Lightning Experience Development

Understand how our new UX impacts Apex, Visualforce, APIs, and more.

UNIT - User Interface Development Considerations

User Interface Development Considerations

Lightning Experience is a brand new, amazing opportunity for developers on the Force.com platform. In this module we’re going to look at many of the aspects that make Lightning Experience different for developers. Visualforce renders the page on the server. Before we get to that, though, we want to have a quick chat about a couple higher-level concerns. Visualforce will be fully supported by Salesforce for years to come.

Visualforce is conceptually similar to other page-centric technologies like PHP, ASP, JSP, and Ruby on Rails. Visualforce is designed primarily for page-centric web apps. The Visualforce framework provides a robust set of tags that are resolved at the server-side and that work alongside standard or custom controllers to make database and other operations simple to implement.

Lightning components are part of the new Salesforce user interface framework for developing dynamic web applications for desktop and mobile devices. Visualforce pages still work on Salesforce. It doesn’t matter if you’re using the new interface or your old friend Salesforce Classic—Visualforce is able to work with both. You don’t have to convert any existing Visualforce pages to keep using them for a long time.

All of these are good use cases for Lightning Components.

1) Using Lightning components developing an app Salesforce1.
2) Developing a highly interactive app with an innovative user interface.
3) Developing widgets for use in Lightning App Builder.
1) **Which of the following statements is true about creating apps with Visualforce:**

A) Visualforce is designed primarily for page-centric web apps.

B) Visualforce renders the page on the server.

C) Visualforce will be fully supported by Salesforce for years to come.

D) All of the above.

2) **Which of the following statements is NOT true about creating apps with Lightning components:**

A) Lightning components are designed primarily for app-centric web apps.

B) Lightning components can be used everywhere Visualforce can be used.

C) Lightning components render the page on the client.

D) All of the above.

3) **Which of the following is a poor use of Lightning Components:**

A) Developing an app for Salesforce1.

B) Developing a highly interactive app with an innovative user interface.

C) Developing widgets for use in Lightning App Builder.

D) None of the above. All of these are good use cases for Lightning Components.

**UNIT II - Using Visualforce in Lightning Experience.**

**What Doesn’t Work**

And so we come to the, shall we say, less pleasant part of our conversation. Fortunately, the list of what doesn’t work in Visualforce for Lightning Experience is short, and we can get through it quickly. Perhaps the most significant change, in terms of things that might be hard to work around, Using window.location JavaScript code in Lightning components do not work Properly. Visualforce overrides of standard actions are slightly different in Lightning Experience compared to Salesforce Classic. Any override for the object list action won’t be accessible in Lightning Experience, And, coming down to the really minor issues, rendering Visualforce pages as PDFs
works exactly as in Salesforce Classic, without any of the Lightning Experience visual design. This is probably what you want anyway, but if you wanted to render pages into PDFs that include the Lightning Experience design, that’s not possible today.

1) Which of the following features of Visualforce do NOT work in Lightning Experience:
   A) Creating custom apps and tabs.
   B) Overriding standard actions with Visualforce pages.
   C) Using window.location in JavaScript code.
   D) Remote Objects.

2) Which of the following is NOT true about the user interface and visual design in Lightning Experience:
   A) PDFs render with the Lightning Experience visual design.
   B) You can't hide the Lightning Experience main navigation header or sidebar.
   C) The <apex:inputField> tag renders with the Salesforce Classic appearance.
   D) The standard Visualforce header and sidebar are hidden.

UNIT III - Using Lightning Components in Lightning Experience

The word “Lightning” so many times it’s probably lost all meaning. To worsen the storm, we’ve been talking about both “Lightning Experience” and “Lightning components.” Let’s clear up the relationship between the two. Lightning components are still in their infancy and not all the features you’re used to in Visualforce are fully supported.

The following statements are about Lightning Components and Lightning Experience.

1) Lightning Experience is something you use directly, Lightning Components are something you build apps with.
2) Lightning Experience is (mostly) built with Lightning Components.
3) Lightning Experience uses an app-centric development model using Lightning Components.

1) **Which of the following statements about Lightning Components and Lightning Experience is true?**

   A) Lightning Experience is something you use directly, Lightning Components are something you build apps with.
   B) Lightning Experience is (mostly) built with Lightning Components.
   C) Lightning Experience uses an app-centric development model using Lightning Components.
   D) All of the above.

2) **Which of the following is true of Lightning Components:**

   A) Lightning Components can only be used in the Lightning Experience and not the Salesforce mobile app.
   B) Lightning Components can be used in Visualforce pages.
   C) Lightning Components are only optimized for the desktop experience.
   D) All of the above.

**UNIT IV - ISVs, Packaging, and AppExchange**

If you’re a Salesforce ISV partner, you probably have some concerns about how your app development and release processes are affected by Lightning Experience. At the moment, most parts of the ISV experience are the same. There are a few speed bumps to look out for along the way. But overall, we hope that Lightning Experience opens more doors for innovative app development than it does close them.
Similar to the way that your app undergoes a security review when you list it on AppExchange, it’s also reviewed for Lightning Experience readiness. Apps supported in Lightning Experience get a “Lightning Ready” sash on their listing. Apps that aren’t certified as Lightning Ready can still be used in Lightning Experience but there’s no guarantee that they’ll work as expected. They might also be visually inconsistent with Lightning Experience. It’s best to use these apps in Salesforce Classic. The good news here is that, as an ISV, you’re not expected to change all your listed apps so that they’re Lightning Ready. You can make this transition over time or expect that your users continue using your apps in Salesforce Classic.

1) Which of the following ISV features are available in Lightning Experience?
   A) Channel Order App.
   B) Trialforce.
   C) Usage Metrics Visualization app.
   D) Package creation.
   E) None of the above are available.

2) As an ISV, which of the following is true about AppExchange:
   A) Your apps undergo a review for Lighting Experience readiness.
   B) Your apps are available to all customers, whether they have enabled Lightning Experience or not.
   C) The 'Lighting Ready' sash lets AppExchange visitors know your app is verified for Lighting Experience.
   D) All of the above.

UNIT V – The API and Apex in Lightning Experience

As a developer, one of your most important tools on any platform is the API. As a Salesforce developer, Apex is just as important to your success. Your Apex code and queries continue to function as expected, regardless of whether you’re using Lightning Experience or Salesforce
Classic. It really is just that simple. The best answer we can give is maybe. Moving forward, apps listed on AppExchange are marked with a “Lightning Ready” sash if they’re fully compatible with Lightning Experience. Check the listing to see if an app is Lightning Ready.

1) Which of the following is true of Apex in the new Lightning Experience:
   A) You have to update the API version for all your Apex classes in order for them to work in Lightning Experience.
   B) Apex is not supported in the new Lightning Experience.
   C) Your Apex code and queries continue to function as before.
   D) All of the above

2) What should you look for when reviewing your installed packages before using them with Lightning Experience?
   A) A 'Lightning Ready' sash for the package on AppExchange.
   B) An error message saying the package isn't 'Lightning Ready'.
   C) A 'Lightning Ready' check mark on the Installed Packages page.
   D) Errors in the JavaScript console when using the package.