Atuin tools documentation

Release 2.0.0

Paolo Casciello, Luca Zarotti, Stefano Cilloni see contributors

Nov 29, 2018

Contents

Version: 2.0.0

CHAPTER 1

Features

- Python modules update
- SASS to CSS compilation
- CSS concatenation and minification
- JS concatenation, minification and obfuscation
- Images optimization without quality loss
- Translations management (translations defined through Babel)
- Pre-deploy preparation task (CSS, JS, images fine optimizations)

CHAPTER 2

Container purpose

In modern days we use lots of different tools to build assets, compile js, minification and such. This container simplifies it by creating an environment which runs gulp and all the tools needed to build the `app/static/min` folder of Atuin Web Framework.

CHAPTER 3

Run the tasks

You can run any of the tasks from outside the container

```
docker-compose run --rm tools gulp prepare-deploy
```

or by entering the container

```
docker-compose run --rm tools bash

root@7dd8dbadea38:/workspace# gulp prepare-deploy
```

The above commands do assume you're using this container through a docker-compose.yaml file defined by one of the Atuin Web Framework projects.

CHAPTER 4

Tasks

default | help
Shows the available tasks

update
Dependencies management.
It updates the dependencies specified in the requirements.txt file.

monitor [–type production]
Real time check for css and js.
It handles errors and rebuilds the minified and compiled files.
–type production compress css and obfuscate js.

prepare-deploy
Preare static files to being deployed: minification and uglification of files.

clean[:min|:css|:js|:img]
Cleans files.
From all project clean: *.pyc *.pyo *.~
:min Clean all minified fiels app/static/min
:css Clean minified CSS app/static/min/css
:js Clean minified JS app/static/min/js
:img Clean optimized img app/static/min/img

LOCALIZATION SUBSYSTEM

translations<:extract|:update|:compile|:init>
Manages translations. Each language must be initialized using **:init**.

:extract Extracts translations from source
:update Updates translations messages files for every language
:compile Compiles messages.po files for every language
:init –lang <code> Initialize a new language. Code must be language code like 'en', 'de', ...

4.1 gulp update

This task looks for the files

- `app/atuin/requirements.txt`
- `app/atuincms/requirements.txt`
- `app/requirements.txt`

and install/update the modules from all the requirement's files into `app/libs`.

4.2 gulp monitor

This task keeps on watching the `app/static/src` looking for changes into SASS, JS and images files. Each time a file is changed, it triggers the recompilation and minification process in order to keep up to date the corresponding files in the `app/static/min` directory.

This task is executed by default when the atuin-tools container is run.

4.3 gulp prepare-deploy

It prepares all the CSS, JS and images files to be deployed by minifying as much as possible the CSS and JS files (white spaces and break lines removed).

It should be run before each application deployment.

Clashes with the `gulp monitor` task may happen because they both edit the same files.

Be sure to stop the `gulp monitor` task before run the `gulp prepare-deploy`.

4.4 gulp clean

Remove files matching *.pyc, *.pyo, *.~ from all the project.

4.5 gulp clean[:min|:css|:js|:img]

Delete directories of minified files.

command	deleted directory
gulp clean:min	app/static/min
gulp clean:css	app/static/min/css
gulp clean:js	app/static/min/js
gulp clean:img	app/static/min/img

4.6 gulp translations:init -lang <code>

Initialize a new translation language.

4.7 gulp translations:extract

Extracts translations from the whole project according to the configurations found at `config/Babel.cfg`. This file should be created according to the Extraction Method Mapping and Configuration.

4.8 gulp translations:update

Update the extracted translations to the **.po* files for the initialized languages.

4.9 gulp translations:compile

Compile the translation messages into **.mo* corresponding files for the initialized languages.

CHAPTER 5

Suggested use of the container

This container is not meant to be used as is but part of the docker-compose environment started for development in Atuin Web Framework.

```
services:
  tools:
    image: atuinframework/atuin-tools:2.0.0
    volumes:
      - ./app:/workspace/app
      - ./config:/workspace/config
```

CHAPTER 6

Automatic container build

This container is automatically built on Docker Hub.

CHAPTER 7

Documentation

To work on the project documentation and checkout the compiled result run

```
docker-compose -f docker-compose.docs.yaml up
```

and visit http://localhost:9999/.

CHAPTER 8

Contributions

Do you want to take part to this project? Just send a pull request to the official repository.

CHAPTER 9

Links

- Documentation
- Repository
- DockerHub