

# Cheat Sheet

This is a summary of the docs, as of Streamlit v1.18.0.

## Install & Import

```
streamlit run first_app.py
```

```
# Import convention
```

```
>>> import streamlit as st
```

## Command line

```
streamlit --help
```

```
streamlit run your_script.py
```

```
streamlit hello
```

```
streamlit config show
```

```
streamlit cache clear
```

```
streamlit docs
```

```
streamlit --version
```

## Pre-release features

```
pip uninstall streamlit
```

```
pip install streamlit-nightly --upgrade
```

Learn more about experimental features

## Magic commands

```
# Magic commands implicitly
```

```
# call st.write().
```

```
'_This_ is some **Markdown**'
```

```
my_variable  
'dataframe:', my_data_frame
```

## Display text

```
st.text('Fixed width text')  
st.markdown('_Markdown_') # see *  
st.latex(r''' e^{i\pi} + 1 = 0 ''')  
st.write('Most objects') # df, err, func, keras!  
st.write(['st', 'is <', 3]) # see *  
st.title('My title')  
st.header('My header')  
st.subheader('My sub')  
st.code('for i in range(8): foo()')  
* optional kwarg unsafe_allow_html = True
```

## Display data

```
st.dataframe(my_dataframe)  
st.table(data.iloc[0:10])  
st.json({'foo': 'bar', 'fu': 'ba'})  
st.metric('My metric', 42, 2)
```

## Display media

```
st.image('./header.png')  
st.audio(data)  
st.video(data)
```

## Add widgets to sidebar

*# Just add it after st.sidebar:*

```
>>> a = st.sidebar.radio('Select one:', [1, 2])
```

*# Or use "with" notation:*

```
>>> with st.sidebar:
```

```
>>>     st.radio('Select one:', [1, 2])
```

## Columns

*# Two equal columns:*

```
>>> col1, col2 = st.columns(2)
```

```
>>> col1.write("This is column 1")
```

```
>>> col2.write("This is column 2")
```

*# Three different columns:*

```
>>> col1, col2, col3 = st.columns([3, 1, 1])
```

*# col1 is larger.*

*# You can also use "with" notation:*

```
>>> with col1:
```

```
>>>     st.radio('Select one:', [1, 2])
```

## 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")
```

*# You can also use "with" notation:*

```
>>> with tab1:
```

```
>>>     st.radio('Select one:', [1, 2])
```

## Control flow

*# Stop execution immediately:*

```
st.stop()
```

*# Rerun script immediately:*

```
st.experimental_rerun()
```

*# Group multiple widgets:*

```
>>> with st.form(key='my_form'):
```

```
>>>     username = st.text_input('Username')
```

```
>>>     password = st.text_input('Password')
```

```
>>>     st.form_submit_button('Login')
```

## Display interactive widgets

```
st.button('Click me')
st.checkbox('I agree')
st.radio('Pick one', ['cats', 'dogs'])
st.selectbox('Pick one', ['cats', 'dogs'])
st.multiselect('Buy', ['milk', 'apples', 'potatoes'])
st.slider('Pick a number', 0, 100)
st.select_slider('Pick a size', ['S', 'M', 'L'])
st.text_input('First name')
st.number_input('Pick a number', 0, 10)
st.text_area('Text to translate')
st.date_input('Your birthday')
st.time_input('Meeting time')
st.file_uploader('Upload a CSV')
st.download_button('Download file', data)
st.camera_input("Take a picture")
st.color_picker('Pick a color')
```

*# Use widgets' returned values in variables:*

```
>>> for i in range(int(st.number_input('Num:'))):
>>>     foo()
>>> if st.sidebar.selectbox('I:', ['f']) == 'f':
>>>     b()
>>> my_slider_val = st.slider('Quinn Mallory', 1, 88)
>>> st.write(slider_val)
```

*# Disable widgets to remove interactivity:*

```
>>> st.slider('Pick a number', 0, 100, disabled=True)
```

## Mutate data

```
# Add rows to a dataframe after  
# showing it.
```

```
>>> element = st.dataframe(df1)  
>>> element.add_rows(df2)
```

```
# Add rows to a chart after  
# showing it.
```

```
>>> element = st.line_chart(df1)  
>>> element.add_rows(df2)
```

## Display code

```
>>> with st.echo():  
>>>     st.write('Code will be executed and printed')
```

## Placeholders, help, and options

*# Replace any single element.*

```
>>> element = st.empty()
>>> element.line_chart(...)
>>> element.text_input(...) # Replaces previous.
```

*# Insert out of order.*

```
>>> elements = st.container()
>>> elements.line_chart(...)
>>> st.write("Hello")
>>> elements.text_input(...) # Appears above "Hello".
```

```
st.help(pandas.DataFrame)
st.get_option(key)
st.set_option(key, value)
st.set_page_config(layout='wide')
st.experimental_show(objects)
st.experimental_get_query_params()
st.experimental_set_query_params(**params)
```

# Optimize performance

## Cache data objects

*# E.g. Dataframe computation, storing downloaded data, etc.*

```
>>> @st.cache_data
```

```
... def foo(bar):
```

```
...     # Do something expensive and return data
```

```
...     return data
```

*# Executes foo*

```
>>> d1 = foo(ref1)
```

*# Does not execute foo*

*# Returns cached item by value, d1 == d2*

```
>>> d2 = foo(ref1)
```

*# Different arg, so function foo executes*

```
>>> d3 = foo(ref2)
```

*# Clear all cached entries for this function*

```
>>> foo.clear()
```

*# Clear values from \*all\* in-memory or on-disk cached functions*

```
>>> st.cache_data.clear()
```

## Cache global resources

*# E.g. TensorFlow session, database connection, etc.*

```
>>> @st.cache_resource
```

```
... def foo(bar):
```

```
...     # Create and return a non-data object
```

```
...     return session
```

*# Executes foo*

```
>>> s1 = foo(ref1)
```

*# Does not execute foo*

*# Returns cached item by reference, s1 == s2*

```
>>> s2 = foo(ref1)
```

*# Different arg, so function foo executes*

```
>>> s3 = foo(ref2)
```

*# Clear all cached entries for this function*

```
>>> foo.clear()
```



```
# Clear all global resources from cache
```

```
>>> st.cache_resource.clear()
```

## Deprecated caching

```
>>> @st.cache
```

```
... def foo(bar):
```

```
...     # Do something expensive in here...
```

```
...     return data
```

```
>>> # Executes foo
```

```
>>> d1 = foo(ref1)
```

```
>>> # Does not execute foo
```

```
>>> # Returns cached item by reference, d1 == d2
```

```
>>> d2 = foo(ref1)
```

```
>>> # Different arg, so function foo executes
```

```
>>> d3 = foo(ref2)
```

## Display progress and status

```
>>> with st.spinner(text='In progress'):
```

```
>>>     time.sleep(5)
```

```
>>>     st.success('Done')
```

```
st.progress(progress_variable_1_to_100)
```

```
st.balloons()
```

```
st.snow()
```

```
st.error('Error message')
```

```
st.warning('Warning message')
```

```
st.info('Info message')
```

```
st.success('Success message')
```

```
st.exception(e)
```

## Personalize apps for users

```
# Show different content based on the user's email address.
```

```
>>> if st.user.email == 'jane@email.com':
```

```
>>>     display_jane_content()
```

```
>>> elif st.user.email == 'adam@foocorp.io':  
>>>     display_adam_content()  
>>> else:  
>>>     st.write("Please contact us to get access!")
```