gen_stm32 Documentation

Release

https://github.com/vroncevic/gen_stm32/releases

Vladimir Roncevic <elektron.ronca@gmail.com>

Apr 07, 2022

Contents

gen_stm32 is toolset for generation STM32 project skeleton for development of embedded applications.

Developed in python code.

The README is used to introduce the tool and provide instructions on how to install the tool, any machine dependencies it may have and any other information that should be provided before the tool is installed.

CHAPTER 1

gen_stm32

1.1 gen_stm32 package

1.1.1 Subpackages

1.1.1.1 gen_stm32.pro package

Subpackages

gen_stm32.pro.config package

Submodules

gen_stm32.pro.config.pro_name module

Module pro_name.py

Copyright Copyright (C) 2018 Vladimir Roncevic <elektron.ronca@gmail.com> gen_stm8 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen_stm8 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Info Defined class ProName with attribute(s) and method(s). Defined API for project name with preparations for generation.

class gen_stm32.pro.config.pro_name.**ProName**(*verbose=False*)
Bases: object

Defined class ProName with attribute(s) and method(s). Defined API for project name with preparations for generation. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
__verbose - enable/disable verbose option.
__pro_name - project name.

methods

__init__ - initial constructor.
pro_name - property methods for set/get operations.
is_pro_name_ok - checking is project name ok.
__str__ - dunder method for ProName.

GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::PRO_NAME'

is_pro_name_ok()
Checking is project name ok.

Returns boolean status, True (not None) | False.

Return type <bool>

Exceptions None

pro_name
Property method for getting project name.

Returns formatted project name | None.

Return type <str> | <NoneType>

Exceptions None

gen_stm32.pro.config.template_dir module

Module template_dir.py

Copyright Copyright (C) 2018 Vladimir Roncevic <elektron.ronca@gmail.com> gen_stm8 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen_stm8 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Info Defined class TemplateDir with attribute(s) and method(s). Defined API for template directory for generation.

class gen_stm32.pro.config.template_dir.**TemplateDir**(*verbose=False*)
Bases: object

Defined class TemplateDir with attribute(s) and method(s). Defined API for template directory for generation. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
__verbose - enable/disable verbose option.
__template_dir - project template dir.

methods

__init__ - initial constructor.
template_dir - property methods for set/get operations.
is_template_dir_ok - checking is template dir ok.
__str__ - dunder method for TemplateDir.

GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::TEMPLATE_DIR'

is_template_dir_ok()
Checking is project template dir ok.

Returns boolean status, True (not None) | False.

Return type <bool>

Exceptions None

template_dir
Property method for getting template dir.

Returns formatted template dir | None.

Return type <str> | <NoneType>

Exceptions None

Module contents

Module __init__.py

Info Defined class ProConfig with attribute(s) and method(s). Defined project configuration container.

class gen_stm32.pro.config.**ProConfig**(*verbose=False*)
Bases: object

Defined class ProConfig with attribute(s) and method(s). Defined project configuration container. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
TEMPLATES - templates key.
MODULES - modules key.
FORMAT - format for template file.
__verbose - enable/disable verbose option.
__config - configuration dictionary.

methods

__init__ - initial constructor.
pro_name - property methods for set/get operations.
is_config_ok - checking is project configuration ok.

__str__ - dunder method for ProConfig.

FORMAT = 'template'

GEN_VERBOSE = 'GEN_STM32::PRO::CONFIG::PRO_CONFIG'

MODULES = 'modules'

TEMPLATES = 'templates'

config
Property method for getting project configuration.

Returns formatted project configuration | None.

Return type <dict> | <NoneType>

Exceptions None

is_config_ok()
Checking is project configuration ok.

Returns boolean status, True (ok) | False.

Return type <bool>

Exceptions None

Submodules

gen_stm32.pro.read_template module

Module read_template.py

Copyright Copyright (C) 2018 Vladimir Roncevic <elektron.ronca@gmail.com> gen_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Info Define class ReadTemplate with attribute(s) and method(s). Created API for read a templates and return a string representations.

class gen_stm32.pro.read_template.**ReadTemplate**(*verbose=False*)
Bases: ats_utilities.config_io.base_check.FileChecking, *gen_stm32.pro.config.template_dir.TemplateDir*

Define class ReadTemplate with attribute(s) and method(s). Created API for read a templates and return a string representations. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
TEMPLATE_DIR - prefix path to templates.

methods

__init__ - initial constructor.
read - read a templates and return a content.
__str__ - dunder method for ReadTemplate.

GEN_VERBOSE = 'GEN_STM32::PRO::READ_TEMPLATE'

TEMPLATE_DIR = '/../conf/template/'

VERBOSE = 'ATS_UTILITIES'

read(*config*, *verbose=False*)
Read a templates and return a content.

Parameters

- **config** (*<dict>*) – parameter file name.
- **verbose** (*<bool>*) – enable/disable verbose option.

Returns template content list | empty list.

Return type <list>

Exceptions ATSTypeError | ATSBadCallError

gen_stm32.pro.write_template module

Module write_template.py

Copyright Copyright (C) 2018 Vladimir Roncevic <elektron.ronca@gmail.com> gen_stm32 is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version. gen_stm32 is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details. You should have received a copy of the GNU General Public License along with this program. If not, see <http://www.gnu.org/licenses/>.

Info Define class WriteTemplate with attribute(s) and method(s). Created API for writing template content with parameters.

class gen_stm32.pro.write_template.**WriteTemplate**(*verbose=False*)
Bases: ats_utilities.config_io.base_check.FileChecking

Define class WriteTemplate with attribute(s) and method(s). Created API for writing template content with parameters. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.

methods

__init__ - initial constructor.
write - write a templates with parameters.
__str__ - dunder method for WriteTemplate.

GEN_VERBOSE = 'GEN_STM32::PRO::WRITE_TEMPLATE'

VERBOSE = 'ATS_UTILITIES'

write(*templates*, *project_name*, *verbose=False*)
Write a templates with parameters.

Parameters

- **templates** (*<list>*) – parameter templates.
- **project_name** (*<str>*) – parameter project name.

- **verbose** (*<bool>*) – enable/disable verbose option.

Returns boolean status, True (success) | False.

Return type <bool>

Exceptions ATSTypeError | ATSBadCallError

Module contents

Module __init__.py

Info Defined class STM32Setup with attribute(s) and method(s). Generate project content with parameters from a project setup.

class gen_stm32.pro.**STM32Setup**(*verbose=False*)

Bases: ats_utilities.config_io.base_check.FileChecking, *gen_stm32.pro.config.ProConfig*, *gen_stm32.pro.config.pro_name.ProName*

Defined class STM32Setup with attribute(s) and method(s). Generate project content with parameters from a project setup. It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
PRO_STRUCTURE - project structure.
__reader - reader API.
__writer - writer API.

methods

__init__ - initial constructor.
get_reader - getter for reader object.
get_writer - getter for writer object.
gen_pro_setup - generate STM32 project structure.
__str__ - dunder method for STM32Setup.

PRO_STRUCTURE = '/../conf/project.yaml'

VERBOSE = 'ATS_UTILITIES'

gen_pro_setup(*project_name*, *verbose=False*)

Generate project structure.

Parameters

- **project_name** (*<str>*) – parameter tool name.
- **verbose** (*<bool>*) – enable/disable verbose option.

Returns boolean status, True (success) | False.

Return type <bool>

Exceptions None

get_reader()
Getter for reader object. :return: Read template object :rtype: <ReadTemplate> :exceptions: None

get_writer()
Getter for writer object. :return: Write template object :rtype: <WriteTemplate> :exceptions: None

1.1.2 Module contents

Module __init__.py

Info Defined class GenSTM32 with attribute(s) and method(s). Load a base info, create an CLI interface and run operation(s).

class gen_stm32.**GenSTM32**(*verbose=False*)
Bases: ats_utilities.cli.cfg_cli.CfgCLI

Defined class GenSTM32 with attribute(s) and method(s). Load a base info, create an CLI interface and run operation(s). It defines:

attributes

GEN_VERBOSE - console text indicator for process-phase.
CONFIG - configuration file path.
LOG - tool log file path.
LOGO - logo for splash screen.
OPS - list of tool options.
logger - logger object API.

methods

__init__ - initial constructor.
process - process and run tool option(s).
__str__ - dunder method for GenSTM32.

CONFIG = '/conf/gen_stm32.cfg'

GEN_VERBOSE = 'GEN_STM32'

LOG = '/log/gen_stm32.log'

LOGO = '/conf/gen_stm32.logo'

OPS = ['-g', '--gen', '-v', '--verbose', '--version']

VERBOSE = 'ATS_UTILITIES'

process(*verbose=False*)
Process and run operation.

Parameters **verbose** (*<bool>*) – enable/disable verbose option.

Returns boolean status, True (success) | False.

Return type <bool>

Exceptions None

CHAPTER 2

Installation

Navigate to release page download and extract release archive.

To install **gen_stm32** type the following

```
tar xvzf gen_stm32-x.y.z.tar.gz
cd gen_stm32-x.y.z/
# python2
pip install -r requirements.txt
python setup.py install_lib
python setup.py install_data
python setup.py install_egg_info
# pyton3
pip3 install -r requirements.txt
python3 setup.py install_lib
python3 setup.py install_data
python3 setup.py install_egg_info
```

You can use Docker to create image/container, or You can use pip to install

```
# pyton2
pip install gen-stm32
# pyton3
pip3 install gen-stm32
```

CHAPTER 3

Dependencies

gen_stm32 requires next modules and libraries

- ats-utilities - Python App/Tool/Script Utilities

CHAPTER 4

Generation flow of project setup

Base flow of generation process

CHAPTER 5

Tool structure

gen_stm32 is based on Template mechanism

Code structure:

```
gen_stm32/
├── conf/
│   ├── gen_stm32.logo
│   ├── gen_stm32.cfg
│   ├── gen_stm32_util.cfg
│   ├── project.yaml
│   └── template/
│       ├── build/
│       │   ├── includes/
│       │   │   └── STM32F4xx_StdPeriph_Driver/
│       │   │       └── src/
│       │   │           └── subdir.template
│       │   ├── Makefile.template
│       │   ├── objects.template
│       │   ├── source/
│       │   │   └── subdir.template
│       │   └── sources.template
│       ├── includes/
│       │   ├── CMSIS/
│       │   │   ├── arm_common_tables.template
│       │   │   ├── arm_math.template
│       │   │   ├── core_cm0.template
│       │   │   ├── core_cm3.template
│       │   │   ├── core_cm4_simd.template
│       │   │   ├── core_cm4.template
│       │   │   ├── core_cmFunc.template
│       │   │   └── core_cmInstr.template
│       │   ├── STM32F4xx/
│       │   │   ├── stm32f4xx_conf.template
│       │   │   ├── stm32f4xx.template
│       │   │   └── system_stm32f4xx.template
```

(continues on next page)

(continued from previous page)

```
└── STM32F4xx_StdPeriph_Driver/
    ├── inc/
    │   ├── misc.template
    │   ├── stm32f4xx_adc.template
    │   ├── stm32f4xx_can.template
    │   ├── stm32f4xx_crc.template
    │   ├── stm32f4xx_cryp.template
    │   ├── stm32f4xx_dac.template
    │   ├── stm32f4xx_dbgmcu.template
    │   ├── stm32f4xx_dcmi.template
    │   ├── stm32f4xx_dma.template
    │   ├── stm32f4xx_exti.template
    │   ├── stm32f4xx_flash.template
    │   ├── stm32f4xx_fsmc.template
    │   ├── stm32f4xx_gpio.template
    │   ├── stm32f4xx_hash.template
    │   ├── stm32f4xx_i2c.template
    │   ├── stm32f4xx_iwdg.template
    │   ├── stm32f4xx_pwr.template
    │   ├── stm32f4xx_rcc.template
    │   ├── stm32f4xx_rng.template
    │   ├── stm32f4xx_rtc.template
    │   ├── stm32f4xx_sdio.template
    │   ├── stm32f4xx_spi.template
    │   ├── stm32f4xx_syscfg.template
    │   ├── stm32f4xx_tim.template
    │   ├── stm32f4xx_usart.template
    │   └── stm32f4xx_wwdg.template
    └── src/
        ├── misc.template
        ├── stm32f4xx_adc.template
        ├── stm32f4xx_can.template
        ├── stm32f4xx_crc.template
        ├── stm32f4xx_cryp_aes.template
        ├── stm32f4xx_cryp_des.template
        ├── stm32f4xx_cryp_tdes.template
        ├── stm32f4xx_cryp.template
        ├── stm32f4xx_dac.template
        ├── stm32f4xx_dbgmcu.template
        ├── stm32f4xx_dcmi.template
        ├── stm32f4xx_dma.template
        ├── stm32f4xx_exti.template
        ├── stm32f4xx_flash.template
        ├── stm32f4xx_fsmc.template
        ├── stm32f4xx_gpio.template
        ├── stm32f4xx_hash_md5.template
        ├── stm32f4xx_hash_sha1.template
        ├── stm32f4xx_hash.template
        ├── stm32f4xx_i2c.template
        ├── stm32f4xx_iwdg.template
        ├── stm32f4xx_pwr.template
        ├── stm32f4xx_rcc.template
        ├── stm32f4xx_rng.template
        ├── stm32f4xx_rtc.template
        ├── stm32f4xx_sdio.template
        ├── stm32f4xx_spi.template
        ├── stm32f4xx_syscfg.template
```

(continues on next page)

(continued from previous page)

```
│           │               ├── stm32f4xx_tim.template
│           │               ├── stm32f4xx_usart.template
│           │               └── stm32f4xx_wwdg.template
│           ├── scripts/
│           │   └── arm_cortex_m4_512.template
│           └── source/
│               ├── main.template
│               ├── startup_stm32f4xx.template
│               ├── syscall.template
│               ├── system_stm32f4xx.template
│               └── tinynew.template
├── __init__.py
├── log/
│   └── gen_stm32.log
├── pro/
│   ├── config
│   │   ├── __init__.py
│   │   ├── pro_name.py
│   │   └── template_dir.py
│   ├── __init__.py
│   ├── read_template.py
│   └── write_template.py
└── run/
    └── gen_stm32_run.py
```

CHAPTER 6

Copyright and licence

Lets help and support PSF.

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

g

Index

V

W