Lightning Components are the preferred way to build UI

<https://trailhead.salesforce.com/en/content/learn/trails/migrate-from-visualforce-to-lightning-web-components>

<apex:page standardController="Contact" >

<apex:form >

<apex:pageBlock title="Edit Contact">

<apex:pageBlockSection columns="1">

<apex:inputField value="{!Contact.FirstName}"/>

<apex:inputField value="{!Contact.LastName}"/>

<apex:inputField value="{!Contact.Email}"/>

<apex:inputField value="{!Contact.Birthdate}"/>

</apex:pageBlockSection>

<apex:pageBlockButtons >

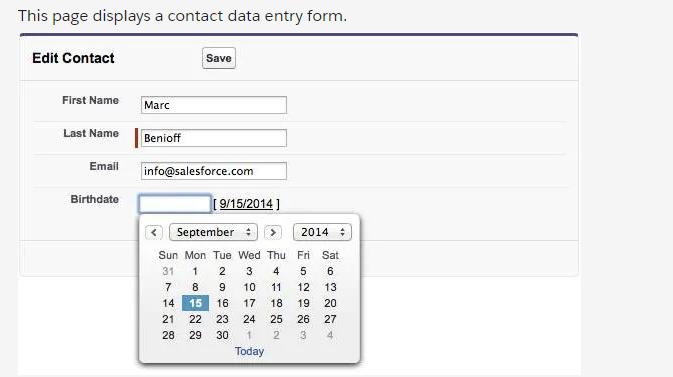
<apex:commandButton action="{!save}" value="Save"/>

</apex:pageBlockButtons>

</apex:pageBlock>

</apex:form>

</apex:page>



To open Visualforce pages:

App Launcher > View All >

Custom Visualforce App appear as different apps in the list of all apps

VF pages can be used as components in the Lightning app builder

<https://developer.salesforce.com/docs/atlas.en-us.pages.meta/pages/pages_intro.htm>

<https://www.youtube.com/watch?v=nzvqDUtwdcY>

<https://developer.salesforce.com/events/webinars/Lightning-App-Builder>

<https://trailhead.salesforce.com/module/lightning_app_builder>

<https://trailhead.salesforce.com/en/modules/lex_dev_visualforce/units/lex_dev_visualforce_intro>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_compref.htm>

<https://help.salesforce.com/apex/HTViewHelpDoc?id=pages_creating.htm&language=en_US>

<https://help.salesforce.com/HTViewHelpDoc?id=code_system_log.htm&language=en_US>

<https://help.salesforce.com/apex/HTViewHelpDoc?id=pages_using_the_editor.htm&language=en_US>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_tools_intro.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_compref.htm>

<https://forcedotcom.github.io/salesforcedx-vscode/>

<https://developer.salesforce.com/page/Tools>

<https://developer.salesforce.com/blogs/developer-relations/2011/09/developing-visualforce-with-your-browser.html>

Global Variables (built in functions)

<apex:page>

<apex:pageBlock title="User Status">

<apex:pageBlockSection columns="1">

{! $User.FirstName } {! $User.LastName }

({! $User.Username })

</apex:pageBlockSection>

</apex:pageBlock>

</apex:page>

Other syntax

{! $User.FirstName & ' ' & $User.LastName }

<p> Today's Date is {! TODAY() } </p>

<p> Next week it will be {! TODAY() + 7 } </p>

<p>The year today is {! YEAR(TODAY()) }</p>

<p>Tomorrow will be day number {! DAY(TODAY() + 1) }</p>

<p>Let's find a maximum: {! MAX(1,2,3,4,5,6,5,4,3,2,1) } </p>

<p>The square root of 49 is {! SQRT(49) }</p>

<p>Is it true? {! CONTAINS('salesforce.com', 'force.com') }</p>

<p>{! IF( CONTAINS('salesforce.com','force.com'),

'Yep', 'Nope') }</p>

<p>{! IF( DAY(TODAY()) < 15,

'Before the 15th', 'The 15th or after') }</p>

({! IF($User.isActive, $User.Username, 'inactive') })

<https://help.salesforce.com/HTViewHelpDoc?id=dev_understanding_global_variables.htm&language=en_US>

<https://help.salesforce.com/HTViewHelpDoc?id=elements_of_a_formula.htm&language=en_US>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_variables_global.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_variables_functions.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_variables_operators.htm>

<https://trailhead.salesforce.com/en/modules/point_click_business_logic>

This code will pull data from the record if you add that same ID parm to the URL for your new page:

<apex:page standardController="Account">

<apex:pageBlock title="Account Summary">

<apex:pageBlockSection>

Name: {! Account.Name } <br/>

Phone: {! Account.Phone } <br/>

Industry: {! Account.Industry } <br/>

Revenue: {! Account.AnnualRevenue } <br/>

</apex:pageBlockSection>

</apex:pageBlock>

</apex:page>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_std.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_sosc_about.htm>

<https://www.developerforce.com/guides/Visualforce_in_Practice.pdf>

<https://www.developerforce.com/guides/Visualforce_in_Practice.pdf>

<https://developer.salesforce.com/blogs/developer-relations/2014/03/twitter-bootstrap-and-visualforce-in-minutes.html>

Sample preview url for a specific contact

<https://resourceful-narwhal-s9j7rs-dev-ed--c.visualforce.com/apex/ContactView?core.apexpages.request.devconsole=1&id=0038c00002gp0ZiAAI>

Automatically display details for an object displayed using a standard controller

<apex:detail />

Other outputs:

<apex:relatedList list="Contacts"/>

<apex:relatedList list="Opportunities" pageSize="5"/>

Output field adds some styling

<apex:pageBlock title="Account Details">

<apex:pageBlockSection>

<apex:outputField value="{! Account.Name }"/>

<apex:outputField value="{! Account.Phone }"/>

<apex:outputField value="{! Account.Industry }"/>

<apex:outputField value="{! Account.AnnualRevenue }"/>

</apex:pageBlockSection>

</apex:pageBlock>

Pageblock table iterates

<apex:pageBlock title="Contacts">

<apex:pageBlockTable value="{!Account.contacts}" var="contact">

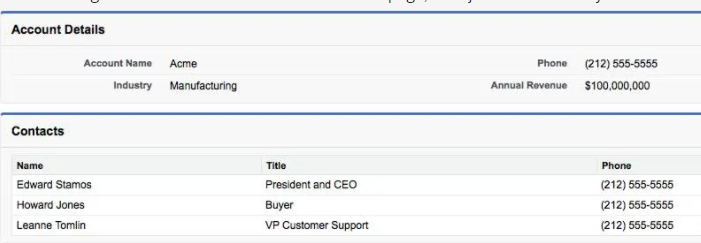
<apex:column value="{!contact.Name}"/>

<apex:column value="{!contact.Title}"/>

<apex:column value="{!contact.Phone}"/>

</apex:pageBlockTable>

</apex:pageBlock>



<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_compref.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_display_field_values.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_component_library.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_custom_mapping.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_iteration_components.htm>

<apex:page standardController="Account">

<apex:form>

<apex:pageBlock title="Edit Account">

<apex:pageBlockSection columns="1">

<apex:inputField value="{! Account.Name }"/>

<apex:inputField value="{! Account.Phone }"/>

<apex:inputField value="{! Account.Industry }"/>

<apex:inputField value="{! Account.AnnualRevenue }"/>

</apex:pageBlockSection>

<apex:pageBlockButtons>

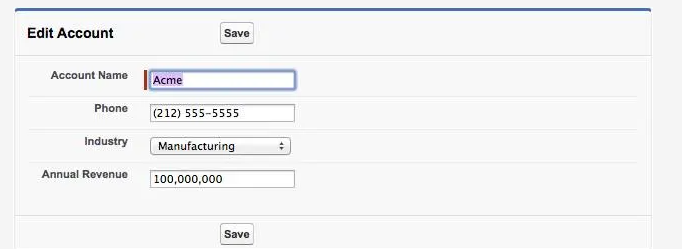
<apex:commandButton action="{! save }" value="Save" />

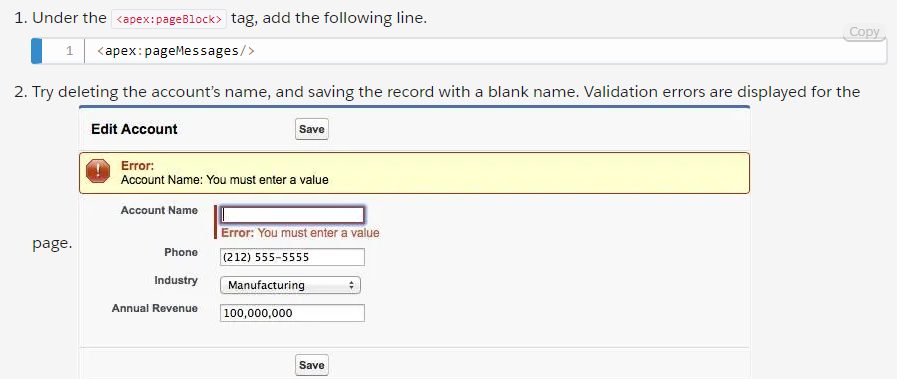
</apex:pageBlockButtons>

</apex:pageBlock>

</apex:form>

</apex:page>





Editing Related Records:

<apex:pageBlock title="Contacts">

<apex:pageBlockTable value="{!Account.contacts}" var="contact">

<apex:column>

<apex:outputLink

value="{! URLFOR($Action.Contact.Edit, contact.Id) }">

Edit

</apex:outputLink>

&nbsp;

<apex:outputLink

value="{! URLFOR($Action.Contact.Delete, contact.Id) }">

Del

</apex:outputLink>

</apex:column>

<apex:column value="{!contact.Name}"/>

<apex:column value="{!contact.Title}"/>

<apex:column value="{!contact.Phone}"/>

</apex:pageBlockTable>

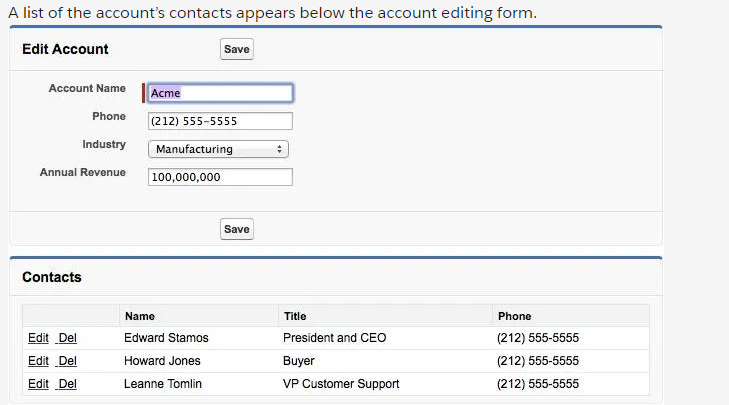
</apex:pageBlock>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_input_components.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_std_actions.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_variables_global_action_valid_values.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_std.htm>



Standard List Controller

<apex:page standardController="Contact" recordSetVar="contacts">

<apex:pageBlock title="Contacts List">

<!-- Contacts List -->

<apex:pageBlockTable value="{! contacts }" var="ct">

<apex:column value="{! ct.FirstName }"/>

<apex:column value="{! ct.LastName }"/>

<apex:column value="{! ct.Email }"/>

<apex:column value="{! ct.Account.Name }"/>

</apex:pageBlockTable>

</apex:pageBlock>

</apex:page>

OR (with filter control)

<apex:page standardController="Contact" recordSetVar="contacts">

<apex:form>

<apex:pageBlock title="Contacts List" id="contacts\_list">

Filter:

<apex:selectList value="{! filterId }" size="1">

<apex:selectOptions value="{! listViewOptions }"/>

<apex:actionSupport event="onchange" reRender="contacts\_list"/>

</apex:selectList>

<!-- Contacts List -->

<apex:pageBlockTable value="{! contacts }" var="ct">

<apex:column value="{! ct.FirstName }"/>

<apex:column value="{! ct.LastName }"/>

<apex:column value="{! ct.Email }"/>

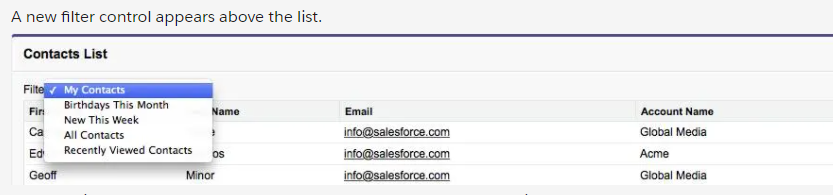
<apex:column value="{! ct.Account.Name }"/>

</apex:pageBlockTable>

</apex:pageBlock>

</apex:form>

</apex:page>



The full lifecycle for the new features on this page goes something like this.

* When the page loads, the <apex:selectList> builds a menu of available filters, by getting the list from the {! listViewOptions } expression. listViewOptions is a property provided by the standard list controller.
* When you choose a new option from the menu, the onchange event is fired by the <apex:actionSupport> component.
* When the onchange fires, the page submits back the new list view selected, by submitting the new item to the filterId property, set in the <apex:selectList>.
* When the property is updated, the page gets a new response from the server, with a new collection of matching records in the contacts variable.
* But because <apex:actionSupport> specifies rerendering only a portion of the page, the page is updated using Ajax—asynchronous JavaScript—instead of by a full page reload.

Pagination:

<!-- Pagination -->

<table style="width: 100%"><tr>

<td>

Page: <apex:outputText

value=" {!PageNumber} of {! CEILING(ResultSize / PageSize) }"/>

</td>

<td align="center">

<!-- Previous page -->

<!-- active -->

<apex:commandLink action="{! Previous }" value="« Previous"

rendered="{! HasPrevious }"/>

<!-- inactive (no earlier pages) -->

<apex:outputText style="color: #ccc;" value="« Previous"

rendered="{! NOT(HasPrevious) }"/>

&nbsp;&nbsp;

<!-- Next page -->

<!-- active -->

<apex:commandLink action="{! Next }" value="Next »"

rendered="{! HasNext }"/>

<!-- inactive (no more pages) -->

<apex:outputText style="color: #ccc;" value="Next »"

rendered="{! NOT(HasNext) }"/> </td>

<td align="right">

Records per page:

<apex:selectList value="{! PageSize }" size="1">

<apex:selectOption itemValue="5" itemLabel="5"/>

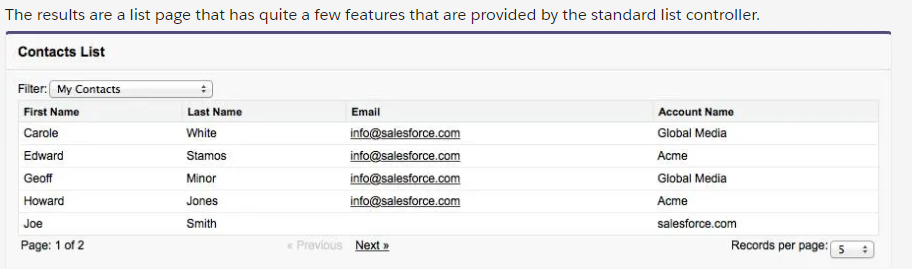
<apex:selectOption itemValue="20" itemLabel="20"/>

<apex:actionSupport event="onchange" reRender="contacts\_list"/>

</apex:selectList>

</td>

</tr></table>



<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_std.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller_sosc_about.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.apexcode.meta/apexcode/apex_pages_standardcontroller.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.apexcode.meta/apexcode/apex_pages_standardsetcontroller.htm>

<https://help.salesforce.com/HTViewHelpDoc?id=customviews.htm&language=en_US>

<https://www.developerforce.com/guides/Visualforce_in_Practice.pdf>

<https://www.developerforce.com/guides/Visualforce_in_Practice.pdf>

<https://developer.salesforce.com/blogs/developer-relations/2014/03/twitter-bootstrap-and-visualforce-in-minutes.html>

Example using repeat with a link to the detail page

<apex:page standardController="Account" recordSetVar="accounts">

<apex:pageBlock title="Account" >

<ul>

<apex:repeat value="{!accounts}" var="a">

<li> <apex:outputLink value="/{!a.Id}"> {!a.name} </apex:outputLink> </li>

</apex:repeat>

</ul>

</apex:pageBlock>

</apex:page>

**Static resources**

Setup > Static Resource

New

Cache Control = Public (if you can see it)

In the below example, the static resource name is jQuery

<apex:page>

<!-- Add the static resource to page's <head> -->

<apex:includeScript value="{! $Resource.jQuery }"/>

<!-- A short bit of jQuery to test it's there -->

<script type="text/javascript">

jQuery.noConflict();

jQuery(document).ready(function() {

jQuery("#message").html("Hello from jQuery!");

});

</script>

<!-- Where the jQuery message will appear -->

<h1 id="message"></h1>

</apex:page>

<https://developer.salesforce.com/mobile/services/mobile-packs>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_resources.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_resources_reference.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_styling.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_javascript_intro.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_variables_global_resource.htm>

<https://developer.salesforce.com/blogs/developer-relations/2013/05/instantly-reloading-visualforce-static-resources.html>

**Custom Controllers**

<apex:page controller="ContactsListWithController">

<apex:form>

<apex:pageBlock title="Contacts List" id="contacts\_list">

<!-- Contacts List -->

<apex:pageBlockTable value="{! contacts }" var="ct">

<apex:column value="{! ct.FirstName }"/>

<apex:column value="{! ct.LastName }"/>

<apex:column value="{! ct.Title }"/>

<apex:column value="{! ct.Email }"/>

</apex:pageBlockTable>

</apex:pageBlock>

</apex:form>

</apex:page>

private String sortOrder = 'LastName';

public List<Contact> getContacts() {

List<Contact> results = Database.query(

'SELECT Id, FirstName, LastName, Title, Email ' +

'FROM Contact ' +

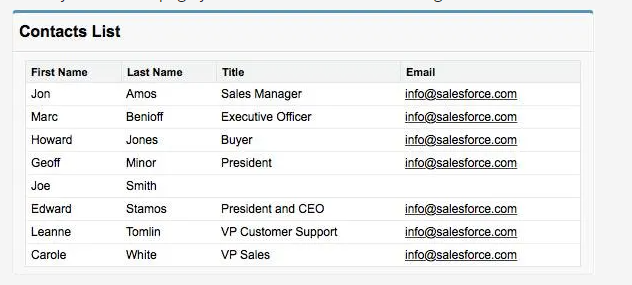
'ORDER BY ' + sortOrder + ' ASC ' +

'LIMIT 10'

);

return results;

}



**Custom Actions:**

Add this to the controller class:

public void sortByLastName() {

this.sortOrder = 'LastName';

}

public void sortByFirstName() {

this.sortOrder = 'FirstName';

}

To invoke them:

<apex:column value="{! ct.FirstName }">

<apex:facet name="header">

<apex:commandLink action="{! sortByFirstName }"

reRender="contacts\_list">First Name

</apex:commandLink>

</apex:facet>

</apex:column>

<apex:column value="{! ct.LastName }">

<apex:facet name="header">

<apex:commandLink action="{! sortByLastName }"

reRender="contacts\_list">Last Name

</apex:commandLink>

</apex:facet>

</apex:column>

To reference a custom object:

public MyObject\_\_c myVariable { get; set; }

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_quick_start_controller.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.pages.meta/pages/pages_controller.htm>

<https://developer.salesforce.com/docs/atlas.en-us.224.0.apexcode.meta/apexcode/>

<https://developer.salesforce.com/page/Force.com_Books>

<https://developer.salesforce.com/blogs/developer-relations/2013/01/apex-template-visualforce-controller.html>

<https://developer.salesforce.com/blogs/tech-pubs/2011/11/controller-for-visualforce-charting.html>

**Display list of cases (Id and number) that link to each case**

public class NewCaseListController {

public List<Case> getNewCases() {

List<Case> cases = [select id, CaseNumber from Case where status='New'];

return cases;

}

}

----------------

<apex:page controller="NewCaseListController">

<apex:pageblock title="New Cases List" id="cases\_list">

<ul>

<apex:repeat value="{!newCases}" var="case" rendered="true" id="case\_list" >

<li> <apex:outputLink value="/{!case.Id}"> {!case.Id} </apex:outputLink>

<apex:outputLink value="/{!case.Id}"> {!case.CaseNumber} </apex:outputLink> </li>

</apex:repeat>

</ul>

</apex:pageBlock>

</apex:page>