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
      remove
      - st.cache_data
      - st.cache_resource
      - st.session_state

-
    -
        -
            - st.query_params
            - st.experimental_get_query_params*delete*
            - st.experimental_set_query_params*delete*
        - 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/
- Caching and state

Caching and state

Optimize performance and add statefulness to your app!

Caching

Streamlit provides powerful cache primitives for data and global resources. They allow your app to stay performant even when loading data from the web, manipulating large datasets, or performing expensive computations.

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" )
```

Manage state

Streamlit re-executes your script with each user interaction. Widgets have built-in statefulness between reruns, but Session State lets you do more!

Session State

Save data between reruns and across pages.

```
st.session_state["foo"] = "bar"
```

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()
```

Deprecated commands

delete

> This command was deprecated in version 1.18.0. Use `st.cache_data` instead.

Memo

Experimental function decorator to memoize function executions.

```
@st.experimental_memo def fetch_and_clean_data(url): # Fetch data from URL here, and then clean it up. return data
```

delete

> This command was deprecated in version 1.18.0. Use `st.cache_resource` instead.

Singleton

Experimental function decorator to store singleton objects.

```
@st.experimental_singleton def get_database_session(url): # Create a database session object that points to the URL. return session
```

delete

Get query parameters

Get query parameters that are shown in the browser's URL bar.

```
param_dict = st.experimental_get_query_params()
```

delete

Set query parameters

Set query parameters that are shown in the browser's URL bar.

```
st.experimental_set_query_params({"show_all"=True, "selected"=["asia", "america"]})
```

←Previous: Execution flowNext: st.cache_data→

forum

Still have questions?

Our forums are full of helpful information and Streamlit experts.

Cookie policy

forum Ask AI