Documentation

search

Search

- *rocket_launch*

  Get started

  - Installation
    add
  - Fundamentals
    add
  - First steps
    add
- *code*

  Develop

  - Concepts
    add
  - API reference
    remove
    - PAGE ELEMENTS

    - Write and magic
      add
    - Text elements
      add
    - Data elements
      add
    - Chart elements
      add
    - Input widgets
      remove
      - BUTTONS

      - st.button
      - st.download_button
      - st.form_submit_button*link*
      - st.link_button
      - st.page_link
      - SELECTIONS

      - st.checkbox
      - st.color_picker
      - st.feedback
      - st.multiselect
      - st.pills
      - st.radio
      - st.segmented_control
      - st.selectbox
      - st.select_slider
      - st.toggle
      - NUMERIC

- st.number_input
- st.slider
- DATE & TIME

- st.date_input
- st.time_input
- TEXT

- st.chat_input*link*
- st.text_area
- st.text_input
- MEDIA & FILES

- st.audio_input
- st.camera_input
- st.data_editor*link*
- st.file_uploader

- Media elements
  add
- Layouts and containers
  add
- Chat elements
  add
- Status elements
  add
- Third-party components*open_in_new*
- APPLICATION LOGIC

- Navigation and pages
  add
- Execution flow
  add
- Caching and state
  add
- Connections and secrets
  add
- Custom components
  add
- Utilities
  add
- Configuration
  add
- TOOLS

- App testing
  add
- Command line
  add

- Tutorials
  add
- Quick reference
  add

- *web_asset*

  Deploy

  - Concepts
    add

st.segmented_control

Streamlit Version Version 1.41.0

Display a segmented control widget.

A segmented control widget is a linear set of segments where each of the passed `options` functions like a toggle button.

Function signature[source]

st.segmented_control(label, options, *, selection_mode="single", default=None, format_func=None, key=None, help=None, on_change=None, args=None, kwargs=None, disabled=False, label_visibility="visible")

Parameters

label (str)

A short label explaining to the user what this widget is for. The label can optionally contain GitHub-flavored Markdown of the following types: Bold, Italics, Strikethroughs, Inline Code, Links, and Images. Images display like icons, with a max height equal to the font height.

Unsupported Markdown elements are unwrapped so only their children (text contents) render. Display unsupported elements as literal characters by backslash-escaping them. E.g., `"1\. Not an ordered list"`.

See the `body` parameter of `st.markdown` for additional, supported Markdown directives.

For accessibility reasons, you should never set an empty label, but you can hide it with `label_visibility` if needed. In the future, we may disallow empty labels by raising an exception.

options (Iterable of V)

Labels for the select options in an `Iterable`. This can be a `list`, `set`, or anything supported by `st.dataframe`. If `options` is dataframe-like, the first column will be used. Each label will be cast

Returns

(list of V, V, or None)

If the `selection_mode` is `multi`, this is a list of selected options or an empty list. If the `selection_mode` is `"single"`, this is a selected option or `None`.

Function signature[source]

st.segmented_control(label, options, *, selection_mode="single", default=None, format_func=None, key=None, help=None, on_change=None, args=None, kwargs=None, disabled=False, label_visibility="visible")

	to `str` internally by default.
selection_mode ("single" or "multi")	The selection mode for the widget. If this is `"single"` (default), only one option can be selected. If this is `"multi"`, multiple options can be selected.
default (Iterable of V, V, or None)	The value of the widget when it first renders. If the `selection_mode` is `multi`, this can be a list of values, a single value, or `None`. If the `selection_mode` is `"single"`, this can be a single value or `None`.
format_func (function)	Function to modify the display of the options. It receives the raw option as an argument and should output the label to be shown for that option. This has no impact on the return value of the command.
key (str or int)	An optional string or integer to use as the unique key for the widget. If this is omitted, a key will be generated for the widget based on its content. Multiple widgets of the same type may not share the same key.
help (str)	An optional tooltip that gets displayed next to the widget label. Streamlit only displays the tooltip when `label_visibility="visible"`.
on_change (callable)	An optional callback invoked when this widget's value changes.
args (tuple)	An optional tuple of args to pass to the callback.
kwargs (dict)	An optional dict of kwargs to pass to the callback.
disabled (bool)	An optional boolean that disables the widget if set to `True`. The default is `False`.
label_visibility ("visible", "hidden", or "collapsed")	The visibility of the label. The default is `"visible"`. If this is `"hidden"`, Streamlit displays an empty spacer instead of the label, which can help keep the widget alligned with other widgets. If this is `"collapsed"`, Streamlit displays no label or spacer.
Returns	
(list of V, V, or None)	If the `selection_mode` is `multi`, this is a list of selected options or an empty list. If the `selection_mode` is `"single"`, this is a selected option or `None`.

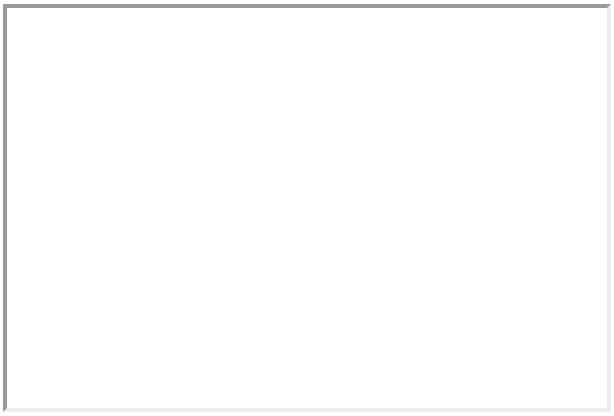
Examples

Example 1: Multi-select segmented control

Display a multi-select segmented control widget, and show the selection:

```
import streamlit as st

options = ["North", "East", "South", "West"]
selection = st.segmented_control(
    "Directions", options, selection_mode="multi"
)
st.markdown(f"Your selected options: {selection}.")
```



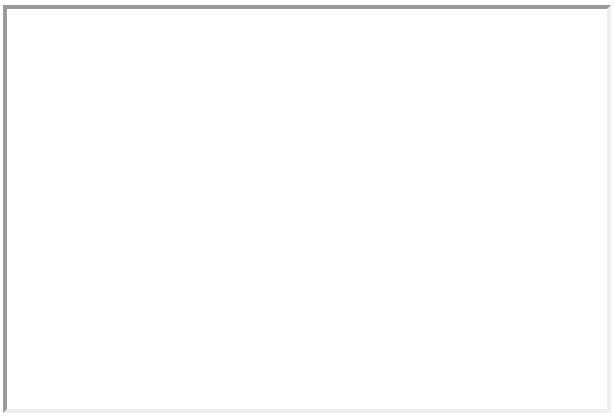
Built with Streamlit 🎈
Fullscreen *open_in_new*

Example 2: Single-select segmented control with icons

Display a single-select segmented control widget with icons:

```
import streamlit as st

option_map = {
    0: ":material/add:",
    1: ":material/zoom_in:",
    2: ":material/zoom_out:",
    3: ":material/zoom_out_map:",
}
selection = st.segmented_control(
    "Tool",
    options=option_map.keys(),
    format_func=lambda option: option_map[option],
    selection_mode="single",
)
st.write(
    "Your selected option: "
    f"{None if selection is None else option_map[selection]}"
)
```



Built with Streamlit 🎈
Fullscreen *open_in_new*

←Previous: st.radioNext: st.selectbox→
forum

Still have questions?

Our forums are full of helpful information and Streamlit experts.

forum Ask AI