Zoom Keyhole

Release 0.12.0b1

Fred Dulwich, Ben Mort, Ferdl Graser, Ugur Yilmaz

Nov 25, 2021

CONTENTS

A simple web application that displays Zoom rooms participants on Miro boards.

Created for PI8 planning in the IP sprint. Updated to a web application in the PI9 IP sprint.

CHAPTER

ONE

QUICK-START

1.1 Required environment variables

Zoom Keyhole requires that the following environment variables are set:

- `ZOOM_KEYHOLE_CONFIG_FILE`: Location of the YAML configuration file (see below). (default: `./config.yaml`)
- `ZOOM_KEYHOLE_MIRO_API_TOKEN`: Miro API token to use. A semi-colon separated list can be used to specify more than one token. If multiple tokens are provided, Miro API calls are shared between them.
- `ZOOM_KEYHOLE_ZOOM_API_KEY`: Zoom API key. Needed for polling the zoom API for room participants. Used when getting initial room list and periodically for correcting for errors zoom events.
- `ZOOM_KEYHOLE_ZOOM_SECRET`: Zoom API secret. Used in combination with the Zoom API key.
- `ZOOM_KEYHOLE_GSHEET_KEY`: API Key for google sheets. Used to load google sheets configuration.
- `ZOOM_KEYHOLE_GSHEET_ID`: Google sheet ID where the configuration is found.

1.2 Configuration file

It also requires a YAML configuration file which describes the location of the miro board being updated, as well as ....

An example file is:

```
---
# List of boards to update (NOTE: Replace with actual board ids!)
miro_board_ids:
    - o9J_x......=
    - o9J_x......=
header_particpant_list: Currently in the room
symbol:
    mode: random
    position: left
    default: "\\U00002795"
    show_default: false
    show_bullet_point: true
```

1.3 Google sheets configuration

Zoom Keyhole loads its zoom room, users, and user roles configuration from a Google sheets spreadsheet, specified by the ID stored in the environment variable `ZOOM_KEYHOLE_GSHEET_ID`.

This spreadsheet should have three pages, ... TODO!

Configuration from this spreadsheet is loaded on application start up, and then reloaded periodically to pull in changes. The update period can be set by the optional environment variables `GSHEET_USERS_SCAN_PERIOD`, `GSHEET_ROOMS_SCAN_PERIOD`, `GSHEET_ROLES_SCAN_PERIOD`. These are minimum periods in seconds between which the application will attempt to load settings from the respective sheet pages.

1.4 Setting up Miro

Zoom Keyhole finds and populates widgets on the list of Miro boards, provided in the configuration file whose location can be set by the environment variable `ZOOM_KEYHOLE_CONFIG_FILE` and whose format is described above.

In order to create a new widget to update ... TODO!

1.5 Installing dependencies

Skip this is running using Docker or continuous deployment!

```
pipenv install
```

1.6 Starting the application

The web application can be started using:

```
make run
```

It can also be run for local testing with:

```
make run-local
```

1.7 Optional settings

A number of other environment variables provide addition configuration:

- `GSHEET_USERS_SCAN_PERIOD`: Minimum period between loading user settings from Google sheets, in seconds (default: 20s).
- `GSHEET_ROOMS_SCAN_PERIOD`: Minimum period between zoom room settings from Google sheets, in seconds (default: 60s).
- `GSHEET_ROLES_SCAN_PERIOD`: Minimum period between loading role settings from Google sheets, in seconds (default: 60s).
- `MIRO_BOARD_SCAN_PERIOD`: Minimum period between scanning for changes to Zoom Keyhole Miro widgets, in seconds (default: 120s).

- `ZOOM_ROOM_SCAN_PERIOD`: Minimum period between scanning for changes to Zoom room attendance, in seconds (default: 120s). This is used to correct for errors in Zoom webhook events.
- `MIRO_BOARD_UPDATE_PERIOD`: Minimum period between Miro board widget updates, in seconds (default: 2s).

CHAPTER

TWO

DOCKER USAGE

****Note: these instructions are out of date!****

You can also run the development environment with the local docker image as well.

`make development` will mount your `app` directory with the location of `config.yaml` file (default: `CONFIG_FILE=config.yaml`) so that you can use the app exposed by docker at `PORT` (default: 8090) with auto-reload enabled. i.e. `make development CONFIG_FILE=/temp/config.yaml PORT=3000` will run the app at port 3000 with the `temp/config.yaml` file.

`make development` starts the latest built docker image (`make docker-build`) with app folder mounted into it and the server is set to `--reload` flag. This enables local development by reflecting any change in your `app/` folder to loaded into the api server as well.

CHAPTER

THREE

WITH KUBERNETES

****Note: these instructions need review and may be out of date!****

An example minikube installation with loadbalancer enabled could be found here - this is the suggested starting point for testing locally with Minikube.

You want to install charts using the docker image created with `make docker-build`. If you ran `eval $(minikube docker-env)` before building, the image will be pulled from your local cache.

Next, you want to deploy your charts. `make install-chart` deploys the helm chart into a k8s environment with the following ingress controllers:

- NGINX
- Traefik

By default, it uses `traefik` for local development and testing. You can override this by providing `INGRESS` variable like `make install-chart INGRESS=nginx`. In deployment correct ingress is automatically selected.

CHAPTER

FOUR

DEVELOPER NOTES

4.1 Configuration data structure

The application state and configuration is stored in a global Python dictionary called CONFIG, which has the following structure:

```
---
header_particpant_list: Currently in the room
miro_board_ids:
  # List of boards to update, loaded once from the config file.
  - o9J_x......=
  - o9J_x......=
symbol:
  # Symbol (emoji) display configuration, loaded once from the config file.
  mode: random
  position: left
  default: "\\U00002795"
  show_default: false
  show_bullet_point: true
  roles:
    # Role name/symbol dictionary, loaded periodically from a Google Sheet.
    SM: symbol
    PO: symbol
users:
  # User configuration, loaded periodically from a Google Sheet.
  # List of dictionaries.
  - name: Alice  # Full name for display.
    id: Ali  # Unique partial Zoom name; can be different to display name.
    roles: ["PO"]  # List of roles for this user.
    team: Team Name
    ART: DP
    emoji: symbol
  - name: Bob  # Full name for display.
    id: Bob  # Unique partial Zoom name; can be different to display name.
    roles: ["SM"]  # List of roles for this user.
    team: Team Name
    ART: OMC
    emoji: symbol
zoom_config:
  # Zoom room configuration, loaded periodically from a Google Sheet.
  # Dictionary of dictionaries.
  Team Room A:
    name: Team Room A  # Meeting room name.
    id: 12345678  # Zoom meeting ID.
```

(continues on next page)

(continued from previous page)

```
    url: https://...  # Zoom meeting URL.
    utc_offset: +1
    comment: This will be added to the widget
  Coffee Room 1:
    name: Coffee Room 1  # Meeting room name.
    id: 23456789  # Zoom meeting ID.
    url: https://...  # Zoom meeting URL.
    utc_offset: 0
    comment: This will be added to the widget
zoom_data:
  # Zoom room data and state, populated dynamically from Miro and Zoom.
  # Dictionary of dictionaries.
  Team Room A:
    participants:
      # List of tuples of meeting participant Zoom name and user ID.
      - ("Ali", "1234")
    keyholes:
      # List of keyhole widgets for this meeting room, found by scanning Miro.
      - board: Miro board ID
        widget: Miro widget ID
        text: Current keyhole widget text string
  Coffee Room 1:
    participants:
      # List of tuples of meeting participant Zoom name and user ID.
      - ("Bob", "5678")
    keyholes:
      # List of keyhole widgets for this meeting room, found by scanning Miro.
      - board: Miro board ID
        widget: Miro widget ID
        text: Current keyhole widget text string
zoom_participant_location:
  # Dictionary mapping Zoom user name to room name, populated dynamically.
  # Used for the directory widgets.
  Ali: Team Room A
  Bob: Coffee Room 1
directories:
  # List of directory widgets to update, found by scanning Miro.
  - board: Miro board ID
    widget: Miro widget ID
    roles: ["PO", "SM"]  # List of user roles to display in this directory.
    ART: ["DP"]  # List of ARTs to display in this directory.
    text: Current directory text string
mutex: (Object)              # Global mutex for thread synchronisation.
rooms_scanned: True | False  # Boolean flag set if initial scan done.
shutdown: True | False       # Boolean flag set if shutdown event triggered.
```

4.2 Building documentation

Sphinx documentation can be built locally using:

```
make docs
make docs-autobuild
```