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.file_uploader

Streamlit Version Version 1.41.0

Display a file uploader widget.

By default, uploaded files are limited to 200MB. You can configure this using the `server.maxUploadSize` config option. For more info on how to set config options, see https://docs.streamlit.io/develop/api-reference/configuration/config.toml

Function signature[source]

st.file_uploader(label, type=None, accept_multiple_files=False, 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 file uploader 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.

Returns

(None or UploadedFile or list of UploadedFile)

- If accept_multiple_files is False, returns either None or an UploadedFile object.
- If accept_multiple_files is True, returns a list with the uploaded files as UploadedFile objects. If no files were uploaded, returns an empty list.

The UploadedFile class is a subclass of BytesIO, and therefore is "file-like". This means you can pass an instance of it anywhere a file is expected.

Function signature[source]

st.file_uploader(label, type=None, accept_multiple_files=False, key=None, help=None, on_change=None, args=None, kwargs=None, *, disabled=False, label_visibility="visible")

	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.
type (str or list of str or None)	Array of allowed extensions. ['png', 'jpg'] The default is None, which means all extensions are allowed.
accept_multiple_files (bool)	If True, allows the user to upload multiple files at the same time, in which case the return value will be a list of files. Default: False
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. No two widgets may have 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 file_uploader'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 file uploader 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	
(None or UploadedFile or list of UploadedFile)	• If accept_multiple_files is False, returns either None or an UploadedFile object. • If accept_multiple_files is True, returns a list with the uploaded files as UploadedFile objects. If no files were uploaded, returns an empty list. The UploadedFile class is a subclass of BytesIO, and therefore is "file-like". This means you can pass an instance of it anywhere a file is expected.

Examples

Insert a file uploader that accepts a single file at a time:

```
import streamlit as st
import pandas as pd
from io import StringIO
```

```
uploaded_file = st.file_uploader("Choose a file")
if uploaded_file is not None:
    # To read file as bytes:
    bytes_data = uploaded_file.getvalue()
    st.write(bytes_data)

    # To convert to a string based IO:
    stringio = StringIO(uploaded_file.getvalue().decode("utf-8"))
    st.write(stringio)

    # To read file as string:
    string_data = stringio.read()
    st.write(string_data)

    # Can be used wherever a "file-like" object is accepted:
    dataframe = pd.read_csv(uploaded_file)
    st.write(dataframe)
```

Insert a file uploader that accepts multiple files at a time:

```
import streamlit as st

uploaded_files = st.file_uploader(
    "Choose a CSV file", accept_multiple_files=True
)
for uploaded_file in uploaded_files:
    bytes_data = uploaded_file.read()
    st.write("filename:", uploaded_file.name)
    st.write(bytes_data)
```

[Built with Streamlit](#)

[Fullscreen *open_in_new*](#)

[Previous: st.data_editor](#)[Next: Media elements](#)

forum

Still have questions?

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