# Get Familiar with FlexCards

# OmniStudio FlexCards display contextual information in an at-a-glance format and provide access to relevant tasks for the displayed data.

# The Interaction Console provides a holistic view of the customer's information—and FlexCards are important components of these 360-degree views. The FlexCards you see on the console are Lightning web components (LWC) based on FlexCards; the console itself is a Salesforce Lightning console.

# Key Capabilities

# OK, so let’s say you have 2 minutes to describe what FlexCards are and what they can do. What do you say? Here are the key capabilities of FlexCards.

# FlexCards summarize contextual information at a glance.

# FlexCards are the beginning and ending points for customer transactions.

# FlexCards are viewable on any device or channel.

# FlexCards can display data from multiple data sources.

# FlexCards are built quickly using drag-and-drop elements.

# FlexCards have a what-you-see-is-what-you-get (WYSIWYG) editor for controlling their layout and style.

# FlexCard actions are relevant to the context of the card.

# FlexCards are embeddable in other FlexCards.

# FlexCards are embeddable inside an LWC OmniScript.

# FlexCards display more detail on demand with flyouts.

# FlexCards have multiple states that display based on conditions.

# Where to Find the FlexCard Designer

# FlexCard Designer lets you quickly configure, preview, and debug cards. You can do all sorts of awesome things like drag and drop, position, and resize user interface (UI) elements onto a canvas to format text, buttons, icons, images, links, charts, tables, and even other FlexCards.

# Where do you find this super tool? Use the App Launcher to find the OmniStudio app (formerly called Vlocity Digital Studio). Click the dropdown menu and select **FlexCards**. Now, you're ready to explore the FlexCard Designer header and canvas.

# Header and Canvas

# The header is where you view metadata and perform actions related to your FlexCard, such as the following.

# View basic metadata about your FlexCard, such as Author, Version, Status, the FlexCard it has been Cloned From, whether it is a Child Card, Last Modified date, and Theme.

# Toggle between Design view and Preview, create a new version of your FlexCard, and clone, activate, or deactivate your FlexCard.

# Configure Publish Options for an activated FlexCard, and export your FlexCard.

# Access FlexCard documentation via the Help link.

# The FlexCard Designer header.

# Build your FlexCard by dragging elements onto the canvas.

# Drag elements such as fields, actions, images, states, child FlexCards, and custom Lightning web components (LWC) from the Build panel onto the canvas.

# Rearrange, clone, delete, and adjust the widths of your elements as needed.

# In Design view, test the responsiveness of your FlexCard with the viewport dropdown (1). This feature allows you to see how elements are positioned at different viewport breakpoints.

# Build, Properties, Style, and Setup Panels

# To build your FlexCard, drag fields and elements onto the canvas.

# Fields: Drag fields based on the FlexCard’s configured data source onto the canvas.

# Elements:

# Add simple elements to your FlexCard such as text, images, icons, and blocks.

# Add more complex elements such as actions, charts, menus, and datatables.

# Add states to your FlexCard, embed custom Lightning web components, and embed reusable child FlexCards.

# Fields and Elements are found in the Build panel (1)

# Properties Panel

# When you select an element on the canvas, configure the element’s properties from the Properties panel (1) to determine how the element behaves. For example, when you select a Field element, update the label, choose the data field value to display, and select the field type.

# Style Panel

# Use the Style panel (1) to style your FlexCard elements and update their appearance in real time. This panel has settings for configuring backgrounds, sizes, borders, padding, margins, height, fonts, and responsiveness. When you create a style for an element, save it for use on multiple elements on the FlexCard. For custom designs, create and apply custom CSS.

# Setup Panel

# You configure multiple settings when you create a FlexCard using the data source. For existing FlexCards, go to the Setup panel (1) to update these settings or configure more settings.

# Update your Data Source.

# Apply custom permissions to limit access to your FlexCard.

# Track Custom Data on elements with tracking enabled.

# Enable Multi-Language Support, set Session Variables, and create Event Listeners.

# A Session Variable is a special type of variable that allows us to store values from data sources or external systems and access them globally on a FlexCard.

# An Event Listener is a function that ‘listens’ or waits for an event to occur and performs an action in response.

# Preview and Publish

# Preview your FlexCard in real time to test its design and functionality.

# Select options in the viewport dropdown menu (1) to preview how a FlexCard appears on different devices, such as mobile, desktop, and tablet.

# Click Add Test Parameters (2) to preview your FlexCard with different parameters, such as record Ids and pagination limits.

# Then, click Activate in the header to activate your completed FlexCard. This process compiles and deploys your Lightning web component.

# After you activate your FlexCard, configure your generated Lightning web component’s metadata values, such as where it’s published (Targets).

# Add your own custom component SVG icon to identify your generated Lightning web component from the Experience Builder for Communities and from the Lightning App Builder for Lightning pages.

# Create FlexCards with the Data Source Wizard

# When you create a FlexCard, the Data Source Wizard walks through a series of steps.

# Define the FlexCard.

# Select the data source type.

# Select the data source.

# Configure the data source inputs.

# Steps for creating a new FlexCard.

# FlexCard Elements

# You’re probably excited to dig even deeper into FlexCards, so let’s jump right into it.

# The following FlexCard has two distinct areas. The left area displays basic account information for a customer account: an icon, a title, four fields, and a menu for actions. The right area, although blank in the following example, will eventually display weather information. In this unit, however, you focus on the account information side.

# View of a FlexCard, focusing on the account information.

# First, let’s strip away the data and the style of the FlexCard to show how the elements are working underneath.

# The elements that make up the FlexCard with account details.

# Block Elements

# There are two Block elements that make up this FlexCard. A Block element combines logical groups of other elements inside a FlexCard. You can make a Block collapsible to hide and expand its content. You can also place a Block inside another Block.

# The Account Block contains a few simple user interface (UI) elements: Text, Icon, and Menu UI elements. It also includes another Block, which groups an Icon and Text element to make a header in the FlexCard.

# Text and Field Elements

# In the previous example, the Text element is used to add the data fields returned from the data source. The Text element combines text and parsed merge fields using a rich text editor. The rich text editor adds an HTML div for each section of text to build the layout of the text element. This element also includes a Style menu for formatting the text.

# You also use Field elements to display data returned from a data source on a FlexCard.

# There are two ways to add data fields using this element. The simplest way is to drag a field from the Fields list onto the canvas. The Fields list shows the list of available data fields.

# The second way is to drag the Field element from the Elements list onto the canvas. The Field element displays the output from a data field.

# The Field properties include a Field Type setting, which is useful for showing data in its correct format. For example, a Date field type displays a date in its proper format.

# Icon Elements

# The Icon element displays a Salesforce SVG icon that you access from the Salesforce Lightning Design System library of icons or a custom icon that you upload.

# You can configure icons to perform actions when clicked. Image, Block, and Toggle elements also have this capability.

# Menu and Action Elements

# Menu and Action elements.

# The Menu element (1) creates a menu from a list of actions on a FlexCard. Style the menu button and each action in the menu’s dropdown. This element is a useful option when you want to add a lot of actions to your FlexCard but you don’t want to clutter the card. The Action element (2) renders text or a button that executes an action when clicked.

# There are several types of actions available.

| Action Type Name | Description |
| --- | --- |
| Card | Perform card-level actions, such as reload, update data source, and remove. |
| Event | Custom: Fire a Custom Event to notify the parent FlexCard of an event occurring.PubSub: Fire a PubSub Event to notify another component on a page or application of an event occurring. |
| Flyout | Display additional information from a child card, OmniScript, or custom Lightning web component (LWC) in a modal or popover. |
| Navigate | Select a target URL or a PageReference type that enables navigation within Lightning Experience, within Communities, or to an external web address. |
| OmniScript | Launch an OmniScript from the FlexCard. |
| Update OmniScript | Update an OmniScript from a FlexCard embedded as a custom Lightning web component in an OmniScript. |
| OmniStudio Action | Launch OmniScripts, components, web pages, or external applications from a reusable OmniStudio Action. An OmniStudio Action can launch the same OmniScript from multiple FlexCards. |

# Customizable FlexCards help you see all sorts of awesome information at a glance, right? Next up, kick your cards up a notch and learn how to style the displayed data and Action elements you add to FlexCards.

# Features of the Style Panel

# You configure the look and feel of a FlexCard by styling individual elements and the fields within elements. Using the Style panel, we design backgrounds, text, and borders, and adjust heights, widths, and the spacing inside and between elements. There are also specific style options available for the following elements.

# Action

# Datatable

# Field

# Icon

# Menu

# Toggle

# The Style panel is made up of these sections.

| Section | Description |
| --- | --- |
| Alignment | Update the padding, margin, and text alignment of an element. |
| Appearance | Configure the color, background, and border of an element. |
| Custom | Create and apply custom classes, and add inline styles to an element. |
| Custom CSS | Save style settings on a FlexCard element as a custom style to use on multiple elements of a FlexCard. |
| Dimensions | Set the height and width, and configure the responsiveness of your element. |

# So let’s get to stylin’, shall we?

# Enable Responsiveness

# Enable **Responsiveness** on elements to set the widths to change as the width of the visible area of a page (the viewport) changes. The responsive sizing of elements is a mobile-first approach. Sizing starts at the smallest visible viewport breakpoint and is continuously applied upwards until the next wider breakpoint overrides it. The default width becomes the width of the smallest viewport breakpoint.

# Styling dimensions and responsiveness for a FlexCard.

# Width

# The FlexCard canvas is a 12-column grid. The width of an element is 1 to 12 parts of this grid. When you drag a field element onto the canvas for the first time, its default width is 12 columns. If you want to position other fields on the same line, you need to adjust the widths. For example, a field element that is 6 columns wide takes up 50% of the available horizontal space, so you can make 2 field elements that are 6 columns wide appear on the same line. If you then change one of the field elements to a width of 7, the fields display on 2 separate lines again.

# In the example above, the width of the Primary Contact field element is changed to 7 and the width of the Menu element is changed to 4.

# Padding and Margin

# Add padding to create space between the boundary of your element and the contents inside your element. The Around padding type allows you to add padding on all sides of the content at once.

# Add margins between the adjacent elements on your FlexCard to create space around the container of a FlexCard element. The Around margin type allows you to add space around the entire element.

# In the example above, the Margin Type is changed to Left and the Margin Size to Small. The Padding Type is changed to Around and the Padding Size to X-Small. After these changes were made, the style was saved and then applied to the other elements.

# Save and Apply Styles

# Save styles in a FlexCard to reuse on other elements in the FlexCard. Save the style as a custom style in the Style panel, then apply the saved style to other elements of the same type.

# Saved styles are only available within the FlexCard. You can only save the styling that you’ve done within the Style panel, not any styling done in the Properties panel of the element.

# Saving and applying styles in the Style panel.

# You’ve brushed up on FlexCard basics, learned the ins and outs of the Data Source Wizard, and discovered how to add data and style your FlexCards. Remember, FlexCards are chock full of contextual info that provides an at-a-glance, 360-degree view of customers. And they’re just as their name suggests—flexible!