Tables

> *This page is still evolving and is subject to change in the future. We do not yet have coded examples for everything shown on this page. Let us know what you think.*

[Design System Image]

Similar to lists, tables are used to display structured data. Tables are especially powerful in desktop applications, where users can easily compare and analyze a large set of data and find what they are looking for with sorting and filtering options.

[MaterialDesignDescription Component - Interactive React component]

[TOC Component - Interactive React component]

Appearance

[Design System Image]

Table cells may be dynamic and interactive.

[Design System Image]

You should seek to make your tables both easy to navigate and effective to use.

[Design System Image]

Behaviors

Row Actions

Similar to lists, tables can also have row-level actions. Common actions include edit, delete, and view details. They are usually placed at the end of a table row.

[Design System Image]

[Design System Image]

Batch Actions

If your application allows users to perform batch actions on their data set, such as deleting 15 items at once, you can add a checkbox column on the left side of the table for handling selection.

[Design System Image]

Sticky Header Row

When a table gets long, your users should not need to scroll back to the top to view the table header.

[Design System Image]

Expanded Row

Occasionally, table rows may expand to allow users to take a quick peek into additional row details.

[Design System Image]

[Design System Image]

Horizontal Scrolling

We strongly encourage you to design your application to only show as much data as is necessary.

When your table overflows horizontally, you should allow your users to scroll horizontally to view more about the table. However, this interaction is less ergonomically convenient when users are not using a touch screen or a trackpad. See the Responsive section below.

The application may display a scroll bar to suggest that there are more contents to the right.

[Design System Image]

[Design System Image]

Searching and Filtering

There are many different ways to perform searching and filtering with table data. Below are a few examples:

[Design System Image]

[Design System Image]

Transition Animation

Table rows often look very similar and can be difficult for users to differentiate. You should make use of animations and transitions to help users understand where certain actions are taking place.

[Design System Image]

Tables vs. Lists

[Design System Image]

In most cases, we encourage you to organize your data into lists instead of tables. However, the distinction between the two is not black and white. Below are a few factors you should consider when deciding how to present your data.

Arrangement

Tables place more emphasis on columns, allowing users to easily find a row by any of its associated data points. Lists, on the other hand, can be less structured and tend to treat each item as a cohesive whole rather than a collection of individual data values.

[Design System Image]

Because of this, tables tend to take up much more horizontal space. Therefore, they are almost always the primary content on the page where they reside. You should not use tables in dashboards, where many widgets are competing for the space; you should also avoid using tables on mobile applications, where the viewport size is limited.

[Design System Image]

A "tabular list" or a structured list that attempts to combine tables and lists together might result in a design that does not pick up the advantages of either pattern.

[Design System Image]

Eye Movement

Tables facilitate users who want to work across multiple different properties of an item. Users often navigate down a column, find the desired value, jump to another column, navigating further down, before they exit their path. This is called a lawn mower pattern. On the other hand, lists often have a primary property — these can be icons, avatars, or primary list text — that the user would rely on for navigation, a scanning behavior known as the layer cake pattern.

[Design System Image]

If you expect your users to compare across multiple parameters before deciding which row to interact with, you should use a table. If they primarily read your data row-by-row, you should try structuring your data as a list.

[Design System Image]

Data Labels

In tables, each data point is labeled only once in the table header. In lists, however, the values are either clear enough on their own without explanations, or they have to be included in each list item.

[Design System Image]

[Design System Image]

Consumer-Facing Applications

Avoid using tables in consumer-facing applications. Lists mentally match real-world individual objects, while tables can be more abstract. Use tables only if your users really need to examine technical details.

Responsive

In general, **we do not encourage tables on mobile platforms**, as table-related interactions tend to be complicated for small viewports, and mobile phones also often lack the computational power to process massive data sets.

[Design System Image]

Understanding Users' Goals

[Design System Image]

Before you design a table structure for a mobile application, you should conduct user research and have a clear understanding of users' intentions. When they land on this page, do they need to do complicated cross-comparisons between multiple rows and columns? Is a full table necessary?

For applications where users will be on foot and out in the field while they interact with the app, keep the interactions minimal so users can pay sufficient attention to their surroundings.

Breaking Down Tables into Lists

If a user only needs a few parameters to identify the correct data entry and make decisions, a list with a few selected data points would suffice.

[Design System Image]

Infinite Scroll and Pagination

If tables become necessary in your mobile application, infinite scroll and pagination are especially helpful to reduce the load of both the server and the client-side render.

[Design System Image]

> *Be careful using pagination with multi-selection, as it is unclear whether the selection gets preserved when the page number changes.*

Native Mobile Gestures

In native mobile applications, you may also consider embedding action gestures common for touch screens, such as long-press and swipe.

[MaterialDesignDescription Component - Interactive React component]

However, these gestures should never be the only way to perform an action. For example, if your application is able to delete a table row, users may be able to either (1) left-swipe to bring up a contextual menu with a delete option, or (2) click into the target row, which opens a new page that contains a delete button.

Landscape Mode

If a full table view is desired on mobile devices, consider offering a landscape view so that more columns are visible at once.

[Design System Image]

However, many applications are meant to be used primarily in portrait mode. In this case, you need to call out the option.

[Design System Image]

Design Specifications

[Design System Image]

[MaterialDesignDescription Component - Interactive React component]

Developers

Use the following components to implement this pattern:

Angular:
- Angular Material - Table

React:
- MUI - Table - Data Grid

React Native:
- React Native Paper - Data Table