

🔔 Attention Salesforce Certified Trailblazers! Maintain your credentials and link your [Trailhead](#) and Webassessor accounts by December 6th. [Learn more](#).

1. [Lightning Design System for Developers](#)-



2. [Use Images, Icons, and Avatars](#)-

Use Images, Icons, and Avatars-

Learning Objectives

After completing this unit, you'll be able to:

- Use the Image component to add avatars to your design.
- Use the Design System SVG sprite maps to include icons in your designs.
- Describe markup gotchas when working within Visualforce.

An Avatar Tells a Thousand Words

In this section, we're going to brighten things up with images and an all new set of beautiful icons (more on those below). This may sound straightforward... an entire unit on images, really? However, these are key components and in particular you'll see that we are using some cutting-edge technology for our icon markup. Plus you need to brighten up your sample pages to earn those Trailhead badges.

Before we get into the code, a note on accessibility. Remember that every `` tag should have an alt attribute for assistive technology users. If the image carries information, set the alt to a clear concise description. If the image is merely decorative, or redundant to adjacent text, set an empty `alt` attribute like so: ``.

Image-wise, the Design System includes special styling for avatars. You know, those hip circular images you see around the internet these days.

Avatar



```
<span class="slds-avatar slds-avatar--medium">
  
</span>
```

An avatar can be circular or a rounded rectangle, depending on usage. The default is a rounded rectangle and requires `.slds-avatar` as the base class. Use a circle for all people-oriented objects that could potentially render as avatars. For a fully round avatar, add the `.slds-avatar--circle` class. Four additional classes are available for sizing.

An [Avatar](#) is created by wrapping your `` element in a `` element with the class `slds-avatar`. Additional sizing helper classes can be applied, for example `slds-avatar_large`. The Design System download actually includes several avatar examples under `/assets/images`. Please feel free to include them in your app. Here's an example of the markup:

```
<span class="slds-avatar slds-avatar x-small">
  
</span>
```

Copy

Copy

Media Objects

A common pattern for including images in web apps is to include an image and text side by side. The Design System has a component to make this super easy, the [Media Object](#).

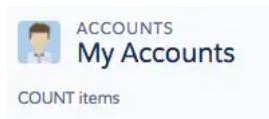
Let's dive straight back into our code from the last unit. We're going to use a **Media Object** to include an avatar in our page header. Replace the `<!-- HEADING AREA -->` part of the page header with the following code:

```
<!-- HEADING AREA -->
<div class="slds-media slds-no-space slds-grow">
  <div class="slds-media figure">
    <span class="slds-avatar slds-avatar-medium">
      
    </span>
  </div>
  <div class="slds-media body">
    <p class="slds-text-title caps slds-line-height reset">Accounts</p>
    <h1 class="slds-page-header__title slds-m-right_small slds-align-middle slds-truncate" title="My Accounts">My Accounts</h1>
  </div>
</div>
<!-- / HEADING AREA -->
```

Copy

Copy

Preview your page and notice how it has *started* to get more visually appealing.



The base class of the **Media Object** component is `slds-media`. It can be applied to any container element, here we use a `<div>`. Inside the container, we provide a figure (the image) and a body (the content).

The figure, i.e. our avatar, is contained inside a `` with the class `slds-media__figure`. The avatar image is specified with a standard `` element. You could also include an icon here (see below).

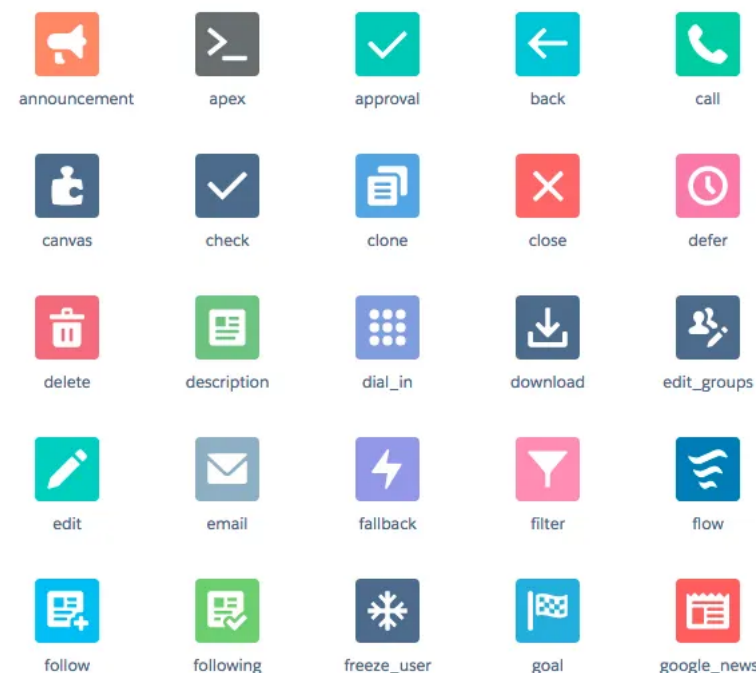
The body is a `<div>` with class `slds-media__body`. This wraps the header text we used earlier.

Icons

Updating the Salesforce icons was a huge priority for us as we embarked upon this work. Let's face it, the existing "clip art" was looking a bit haggard and 90's. Hence we are very excited to offer this set of exciting Technicolor icons covering a wide range of use cases for you to use in your own apps.

The Design System includes a varied supply of [new icons](#) divided into five categories:

- **Custom** - These icons represent Custom Salesforce objects in our UI. They are the icons we make available to Salesforce Administrators when they are creating their Custom objects
- **Doctype** - Common document and file formats
- **Standard** - These icons cover all the Standard Salesforce objects in our UI
- **Utility** - We use these icons to represent interactions that the user can engage with in the UI - things like closing a modal, going back to a previous page, or opening a dropdown menu
- **Action** - And last, but not least, the good ol' action category. We utilize these icons for a fairly specific use case within our mobile UI. You'll notice some duplication here with the Standard and Custom sets - don't panic, you're not going crazy. It's likely you can ignore this category altogether and still have enough icons for any use case you can dream up!



The icons are supplied both as individual PNGs and SVGs, as well packaged up inside [SVG sprite maps](#). Each of the above icon categories has its own sprite map under `/assets/icons`. Sprite maps are our recommended technique for including icons in pages. The advantages of SVG sprite maps over traditional icon fonts

include more fine-grained CSS control and easier positioning in components, as well as better resizability of vector-based SVGs. This final advantage is a boon for responsive design. Vector-based images make clean art at any size.

PLEASE NOTE: Current versions of Edge, Google Chrome, Safari and Firefox already support SVG sprite maps. **To use SVG spritemap image icons with Microsoft Internet Explorer 11 you will need to download a small script called [svg4everybody](#).** After you download [svg4everybody](#), add the `svg4everybody.min.js` script as a static resource, include it in your pages, and call it in a `<script>` tag. Please refer to the full instructions on the [svg4everybody](#) website for more details. The advantages of using this icon technique (as outlined in the previous paragraph) more than make up for this extra step.

```
<apex:includeScript value="{!$Resource.REPLACE_WITH_NAME_OF_SVG4EVERYBODY_STATIC_RESOURCE}" />
<script>
  svg4everybody();
</script>
</head>
```

Copy

Copy

SVG sprite maps are also why we had you add the `xmlns` and `xmlns:xlink` attributes to your `<html>` element. The reason for this is to configure the SVG and `xlink` namespaces within Visualforce. Again, a tiny bit of work for a big payoff.

```
<html xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" lang="en">
```

Copy

Copy

The Design System's icon markup will be new to many readers so let's review an example. Here is the markup for a stand-alone icon. Let's imagine that the placement in the UI carries some meaning to the user (i.e. the icon is not decorative). This means we need to provide some assistive text along with our icon:

```
<span class="slds-icon container slds-icon-standard-account" title="description of icon when needed">
  <svg aria-hidden="true" class="slds-icon">
    <use xlink:href="{!URLFOR($Asset.Slds, 'assets/icons/standard-sprite/svg/symbols.svg#account')}"></use>
  </svg>
  <span class="slds-assistive-text">Account Icon</span>
</span>
```

Copy

Copy

Let's step through this element by element, and class by class. Since the icon is stand-alone and carries meaning we place it inside an outer span with the [slds-icon-container](#) class.

The icons have no background color out of the box. In order to set one, we apply a second class to the span. To use the default color for a particular icon, construct the name of the icon's specific utility class by concatenating `slds-icon-`, the sprite map name, and `-icon`. Apply that class to the `` element. In the example we are using the "standard" sprite map and the "account" icon so the class is `slds-icon-standard-account`.

Inside the ``, we have a `<svg>` element with the `slds-icon` class. The `<svg>` element in turn contains a `<use>` tag that specifies the icon to display based on its `xlink:href` attribute.

In order to set the `xlink:href` path go through these steps:

1. First select the icon you want to use from the [icons page](#). Make a note of which category it is in (action, custom, doctype, standard or utility).
2. Complete the `xlink:href` attribute by concatenating the category sprite (e.g. "standard-sprite"), `/svg/symbols.svg#` and the specific icon within it (e.g. "account"). Hence, the path `assets/icons/standard-sprite/svg/symbols.svg#account` in the above example.

If the icon does not appear, check the following:

- Check you have applied the `xmlns` and `xmlns:xlink` attributes to your `<html>` element
- If you are using MSIE, be sure to use a recent version and include the [svg4everybody](#) script in your page as described above.
- Make sure you have specified the sprite type and icon correctly in the svg path
- Double check that you have the correct URLFOR path to your static resource
- Finally, check the page in your browser Developer Tools for really crazy esoteric issues. Good luck.

After the `<svg>` tag, the assistive text is contained within a span with the `slds-assistive-text` class.

For more information on using icons, such as changing icon colors or size, see the [Icons](#) component documentation on the Design System website.

Adding Icons to the List View Data Table

Now lets brighten up the Data Table with some account icons. Replace the `updateOutputDiv` function with the following to add an additional column with an icon:

```
var updateOutputDiv = function() {
  account.retrieve(
    { orderby: [{ LastModifiedDate: 'DESC' }], limit: 10 },
    function(error, records) {
      if (error) {
        alert(error.message);
      } else {
        // create data table
        var dataTable = document.createElement('table');
        dataTable.className = 'slds-table slds-table-bordered slds-table_cell-buffer slds-no-row-hover';
        // add header row
        var tableHeader = dataTable.createTHead();
        var tableHeaderRow = tableHeader.insertRow();
        var tableHeaderRowCellIcon = tableHeaderRow.insertCell(0);
        tableHeaderRowCellIcon.setAttribute('class', 'slds-cell-shrink');
        var tableHeaderRowCell1 = tableHeaderRow.insertCell(1);
        tableHeaderRowCell1.appendChild(document.createTextNode('Account name'));
        tableHeaderRowCell1.setAttribute('scope', 'col');
        tableHeaderRowCell1.setAttribute('class', 'slds-text-heading_label');
        var tableHeaderRowCell2 = tableHeaderRow.insertCell(2);
```

```

tableHeaderRowCell12.appendChild(document.createTextNode('Account ID'));
tableHeaderRowCell12.setAttribute('scope', 'col');
tableHeaderRowCell12.setAttribute('class', 'slds-text-heading_label');
// build table body
var tableBody = dataTable.appendChild(document.createElement('tbody'))
var dataRow, dataRowCell11, dataRowCell12, recordName, recordId;
records.forEach(function(record) {
  dataRow = tableBody.insertRow();
  var sldsSpriteReference = document.createElementNS('http://www.w3.org/2000/svg', 'use');
  sldsSpriteReference.setAttributeNS('http://www.w3.org/1999/xlink', 'xlink:href', '{!URLFOR($Asset.SLDS, 'assets/icons/standard-sprite/svg/symbols.svg#account')}');
  var accountIcon = document.createElementNS('http://www.w3.org/2000/svg', 'svg');
  accountIcon.setAttributeNS('http://www.w3.org/2000/svg', 'aria-hidden', true);
  accountIcon.classList.add('slds-icon');
  accountIcon.appendChild(sldsSpriteReference);
  accountIconWrapper = document.createElement('span');
  accountIconWrapper.classList.add('slds-icon_container', 'slds-icon-standard-account');
  accountIconWrapper.appendChild(accountIcon);
  dataRowCellIcon = dataRow.insertCell(0);
  dataRowCellIcon.appendChild(accountIconWrapper);
  dataRowCell11 = dataRow.insertCell(1);
  recordName = document.createTextNode(record.get('Name'));
  dataRowCell11.appendChild(recordName);
  dataRowCell12 = dataRow.insertCell(2);
  recordId = document.createTextNode(record.get('Id'));
  dataRowCell12.appendChild(recordId);
});
if (outputDiv.firstChild) {
  // replace table if it already exists
  // see later in tutorial
  outputDiv.replaceChild(dataTable, outputDiv.firstChild);
} else {
  outputDiv.appendChild(dataTable);
}
}
}
);
}


```

Copy

Copy

Preview your page and you should see something approximating a real user interface. Now review the unit and note how many lines of CSS this took.

SALESFORCE LIGHTNING DESIGN SYSTEM TRAILHEAD MODULE



ACCOUNTS
My Accounts

COUNT items





New Account

Add a new account

Name

New account

Create Account

ACCOUNT NAME	ACCOUNT ID
 A great Trailhead module!	001R0000002LT5kiAG
 salesforce.com	001R0000002LT5hiAG
 Acme	001R0000002LT5giAG
 Global Media	001R0000002LT5fiAG

In the next unit we work on one of the most renowned Salesforce pages: the infamous record home page. This will draw together the skills and components we have learned in the units so far, as well as introduce some additional Design System components.

Resources

- [Introduction to SVG sprite maps](#)
- [svg4everybody: using sprite maps with MSIE11](#)
- [Icon component documentation](#)
- [Design Systems icons catalogue](#)

1 Which browser requires an additional JavaScript library before SVG sprite map icons work?

- ☐ A. Mozilla Firefox.
☐ B. MSIE.
☐ C. Google Chrome.
☐ D. Safari.

- ☐ E.None of the above
- ☐ F.All of the above

2The core markup for a Design System SVG sprite map icon consists of: _____

- ☐ A.A <svg> element inside a <use> element.
- ☐ B.A single <svg> element.
- ☐ C.A <use> element inside a <svg> element.
- ☐ D.A single element.
- ☐ E.None of the above.

3Which of the following does NOT appear in the xlink:href icon path? _____

- ☐ A.The icon name
- ☐ B.The icon color
- ☐ C.The sprite map name
- ☐ D.The sub-string assets/icons