

[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](#)  
*remove*
    - HEADINGS & BODY

---
    - [st.title](#)
    - [st.header](#)
    - [st.subheader](#)
    - [st.markdown](#)
    - FORMATTED TEXT

---
    - [st.caption](#)
    - [st.code](#)
    - [st.divider](#)
    - [st.echo](#)
    - [st.latex](#)
    - [st.text](#)
    - UTILITIES

---
  - [st.htmllink](#)
- [Data elements](#)  
*add*
- [Chart elements](#)  
*add*
- [Input widgets](#)  
*add*
- [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*
- [Streamlit Community Cloud](#)  
*add*
- [Snowflake](#)
- [Other platforms](#)  
*add*

- [school](#)

## [Knowledge base](#)

- [FAQ](#)
- [Installing dependencies](#)
- [Deployment issues](#)

- [Home/](#)
- [Develop/](#)
- [API reference/](#)
- [Text elements/](#)
- [st.markdown](#)



Display string formatted as Markdown.

Function signature[\[source\]](#)

`st.markdown(body, unsafe_allow_html=False, *, help=None)`

Parameters

The text to display as GitHub-flavored Markdown. Syntax information can be found at: <https://github.github.com/gfm>. If anything other than a string is passed, it will be converted into a string behind the scenes using `str(body)`.

This also supports:

- body (any)
- Emoji shortcodes, such as `:+1:` and `:sunglasses:`. For a list of all supported codes, see <https://share.streamlit.io/streamlit/emoji-shortcodes>.
  - Streamlit logo shortcode. Use `:streamlit:` to add a little Streamlit flair to your text.
  - A limited set of typographical symbols. "`<- -> <-> -- >= <= ~=`" becomes "`← → ↔ — ≥ ≤ ≈`" when parsed as Markdown.
  - Google Material Symbols (rounded style), using the syntax `:material/icon_name:`, where "icon\_name" is the name of the icon in snake case. For a complete list of icons, see Google's [Material Symbols](#) font library.
  - LaTeX expressions, by wrapping them in `"$"` or `"$$"` (the `"$$"` must be on their own lines). Supported LaTeX functions are listed at <https://katex.org/docs/supported.html>.
  - Colored text and background colors for text, using the syntax `:color[text to be colored]` and `:color-background[text to be colored]`, respectively. color must be replaced with any of the following supported colors: blue, green, orange, red, violet, gray/grey, rainbow, or primary. For example, you can use `:orange[your text here]` or `:blue-background[your text here]`. If you use "primary" for color, Streamlit will use the default primary accent color unless you set the `theme.primaryColor` configuration option.

Whether to render HTML within body. If this is `False` (default), any HTML tags found in body will be escaped and therefore treated as raw text. If this is `True`, any HTML expressions within body will be rendered.

unsafe\_allow\_html (bool) Adding custom HTML to your app impacts safety, styling, and maintainability.

Note

If you only want to insert HTML or CSS without Markdown text, we recommend using `st.html` instead.

help (str) An optional tooltip that gets displayed next to the Markdown.

Examples

```
import streamlit as st

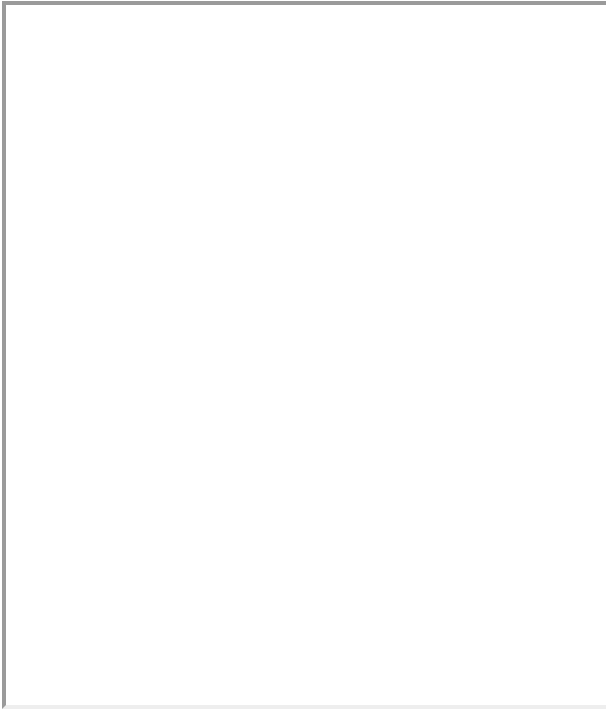
st.markdown("*Streamlit* is **really** ***cool***.")
st.markdown('''
    :red[Streamlit] :orange[can] :green[write] :blue[text] :violet[in]
    :gray[pretty] :rainbow[colors] and :blue-background[highlight] text.''' )
```

```
st.markdown("Here's a bouquet &mdash;\n            :tulip::cherry_blossom::rose::hibiscus::sunflower::blossom:")
```

```
multi = '''If you end a line with two spaces,\na soft return is used for the next line.
```

```
Two (or more) newline characters in a row will result in a hard return.\n'''
```

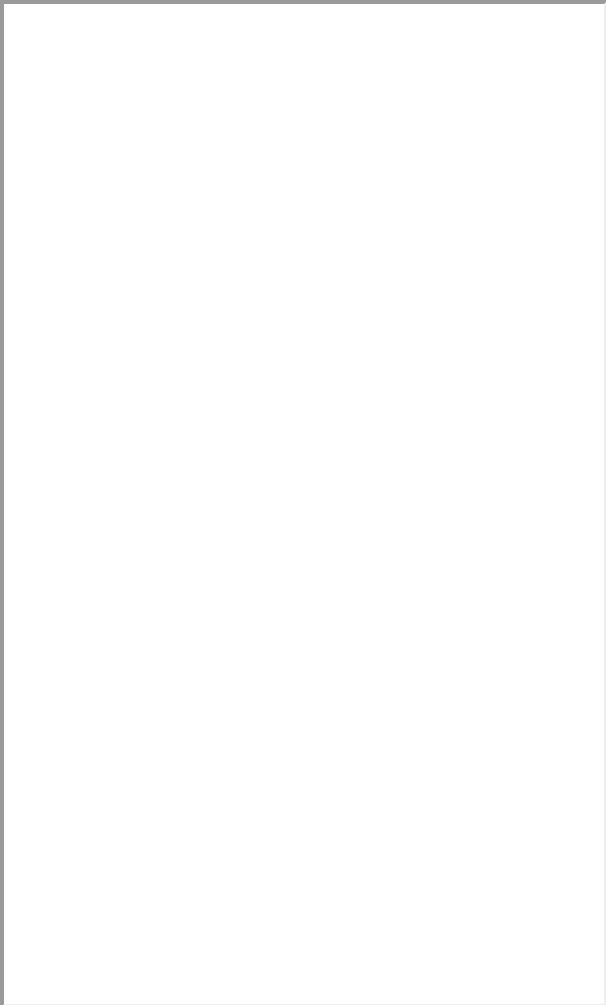
```
st.markdown(multi)
```



[Built with Streamlit](#) 🍷

[Fullscreen open in new](#)

```
import streamlit as st md = st.text_area('Type in your markdown string (without outer quotes)', "Happy\nStreamlit-ing! :balloon:") st.code(f""" import streamlit as st st.markdown('{md}') """)\nst.markdown(md)
```



[Built with Streamlit](#)   
[Fullscreen open in new](#)  
[←Previous: st.subheader](#)[Next: st.caption→](#)  
*forum*

### Still have questions?

Our [forums](#) are full of helpful information and Streamlit experts.

---

[Home](#)[Contact Us](#)[Community](#)



© 2025 Snowflake Inc. [Cookie policy](#)

[forum](#) [Ask AI](#)