

[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](#)
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](#)

API reference

Streamlit makes it easy for you to visualize, mutate, and share data. The API reference is organized by activity type, like displaying data or optimizing performance. Each section includes methods associated with the activity type, including examples.

Browse our API below and click to learn more about any of our available commands! 

Display almost anything

Write and magic

[st.write](#)

[Write arguments to the app.](#)

```
st.write("Hello **world**!") st.write(my_data_frame) st.write(my_mpl_figure).
```

[st.write_stream](#)

[Write generators or streams to the app with a typewriter effect.](#)

```
st.write_stream(my_generator) st.write_stream(my_llm_stream).
```

[Magic](#)

[Any time Streamlit sees either a variable or literal value on its own line, it automatically writes that to your app using st.write](#)

```
"Hello **world**!" my_data_frame my_mpl_figure
```

Text elements



[Markdown](#)

[Display string formatted as Markdown.](#)

```
st.markdown("Hello **world**!")
```



[Title](#)

[Display text in title formatting.](#)

```
st.title("The app title").
```



[Header](#)

[Display text in header formatting.](#)

```
st.header("This is a header").
```



[Subheader](#)

[Display text in subheader formatting.](#)

```
st.subheader("This is a subheader").
```



[Caption](#)

[Display text in small font.](#)

```
st.caption("This is written small caption text").
```

 [screenshot](#)

[Code block](#)

[Display a code block with optional syntax highlighting.](#)

```
st.code("a = 1234").
```

 [screenshot](#)

[Echo](#)

[Display some code in the app, then execute it. Useful for tutorials.](#)


```
with st.echo(): st.write('This code will be printed').
```

 [screenshot](#)

[LaTeX](#)

[Display mathematical expressions formatted as LaTeX.](#)

```
st.latex("\int a x^2 \, dx").
```

 [screenshot](#)

[Preformatted text](#)

[Write fixed-width and preformatted text.](#)

```
st.text("Hello world").
```

 [screenshot](#)

[Divider](#)

[Display a horizontal rule.](#)

```
st.divider().
```

Third-party components



These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!


[Previous](#)

 [screenshot](#)

[Tags](#)

[Add tags to your Streamlit apps. Created by @gagan3012.](#)

```
st_tags(label='# Enter Keywords:', text='Press  
enter to add more', value=['zero', 'One', 'Two'],  
suggestions=['five', 'six', 'seven', 'eight',  
'nine', 'three', 'eleven', 'ten', 'four'], maxtags  
= 4, key='1')
```

 [screenshot](#)

[NLU](#)

[Apply text mining on a dataframe. Created by @JohnSnowLabs.](#)


```
nlu.load('sentiment').predict('I love NLU! <3')
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)


```
mention(label="An awesome Streamlit App",  
icon="streamlit",  
url="https://extras.streamlit.app",)
```

 [screenshot](#)

[Annotated text](#)

[Display annotated text in Streamlit apps. Created by @tvst.](#)

```
annotated_text("This ", ("is", "verb"), " some ",  
("annotated", "adj"), ("text", "noun"), " for those  
of ", ("you", "pronoun"), " who ", ("like",  
"verb"), " this sort of ", ("thing", "noun"), ".")
```

 [screenshot](#)

[Drawable Canvas](#)

[Provides a sketching canvas using Fabric.js.](#) Created by [@andfanilo](#).

```
st_canvas(fill_color="rgba(255, 165, 0, 0.3)",
stroke_width=stroke_width,
stroke_color=stroke_color,
background_color=bg_color,
background_image=Image.open(bg_image) if bg_image
else None, update_streamlit=realtime_update,
height=150, drawing_mode=drawing_mode,
point_display_radius=point_display_radius if
drawing_mode == 'point' else 0, key="canvas",)
```



Tags

[Add tags to your Streamlit apps.](#) Created by [@gagan3012](#).

```
st_tags(label='# Enter Keywords:', text='Press
enter to add more', value=['Zero', 'One', 'Two'],
suggestions=['five', 'six', 'seven', 'eight',
'nine', 'three', 'eleven', 'ten', 'four'], maxtags
= 4, key='1')
```



NLU

[Apply text mining on a dataframe.](#) Created by [@JohnSnowLabs](#).

```
nl.load('sentiment').predict('I love NLU! <3')
```



Streamlit Extras

[A library with useful Streamlit extras.](#) Created by [@arnaudmiribel](#).

```
mention(label="An awesome Streamlit App",
icon="streamlit",
url="https://extras.streamlit.app",)
```



Annotated text

[Display annotated text in Streamlit apps.](#) Created by [@tvst](#).

```
annotated_text("This ", ("is", "verb"), " some ",
("annotated", "adj"), ("text", "noun"), " for those
of ", ("you", "pronoun"), " who ", ("like",
"verb"), " this sort of ", ("thing", "noun"), ".")
```



Drawable Canvas

[Provides a sketching canvas using Fabric.js.](#) Created by [@andfanilo](#).

```
st_canvas(fill_color="rgba(255, 165, 0, 0.3)",
stroke_width=stroke_width,
stroke_color=stroke_color,
background_color=bg_color,
background_image=Image.open(bg_image) if bg_image
else None, update_streamlit=realtime_update,
height=150, drawing_mode=drawing_mode,
point_display_radius=point_display_radius if
drawing_mode == 'point' else 0, key="canvas",)
```



Tags

[Add tags to your Streamlit apps.](#) Created by [@gagan3012](#).

```
st_tags(label='# Enter Keywords:', text='Press
enter to add more', value=['Zero', 'One', 'Two'],
suggestions=['five', 'six', 'seven', 'eight',
'nine', 'three', 'eleven', 'ten', 'four'], maxtags
= 4, key='1')
```



NLU

[Apply text mining on a dataframe.](#) Created by [@JohnSnowLabs](#).

```
ml.load('sentiment').predict('I love NLU! <3')
```



[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
mention(label="An awesome Streamlit App",
icon="streamlit",
url="https://extras.streamlit.app",)
```

Data elements





Dataframes

[Display a dataframe as an interactive table.](#)

```
st.dataframe(my_data_frame).
```



Data editor

[Display a data editor widget.](#)

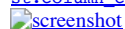
```
edited = st.data_editor(df, num_rows="dynamic").
```



Column configuration

[Configure the display and editing behavior of dataframes and data editors.](#)

```
st.column_config.NumberColumn("Price (in USD)", min_value=0, format="$%d").
```



Static tables

[Display a static table.](#)

```
st.table(my_data_frame).
```



Metrics

[Display a metric in big bold font, with an optional indicator of how the metric changed.](#)

```
st.metric("My metric", 42, 2).
```



Dicts and JSON

[Display object or string as a pretty-printed JSON string.](#)

```
st.json(my_dict).
```

Third-party components



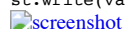
These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!



Image Coordinates


[Get the coordinates of clicks on an image. Created by @blackary.](#)

```
from streamlit_image_coordinates import
streamlit_image_coordinates value =
streamlit_image_coordinates("https://placekitten.com/200/300")
st.write(value)
```




Plotly Events

[Make Plotly charts interactive!. Created by @null-jones.](#)

```
from streamlit_plotly_events import plotly_events
fig = px.line(x=[1], y=[1]) selected_points =
plotly_events(fig)

```


[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
from streamlit_extras.metric_cards import
style_metric_cards col3.metric(label="No Change",
value=5000, delta=0) style_metric_cards()

```


[Streamlit Aggrid](#)

[Implementation of Ag-Grid component for Streamlit. Created by @PablocFonseca.](#)

```
df = pd.DataFrame({'col1': [1, 2, 3], 'col2': [4,
5, 6]}) grid_return = AgGrid(df, editable=True)
new_df = grid_return['data']

```


[Streamlit Folium](#)

[Streamlit Component for rendering Folium maps. Created by @randyzwitch.](#)

```
m = folium.Map(location=[39.949610, -75.150282],
zoom_start=16) folium.Marker([39.949610,
-75.150282], popup="Liberty Bell", tooltip="Liberty
Bell").add_to(m) st_data = st_folium(m, width=725)

```


[Pandas Profiling](#)

[Pandas profiling component for Streamlit. Created by @okld.](#)

```
df =
pd.read_csv("https://storage.googleapis.com/tf-
datasets/titanic/train.csv") pr =
df.profile_report() st_profile_report(pr)

```


[Image Coordinates](#)

[Get the coordinates of clicks on an image. Created by @blackary.](#)

```
from streamlit_image_coordinates import
streamlit_image_coordinates value =
streamlit_image_coordinates("https://placekitten.com/200/300")
st.write(value)

```


[Plotly Events](#)

[Make Plotly charts interactive!. Created by @null-jones.](#)

```
from streamlit_plotly_events import plotly_events
fig = px.line(x=[1], y=[1]) selected_points =
plotly_events(fig)

```

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
from streamlit_extras.metric_cards import
style_metric_cards col3.metric(label="No Change",
value=5000, delta=0) style_metric_cards()

```

[Streamlit Aggrid](#)

[Implementation of Ag-Grid component for Streamlit. Created by @PablocFonseca.](#)


```
df = pd.DataFrame({'col1': [1, 2, 3], 'col2': [4,
5, 6]}) grid_return = AgGrid(df, editable=True)
new_df = grid_return['data']
```

 [screenshot](#)

[Streamlit Folium](#)

[Streamlit Component for rendering Folium maps. Created by @randyzwitch.](#)


```
m = folium.Map(location=[39.949610, -75.150282],
zoom_start=16) folium.Marker([39.949610,
-75.150282], popup="Liberty Bell", tooltip="Liberty
Bell").add_to(m) st_data = st_folium(m, width=725)
```

 [screenshot](#)

[Pandas Profiling](#)

[Pandas profiling component for Streamlit. Created by @okld.](#)


```
df =
pd.read_csv("https://storage.googleapis.com/tf-
datasets/titanic/train.csv") pr =
df.profile_report() st_profile_report(pr)
```

 [screenshot](#)

[Image Coordinates](#)

[Get the coordinates of clicks on an image. Created by @blackary.](#)


```
from streamlit_image_coordinates import
streamlit_image_coordinates value =
streamlit_image_coordinates("https://placekitten.com/200/300")
st.write(value)
```

 [screenshot](#)

[Plotly Events](#)

[Make Plotly charts interactive!. Created by @null-jones.](#)

```
from streamlit_plotly_events import plotly_events
fig = px.line(x=[1], y=[1]) selected_points =
plotly_events(fig)
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
from streamlit_extras.metric_cards import
style_metric_cards col3.metric(label="No Change",
value=5000, delta=0) style_metric_cards()
```

Chart elements



 [screenshot](#)

[Simple area charts](#)

[Display an area chart.](#)


```
st.area_chart(my_data_frame).
```

 [screenshot](#)

[Simple bar charts](#)

[Display a bar chart.](#)


```
st.bar_chart(my_data_frame).
```

 [screenshot](#)

[Simple line charts](#)

[Display a line chart.](#)

```
st.line_chart(my_data_frame).
```

 [screenshot](#)

[Simple scatter charts](#)

[Display a line chart.](#)

```
st.scatter_chart(my_data_frame).
```

 [screenshot](#)

[Scatterplots on maps](#)

[Display a map with points on it.](#)

```
st.map(my_data_frame).
```

 [screenshot](#)

[Matplotlib](#)

[Display a matplotlib.pyplot figure.](#)


```
st.pyplot(my_mpl_figure).
```

 [screenshot](#)

[Altair](#)

[Display a chart using the Altair library.](#)

```
st.altair_chart(my_altair_chart).
```

 [screenshot](#)

[Vega-Lite](#)

[Display a chart using the Vega-Lite library.](#)


```
st.vega_lite_chart(my_vega_lite_chart).
```

 [screenshot](#)

[Plotly](#)

[Display an interactive Plotly chart.](#)


```
st.plotly_chart(my_plotly_chart).
```

 [screenshot](#)

[Bokeh](#)

[Display an interactive Bokeh chart.](#)

```
st.bokeh_chart(my_bokeh_chart).
```

 [screenshot](#)

[PyDeck](#)

[Display a chart using the PyDeck library.](#)

```
st.pydeck_chart(my_pydeck_chart).
```

 [screenshot](#)

[GraphViz](#)

[Display a graph using the dagre-d3 library.](#)


```
st.graphviz_chart(my_graphviz_spec).
```

Third-party components



These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!


[Previous](#)

 [screenshot](#)

[Streamlit Lottie](#)

[Integrate Lottie](#) animations inside your Streamlit app. Created by [@andfanilo](#).


```
lottie_hello =  
load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_v9t630.json")  
st_lottie(lottie_hello, key="hello")
```

 [screenshot](#)

[Plotly Events](#)

[Make Plotly charts interactive!](#). Created by [@null-jones](#).

```
fig = px.line(x=[1], y=[1]) selected_points =  
plotly_events(fig)
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
chart += get_annotations_chart(annotations=[("Mar 01, 2008", "Pretty good day for GOOG"), ("Dec 01, 2007", "Something's going wrong for GOOG & AAPL"), ("Nov 01, 2008", "Market starts again thanks to..."), ("Dec 01, 2009", "Small crash for GOOG after..."),],),) st.altair_chart(chart, use_container_width=True)
```



[Plotst](#)

[A deceptively simple plotting library for Streamlit. Created by @tvst.](#)

```
import plotst plotst.line_chart(my_dataframe, x='time', y='stock_value', color='stock_name',)
```



[HiPlot](#)

[High dimensional Interactive Plotting. Created by @facebookresearch.](#)

```
data = [{ 'dropout':0.1, 'lr': 0.001, 'loss': 10.0, 'optimizer': 'SGD'}, { 'dropout':0.15, 'lr': 0.01, 'loss': 3.5, 'optimizer': 'Adam'}, { 'dropout':0.3, 'lr': 0.1, 'loss': 4.5, 'optimizer': 'Adam'}] hip.Experiment.from_iterable(data).display()
```



[ECharts](#)

[High dimensional Interactive Plotting. Created by @andfanilo.](#)

```
from streamlit_echarts import st_echarts st_echarts(options=options)
```



[Streamlit Folium](#)

[Streamlit Component for rendering Folium maps. Created by @randyzwitch.](#)

```
m = folium.Map(location=[39.949610, -75.150282], zoom_start=16) st_data = st_folium(m, width=725)
```



[Spacy-Streamlit](#)

[spaCy building blocks and visualizers for Streamlit apps. Created by @explosion.](#)

```
models = ["en_core_web_sm", "en_core_web_md"] spacy_streamlit.visualize(models, "Sundar Pichai is the CEO of Google.")
```



[Streamlit Agraph](#)

[A Streamlit Graph Vis. based on react-grah-vis. Created by @ChrisDeIClea.](#)

```
from streamlit_agraph import agraph, Node, Edge, Config agraph(nodes=nodes, edges=edges, config=config)
```



[Streamlit Lottie](#)


[Integrate Lottie animations inside your Streamlit app. Created by @andfanilo.](#)

```
lottie_hello = load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_V9t630.json") st_lottie(lottie_hello, key="hello")
```



[Plotly Events](#)

[Make Plotly charts interactive!. Created by @null-jones.](#)

```
fig = px.line(x=[1], y=[1]) selected_points =
plotly_events(fig)

```

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

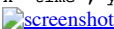
```
chart += get_annotations_chart(annotations=[("Mar
01, 2008", "Pretty good day for GOOG"), ("Dec 01,
2007", "Something's going wrong for GOOG & AAPL"),
("Nov 01, 2008", "Market starts again thanks
to..."), ("Dec 01, 2009", "Small crash for GOOG
after..."),],) st.altair_chart(chart,
use_container_width=True)
```



[Plotst](#)

[A deceptively simple plotting library for Streamlit. Created by @tvst.](#)

```
import plotst
plotst.line_chart(my_dataframe,
x='time', y='stock_value', color='stock_name',)
```



[HiPlot](#)

[High dimensional Interactive Plotting. Created by @facebookresearch.](#)

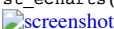
```
data = [{ 'dropout': 0.1, 'lr': 0.001, 'loss': 10.0,
'optimizer': 'SGD'}, { 'dropout': 0.15, 'lr': 0.01,
'loss': 3.5, 'optimizer': 'Adam'}, { 'dropout': 0.3,
'lr': 0.1, 'loss': 4.5, 'optimizer': 'Adam'}]
hip.Experiment.from_iterable(data).display()
```



[ECharts](#)

[High dimensional Interactive Plotting. Created by @andfanilo.](#)

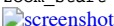
```
from streamlit_echarts import st_echarts
st_echarts(options=options)
```



[Streamlit Folium](#)

[Streamlit Component for rendering Folium maps. Created by @randyzwitch.](#)


```
m = folium.Map(location=[39.949610, -75.150282],
zoom_start=16) st_data = st_folium(m, width=725)
```



[Spacy-Streamlit](#)

[spaCy building blocks and visualizers for Streamlit apps. Created by @explosion.](#)

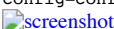
```
models = ["en_core_web_sm", "en_core_web_md"]
spacy_streamlit.visualize(models, "Sundar Pichai is
the CEO of Google.")
```



[Streamlit Agraph](#)

[A Streamlit Graph Vis, based on react-grah-vis. Created by @ChrisDelClea.](#)

```
from streamlit_agraph import agraph, Node, Edge,
Config
agraph(nodes=nodes, edges=edges,
config=config)
```



[Streamlit Lottie](#)

[Integrate Lottie animations inside your Streamlit app. Created by @andfanilo.](#)


```
lottie_hello =
load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_v9t630.json")
st_lottie(lottie_hello, key="hello")
```

 [screenshot](#)

[Plotly Events](#)

[Make Plotly charts interactive!. Created by @null-jones.](#)

```
fig = px.line(x=[1], y=[1]) selected_points =  
plotly_events(fig)
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
chart += get_annotations_chart(annotations=[("Mar  
01, 2008", "Pretty good day for GOOG"), ("Dec 01,  
2007", "Something's going wrong for GOOG & AAPL"),  
("Nov 01, 2008", "Market starts again thanks  
to..."), ("Dec 01, 2009", "Small crash for GOOG  
after..."),],) st.altair_chart(chart,  
use_container_width=True)
```

Input widgets

8

 [screenshot](#)

[Button](#)

[Display a button widget.](#)

```
clicked = st.button("Click me").
```

 [screenshot](#)

[Download button](#)

[Display a download button widget.](#)

```
st.download_button("Download file", _file).
```

 [screenshot](#)

[Form button](#)

[Display a form submit button. For use with `st.form`.](#)

```
st.form_submit_button("Sign up").
```

 [screenshot](#)

[Link button](#)

[Display a link button.](#)

```
st.link_button("Go to gallery", url).
```

 [screenshot](#)

[Page link](#)

[Display a link to another page in a multipage app.](#)


```
st.page_link("app.py", label="Home", icon="🏠") st.page_link("pages/profile.py", label="My profile").
```

 [screenshot](#)

[Checkbox](#)

[Display a checkbox widget.](#)


```
selected = st.checkbox("I agree").
```

 [screenshot](#)

[Color picker](#)

[Display a color picker widget.](#)

```
color = st.color_picker("Pick a color").
```

 [screenshot](#)

[Feedback](#)

[Display a rating or sentiment button group.](#)


```
st.feedback("stars").
```

 [screenshot](#)

Multiselect

[Display a multiselect widget. The multiselect widget starts as empty.](#)


```
choices = st.multiselect("Buy",_["milk", "apples", "potatoes"]).
```

 [screenshot](#)

Pills

[Display a pill-button selection widget.](#)


```
st.pills("Tags",_["Sports", "AI", "Politics"]).
```

 [screenshot](#)

Radio

[Display a radio button widget.](#)


```
choice = st.radio("Pick one",_["cats", "dogs"]).
```

 [screenshot](#)

Segmented control

[Display a segmented-button selection widget.](#)


```
st.segmented_control("Filter",_["Open", "Closed", "All"]).
```

 [screenshot](#)

Selectbox

[Display a select widget.](#)


```
choice = st.selectbox("Pick one",_["cats", "dogs"]).
```

 [screenshot](#)

Select-slider

[Display a slider widget to select items from a list.](#)

```
size = st.select_slider("Pick a size",_["S", "M", "L"]).
```

 [screenshot](#)

Toggle

[Display a toggle widget.](#)


```
activated = st.toggle("Activate").
```

 [screenshot](#)

Number input

[Display a numeric input widget.](#)


```
choice = st.number_input("Pick a number",_0, 10).
```

 [screenshot](#)

Slider

[Display a slider widget.](#)


```
number = st.slider("Pick a number",_0, 100).
```

 [screenshot](#)

Date input

[Display a date input widget.](#)

```
date = st.date_input("Your birthday").
```

 [screenshot](#)

Time input

[Display a time input widget.](#)

```
time = st.time_input("Meeting time").
```

 [screenshot](#)

Chat input

[Display a chat input widget.](#)


```
prompt = st.chat_input("Say something") if prompt: st.write(f"The user has sent: {prompt}").
```

 [screenshot](#)

[Text-area](#)

[Display a multi-line text input widget.](#)


```
text = st.text_area("Text to translate").
```

 [screenshot](#)

[Text input](#)

[Display a single-line text input widget.](#)


```
name = st.text_input("First name").
```

 [screenshot](#)

[Audio input](#)

[Display a widget that allows users to record with their microphone.](#)


```
speech = st.audio_input("Record a voice message").
```

 [screenshot](#)

[Data editor](#)

[Display a data editor widget.](#)


```
edited = st.data_editor(df, num_rows="dynamic").
```

 [screenshot](#)

[File uploader](#)

[Display a file uploader widget.](#)

```
data = st.file_uploader("Upload a CSV").
```

 [screenshot](#)

[Camera input](#)

[Display a widget that allows users to upload images directly from a camera.](#)

```
image = st.camera_input("Take a picture").
```

Third-party components



These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!


[Previous](#)

 [screenshot](#)

[Streamlit Chat](#)

[Streamlit Component for a Chatbot UI. Created by @AI-Yash.](#)


```
from streamlit_chat import message message("My  
message") message("Hello bot!", is_user=True) #  
align's the message to the right
```

 [screenshot](#)

[Streamlit Option Menu](#)

[Select a single item from a list of options in a menu. Created by @victoryhb.](#)


```
from streamlit_option_menu import option_menu  
option_menu("Main Menu", ["Home", 'Settings'],  
icons=['house', 'gear'], menu_icon="cast",  
default_index=1)
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)


```
from streamlit_extras.stoggle import stoggle  
stoggle( "Click me!", ""👤 Surprise! Here's some  
additional content"",)
```

 [screenshot](#)

[Streamlit Elements](#)

[Create a draggable and resizable dashboard in Streamlit.](#)
Created by [@okls](#).

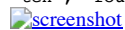
```
from streamlit_elements import elements, mui, html
with elements("new_element"): mui.Typography("Hello
world")
```



[Tags](#)

[Add tags to your Streamlit apps.](#) Created by [@gagan3012](#).

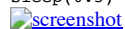
```
from streamlit_tags import st_tags st_tags(label='#
Enter Keywords:', text='Press enter to add more',
value=['Zero', 'One', 'Two'], suggestions=['five',
'six', 'seven', 'eight', 'nine', 'three', 'eleven',
'ten', 'four'], maxtags = 4, key='1')
```



[Stqdm](#)

[The simplest way to handle a progress bar in streamlit app.](#)
Created by [@Wirg](#).

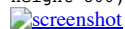
```
from tqdm import tqdm for _ in tqdm(range(50)):
sleep(0.5)
```



[Timeline](#)

[Display a Timeline in Streamlit apps using TimelineJS.](#)
Created by [@innerdoc](#).

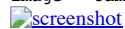
```
from streamlit_timeline import timeline with
open('example.json', "r") as f: timeline(f.read(),
height=800)
```



[Camera input live](#)

[Alternative for st.camera_input which returns the webcam images live.](#) Created by [@blackary](#).

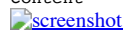
```
from camera_input_live import camera_input_live
image = camera_input_live() st.image(value)
```



[Streamlit Ace](#)

[Ace editor component for Streamlit.](#) Created by [@okld](#).

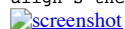
```
from streamlit_ace import st_ace content = st_ace()
content
```



[Streamlit Chat](#)

[Streamlit Component for a Chatbot UI.](#) Created by [@AI-Yash](#).

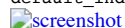
```
from streamlit_chat import message message("My
message") message("Hello bot!", is_user=True) #
align's the message to the right
```



[Streamlit Option Menu](#)

[Select a single item from a list of options in a menu.](#) Created by [@victoryhb](#).

```
from streamlit_option_menu import option_menu
option_menu("Main Menu", ["Home", 'Settings'],
icons=['house', 'gear'], menu_icon="cast",
default_index=1)
```



[Streamlit Extras](#)

[A library with useful Streamlit extras.](#) Created by [@arnaudmiribel](#).


```
from streamlit_extras.stoggle import stoggle
stoggle( "Click me!", ""👤 Surprise! Here's some
additional content"", )
```

 [screenshot](#)

[Streamlit Elements](#)

[Create a draggable and resizable dashboard in Streamlit.](#)
Created by [@okls](#).


```
from streamlit_elements import elements, mui, html
with elements("new_element"): mui.Typography("Hello
world")
```

 [screenshot](#)

[Tags](#)

[Add tags to your Streamlit apps. Created by @gagan3012.](#)


```
from streamlit_tags import st_tags st_tags(label='#
Enter Keywords:', text='Press enter to add more',
value=['zero', 'One', 'Two'], suggestions=['five',
'six', 'seven', 'eight', 'nine', 'three', 'eleven',
'ten', 'four'], maxtags = 4, key='1')
```

 [screenshot](#)

[Stqdm](#)

[The simplest way to handle a progress bar in streamlit app.](#)
Created by [@Wirg](#).


```
from tqdm import tqdm for _ in tqdm(range(50)):
sleep(0.5)
```

 [screenshot](#)

[Timeline](#)

[Display a Timeline in Streamlit apps using TimelineJS.](#)
Created by [@innerdoc](#).


```
from streamlit_timeline import timeline with
open('example.json', "r") as f: timeline(f.read(),
height=800)
```

 [screenshot](#)

[Camera input live](#)

[Alternative for st.camera_input which returns the webcam images live. Created by @blackary.](#)


```
from camera_input_live import camera_input_live
image = camera_input_live() st.image(value)
```

 [screenshot](#)

[Streamlit Ace](#)

[Ace editor component for Streamlit. Created by @okld.](#)


```
from streamlit_ace import st_ace content = st_ace()
content
```

 [screenshot](#)

[Streamlit Chat](#)

[Streamlit Component for a Chatbot UI. Created by @AI-Yash.](#)


```
from streamlit_chat import message message("My
message") message("Hello bot!", is_user=True) #
align's the message to the right
```

 [screenshot](#)

[Streamlit Option Menu](#)

[Select a single item from a list of options in a menu. Created by @victoryhb.](#)

```
from streamlit_option_menu import option_menu
option_menu("Main Menu", ["Home", 'Settings'],
icons=['house', 'gear'], menu_icon="cast",
default_index=1)
```

 [screenshot](#)

[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by @arnaudmiribel.](#)

```
from streamlit_extras.stoggle import stoggle
stoggle( "Click me!", ""👤 Surprise! Here's some
additional content"", )
```

Next

Media elements

8

 [screenshot](#)

Image

[Display an image or list of images.](#)


```
st.image(numpy_array) st.image(image_bytes) st.image(file) st.image("https://example.com/myimage.jpg").
```

 [screenshot](#)

Logo

[Display a logo in the upper-left corner of your app and its sidebar.](#)

```
st.logo("logo.jpg").
```

 [screenshot](#)

Audio

[Display an audio player.](#)

```
st.audio(numpy_array) st.audio(audio_bytes) st.audio(file) st.audio("https://example.com/myaudio.mp3", format="audio/mp3").
```

 [screenshot](#)

Video

[Display a video player.](#)

```
st.video(numpy_array) st.video(video_bytes) st.video(file) st.video("https://example.com/myvideo.mp4", format="video/mp4").
```

Third-party components



These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!

Previous

 [screenshot](#)

Streamlit Cropper

[A simple image cropper for Streamlit. Created by @turner-anderson.](#)

```
from streamlit_cropper import st_cropper
st_cropper(img, realtime_update=realtime_update,
box_color=box_color, aspect_ratio=aspect_ratio)
```


 [screenshot](#)

Image Coordinates

[Get the coordinates of clicks on an image. Created by @blackary.](#)


```
from streamlit_image_coordinates import
streamlit_image_coordinates
streamlit_image_coordinates("https://placekitten.com/200/300")
```

 [screenshot](#)

Streamlit Lottie

[Integrate Lottie](#) animations inside your Streamlit app. Created by [@andfanilo](#).

```
lottie_hello =
load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_v9t630.json")
st_lottie(lottie_hello, key="hello")
```

 [screenshot](#)

Streamlit Webrtc

[Handling and transmitting real-time video/audio streams with Streamlit.](#) Created by [@whitphx](#).

```
from streamlit_webrtc import webrtc_streamer
webrtc_streamer(key="sample")
```


 [screenshot](#)

[Drawable Canvas](#)

[Provides a sketching canvas using Fabric.js](#). Created by [@andfanilo](#).

```
from streamlit_drawable_canvas import st_canvas
st_canvas(fill_color="rgba(255, 165, 0, 0.3)",
stroke_width=stroke_width,
stroke_color=stroke_color,
background_color=bg_color,
background_image=Image.open(bg_image) if bg_image
else None, update_streamlit=realtime_update,
height=150, drawing_mode=drawing_mode,
point_display_radius=point_display_radius if
drawing_mode == 'point' else 0, key="canvas",)
```

 [screenshot](#)

[Image Comparison](#)

[Compare images with a slider using JuxtaposeJS](#). Created by [@fcakyon](#).


```
from streamlit_image_comparison import
image_comparison
image_comparison(img1="image1.jpg",
img2="image2.jpg",)
```

 [screenshot](#)

[Streamlit Cropper](#)

[A simple image cropper for Streamlit](#). Created by [@turner-anderson](#).


```
from streamlit_cropper import st_cropper
st_cropper(img, realtime_update=realtime_update,
box_color=box_color, aspect_ratio=aspect_ratio)
```

 [screenshot](#)

[Image Coordinates](#)

[Get the coordinates of clicks on an image](#). Created by [@blackary](#).


```
from streamlit_image_coordinates import
streamlit_image_coordinates
streamlit_image_coordinates("https://placekitten.com/200/300")
```

 [screenshot](#)

[Streamlit Lottie](#)

[Integrate Lottie](#) animations inside your Streamlit app. Created by [@andfanilo](#).


```
lottie_hello =
load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_v9t630.json")
st_lottie(lottie_hello, key="hello")
```

 [screenshot](#)

[Streamlit Webrtc](#)

[Handling and transmitting real-time video/audio streams with Streamlit](#). Created by [@whitphx](#).

```
from streamlit_webrtc import webrtc_streamer
webrtc_streamer(key="sample")
```

 [screenshot](#)

[Drawable Canvas](#)

[Provides a sketching canvas using Fabric.js](#). Created by [@andfanilo](#).


```
from streamlit_drawable_canvas import st_canvas
st_canvas(fill_color="rgba(255, 165, 0, 0.3)",
stroke_width=stroke_width,
stroke_color=stroke_color,
background_color=bg_color,
background_image=Image.open(bg_image) if bg_image
else None, update_streamlit=realtime_update,
height=150, drawing_mode=drawing_mode,
point_display_radius=point_display_radius if
drawing_mode == 'point' else 0, key="canvas",)
```

 [screenshot](#)

[Image Comparison](#)

[Compare images with a slider using JuxtaposeJS](#). Created by [@fcakyon](#).


```
from streamlit_image_comparison import
image_comparison
image_comparison(img1="image1.jpg",
img2="image2.jpg",)
```



[Streamlit Cropper](#)

[A simple image cropper for Streamlit](#). Created by [@turner-anderson](#).


```
from streamlit_cropper import st_cropper
st_cropper(img, realtime_update=realtime_update,
box_color=box_color, aspect_ratio=aspect_ratio)
```



[Image Coordinates](#)

[Get the coordinates of clicks on an image](#). Created by [@blackary](#).

```
from streamlit_image_coordinates import
streamlit_image_coordinates
streamlit_image_coordinates("https://placekitten.com/200/300")
```



[Streamlit Lottie](#)

[Integrate Lottie](#) animations inside your Streamlit app. Created by [@andfanilo](#).

```
lottie_hello =
load_lottieurl("https://assets5.lottiefiles.com/packages/lf20_V9t630.json")
st_lottie(lottie_hello, key="hello")
```

[Layouts and containers](#)




 [screenshot](#)

[Columns](#)

[Insert containers laid out as side-by-side columns](#).


```
col1, col2 = st.columns(2) col1.write("this is column 1") col2.write("this is column 2")
```



[Container](#)

[Insert a multi-element container](#).


```
c = st.container() st.write("This will show last") c.write("This will show first") c.write("This will show second")
```



[Modal dialog](#)

[Insert a modal dialog that can rerun independently from the rest of the script](#).


```
@st.dialog("Sign up") def email_form(): name = st.text_input("Name") email = st.text_input("Email")
```



[Empty](#)

[Insert a single-element container](#).

```
c = st.empty() st.write("This will show last") c.write("This will be replaced") c.write("This will show first")
```



[Expander](#)

[Insert a multi-element container that can be expanded/collapsed](#).


```
with st.expander("Open to see more"): st.write("This is more content")
```

 [screenshot](#)

[Popover](#)

[Insert a multi-element popover container that can be opened/closed.](#)


```
with st.popover("Settings"): st.checkbox("Show completed").
```

 [screenshot](#)

[Sidebar](#)

[Display items in a sidebar.](#)

```
st.sidebar.write("This lives in the sidebar") st.sidebar.button("Click me!").
```

 [screenshot](#)

[Tabs](#)

[Insert containers separated into tabs.](#)

```
tab1, tab2 = st.tabs(["Tab 1", "Tab2"]) tab1.write("this is tab 1") tab2.write("this is tab 2").
```

Third-party components




These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!

 [screenshot](#)

[Streamlit Elements](#)

[Create a draggable and resizable dashboard in Streamlit.](#)
[Created by @okls.](#)


```
from streamlit_elements import elements, mui, html
with elements("new_element"): mui.Typography("Hello
world")
```

 [screenshot](#)

[Pydantic](#)

[Auto-generate Streamlit UI from Pydantic Models and Dataclasses.](#) [Created by @lukasmasuch.](#)

```
import streamlit_pydantic as sp
sp.pydantic_form(key="my_form", model=ExampleModel)
```

 [screenshot](#)

[Streamlit Pages](#)

[An experimental version of Streamlit Multi-Page Apps.](#)
[Created by @blackary.](#)

```
from st_pages import Page, show_pages,
add_page_title show_pages([
Page("streamlit_app.py", "Home", "🏠"),
Page("other_pages/page2.py", "Page 2", ":books:"),
])
```

[Chat elements](#)



Streamlit provides a few commands to help you build conversational apps. These chat elements are designed to be used in conjunction with each other, but you can also use them separately.

`st.chat_message` lets you insert a chat message container into the app so you can display messages from the user or the app. Chat containers can contain other Streamlit elements, including charts, tables, text, and more. `st.chat_input` lets you display a chat input widget so the user can type in a message.

 [screenshot](#)

[Chat input](#)

[Display a chat input widget.](#)

```
prompt = st.chat_input("Say something") if prompt: st.write(f"The user has sent: {prompt}").
```

 [screenshot](#)

[Chat message](#)

[Insert a chat message container.](#)

```
import numpy as np with st.chat_message("user"): st.write("Hello 🐼") st.line_chart(np.random.randn(30, 3)).
```

 [screenshot](#)

Status container

[Display output of long-running tasks in a container.](#)

```
with st.status('Running'): do_something_slow().
```

st.write stream

[Write generators or streams to the app with a typewriter effect.](#)

```
st.write_stream(my_generator) st.write_stream(my_llm_stream).
```

Status elements

8

 [screenshot](#)

Progress bar

[Display a progress bar.](#)

```
for i in range(101): st.progress(i) do_something_slow().
```

 [screenshot](#)

Spinner

[Temporarily displays a message while executing a block of code.](#)

```
with st.spinner("Please wait..."): do_something_slow().
```

 [screenshot](#)

Status container

[Display output of long-running tasks in a container.](#)

```
with st.status('Running'): do_something_slow().
```

 [screenshot](#)

Toast

[Briefly displays a toast message in the bottom-right corner.](#)

```
st.toast('Butter!', icon='🍞').
```

 [screenshot](#)

Balloons

[Display celebratory balloons!](#)

```
do_something(.) # Celebrate when all done! st.balloons().
```

 [screenshot](#)

Snowflakes

[Display celebratory snowflakes!](#)

```
do_something(.) # Celebrate when all done! st.snow().
```

 [screenshot](#)

Success box

[Display a success message.](#)

```
st.success("Match found!").
```

 [screenshot](#)

Info box

[Display an informational message.](#)

```
st.info("Dataset is updated every day at midnight..").
```

 [screenshot](#)

Warning box

[Display warning message.](#)

```
st.warning("Unable to fetch image. Skipping...").
```

 [screenshot](#)

Error box

[Display error message.](#)

```
st.error("We encountered an error").
```



[Exception output](#)

[Display an exception.](#)

```
e = RuntimeError("This is an exception of type RuntimeError") st.exception(e).
```

Third-party components



These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!

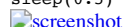


[Stqdm](#)

[The simplest way to handle a progress bar in streamlit app.](#)

[Created by @Wirg.](#)

```
from tqdm import tqdm for _ in tqdm(range(50)):
sleep(0.5)
```



[Custom notification box](#)

[A custom notification box with the ability to close it out.](#)

[Created by @Socvest.](#)

```
from streamlit_custom_notification_box import
custom_notification_box styles = {'material-icons':
{'color': 'red'}, 'text-icon-link-close-container':
{'box-shadow': '#3896de 0px 4px'}, 'notification-
text': {'': ''}, 'close-button': {'': ''}, 'link':
{'': ''}} custom_notification_box(icon='info',
textDisplay='We are almost done with your
registration...', externalLink='more info',
url='#', styles=styles, key="foo")
```



[Streamlit Extras](#)

[A library with useful Streamlit extras. Created by](#)

[@arnaudmiribel.](#)

```
from streamlit_extras.let_it_rain import rain
rain(emoji="🍷", font_size=54, falling_speed=5,
animation_length="infinite",)
```

App logic and configuration



Navigation and pages





[Navigation](#)

[Configure the available pages in a multipage app.](#)

```
st.navigation({"Your account" : [log_out, settings], "Reports" : [overview, usage], "Tools" : [search]}).
```



[Page](#)

[Define a page in a multipage app.](#)

```
home = st.Page("home.py", title="Home", icon="material/home:").
```



[Page link](#)

[Display a link to another page in a multipage app.](#)

```
st.page_link("app.py", label="Home", icon="🏠") st.page_link("pages/profile.py", label="My profile").
```

[Switch page](#)

[Programmatically navigates to a specified page.](#)

`st.switch_page("pages/my_page.py").`

Execution flow

8



Modal dialog

[Insert a modal dialog that can rerun independently from the rest of the script.](#)

`@st.dialog("Sign up") def email_form(): name = st.text_input("Name") email = st.text_input("Email").`

Forms

[Create a form that batches elements together with a "Submit" button.](#)

`with st.form(key='my_form'): name = st.text_input("Name") email = st.text_input("Email") st.form_submit_button("Sign up").`

Fragments

[Define a fragment to rerun independently from the rest of the script.](#)

`@st.fragment(run_every="10s") def fragment(): df = get_data() st.line_chart(df).`

Rerun script

[Rerun the script immediately.](#)

`st.rerun().`

Stop execution

[Stops execution immediately.](#)

`st.stop().`

Third-party components



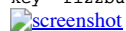
These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras!](#)



Autorefresh

[Force a refresh without tying up a script. Created by @kmcgrady.](#)

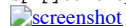
```
from streamlit_autorefresh import st_autorefresh
st_autorefresh(interval=2000, limit=100,
key="fizzbuzzcounter")
```



Pydantic

[Auto-generate Streamlit UI from Pydantic Models and Dataclasses. Created by @lukasmasuch.](#)

```
import streamlit_pydantic as sp
sp.pydantic_form(key="my_form", model=ExampleModel)
```



Streamlit Pages

[An experimental version of Streamlit Multi-Page Apps. Created by @blackary.](#)

```
from st_pages import Page, show_pages,
add_page_title show_pages([
Page("streamlit_app.py", "Home", "🏠"),
Page("other_pages/page2.py", "Page 2", ":books:"),
])
```

Caching and state

8

Cache data

Function decorator to cache functions that return data (e.g. dataframe transforms, database queries, ML inference).

```
@st.cache_data def long_function(param1, _param2): # Perform expensive computation here or # fetch data from the web here return data
```

Cache resource

Function decorator to cache functions that return global resources (e.g. database connections, ML models).

```
@st.cache_resource def init_model(): # Return a global resource here return pipeline( "sentiment-analysis", model="distilbert-base-uncased-finetuned-sst-2-english").
```

Session state

Session state is a way to share variables between reruns, for each user session.

```
st.session_state['key'] = value
```

Query parameters

Get, set, or clear the query parameters that are shown in the browser's URL bar.

```
st.query_params[key] = value st.query_params.clear().
```

Connections and databases

8

Setup your connection



Create a connection

Connect to a data source or API

```
conn = st.connection('pets_db', type='sql') pet_owners = conn.query('select * from pet_owners') st.dataframe(pet_owners).
```

Built-in connections



SnowflakeConnection

A connection to Snowflake.

```
conn = st.connection('snowflake').
```



SQLConnection

A connection to a SQL database using SQLAlchemy.

```
conn = st.connection('sql').
```

Build your own connections

Connection base class

Build your own connection with BaseConnection.

```
class MyConnection(BaseConnection[myconn.MyConnection]): def _connect(self, **kwargs) -> MyConnection: return myconn.connect(**self._secrets, **kwargs). def query(self, query): return self._instance.query(query).
```

Secrets management

Secrets singleton

Access secrets from a local TOML file.

```
key = st.secrets["OpenAI_key"].
```

Secrets file

Save your secrets in a per-project or per-profile TOML file.

```
OpenAI_key = "<YOUR_SECRET_KEY>"
```

Third-party components




These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!

 [screenshot](#)

[Authenticator](#)

[A secure authentication module to validate user credentials.](#)
Created by [@mkhorasani](#).


```
import streamlit_authenticator as stauth
authenticator = stauth.Authenticate(
    config['credentials'], config['cookie']['name'],
    config['cookie']['key'], config['cookie']
    ['expiry_days'], config['preauthorized'])
```

 [screenshot](#)

[WS localStorage](#)

[A simple synchronous way of accessing localStorage from your app.](#) Created by [@gagangoku](#).

```
from streamlit_ws_localstorage import
injectWebsocketCode ret =
conn.setLocalStorageVal(key='k1', val='v1')
st.write('ret: ' + ret)
```

 [screenshot](#)

[Streamlit Auth0](#)

[The fastest way to provide comprehensive login inside Streamlit.](#) Created by [@conradbez](#).

```
from auth0_component import login_button user_info
= login_button(clientId, domain = domain)
st.write(user_info)
```

Custom Components



[Declare a component](#)

[Create and register a custom component.](#)

```
from st.components.v1 import declare_component declare_component("custom_slider", "/frontend",_).
```

[HTML](#)

[Display an HTML string in an iframe.](#)

```
from st.components.v1 import html html("<p>Foo bar.</p>").
```

[iframe](#)

[Load a remote URL in an iframe.](#)

```
from st.components.v1 import iframe iframe("docs.streamlit.io").
```

Utilities and user info



[Context](#)

[st.context provides a read-only interface to access cookies and headers.](#)

[st.context.cookies st.context.headers](#)

[Get help](#)

[Display object's doc string, nicely formatted.](#)

```
st.help(st.write) st.help(pd.DataFrame).
```

[Render HTML](#)

[Renders HTML strings to your app.](#)

```
st.html("<p>Foo bar.</p>").
```

[User info](#)

[st.experimental_user returns information about the logged-in user of private apps on Streamlit Community Cloud.](#)


```
if st.experimental_user.email == "foo@corp.com": st.write("Welcome back, ", st.experimental_user.email) else: st.write("You are not authorized to view this page. ").
```

Configuration

8

Configuration file

Configures the default settings for your app.

```
your-project/ |— .streamlit/ |— config.toml |— your_app.py.
```

Get config option

Retrieve a single configuration option.

```
st.get_option("theme.primaryColor").
```

Set config option

Set a single configuration option. (This is very limited.)

```
st.set_option("deprecation.showPyplotGlobalUse", False).
```

Set page title, favicon, and more

Configures the default settings of the page.

```
st.set_page_config(_page_title="My app", _page_icon=":shark:", _).
```

Developer tools

8

App testing

8

st.testing.v1.AppTest

st.testing.v1.AppTest simulates a running Streamlit app for testing.

```
from streamlit.testing.v1 import AppTest at = AppTest.from_file("streamlit_app.py") at.secrets["WORD"] = "Foobar" at.run() assert not at.exception at.text_input("word").input("Bazbat").run() assert at.warning[0].value == "Try again."
```

AppTest.from_file

st.testing.v1.AppTest.from_file initializes a simulated app from a file.

```
from streamlit.testing.v1 import AppTest at = AppTest.from_file("streamlit_app.py") at.run().
```

AppTest.from_string

st.testing.v1.AppTest.from_string initializes a simulated app from a string.

```
from streamlit.testing.v1 import AppTest at = AppTest.from_string(app_script_as_string) at.run().
```

AppTest.from_function

st.testing.v1.AppTest.from_function initializes a simulated app from a function.

```
from streamlit.testing.v1 import AppTest at = AppTest.from_function(app_script_as_callable) at.run().
```

Block

A representation of container elements, including:

- [st.chat_message](#)
- [st.columns](#)
- [st.sidebar](#)
- [st.tabs](#)
- [The main body of the app.](#)

```
# at.sidebar returns a Block at.sidebar.button[0].click().run() assert not at.exception
```

Element

The base class for representation of all elements, including:

- [st.title](#)
- [st.header](#)

- [st.markdown](#)
- [st.dataframe](#)

[# at.title](#) returns a sequence of Title # Title inherits from Element assert `at.title[0].value == "My_awesome_app"`

[Button](#)

A representation of `st.button` and `st.form_submit_button`.

`at.button[0].click().run()`

[ChatInput](#)

A representation of `st.chat_input`.

`at.chat_input[0].set_value("What is Streamlit?").run()`

[Checkbox](#)

A representation of `st.checkbox`.

`at.checkbox[0].check().run()`

[ColorPicker](#)

A representation of `st.color_picker`.

`at.color_picker[0].pick("#FF4B4B").run()`

[DateInput](#)

A representation of `st.date_input`.

`release_date = datetime.date(2023, 10, 26) at.date_input[0].set_value(release_date).run()`

[Multiselect](#)

A representation of `st.multiselect`.

`at.multiselect[0].select("New York").run()`

[NumberInput](#)

A representation of `st.number_input`.

`at.number_input[0].increment().run()`

[Radio](#)

A representation of `st.radio`.

`at.radio[0].set_value("New York").run()`

[SelectSlider](#)

A representation of `st.select_slider`.

`at.select_slider[0].set_range("A","C").run()`

[Selectbox](#)

A representation of `st.selectbox`.

`at.selectbox[0].select("New York").run()`

[Slider](#)

A representation of `st.slider`.

`at.slider[0].set_range(2,5).run()`

[TextArea](#)

A representation of `st.text_area`.

`at.text_area[0].input("Streamlit is awesome!").run()`

[TextInput](#)

A representation of `st.text_input`.

`at.text_input[0].input("Streamlit").run()`

[TimeInput](#)

[A representation of st.time_input.](#)

`at.time_input[0].increment().run().`

[Toggle](#)

[A representation of st.toggle.](#)

`at.toggle[0].set_value("True").run().`

Third-party components




These are featured components created by our lovely community. For more examples and inspiration, check out our [Components Gallery](#) and [Streamlit Extras](#)!

 [screenshot](#)

[Pandas Profiling](#)

[Pandas profiling component for Streamlit. Created by @okld.](#)


```
df =
pd.read_csv("https://storage.googleapis.com/tf-
datasets/titanic/train.csv") pr =
df.profile_report() st_profile_report(pr)
```

 [screenshot](#)

[Streamlit Ace](#)

[Ace editor component for Streamlit. Created by @okld.](#)

```
from streamlit_ace import st_ace content = st_ace()
content
```

 [screenshot](#)

[Streamlit Analytics](#)

[Track & visualize user interactions with your streamlit app. Created by @jrieke.](#)

```
import streamlit_analytics with
streamlit_analytics.track(): st.text_input("Write
something")
```

[←Previous: Concepts](#)[Next: Write and magic→](#)

[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](#)