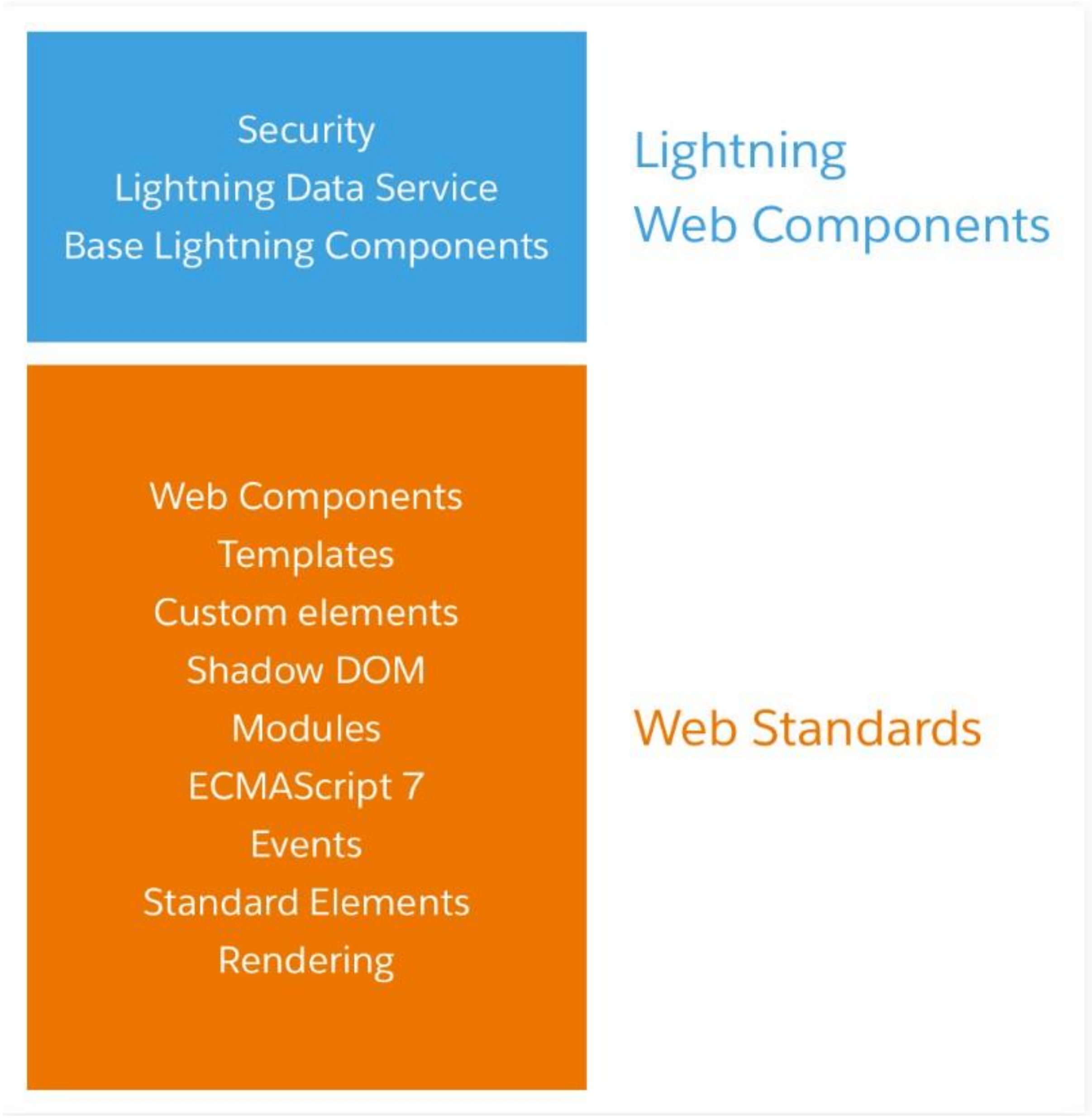
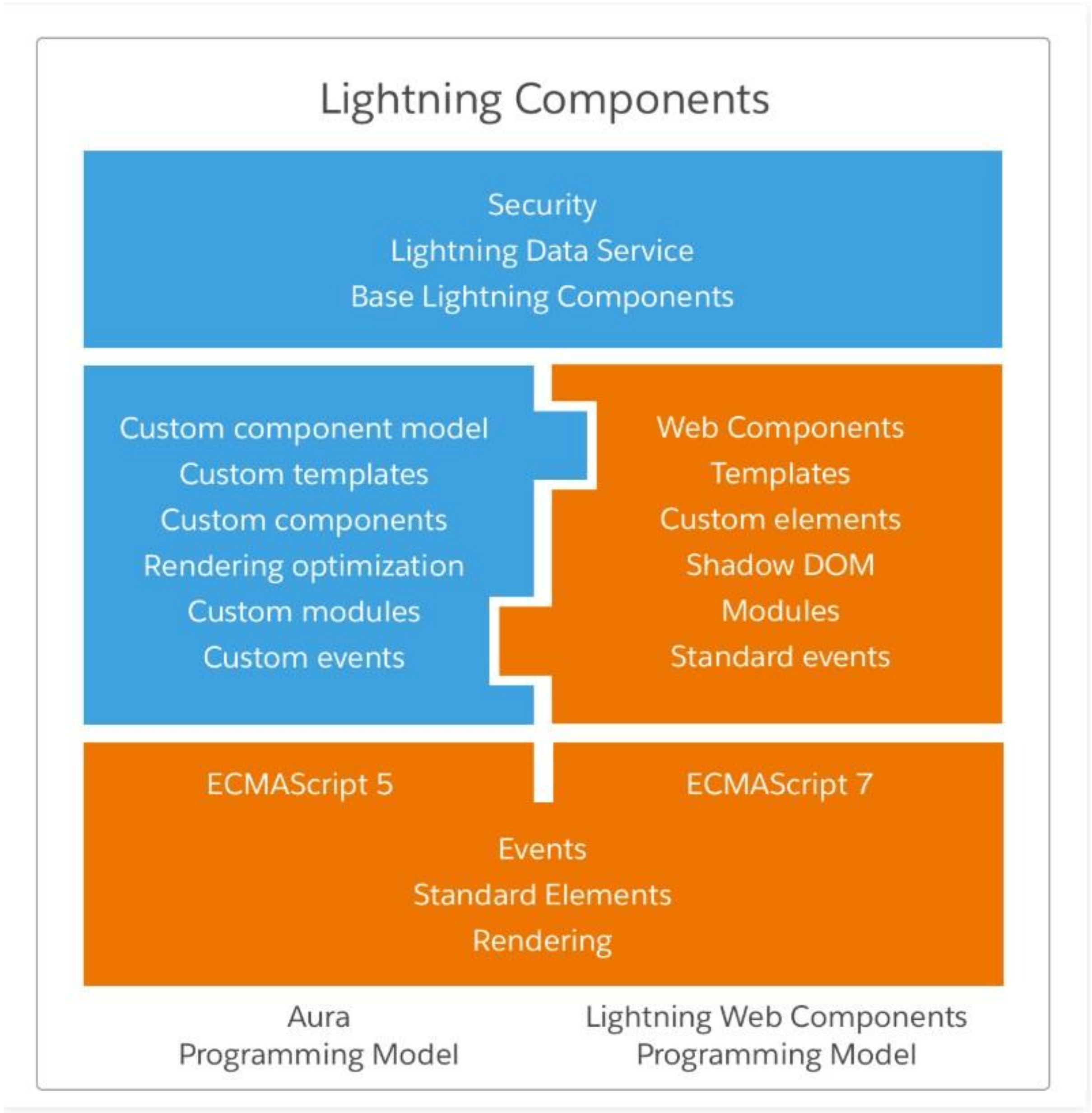
1 What are Lightning Web Components (LWC)?

We can build Lightning components using two programming models: Lightning Web Components, and the original model, Aura Components. Lightning web components are custom HTML elements built using HTML and modern JavaScript. Lightning web components and Aura components can coexist and interoperate on a page. To admins and end users, they both appear as Lightning components.



Lightning Web Components uses core Web Components standards and provides only what's necessary to perform well in browsers supported by Salesforce. Because it's built on code that runs natively in browsers, Lightning Web Components is lightweight and delivers exceptional performance. Most of the code we write is standard JavaScript and HTML.



2 What is the file structure of Lightning Web Components
myComponent folder

myComponent.html

myComponent.js

myComponent.js-meta.xml

myComponent.css

myComponent.svg

The folder and its files must follow these naming rules.

- Can’t contain a hyphen (dash)
- Must begin with a lowercase letter
- Can’t include whitespace
- contain only alphanumeric or underscore characters
- Can’t end with an underscore
- Must be unique in the namespace
- Can’t contain two consecutive underscores

3. How can you display components HTML conditionally
To render HTML conditionally, add the if:true|false directive to a nested <template> tag that encloses the conditional content.

4 How we can bind data in LWC
In the template, surround the property with curly braces, {property}. To compute a value for the property, use a JavaScript getter in the JavaScript class, get property(){}. In the template, the property can be a JavaScript identifier (for example, person) or dot notation that accesses a property from an object (person.firstName). LWC doesn’t allow computed expressions like person[2].name[‘John’].

```
<!-- hello.html -->

<template>

   Hello, {greeting}!

</template>

// hello.js

import { LightningElement } from 'lwc';



export default class Hello extends LightningElement {

   greeting = 'World';

}
```

Don’t add spaces around the property, for example, { data } is not valid HTML.

5. How we can pass data from HTML to JS controller?
We can use the onchange attribute to listen for a change to its value. When the value changes, the handleChange function in the JavaScript file executes. Notice that to bind the handleChange function to the template, we use the same syntax, {handleChange}

```
<!-- helloBinding.html -->

<template>

   <p>Hello, {greeting}!</p>
```

```
    <lightning-input label="Name" value={greeting} onchange={handleChange}></lightning-input>
</template>
// helloBinding.js
import { LightningElement } from 'lwc';


export default class HelloBinding extends LightningElement {
    greeting = 'World';


    handleChange(event) {
        this.greeting = event.target.value;
    }
}
```

We can use same eventHandler with multiple fields as well.

```
<lightning-input name='firstName' label="First Name" onchange={handleChange}></lightning-input>
<lightning-input name='lastName' label="Last Name" onchange={handleChange}></lightning-input>
```

In Controller

```
handleChange(event) {
        const field = event.target.name;
        if (field === 'firstName') {
            this.firstName = event.target.value;
        } else if (field === 'lastName') {
            this.lastName = event.target.value;
        }
    }
```

6. How we can iterate list in Lightning Web Component (LWC)
We have two options available here:

for:each

When using the for:each directive, use for:item="*currentItem*" to access the current item. This example doesn't use it, but to access the current item's index, use for:index="*index*".

To assign a key to the first element in the nested template, use the key={*uniqueId*} directive.

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

Iterator

To apply a special behavior to the first or last item in a list, use the iterator directive, iterator:*iteratorName*={*array*}. Use the iterator directive on a template tag.

Use *iteratorName* to access these properties:

- value—The value of the item in the list. Use this property to access the properties of the array. For example, *iteratorName*.value.*propertyName*.
- index—The index of the item in the list.
- first—A boolean value indicating whether this item is the first item in the list.
- last—A boolean value indicating whether this item is the last item in the list.

```
<template iterator:it={contacts}>
<li key={it.value.Id}>
<div if:true={it.first} class="list-first"></div>
{it.value.Name}, {it.value.Title}
<div if:true={it.last} class="list-last"></div>
</li>
</template>
```

7. Can we display multiple templates
Yes we can, We can import multiple HTML templates and write business logic that renders them conditionally. This pattern is similar to the code splitting used in some JavaScript frameworks.

Create multiple HTML files in the component bundle. Import them all and add a condition in the render() method to return the correct template depending on the component’s state. The returned value from the render() method must be a template reference, which is the imported default export from an HTML file.

```
// MultipleTemplates.js



import { LightningElement } from 'lwc';

import templateOne from './templateOne.html';

import templateTwo from './templateTwo.html';



export default class MultipleTemplates extends LightningElement {



   templateOne = true;



   render() {

      return this.templateOne ? templateOne : templateTwo;

   }



   switchTemplate(){

      this.templateOne = this.templateOne === true ? false : true;

   }

}
```

Component File structure

```
myComponent

   ├──myComponent.html
```

```
├──myComponent.js
├──myComponent.js-meta.xml
├──myComponent.css
├──secondTemplate.html
└──secondTemplate.css
```

8. What are the public properties in Lightning Web Component
Public properties are reactive. If the value of a public property changes, the component rerenders. To expose a public property, decorate a field with @api. Public properties define the API for a component.

9. How to set Property from Parent component to child component
To communicate down the containment hierarchy, an owner can set a property on a child component. An attribute in HTML turns into a property assignment in JavaScript.

```
// todoItem.js

import { LightningElement, api } from 'lwc';

export default class TodoItem extends LightningElement {

   @api itemName;

}

<c-todo-item item-name="Milk"></c-todo-item>

<c-todo-item item-name="Bread"></c-todo-item>
```

10. How we can pass data from parent component to child component
LWC support one way data transfer from parent to child. A non-primitive value (like an object or array) passed to a component is read-only. So the component cannot change the content of the object or array. As a result if the component tries to change the content, we will get errors in console.

We can pass primitive data types as most of the components supports this.

……………………………………………………………………………………………………

1. What is Lightning Framework in Salesforce?

Answer: Salesforce Lightning is an app development framework developed by the salesforce team to simplify business processes for their users. It is a UI framework for creating single-page applications for desktop and mobile devices. Lightning Components are built using two programming models: the Aura Components model and the Lightning Web Components Model.

Note: This is one of the most important question from LWC Interview Questions.

2. What are the components in Salesforce Lightning?

Answer: The components in Salesforce Lightning include:

- **Lightning Component Framework:** a javascript framework that allows the development of recyclable components for personalizing the Salesforce1 Mobile App and lightning experience.
- **Lightning Experience:** a new and modern user experience and interface for Salesforce.
- **Lightning Exchange:** a part of AppExchange, that has over 70 partner components to start the development process.
- **Lightning Design System:** It helps build applications without writing a single line of CSS but with the look and feel of Lightning Experience.
- **Visual Building tools:** relates to drag-and-drop features.
  Note: This is one of the most important question from LWC Interview Questions.

3. What is the use of aura: namespace?

Answer: In Salesforce LWC, the aura: namespace includes all the main building blocks to define applications and components for the application.

Note: Important question for LWC and Salesforce Lightning

4. What are the different types of events of Salesforce Lightning components?

Answer: In Salesforce Lightning components, we observe three main event types. They are as follows:

- System Events.

- Application Events.
- Component Events.

5. What are the component bundles of the Lightning Components?

Answer: The below bundles are the component bundles of the Lightning Components:

- Controller.
- Documentation.
- Style.
- Renderer.
- Helper.

Note: This is one of the most important question from LWC Interview Questions.

6. How can we use Lightning components with Salesforce1 Mobile App?

Answer: We can use Lightning Components with the Salesforce1 Mobile app by creating a traditional Lightning tab that refers to the component. This Lightning tab can be added to the Salesforce1 Mobile Navigation.

7. What is the difference between a component event and an application event?

Answer: A component event is used for interaction between the parent and child component, whereas the application event sends modifications in the part to a broad audience. The change in the child component can be communicated to the parent component through the component event. In contrast, in the application event, the part that has registered for this event will get an alert.

Note: Important question for LWC and Salesforce Lightning.

8. Explain Namespace in the Salesforce Lightning Components.

Answer: In Salesforce Lightning Components, the namespace is used to group components into an organization. C is the default namespace. If the namespace is created with any name for an organization, then that namespace can be used for the Lightning components.

9. Where can the Lightning components be used?

Answer: Lightning Components can be used in:

- I was dragging and dragging in the community and Lightning App Builder.
- I am adding it to the Lightning pages.
- I am starting it as a quick action.
- We are creating stand-alone applications.

10. Differentiate between Salesforce Classic and Salesforce Lightning.

Answer: For Salesforce Classic, we need to hire a Salesforce Developer to operate, whereas Salesforce Lightning is much easier to work with. And also, with Lightning, you don't need Visualforce for every task. Compared to Salesforce Classic, Salesforce Lightning has advanced security features.

Note: Important question for LWC and Salesforce Lightning

Note: This is one of the most important question from LWC Interview Questions.

11. What are Component events?

Answer: Component events are the events that have child components dismissed and parent components handled. Component events are used when we need to send some value from a child component to the parent component.

Note: Important question for LWC and Salesforce Lightning

12. What are Application events?

Answer: Other than component events, we can also have application events, which can be dismissed and handled by any component. An application event does not require any type of relationship among the members. These components should be a part of one application.

13. What are bound and Unbound Expressions?

Answer: The data binding in our application is done using the bound and Unbound expressions in LWC. The value can be passed from an attribute of a parent component to an attribute of a child component in our application whenever a child component is called from the parent component.

Note: Crucial query for Salesforce Lightning and LWC

14. What are the lifecycle hooks in LWC?

Answer: A lifecycle hook is a callback method triggered at a specific phase of a component instance's lifecycle. The following themes are supported in LWC:

- Constructor.
- Connectedcallback.
- Renderedcallback.
- Disconnectedcallback.
- Error callback.

15. What tools are available in Salesforce Lightning?

Answer: A user interface tool called Lightning Schema Builder is used to create and view fields, objects, and relationships.

- Lightning Connect: This integration tool makes it easier for the Force.com application to ingest data from the external source, which complies with the OData specification.
- A user interface (UI) tool called Lightning Process Builder is used to create and visualize automated business processes.
- Lightning Component Framework: It includes add-ons and parts that make it possible to customize the Salesforce1 mobile app, create standalone apps, and create reusable components.
- The most recent user interface tool, Lightning App Builder, enables the speedy creation of Lightning apps using components provided by Salesforce platform developers.

Note: This is one of the most important question from LWC Interview Questions.

16. What function does Javascript serve in the LWC?

Answer:

- Javascript defines the behavior of HTML properties.
- The import statement must be used if a function, class, or variable defined in the module has to be imported.
- The ES6 modules are Javascript files in the LWCs, and everything defined in the module is local.
- The main module of Lightning Web Components is lows. LightningElement is imported from the lwc module using the import statement.

Note: Important question for LWC and Salesforce Lightning

17. Describe Scratch org.

Answer: Useful for testing and development, the Scratch org is an extendable Salesforce org. Salesforce code and metadata are deployed using a source-driven, disposable model.

Since a scratch org may be customized, developers can mimic many Salesforce editions with various features and preferences. The scratch org configuration file can be sent to other team members to facilitate development.

Additional options include installing packages and deploying test data that is artificial or fictitious. The Scratch org can be created for 30 days, after which it is deactivated, while it's default length is seven days.

Note: This is one of the most important question from LWC Interview Questions.

18. What does Salesforce Lightning's Lightning Locker do?

Answer: The security architecture for the Lightning components is solid and practical, thanks to the Lightning Locker service. By separating Lightning components from one namespace from those from another, it improves security. It encourages best practices and improves the code's maintainability by restricting access to supported APIs and blocking it for internal framework resources that haven't been made public.

Note: This is one of the most important question from LWC Interview Questions.

19. Describe navigation in Lightning.

Answer: To browse to a page reference or generate a URL from the provided page reference, utilize Lightning: navigation. The page reference object needs to be defined to navigate. The following supported functionalities can be navigated:

- Lightning Component
- Object Page
- Named Page
- Navigation Item Page
- Web Page
- Knowledge Article
- Record Relationship Page
- Record Page

Note: Important question for LWC and Salesforce Lightning

20. What kinds of characteristics are there that can be utilized to store values?

Answer: Values are stored in the following attributes:

- Date
- Integer
- Decimal
- Datetime
- Map
- Set
- Boolean
- String

Note: This is one of the most important question from LWC Interview Questions.

21. What are the various stages of the transmission of application events?

Answer:

- Capture Phase
- Bubble Phase
- Default Phase

22. Regarding Salesforce Lightning, distinguish between component events and application events.

Answer: Parent and child communication is facilitated by component events. Component Events can inform the parent component of a change in the child component.

A significantly larger audience is informed of changes to the component through Application Events. The difference is communicated to every participant who has registered for the event.

Note: This is one of the most important question from LWC Interview Questions.

23. What distinguishes an application event from a component event?

Answer: Parent-child interaction is facilitated through component events. The component event can inform the parent of the change in the child component.

Modifications to the component are broadcast to a large audience using application events. An alert will be sent to the registered members for this event.

24. How to transmit data between a parent and child component?

Answer: LWC allows for one-way data transfer from parent to child. Any non-primitive value supplied to a component (such as an object or an array) is read-only. The object's or array's content cannot, therefore, be changed by the component. As a result, we will see problems in the console if the component tries to update the content.

Primitive data types can be passed because most components allow for this.

25. Is it possible to use LWC JS to call third-party APIs?

Answer: By default, JavaScript programs cannot call external APIs or establish WebSocket connections. Add a distant site as a CSP Trusted Site to do this.

The Lightning Component architecture uses a W3C standard called Content Security Policy (CSP) to limit the sources of content that can be loaded onto a website. As a result, JavaScript code is prohibited from calling APIs by default under the CSP policy. Adding CSP Trusted Sites will alter the policy and the CSP header's content.

Note: This is one of the most important question from LWC Interview Questions.

Conclusion

In this article, we have extensively discussed the most asked interview questions related to the LWS.

How well you prepare for a job interview will greatly affect your chances of success. Researching the position and the firm is a key component of interview preparation, as is carefully analyzing your responses to the interview questions. Check out this video to understand better.

…………………………………………………………………………………………

1. **Can we call the @AuraEnabled function in the apex class using wire ?**
   Ans: *Function also needs to have cacheable = true annotation ie should be like @AuraEnabled(cacheable = true)*
2. **What do you mean by cacheable = true annotations ?**
   Ans: *First of all when you mark function as cacheable = true it can only retrieve data i.e. it cant have any DML.*
   It is used to improve component performance by quickly showing cached data from client side storage without waiting for server trip.
   Remember to refresh the stale data provisioned by Apex we have to use refreshApex as LDS doesn't manage data coming from Apex (In case of wired service)
   Read more: What is Salesforce—Everything You Should Know About This Leading CRM Provider
3. **@wire(getBoats,{boatTypeId : this.boatTypeId})**
   getBoats(error,data){}
   Will the above code deploy? (Assuming all variables and apex class function exists).
   Ans: *No it wont when passing a variable in wire you should always use $ along with variable, it should be written like @wire(getBoats,{boatTypeId : '$boatTypeId})*
4. **Why do we use $ when passing property in wire function, what does it mean?** [Latest Salesforce lightning interview questions and answeres]
   Ans: *$ prefix tells the wire service to treat values passed as a property of the class and evaluate it as this.propertyName and the property is*

reactive. If the property's value changes, new data is provisioned and the component renders.

5. **See the below code and answer the following question:** [Scenario based Salesforce Lightning Interview question with relevant answer]
   If I want to refresh the wired data in the above function, can I call refreshApex(this.someVar) ?

```
@wire(getBoats,{boatTypeId : '$boatTypeId'})
getBoats({error,data}){
if(data){
this.someVar = data;
this.error = undefined;
}
else if(error){
this.error = error;
this.someVar = undefined ;
}
}
```

Ans:

No we cant call refreshApex(this.someVar) rather refreshApex is called on whole result provisioned from the wire service not just the data part, we will have to rewrite it as below :

```
@wire(getBoats,{boatTypeId : '$boatTypeId'})
getBoats(result){
this.mainResult = result;
if(result.data){
this.someVar = result.data;
this.error = undefined;
}
else if(result.error){
this.error = result.error;
this.someVar = undefined ;
}
}
```

Now we can refresh data as refreshApex(this.mainResult)

6. **Can we call a wire function inside a javascript function like below :**

```
searchBoats(event){

@wire(getBoats,{boatTypeId: '$boatTypeId'}

getBoats(data,error){

}

}
```

Assume searchBoats is being called on click of button? Will I be able to deploy the code ?

Ans: *No you cant call it like this , code will not deploy. You will receive error as leading decorators must be attached to class which means decorators like wire can be directly under class only not inside any other function.*

Similarly if you try to define variable with @track or api decorator inside a function it will fail.

7. **When is the wire method/property called in the lifecycle of a component ?**

Ans*: Wired service is called right after component is created ie after constructor and is again called when parameter that you are passing is made available*.

8. **What are lifecycle hooks in LWC ?**

Ans*: A lifecycle hook is a callback method triggered at a specific phase of a component instance's lifecycle.*

There are following hooks supported in LWC :

Constructor : Called when the component is created.

Connectedcallback : Called when the element is inserted into a document. This hook flows from parent to child.

RenderedCallback : Called after every render of the component. This lifecycle hook is specific to Lightning Web Components, it isn't from the HTML custom elements specification. This hook flows from child to parent. Ie its not part of HTMLElement rather defined in LightningElement.

Disconnectedcallback : Called when the element is removed from a document. This hook flows from parent to child.

Errorcallback : Called when a descendant component throws an error. The error argument is a JavaScript native error object, and the stack argument is a *string. This lifecycle hook is specific to Lightning Web Components, it isn’t from the HTML custom elements specification*

9. **Is wire method called multiple times during lifecycle of component ? (True / False)**

Ans*: True*

10. **What is the difference in below two codes , will they both compile ? Give same results ?**
11. [tricky Salesforce Lightning Interview question with relevant answer]

```
Code 1 :

@wire(getBoats)

getBoats({data,error}){

if(data)

console.log(‘print here’);

Else if(error)

console.log(‘print in else’);

}

@wire(getBoats,{})

getBoats({error,data}){

if(data)

console.log(‘print here’);

Else if(error)

console.log(‘print in else’);

}
```

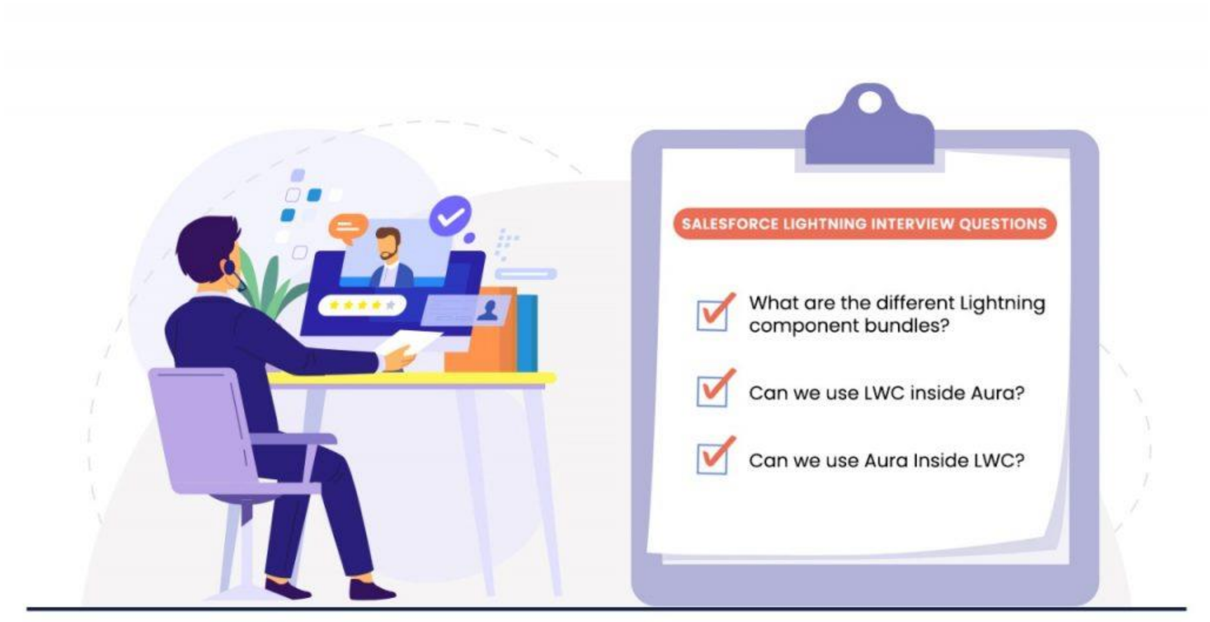
Ans: Both will compile they are same.

12. **Is it mandatory to use data,error only in wired method, can I use some other variable like below :** Scenario based interview question with answers

```
@wire(getBoats)

getBoats({myData,myError}){

if(mydata)

console.log(‘i am in data’);

else if(myError)

console.log(‘I am in error’);
```

}

Will the code deploy successfully or I will receive an error ?


SALESFORCE LIGHTNING INTERVIEW QUESTIONS
What are the different Lightning component bundles?
Can we use LWC inside Aura?
Can we use Aura Inside LWC?

Salesforce Lightning LWC interview questions and answers

Ans: *We cant use any other variable name other than data, error they are hardcoded in wire service. Code will successfully deploy but wire service wont be able to fetch any data.*

13. **What is the difference between event.StopPropogation() and Event.preventDefault()?**
    Ans: *stopPropagation prevents further propagation of the current event in the capturing and bubbling phases.* preventDefault prevents the default action the browser makes on that event.
14. **If we add required attribute in lightning-input , will I get an error on leaving the field empty ?**
    Ans: *No unless you also add logic in backend javascript file to actually throw an error using checkValidity and reportValidity.*
15. **Are quick actions supported for LWC components ?**
    Ans: *Quick actions are supported by LWC in Summer 21 or later orgs. LWC quick actions are only supported on record pages.*
16. **How can i reference record ID of page in my component which is fired from quick action.**
    Ans: There are two types of quick actions in LWC :
    Screen actions : Normal action which open a modal

Headless actions : Which dont have any html file rather just logic written in js ie no modal window will come up
In case of screen actions we will be able to get recordID by just defining recordID as public property in the component.
For headless action you have to define @api invoke method which is auto called and recordID will be set after this function is called.

17. **What is a promise in async transactions? What are it different stages.** [Latest Salesforce lightning interview questions and answers]
Ans:
Promise is object returned in async transaction which notifies you about completion or failure of transaction.
For example when you call apex imperatively it returns you a promise object on the basis of object returned execution either goes into (then) ie transaction was successful or (catch) which means transaction failed.

    Stages of promises are :
    Pending: Waiting for result.
    Fulfilled: Promise results in success.
    Rejected: Promise result in failure.
18. **What is the difference between Promise and Promise.all**
Promise.All takes in multiple promises in it and returns a single promise object.
Promise.all is used when I want to make sure once all promises are resolved then only execute then block.
19. **What are web components, Is LWC based on web components ?**
Ans: In simplest language , web components can be explained as it allows you to create and re-use custom components as html tags in a component with their functionality encapsulated from rest of your code.
Web components has 4 pillars which make it so amazing.
HTML Template : user defined templates which are rendered until called upon.
Custom Elements : With this we can embed as components merely as html tags in our template.
Shadow DOM : With this we can separate DOM of each custom element from each other to make sure nothing from any components leaks into each other.
HTML Imports : You can import other html documents in another

html document for example we can import multiple templates into one component then use render function to display a template.
Yes LWC is based on web components
If you look at LWC architecture Salesforce only adds security , LDS and base components rest is based on web components and es 7.

20. **Why do we extend LightningElement ?**
    Ans: *LightningElement is custom wrapper on HTMLElement which actually contains all the lifecycle hooks methods over which Salesforce did some customization*.
21. **When we create new component why does it say export default ComponentName ?**
    Ans: *Because of export default component only we are able to embed one component into another*
22. **How do I retrieve elements on the basis of ID?**
    Ans: *We should never use "id" in lwc as id that you will define in LWC may get transformed into something entirely different on rendering, rather use data-id in your tags to reference them.*
23. **How to send data from Parent to Child in LWC ?**
    Ans: *For parent to child communication you just need to expose your function or attribute as @api then in your parent component you can use querySelector to actually query the child component access exposed attribute or method*
24. **If we parent component A and there are two component B and C as child components. How can we communicate between B and C ?**
    Ans: *We should fire up event from child component B to parent A then from A call attribute / function exposed (as @api) and pass data into C component.*
25. **What does composed = true does in an event ?**
    Ans: *These type of events can bubble up inside dom and also able to cross shadow boundary.*
26. **When you fire an event, can i capture it in same template/ component ?**
    Ans: No
27. **Is bubble : false and composed : true possible in an event ?**
    Ans: No
28. **What is the difference between below :**
    a. Bubble : false and composed : false

b. Bubble : true and composed : true
c. Bubble : true and composed : false

29. **What are callback functions in LWC ?**
Ans: *Callback functions are nothing but functions passed as parameter into other functions.*
30. **Are callback functions synchronous or asynchronous ?**
Ans: *Functions are neither async or sync in themselves rather depend on where are they being passed for example if a function in passed into reduce method is sync but if function is being passed into setTimeout function its async.*
31. **What is callback hell ?**

Ans: In most simple words callback hell occurs when there are many functions you want to call async and you end up puting them one side each another and as the code grows it becomes very difficult to read for example :

```
getData(function(x)

{

getMoreData(x, function(y)

{

getMoreData(y, function(z)

{ ... }

);

});

});
```

32. **What are decorators in LWC (Lightning web components) in Salesforce? Latest interview question.**
Ans: *The Lightning Web Components programming model has three decorators that add functionality to a property or function. There are 3 decorators for LWC*
@track , @wire, @api
33. **When do I use @track on a property ? Do I still need it considering all properties are by default reactive now?**
Ans: *After Spring 20 all the properties are made by default reactive ie we dont need @track for primitive properties. We still need it for array or object type properties*
34. **Can I use multiple decorators on one property ?**
Ans: *No we cant use multiple decorators on same property.*

35. **Can I deploy component with empty css file ?**
    Ans: *No we cant do that*
36. **What is difference between var and let in JS ?**
    Ans: *In simple words difference is var is function scoped while let is block scoped*
    Var allows you to redeclare same variable while let will throw an error
    Var variables are hoisted that is you can access them even before they are declared in code its just they will return undefined value while with let it will throw an error.
37. **Is there any difference in how we add LWC component inside another LWC component and inside AURA component ?**
    Ans: *Difference is how they are added in the component.*
    LWC follows kebab case ie if you are adding any LWC component inside another component you will have to add in kebab case form while in AURA it will be added without kebab case for example :
    We have component with name "childLWCComponent"
    In LWC it will be added as <c-child-l-w-c-component/>
    In Aura it will be added as <c:childLWCComponent/>
38. **What is LMS ?**
    Ans: *LMS is defined as the standard publish-subscribe library that enables communication with DOM across the components be it Visualforce Pages, Aura components, and Lightning Web Components (LWC) all can use it to publish message and listen to messages published by others*.
39. **Do we have application events in LWC?**
    Ans: *We dont have application event as such in LWC like Aura rather we have LMS in LWC to communicate between components which are not part of same hierarchy*
40. **How can we navigate user from LWC component to record detail page?**
    Ans: *Can be done using NavigationMixin service*
41. **Do we have force:createRecord equivalent in LWC?**
    Ans: *Using navigation mixin only you can also create new record where you define object , action as new and pass any default values you want to set.*
42. **What is render() , is it part of lifecycle hook? Why do we use it ?**
    Ans: *Render is not part of lifecycle hook its a protected function, we*

only use it if we have imported multiple templates in our component and we want to render particular template on meeting certain criteria.

43. **Can I get current user ID in LWC without apex?**
Ans: *Yes we can get current user ID without apex by simply importing import Id from '@salesforce/user/Id'*
44. **What is the difference between '==' and '===' ?**
Ans: Both are used to compare two variables but == doesnt take data type into consideration and does type coercion ie interpreter tries to automatically convert data types to match the values while in === case if data type is not same it will always return false.
For example :
2=="2" will return True (Type coercion happens)
2==="2" will return False (No Type coercion)
45. **What is String interpolation ?** [important interview questions and Salesforce Lightning]
Ans: It means simply when string literal is evaluated all the placeholders added into it are calculated at run time and replaced in string with values. Place holder can be anything a variable , expression even a function call. In javascript its performed using (backtick).
For example:
const greeting = 'Hello';
const who = 'World';

const message = ${greeting}, ${who}!`;
message; // => 'Hello, World!'
46. **What are template literals ? What is the use?**
Ans: I*n javascript string interpolation is performed using template literals.*
Template literals are created using (`) backtick character apart from string interpolation they also are used for adding multi-line strings without having to use "\n"

For example :

```
console.log('string text line 1\n' +

'string text line 2');

console.log(`string text line 1

string text line 2`);

//Both will result in same output
```

47. **Can i call function annotated with @AuraEnabled(cacheable= true) imperatively ?**
Ans: Yes
48. **Can we do DML in method annotated with @AuraEnabled(cacheable= true)?**
Ans: *No we cant do DML inside any method annotated with cacheable = true , you will receive an error as DMLLimit Exception.*
49. **How to refresh cache when calling method imperatively ?**
Ans: *We have to use getRecordNotifyChange(RecordIds) which refreshes the LDS cache providing you the latest data this will work only if cacheable = true was there*.
Otherwise we will have to call the function again from our js to get the latest data.
50. **When do we face error of "Cant assign to Read only property" in LWC?**
Ans: *This error usually occurs when you are trying to make changes to a property which is marked as @api , ideally you should clone the value then make the changes to it.*
51. **How can I evaluate expression in situation like <template if:true = {someVariable % 7==0}**
Ans: *We cant add expressions directly in html file rather what we can do is create getter which will evaluate value for us which we can use in html like below :*

```
get isMod7() { return this.index % 7 == 0; }

<template if:true ={isMod7}
```

52. **Why do we put constants outside of class in LWC?**
Ans: *We cant put constants inside a class or function in javascript its illegal for example below piece of code will throw you an error*

```
export default class someClass extends LightningElement {

const someVar = 'someValue' ;

}
```

53. **How to query all lightning-input , combobox, radio buttons using one querySelector or do I have to use multiple ?**
Ans: *We can query all of them using one query selector only no need to write multiple for each tag. We can pass all tags (,) separated to query all.*

const allValid = […this.template.querySelectorAll(‘lightning-input,lightning-combobox,lightning-radio-group’)];

If you are wondering why are we using (…) spread operator before querySelectorAll its because querySelector returns you the nodelist and using spread operator we convert it to array of items otherwise we wouldnt be able to use array functions like map or reduce.

54. What is reduce method in JS ?

Reduce method calls the reducer function (callback function) for each item of the array to reduce the whole array into single value.

Remember it doesnt change the actual array

Remember it doesnt change the actual array.

```
array.reduce((accumulator , currentVal , currentIndex , array)=>{

},0)
```

As the name suggests accumulator collects all the value

currentVal - denotes the current val of array

currentIndex - denotes the index of current val of array [optional]

Array - array object being passed. [optional]

Remember : If default value is passed accumulator is set to default value and index starts with 0 , if no default value is passes accumulator is assigned 0th index value and currentIndex starts from 1.

55. What would be the output of : 1+3+”4” ?

//A 44 because type coercion will take place.

statu

```
get statusVal(){

return statusVal ;

}

set statusVal(value){

this.statusVal = value;

}

renderedCallback(){

this.statusVal = ‘ABC’

}
```

***Ans**: Yes you will receive an error "Maximum depth exceeded" as in set function you are again setting statusVal itself which means it will keep on setting itself (infinitely).*