atsuoishimoto / filechest

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

filechest / README.md

atsuoishimoto add Qt settings 3e49289 · 6 hours ago

223 lines (149 loc) · 6 KB

FileChest

A zero-configuration file browser for local directories and Amazon S3, built with Python/Django.

```
uvx filechest /path/to/directory
uvx filechest s3://bucket-name/prefix
```

No setup required. Just run the command and start browsing.

Features

- Browse local directories and S3 buckets
- Upload files (drag & drop supported)
- Download files
- Create, rename, delete files and folders
- Copy and move files between directories
- Preview images, videos, audio, PDF, and text files
- List/Grid view toggle

Installation

We recommend using uv for the best experience. See Installing uv for setup instructions.

```
# Using uvx (recommended, no installat
uvx filechest /path/to/directory
```

main filechest / README.md Top

Preview Code Blame Raw

```
pipx install filechest
```

Usage

Browse a local directory:

```
filechest /path/to/directory
```

Browse an S3 bucket:

```
filechest s3://bucket-name/prefix
```

List all accessible S3 buckets:

```
filechest s3://
```

The command starts a web server and opens your browser automatically.

Command Line Options

```
usage: filechest [-h] [-p PORT] [--
no-browser] [-g] [-a AWS_PROFILE]
                 [--max-buckets
MAX_BUCKETS] [--max-entries
MAX_ENTRIES]
                 path

Start a file manager for a directory
or S3 bucket

positional arguments:
  path                  Path to
directory, S3 URL
(s3://bucket/prefix), or "s3://" to
list all buckets

options:
  -h, --help            show this
help message and exit
  -p, --port PORT       Port to run
the server on (default: 8000)
  --no-browser          Do not open
browser automatically
  -g, --gui             Open GUI
window (Experimental)
  -a, --aws-profile AWS_PROFILE
                        AWS profile
name to use (sets AWS_PROFILE
environment variable)
  --max-buckets MAX_BUCKETS
                        Maximum
number of S3 buckets to load
(default: 100)
```

```
    --max-entries MAX_ENTRIES
                          Maximum
  number of files/directories to list
  (default: 1000)
```

S3 Configuration

FileChest uses your existing AWS credentials. Configure them using the AWS CLI:

```
# Standard credentials
aws configure

# SSO authentication
aws sso configure
```

Or set environment variables directly:

```
export AWS_ACCESS_KEY_ID=your-access-k
export AWS_SECRET_ACCESS_KEY=your-secr
export AWS_DEFAULT_REGION=us-east-1
```

Use the `-a` option to specify a named profile:

```
filechest -a my-profile s3://bucket-na
```

GUI Mode (Experimental)

FileChest can run as a standalone GUI application using pywebview.

```
filechest -g /path/to/directory
```

Installation for GUI

Install the appropriate variant for your platform:

```
# Windows / macOS
uv tool install filechest[gui]

# Linux (GTK)
uv tool install filechest[gtk]

# Linux (Qt)
uv tool install filechest[qt]
```

System Dependencies

pywebview may require additional system libraries depending on your platform. See the pywebview installation guide for details.

For example, on **WSL2 Ubuntu 24.04**, the following packages are required:

- for GTK:

```
sudo apt install pkg-config cmake libca
```

- for Qt:

```
sudo apt install -y qt6-base-dev qt6-wa
```

Using as a Django Application

FileChest can also be run as a standard Django web application, enabling multi-user file management with access control.

Setup

```
git clone https://github.com/atsuoishi
cd django-filechest
uv sync
uv run python manage.py migrate
uv run python manage.py createsuperuse
uv run python manage.py runserver
```

Open http://127.0.0.1:8000/admin/ to configure volumes and permissions.

Key Concepts

Volume: A storage location (local directory or S3 bucket) that users can browse.

Role: Access level granted to a user for a volume.

- **Editor** – Full access: browse, upload, download, create, rename, delete, copy, move
- **Viewer** – Read-only: browse and download only

Creating a Volume

In Django Admin, go to **Filechest** > **Volumes** and add a new volume:

Field	Description
`name`	URL-safe identifier used in the URL path
`verbose_name`	Human-readable name shown in UI
`path`	Local path (`/data/files`) or S3 URL (`s3://bucket/prefix`)
`public_read`	If checked, anyone can view without logging in
`max_file_size`	Maximum upload size in bytes (default: 10MB)
`is_active`	Uncheck to disable the volume

Access Control

Access is determined by the following rules, evaluated in order:

1. **Superusers** always have **Editor** access to all volumes.
2. **Users with VolumePermission** get the assigned role:
   - Go to **Filechest** > **Volume permissions** in Django Admin
   - Select a user and volume, then assign `editor` or `viewer` role
3. **Public volumes** (`public_read=True`): Anyone without explicit permission gets **Viewer** access, including anonymous users.
4. **Private volumes** (`public_read=False`): Users without permission cannot access the volume at all.

Examples

Team shared drive: Create a volume, add VolumePermission for each team member with `editor` role.

Public file hosting: Create a volume with `public_read=True` . Anyone can browse and download. Add specific users as `editor` to allow uploads.

Personal storage: Create a volume per user with VolumePermission (`editor`). Leave `public_read=False` so only that user can access it.

License

MIT License

Links

- GitHub: https://github.com/atsuoishimoto/filechest
- Issues: https://github.com/atsuoishimoto/filechest/issues