
Pageflow



Multimedia storytelling for the web. Built in cooperation with WDR.

For a high level introduction and example Pageflow stories see pageflow.io.

- [Getting Started](#)
- [Guides](#)
- [JavaScript API Reference of `pageflow` package](#)
- [List of Plugins](#)

Updating

For instructions on how to update from a prior version of the gem see the [Updating Pageflow wiki page](#).

Ingredients

Pageflow is a Rails engine which roughly consists of the following components:

- A full MVC stack to manage and display Pageflow stories
- User and permission management with support for isolated accounts and user collaboration
- A client side application for live preview editing of stories
- Background jobs to process and encode images, audios and videos
- Generators to quickly bootstrap a new Rails application

Pageflow assumes the following choice of libraries:

- Devise for authentication
- CanCanCan for authorization
- ActiveAdmin for administration
- Resque for as default for background jobs
- FriendlyId for pretty URLs
- Paperclip for attachment handling
- Backbone Marionette for the editor
- React/Redux for the frontend

Requirements

Pageflow runs in environments with:

- Ruby ≥ 3.2
- Node ≥ 18
- Rails 7.1
- Redis server (for Resque)
- A database server supported by Active Record (tested with MySQL)
- ImageMagick
- libvips
- Audio Waveform Image Generator

Accounts of the following cloud services have to be registered:

- Amazon Web Services for S3 file storage and (optionally) Cloudfront content delivery
- Zencoder for video/audio encoding

Installation

Generate a new Rails application using the MySQL database adapter:

```
1 $ rails new my_pageflow --database=mysql
2 $ cd my_pageflow
```

Do not name your application **"pageflow"** since it will cause conflicts with constant names created by Pageflow itself.

Database Setup

Enter valid MySQL credentials inside `config/database.yml` and create the database:

```
1 $ rake db:create
```

Gem Dependencies

Add these lines to your application's Gemfile, replacing `X.Y.Z` with the current Pageflow version number. It is recommended to depend on a specific minor version using the pessimistic version constraint operator. See Pageflow's versioning policy for details.

```
1 # Gemfile
2 gem 'pageflow', '~> X.Y.Z'
3
4 # The install generator sets up Resque as Active Job backend
5 gem 'resque', '~> 1.25'
6 gem 'resque-scheduler', '~> 4.10'
7 gem 'ar_after_transaction', '~> 0.8.0'
8 gem 'redis', '~> 3.0'
9 gem 'redis-namespace', '~> 1.5'
```

Run bundler to install dependencies:

```
1 $ bundle install
```

Running the Generator

Now you can run the generator to setup Pageflow and its dependencies:

```
1 $ rails generate pageflow:install
```

The generator will invoke Active Admin and Devise generators in turn and apply some configuration changes. When asked to overwrite the `db/seeds.rb` file, choose yes.

To better understand Pageflow's configuration choices, you can run the single steps of the `install` generator one by one. See the wiki page [The Install Generator in Detail](#) for more. If you'd rather not look behind the scenes for now, you can safely read on.

Database Migration

Now you can migrate the database.

```
1 $ rake db:migrate
```

Finally, you can populate the database with some example data, so things do not look too blank in development mode.

```
1 $ rake db:seed
```

Configuration

Pageflow stores files in S3 buckets also in development mode. Otherwise there's no way to have Zen-coder encode them. See [setting up external services](#).

The host application can utilize environment variables to configure the API keys for S3 and Zencoder. The variables can be found in the generated Pageflow initializer.

For available configuration options and examples see the inline docs in `config/initializers/pageflow.rb` in your generated rails app.

Ensure you have defined default url options in your environments files. Here is an example of `default_url_options` appropriate for a development environment in `config/environments/development.rb`:

```
1 config.action_mailer.default_url_options = {host: 'localhost:3000'}
```

In production, `:host` should be set to the actual host of your application.

Running Pageflow

In addition to the Rails server, you need to start two Rake tasks for the background job processing. These tasks are listed in `Procfile` which is generated in the project root folder by the Pageflow installer.

Consider using the foreman gem to start all of these processes (including the Rails server) with a single command in your development environment.

The built-in Resque web server is mounted at `/background_jobs`. Use it to inspect the state of background jobs, and restart failed jobs. This functionality is only available for admins.

Troubleshooting

If you run into problems during the installation of Pageflow, please refer to the Troubleshooting docs. If that doesn't help, consider filing an issue.

Security Policy

See `SECURITY.md`.

Contributing

Pull requests are welcome on GitHub at <https://github.com/codevise/pageflow>. Everyone interacting in the project's codebases, issue trackers and mailing lists is expected to follow the code of conduct.

See the Contributing section in the guides list for instructions on how to setup your development environment. The GitHub wiki contains high level guides on common development workflows.

License

The gem is available as open source under the terms of the MIT License.

Special Thanks

Built in cooperation with:



We would like to express our special thanks to the following services for supporting Pageflow through free open source plans:

