

Code\_generator

Generated by Doxygen 1.8.15



---

<b>1 Namespace Index</b>	<b>1</b>
1.1 Packages	1
<b>2 Class Index</b>	<b>3</b>
2.1 Class List	3
<b>3 Namespace Documentation</b>	<b>5</b>
3.1 Package common	5
3.1.1 Detailed Description	5
3.2 Package configurator	5
3.2.1 Detailed Description	6
3.3 Package framework	6
3.3.1 Detailed Description	6
3.4 Package gui	6
3.4.1 Detailed Description	7
3.5 Package microcontroller	7
3.5.1 Detailed Description	7
3.6 Package projectConfiguration	7
3.6.1 Detailed Description	7
3.7 Package xmlCreator	8
3.7.1 Detailed Description	8
3.8 Package xmlParser	8
3.8.1 Detailed Description	8
<b>4 Class Documentation</b>	<b>9</b>
4.1 gui>AboutWindow Class Reference	9
4.1.1 Detailed Description	9
4.1.2 Constructor & Destructor Documentation	9
4.1.2.1 AboutWindow()	9
4.1.3 Member Function Documentation	10
4.1.3.1 main()	10
4.2 framework.CodeGenerator Class Reference	10
4.2.1 Detailed Description	10
4.2.2 Constructor & Destructor Documentation	10
4.2.2.1 CodeGenerator()	10
4.2.3 Member Function Documentation	11
4.2.3.1 Generate()	11
4.3 configurator.GPIO.CodeName Enum Reference	11
4.3.1 Detailed Description	11
4.3.2 Member Data Documentation	12

4.3.2.1 CODE_NAME . . . . .	12
4.3.2.2 STR_NAME . . . . .	12
4.4 framework.Common Class Reference . . . . .	12
4.4.1 Detailed Description . . . . .	13
4.4.2 Member Function Documentation . . . . .	13
4.4.2.1 getCfgFileCPath() . . . . .	13
4.4.2.2 getCfgFileHPath() . . . . .	13
4.4.2.3 getCfgPath() . . . . .	14
4.4.2.4 getCommonIncludes() . . . . .	14
4.4.2.5 getFrameworkCommonFilePath() . . . . .	15
4.4.2.6 getFrameworkIncludesFilePath() . . . . .	15
4.4.2.7 getInstallationFwkPath() . . . . .	15
4.4.2.8 getProjectFwkPath() . . . . .	16
4.4.2.9 setInstallationFwkPath() . . . . .	16
4.4.2.10 setProjectFwkPath() . . . . .	16
4.4.3 Member Data Documentation . . . . .	16
4.4.3.1 NL . . . . .	16
4.4.3.2 STR_DEFINITION . . . . .	17
4.4.3.3 STR_HEADER_EXT . . . . .	17
4.4.3.4 STR_INCLUDE . . . . .	17
4.4.3.5 STR_MODULE_GPIO . . . . .	17
4.5 configurator.ConfigurationFile Class Reference . . . . .	17
4.5.1 Detailed Description . . . . .	17
4.5.2 Member Data Documentation . . . . .	18
4.5.2.1 STR_PROJ_CONF_FILE . . . . .	18
4.6 xmlCreator.ConfXmlWriter Class Reference . . . . .	18
4.6.1 Detailed Description . . . . .	18
4.6.2 Constructor & Destructor Documentation . . . . .	18
4.6.2.1 ConfXmlWriter() . . . . .	18
4.6.3 Member Function Documentation . . . . .	19
4.6.3.1 addPin() . . . . .	19
4.6.3.2 writeXml() . . . . .	19
4.7 common.ErrorCode Enum Reference . . . . .	20
4.7.1 Detailed Description . . . . .	20
4.7.2 Member Data Documentation . . . . .	20
4.7.2.1 EX_ERROR . . . . .	20
4.7.2.2 FILE_CONF_ERROR . . . . .	20
4.7.2.3 FILE_READ_ERROR . . . . .	21
4.7.2.4 FILE_WRITE_ERROR . . . . .	21

---

4.7.2.5 NO_ERROR . . . . .	21
4.7.2.6 STR_INVALID . . . . .	21
4.8 common.Features Class Reference . . . . .	21
4.8.1 Detailed Description . . . . .	22
4.8.2 Member Function Documentation . . . . .	22
4.8.2.1 debugPrint() . . . . .	22
4.8.2.2 verbosePrint() . . . . .	22
4.8.3 Member Data Documentation . . . . .	22
4.8.3.1 DEBUG . . . . .	23
4.8.3.2 DEBUG_STR . . . . .	23
4.8.3.3 SW_VERSION . . . . .	23
4.8.3.4 VERBOSE . . . . .	23
4.8.3.5 VERBOSE_STR . . . . .	23
4.8.3.6 VERSION_NAME . . . . .	23
4.8.3.7 VERSION_STATUS . . . . .	23
4.9 gui.GpioConfWindow Class Reference . . . . .	24
4.9.1 Detailed Description . . . . .	24
4.9.2 Constructor & Destructor Documentation . . . . .	24
4.9.2.1 GpioConfWindow() . . . . .	24
4.9.3 Member Function Documentation . . . . .	24
4.9.3.1 main() . . . . .	25
4.10 gui.MainGui Class Reference . . . . .	25
4.10.1 Detailed Description . . . . .	25
4.10.2 Member Function Documentation . . . . .	26
4.10.2.1 generateCode() . . . . .	26
4.10.2.2 loadProjectFile() . . . . .	26
4.10.2.3 main() . . . . .	26
4.10.2.4 saveUc() . . . . .	26
4.10.2.5 setNewUC() . . . . .	27
4.10.2.6 showAboutWindow() . . . . .	27
4.10.2.7 showErrorDialog() . . . . .	27
4.10.2.8 showGpioConfWindow() . . . . .	27
4.10.3 Member Data Documentation . . . . .	27
4.10.3.1 ProjectFile . . . . .	28
4.10.3.2 ProjectPath . . . . .	28
4.11 gui.MainWindow Class Reference . . . . .	28
4.11.1 Detailed Description . . . . .	28
4.11.2 Constructor & Destructor Documentation . . . . .	29
4.11.2.1 MainWindow() . . . . .	29

4.11.3 Member Function Documentation	29
4.11.3.1 main()	29
4.11.3.2 OpenFileDialogChooser()	29
4.11.3.3 setProjectInformation()	30
4.11.3.4 setVisible()	30
4.11.4 Member Data Documentation	30
4.11.4.1 FrmCodeGenerator	31
4.12 gui.Messages Class Reference	31
4.12.1 Detailed Description	31
4.12.2 Member Function Documentation	31
4.12.2.1 getString()	31
4.13 microcontroller.Microcontroller Class Reference	32
4.13.1 Detailed Description	32
4.13.2 Constructor & Destructor Documentation	32
4.13.2.1 Microcontroller()	32
4.13.3 Member Function Documentation	33
4.13.3.1 getConfiguredPin()	33
4.13.3.2 getPin()	33
4.13.3.3 getUc_gpioNum()	34
4.13.3.4 getUc_manufacturer()	34
4.13.3.5 getUc_model()	34
4.13.3.6 getUc_pinNum()	34
4.13.3.7 getUc_portNum()	35
4.13.3.8 isValid()	35
4.13.3.9 loadPinsConf()	35
4.13.3.10 processDocument()	35
4.13.4 Member Data Documentation	36
4.13.4.1 GpioCfgPin	36
4.13.4.2 Includes_Common	36
4.13.4.3 Includes_Gpio	36
4.13.4.4 MAX_NUMBER_OF_PINS_PER_PORT	36
4.13.4.5 Ports	36
4.14 configurator.GPIO.Mode Enum Reference	37
4.14.1 Detailed Description	37
4.14.2 Member Function Documentation	37
4.14.2.1 getConfFromString()	37
4.14.3 Member Data Documentation	38
4.14.3.1 MODE_ALTERNATE_FUNCTION	38
4.14.3.2 MODE_ANALOG	38

4.14.3.3 MODE_INPUT . . . . .	38
4.14.3.4 MODE_MAX_VALUE . . . . .	38
4.14.3.5 MODE_OUTPUT . . . . .	38
4.14.3.6 STR_NAME . . . . .	39
4.15 configurator.GPIO.OutLevel Enum Reference . . . . .	39
4.15.1 Detailed Description . . . . .	39
4.15.2 Member Function Documentation . . . . .	39
4.15.2.1 getConfFromString() . . . . .	39
4.15.3 Member Data Documentation . . . . .	40
4.15.3.1 HIGH . . . . .	40
4.15.3.2 LOW . . . . .	40
4.15.3.3 MAX_VALUE . . . . .	40
4.15.3.4 STR_NAME . . . . .	40
4.16 configurator.GPIO.OutType Enum Reference . . . . .	41
4.16.1 Detailed Description . . . . .	41
4.16.2 Member Function Documentation . . . . .	41
4.16.2.1 getConfFromString() . . . . .	41
4.16.3 Member Data Documentation . . . . .	42
4.16.3.1 OTYPE_MAX_VALUE . . . . .	42
4.16.3.2 OTYPE_NOT_AVAILABLE . . . . .	42
4.16.3.3 OTYPE_OPEN_DRAIN . . . . .	42
4.16.3.4 OTYPE_PUSH_PULL . . . . .	42
4.16.3.5 STR_NAME . . . . .	42
4.17 microcontroller.Pin Class Reference . . . . .	43
4.17.1 Detailed Description . . . . .	44
4.17.2 Constructor & Destructor Documentation . . . . .	45
4.17.2.1 Pin() . . . . .	45
4.17.3 Member Function Documentation . . . . .	45
4.17.3.1 getAdc() . . . . .	45
4.17.3.2 getClock() . . . . .	45
4.17.3.3 getFeat_adc() . . . . .	46
4.17.3.4 getFeat_clock() . . . . .	46
4.17.3.5 getFeat_i2c() . . . . .	46
4.17.3.6 getFeat_int() . . . . .	46
4.17.3.7 getFeat_reset() . . . . .	47
4.17.3.8 getFeat_spi() . . . . .	47
4.17.3.9 getFeat_timer() . . . . .	47
4.17.3.10 getFeat_uart() . . . . .	47
4.17.3.11 getFunc_gnd() . . . . .	48

---

4.17.3.12 getFunc_gpio()	48
4.17.3.13 getFunc_misc()	48
4.17.3.14 getFunc_reset()	48
4.17.3.15 getFunc_vcc()	49
4.17.3.16 getI2c()	49
4.17.3.17 getInt()	49
4.17.3.18 getName()	49
4.17.3.19 getNumber()	50
4.17.3.20 getPort()	50
4.17.3.21 getReset()	50
4.17.3.22 getSpi()	50
4.17.3.23 getTimer()	51
4.17.3.24 getUart()	51
4.17.3.25 isValid()	51
4.17.3.26 setAdc()	51
4.17.3.27 setClock()	52
4.17.3.28 setFeat_adc()	52
4.17.3.29 setFeat_clock()	52
4.17.3.30 setFeat_i2c()	53
4.17.3.31 setFeat_int()	53
4.17.3.32 setFeat_reset()	53
4.17.3.33 setFeat_spi()	53
4.17.3.34 setFeat_timer()	54
4.17.3.35 setFeat_uart()	54
4.17.3.36 setFunc_gnd()	54
4.17.3.37 setFunc_gpio()	55
4.17.3.38 setFunc_misc()	55
4.17.3.39 setFunc_reset()	55
4.17.3.40 setFunc_vcc()	56
4.17.3.41 setI2c()	56
4.17.3.42 setInt()	56
4.17.3.43 setName()	56
4.17.3.44 setNumber()	57
4.17.3.45 setPort()	57
4.17.3.46 setReset()	57
4.17.3.47 setSpi()	58
4.17.3.48 setTimer()	58
4.17.3.49 setUart()	58
4.17.4 Member Data Documentation	59

---



4.17.4.1 DEF_FEATURE	59
4.17.4.2 DEF_FEATURE_AV	59
4.17.4.3 DEF_FUNCTION	59
4.17.4.4 DEF_NAME	59
4.17.4.5 DEF_NUMBER	59
4.17.4.6 DEF_PORT	59
4.17.4.7 DISABLE	60
4.17.4.8 ENABLE	60
4.18 configurator.PinConf Class Reference	60
4.18.1 Detailed Description	61
4.18.2 Constructor & Destructor Documentation	61
4.18.2.1 PinConf()	61
4.18.3 Member Function Documentation	61
4.18.3.1 getCodeName()	61
4.18.3.2 getMode()	62
4.18.3.3 getOutLevel()	62
4.18.3.4 getOutType()	62
4.18.3.5 getPinName()	62
4.18.3.6 getPort()	63
4.18.3.7 getPull()	63
4.18.3.8 getSpeed()	63
4.18.3.9 isAv_Adc()	63
4.18.3.10 isAv_altFunc()	64
4.18.3.11 isValid()	64
4.18.3.12 setCodeName()	64
4.18.3.13 setMode()	64
4.18.3.14 setOutLevel()	65
4.18.3.15 setOutType()	65
4.18.3.16 setPull()	65
4.18.3.17 setSpeed()	66
4.18.4 Member Data Documentation	66
4.18.4.1 DF_CODE_NAME	66
4.18.4.2 DF_MODE	66
4.18.4.3 DF_OUT_LEVEL	66
4.18.4.4 DF_OUTTYPE	66
4.18.4.5 DF_PULL	67
4.18.4.6 DF_SPEED	67
4.19 projectConfiguration.ProjectSettings Class Reference	67
4.19.1 Detailed Description	67

4.19.2 Constructor & Destructor Documentation	67
4.19.2.1 ProjectSettings()	68
4.19.3 Member Function Documentation	68
4.19.3.1 getConfFile()	68
4.19.3.2 getFrameworkPath()	68
4.19.3.3 getProjectName()	68
4.19.3.4 getUcFile()	69
4.19.3.5 openProjectFile()	69
4.19.3.6 processDocument()	69
4.20 configurator.GPIO.Pull Enum Reference	70
4.20.1 Detailed Description	70
4.20.2 Member Function Documentation	70
4.20.2.1 getConfFromString()	70
4.20.3 Member Data Documentation	71
4.20.3.1 PULL_DOWN	71
4.20.3.2 PULL_MAX_VALUE	71
4.20.3.3 PULL_NOT_AVAILABLE	71
4.20.3.4 PULL_UP	71
4.20.3.5 STR_NAME	71
4.21 configurator.GPIO.Speed Enum Reference	72
4.21.1 Detailed Description	72
4.21.2 Member Function Documentation	72
4.21.2.1 getConfFromString()	72
4.21.3 Member Data Documentation	73
4.21.3.1 SPEED_FAST	73
4.21.3.2 SPEED_HIGH	73
4.21.3.3 SPEED_MAX_VALUE	73
4.21.3.4 SPEED_MEDIUM	73
4.21.3.5 SPEED_NOT_AVAILABLE	73
4.21.3.6 STR_NAME	74
4.22 xmlParser.TestMain Class Reference	74
4.22.1 Detailed Description	74
4.22.2 Member Function Documentation	74
4.22.2.1 main()	74
4.23 xmlParser.XmlOpener Class Reference	75
4.23.1 Detailed Description	75
4.23.2 Constructor & Destructor Documentation	75
4.23.2.1 XmlOpener()	75
4.23.3 Member Function Documentation	75

4.23.3.1 getElementInfo() . . . . .	75
4.23.3.2 getElementInfoFromDoc() . . . . .	76
4.23.3.3 getParsedDoc() . . . . .	76
4.23.3.4 OpenFile() . . . . .	76

<b>Index</b>	<b>79</b>
--------------	-----------



# Chapter 1

## Namespace Index

### 1.1 Packages

Here are the packages with brief descriptions (if available):

<a href="#">common</a>	5
<a href="#">configurator</a>	5
<a href="#">framework</a>	6
<a href="#">gui</a>	6
<a href="#">microcontroller</a>	7
<a href="#">projectConfiguration</a>	7
<a href="#">xmlCreator</a>	8
<a href="#">xmlParser</a>	8



## Chapter 2

# Class Index

### 2.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">gui&gt;AboutWindow</a>	9
<a href="#">framework.CodeGenerator</a>	10
<a href="#">configurator.GPIO.CodeName</a>	11
<a href="#">framework.Common</a>	12
<a href="#">configurator.ConfigurationFile</a>	17
<a href="#">xmlCreator.ConfXmlWriter</a>	18
<a href="#">common.ErrorCode</a>	20
<a href="#">common.Features</a>	21
<a href="#">gui.GpioConfWindow</a>	24
<a href="#">gui.MainGui</a>	25
<a href="#">gui.MainWindow</a>	28
<a href="#">gui.Messages</a>	31
<a href="#">microcontroller.Microcontroller</a>	32
<a href="#">configurator.GPIO.Mode</a>	37
<a href="#">configurator.GPIO.OutLevel</a>	39
<a href="#">configurator.GPIO.OutType</a>	41
<a href="#">microcontroller.Pin</a>	43
<a href="#">configurator.PinConf</a>	60
<a href="#">projectConfiguration.ProjectSettings</a>	67
<a href="#">configurator.GPIO.Pull</a>	70
<a href="#">configurator.GPIO.Speed</a>	72
<a href="#">xmlParser.TestMain</a>	74
<a href="#">xmlParser.XmlOpener</a>	75





## Chapter 3

# Namespace Documentation

### 3.1 Package common

#### Classes

- enum [ErrorCode](#)
- class [Features](#)

#### 3.1.1 Detailed Description

Common information that needs to be accessed across all the project

#### Author

Miguel Diaz

#### Version

0.1

### 3.2 Package configurator

#### Classes

- class [ConfigurationFile](#)
- class [PinConf](#)

### 3.2.1 Detailed Description

Configuration classes

Author

Miguel Diaz

Version

0.1

## 3.3 Package framework

### Classes

- class [CodeGenerator](#)
- class [Common](#)
- class **GpioGenerator**

### 3.3.1 Detailed Description

Framework information

Author

H112943

Version

0.1

## 3.4 Package gui

### Classes

- class [AboutWindow](#)
- class [GpioConfWindow](#)
- class [MainGui](#)
- class [MainWindow](#)
- class [Messages](#)

### 3.4.1 Detailed Description

Author

Miguel Diaz

Version

0.1

## 3.5 Package microcontroller

### Classes

- class [Microcontroller](#)
- class [Pin](#)

### 3.5.1 Detailed Description

[Microcontroller](#) related classes

Author

Miguel Diaz

Version

0.1

## 3.6 Package projectConfiguration

### Classes

- class [ProjectSettings](#)

### 3.6.1 Detailed Description

Project settings and configuration files

Author

Miguel Diaz

Version

0.1

## 3.7 Package xmlCreator

### Classes

- class [ConfXmlWriter](#)

### 3.7.1 Detailed Description

Create configuration XML

#### Author

Miguel Diaz

#### Version

0.1

## 3.8 Package xmlParser

### Classes

- class [TestMain](#)
- class [XmlOpener](#)

### 3.8.1 Detailed Description

XML parser for microcontroller information and project settings

#### Author

Miguel Diaz

#### Version

0.1

## Chapter 4

# Class Documentation

### 4.1 gui.AboutWindow Class Reference

#### Public Member Functions

- [AboutWindow](#) ()

#### Static Public Member Functions

- static void [main](#) (String[ ] args)

#### 4.1.1 Detailed Description

About Window, contains version and contact information

#### Author

ovd

#### 4.1.2 Constructor & Destructor Documentation

##### 4.1.2.1 AboutWindow()

```
gui.AboutWindow.AboutWindow ( )
```

Create the application.

### 4.1.3 Member Function Documentation

#### 4.1.3.1 main()

```
static void gui.AboutWindow.main (
    String [] args ) [static]
```

About window main

##### Parameters

<i>args</i>	Init parameters
-------------	-----------------

The documentation for this class was generated from the following file:

- `src/gui/AboutWindow.java`

## 4.2 framework.CodeGenerator Class Reference

### Public Member Functions

- [CodeGenerator](#) ([Microcontroller](#) uC, [ProjectSettings](#) projectSettings)
- [ErrorCode Generate](#) ()

#### 4.2.1 Detailed Description

Author

ovd

#### 4.2.2 Constructor & Destructor Documentation

##### 4.2.2.1 CodeGenerator()

```
framework.CodeGenerator.CodeGenerator (
    Microcontroller uC,
    ProjectSettings projectSettings )
```

Constructor

## Parameters

<i>uC</i>	Project's microcontroller
<i>projectSettings</i>	Project's settings

### 4.2.3 Member Function Documentation

#### 4.2.3.1 Generate()

```
ErrorCode framework.CodeGenerator.Generate ( )
```

Generate project's configuration files

## Returns

Error code

The documentation for this class was generated from the following file:

- src/framework/CodeGenerator.java

## 4.3 configurator.GPIO.CodeName Enum Reference

## Public Attributes

- [CODE\\_NAME](#)

## Static Public Attributes

- static final String [STR\\_NAME](#) = "codeName"

#### 4.3.1 Detailed Description

## Author

Miguel Diaz

## Version

0.1

### 4.3.2 Member Data Documentation

#### 4.3.2.1 CODE\_NAME

```
configurator.GPIO.CodeName.CODE_NAME
```

Code name for pin

#### 4.3.2.2 STR\_NAME

```
static final String configurator.GPIO.CodeName.STR_NAME = "codeName" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- src/configurator/GPIO/CodeName.java

## 4.4 framework.Common Class Reference

### Static Public Member Functions

- static String [getInstallationFwkPath](#) ()
- static void [setInstallationFwkPath](#) (String installationFwkPath)
- static String [getProjectFwkPath](#) ()
- static void [setProjectFwkPath](#) (String projectFwkPath)
- static String [getCfgPath](#) (String fwkPath, String cfgModule)
- static String [getCfgFileCPath](#) (String fwkPath, String cfgModule)
- static String [getCfgFileHPath](#) (String fwkPath, String cfgModule)
- static String [getFrameworkCommonFilePath](#) (String fwkPath)
- static String [getFrameworkIncludesFilePath](#) (String fwkPath)
- static String [getCommonIncludes](#) ([Microcontroller](#) uC)

### Static Public Attributes

- static final String [STR\\_MODULE\\_GPIO](#) = "gpio"
- static final String [NL](#) = "\r\n"
- static final String [STR\\_DEFINITION](#) = "#define "
- static final String [STR\\_INCLUDE](#) = "#include "
- static final String [STR\\_HEADER\\_EXT](#) = ".h"



### 4.4.1 Detailed Description

Framework common fields and methods

#### Author

Miguel Diaz

#### Version

0.1

### 4.4.2 Member Function Documentation

#### 4.4.2.1 getCfgFileCPath()

```
static String framework.Common.getCfgFileCPath (  
    String fwkPath,  
    String cfgModule ) [static]
```

Get GPIO configuration file path

#### Parameters

<i>fwkPath</i>	Framework folder path
<i>cfgModule</i>	Configuration module name

#### Returns

GPIO configuration file path

#### 4.4.2.2 getCfgFileHPath()

```
static String framework.Common.getCfgFileHPath (  
    String fwkPath,  
    String cfgModule ) [static]
```

Get GPIO configuration header file path

**Parameters**

<i>fwkPath</i>	Framework folder path
<i>cfgModule</i>	Configuration module name

**Returns**

GPIO configuration header file path

**4.4.2.3 getCfgPath()**

```
static String framework.Common.getCfgPath (
    String fwkPath,
    String cfgModule ) [static]
```

Get configuration module files folder path

**Parameters**

<i>fwkPath</i>	Framework folder path
<i>cfgModule</i>	Configuration module name

**Returns**

Configuration files folder path

**4.4.2.4 getCommonIncludes()**

```
static String framework.Common.getCommonIncludes (
    Microcontroller uC ) [static]
```

**Parameters**

<i>uC</i>	Microcontroller used
-----------	----------------------

**Returns**

[Common](#) headers needed for framework

#### 4.4.2.5 getFrameworkCommonFilePath()

```
static String framework.Common.getFrameworkCommonFilePath (  
    String fwkPath ) [static]
```

Get the framework common header path

##### Parameters

<i>fwkPath</i>	Framework folder path
----------------	-----------------------

##### Returns

Framework common header path

#### 4.4.2.6 getFrameworkIncludesFilePath()

```
static String framework.Common.getFrameworkIncludesFilePath (  
    String fwkPath ) [static]
```

Get the framework includes header path

##### Parameters

<i>fwkPath</i>	Framework folder path
----------------	-----------------------

##### Returns

Framework includes header path

#### 4.4.2.7 getInstallationFwkPath()

```
static String framework.Common.getInstallationFwkPath ( ) [static]
```

Get installation framework path

##### Returns

installation framework path

#### 4.4.2.8 `getProjectFwkPath()`

```
static String framework.Common.getProjectFwkPath ( ) [static]
```

Get project's framework path

##### Returns

project's framework path

#### 4.4.2.9 `setInstallationFwkPath()`

```
static void framework.Common.setInstallationFwkPath (
    String installationFwkPath ) [static]
```

Set installation framework path

##### Parameters

<i>installationFwkPath</i>	installation framework path
----------------------------	-----------------------------

#### 4.4.2.10 `setProjectFwkPath()`

```
static void framework.Common.setProjectFwkPath (
    String projectFwkPath ) [static]
```

Set project's framework path

##### Parameters

<i>projectFwkPath</i>	project's framework path
-----------------------	--------------------------

### 4.4.3 Member Data Documentation

#### 4.4.3.1 `NL`

```
final String framework.Common.NL = "\r\n" [static]
```

[Common](#) implementation of New Line

#### 4.4.3.2 STR\_DEFINITION

```
final String framework.Common.STR_DEFINITION = "#define " [static]
```

Macro definition String

#### 4.4.3.3 STR\_HEADER\_EXT

```
final String framework.Common.STR_HEADER_EXT = ".h" [static]
```

Header file extension

#### 4.4.3.4 STR\_INCLUDE

```
final String framework.Common.STR_INCLUDE = "#include " [static]
```

Include header file string

#### 4.4.3.5 STR\_MODULE\_GPIO

```
final String framework.Common.STR_MODULE_GPIO = "gpio" [static]
```

GPIO module name

The documentation for this class was generated from the following file:

- src/framework/Common.java

## 4.5 configurator.ConfigurationFile Class Reference

### Static Public Attributes

- static final String [STR\\_PROJ\\_CONF\\_FILE](#) = "cgs"

### 4.5.1 Detailed Description

Configuration files properties

Author

Miguel Diaz

Version

0.1

## 4.5.2 Member Data Documentation

### 4.5.2.1 STR\_PROJ\_CONF\_FILE

```
final String configurator.ConfigurationFile.STR_PROJ_CONF_FILE = "cgs" [static]
```

Public configuration file extension

The documentation for this class was generated from the following file:

- src/configurator/ConfigurationFile.java

## 4.6 xmlCreator.ConfXmlWriter Class Reference

### Public Member Functions

- [ConfXmlWriter](#) ([Microcontroller](#) uC)
- void [addPin](#) ([PinConf](#) pin, int pinNum)
- [ErrorCode](#) [writeXml](#) (String fileName)

### 4.6.1 Detailed Description

Write a XML file

Author

Miguel Diaz

Version

0.1

### 4.6.2 Constructor & Destructor Documentation

#### 4.6.2.1 ConfXmlWriter()

```
xmlCreator.ConfXmlWriter.ConfXmlWriter (  
    Microcontroller uC )
```

Constructor

**Parameters**

<i>uC</i>	Microcontroller configuration
-----------	-------------------------------

**4.6.3 Member Function Documentation****4.6.3.1 addPin()**

```
void xmlCreator.ConfXmlWriter.addPin (  
    PinConf pin,  
    int pinNum )
```

Add a pin configuration to the file

**Parameters**

<i>pin</i>	Pin configuration
<i>pinNum</i>	Number of GPIO pin

**4.6.3.2 writeXml()**

```
ErrorCode xmlCreator.ConfXmlWriter.writeXml (  
    String fileName )
```

Write the XML file

**Parameters**

<i>fileName</i>	Name of XML configuration file
-----------------	--------------------------------

**Returns**

Error status

The documentation for this class was generated from the following file:

- src/xmlCreator/ConfXmlWriter.java

## 4.7 common.ErrorCode Enum Reference

### Public Attributes

- [NO\\_ERROR](#)
- [EX\\_ERROR](#)
- [FILE\\_READ\\_ERROR](#)
- [FILE\\_WRITE\\_ERROR](#)
- [FILE\\_CONF\\_ERROR](#)

### Static Public Attributes

- static final String [STR\\_INVALID](#) = "STR\_INVALID"

### 4.7.1 Detailed Description

Error codes enum

#### Author

Miguel Diaz

#### Version

0.1

### 4.7.2 Member Data Documentation

#### 4.7.2.1 EX\_ERROR

`common.ErrorCode.EX_ERROR`

Error during execution

#### 4.7.2.2 FILE\_CONF\_ERROR

`common.ErrorCode.FILE_CONF_ERROR`

File configuration error



#### 4.7.2.3 FILE\_READ\_ERROR

```
common.ErrorCode.FILE_READ_ERROR
```

File reading error

#### 4.7.2.4 FILE\_WRITE\_ERROR

```
common.ErrorCode.FILE_WRITE_ERROR
```

File writing error

#### 4.7.2.5 NO\_ERROR

```
common.ErrorCode.NO_ERROR
```

No error message

#### 4.7.2.6 STR\_INVALID

```
static final String common.ErrorCode.STR_INVALID = "STR_INVALID" [static]
```

Error string

The documentation for this enum was generated from the following file:

- `src/common/ErrorCode.java`

## 4.8 common.Features Class Reference

### Static Public Member Functions

- static void [verbosePrint](#) (String verboseMessage)
- static void [debugPrint](#) (String debugMessage)

### Static Public Attributes

- static final boolean [DEBUG](#) = true
- static final boolean [VERBOSE](#) = true
- static final String [VERBOSE\\_STR](#) = "# "
- static final String [DEBUG\\_STR](#) = "\$ "
- static final String [SW\\_VERSION](#) = VERSION\_MAJOR + "." + VERSION\_MINOR + "." + VERSION\_PATCH
- static final String [VERSION\\_STATUS](#) = "Alpha"
- static final String [VERSION\\_NAME](#) = "Bespin"

### 4.8.1 Detailed Description

Class that includes all project features

#### Author

Miguel Diaz

#### Version

0.1

### 4.8.2 Member Function Documentation

#### 4.8.2.1 `debugPrint()`

```
static void common.Features.debugPrint (  
    String debugMessage ) [static]
```

Print Debug message to console

#### Parameters

<i>debugMessage</i>	Message to display
---------------------	--------------------

#### 4.8.2.2 `verbosePrint()`

```
static void common.Features.verbosePrint (  
    String verboseMessage ) [static]
```

Print Verbose message to console

#### Parameters

<i>verboseMessage</i>	Message to display
-----------------------	--------------------

### 4.8.3 Member Data Documentation

#### 4.8.3.1 DEBUG

```
final boolean common.Features.DEBUG = true [static]
```

Enables debug functions

#### 4.8.3.2 DEBUG\_STR

```
final String common.Features.DEBUG_STR = "$ " [static]
```

Debug messages indicator on system console

#### 4.8.3.3 SW\_VERSION

```
final String common.Features.SW_VERSION = VERSION_MAJOR + "." + VERSION_MINOR + "." + VERSION_PATCH [static]
```

Complete Software version

#### 4.8.3.4 VERBOSE

```
final boolean common.Features.VERBOSE = true [static]
```

Enables console messages

#### 4.8.3.5 VERBOSE\_STR

```
final String common.Features.VERBOSE_STR = "$ " [static]
```

Verbose messages indicator on system console

#### 4.8.3.6 VERSION\_NAME

```
final String common.Features.VERSION_NAME = "Bespin" [static]
```

Code name of the software version

#### 4.8.3.7 VERSION\_STATUS

```
final String common.Features.VERSION_STATUS = "Alpha" [static]
```

Status of the software version

The documentation for this class was generated from the following file:

- src/common/Features.java

## 4.9 gui.GpioConfWindow Class Reference

### Public Member Functions

- [GpioConfWindow](#) ([Microcontroller](#) uCtrl)

### Static Public Member Functions

- static void [main](#) (String[] args)

### 4.9.1 Detailed Description

Window for configuring GPIO pins

#### Author

Miguel Diaz

#### Version

0.1

### 4.9.2 Constructor & Destructor Documentation

#### 4.9.2.1 GpioConfWindow()

```
gui.GpioConfWindow.GpioConfWindow (  
    Microcontroller uCtrl )
```

Create the GPIO configuration window and show it

#### Parameters

<i>uCtrl</i>	Microcontroller object containing all pin's information
--------------	---------------------------------------------------------

### 4.9.3 Member Function Documentation

## 4.9.3.1 main()

```
static void gui.GpioConfWindow.main (
    String [] args ) [static]
```

Gpio configuration window main

## Parameters

<i>args</i>	Init parameters
-------------	-----------------

The documentation for this class was generated from the following file:

- src/gui/GpioConfWindow.java

## 4.10 gui.MainGui Class Reference

### Static Public Member Functions

- static void [main](#) (String[] args)
- static [ErrorCode](#) [loadProjectFile](#) (File inFile)
- static void [showErrorDialog](#) (String message)
- static void [showAboutWindow](#) ()
- static void [showGpioConfWindow](#) ()
- static void [setNewUC](#) ([Microcontroller](#) uC)
- static void [saveUc](#) ()
- static void [generateCode](#) ()

### Static Public Attributes

- static File [ProjectFile](#)
- static String [ProjectPath](#)

### 4.10.1 Detailed Description

Main GUI state machine

#### Author

Miguel Diaz

#### Version

0.1

## 4.10.2 Member Function Documentation

### 4.10.2.1 generateCode()

```
static void gui.MainGui.generateCode ( ) [static]
```

Generate source code files

### 4.10.2.2 loadProjectFile()

```
static ErrorCode gui.MainGui.loadProjectFile (
    File inFile ) [static]
```

Load the project settings file

#### Parameters

<i>inFile</i>	Settings file
---------------	---------------

#### Returns

Error status

### 4.10.2.3 main()

```
static void gui.MainGui.main (
    String [ ] args ) [static]
```

#### Parameters

<i>args</i>	TBD
-------------	-----

### 4.10.2.4 saveUc()

```
static void gui.MainGui.saveUc ( ) [static]
```

Save the microcontroller's configuration to disk

#### 4.10.2.5 setNewUC()

```
static void gui.MainGui.setNewUC (
    Microcontroller uC ) [static]
```

Set the project's microcontroller configuration

##### Parameters

<i>uC</i>	Microcontroller configuration
-----------	-------------------------------

#### 4.10.2.6 showAboutWindow()

```
static void gui.MainGui.showAboutWindow ( ) [static]
```

Show about information window

#### 4.10.2.7 showErrorDialog()

```
static void gui.MainGui.showErrorDialog (
    String message ) [static]
```

Show an error dialog

##### Parameters

<i>message</i>	Message to display
----------------	--------------------

#### 4.10.2.8 showGpioConfWindow()

```
static void gui.MainGui.showGpioConfWindow ( ) [static]
```

Show the GPIOs configuration window

### 4.10.3 Member Data Documentation

#### 4.10.3.1 ProjectFile

```
File gui.MainGui.ProjectFile [static]
```

Project configuration file

#### 4.10.3.2 ProjectPath

```
String gui.MainGui.ProjectPath [static]
```

Project's location

The documentation for this class was generated from the following file:

- `src/gui/MainGui.java`

### 4.11 gui.MainWindow Class Reference

#### Public Member Functions

- [MainWindow](#) ()
- void [setVisible](#) (boolean status)
- File [OpenFileChooser](#) (String initialPath, String title, FileNameExtensionFilter fileFilter)
- [ErrorCode](#) [setProjectInformation](#) (String projectName, String ucManufacturer, String ucName)

#### Static Public Member Functions

- static void [main](#) (String[] args)

#### Public Attributes

- JFrame [FrmCodeGenerator](#)

#### 4.11.1 Detailed Description

Main application window

Author

Miguel Diaz

Version

0.1



## 4.11.2 Constructor & Destructor Documentation

### 4.11.2.1 MainWindow()

```
gui.MainWindow.MainWindow ( )
```

Create the application.

## 4.11.3 Member Function Documentation

### 4.11.3.1 main()

```
static void gui.MainWindow.main (
    String [ ] args ) [static]
```

Open main window

#### Parameters

<i>args</i>	To be determined
-------------	------------------

### 4.11.3.2 OpenFileChooser()

```
File gui.MainWindow.OpenFileChooser (
    String initialPath,
    String title,
    FileNameExtensionFilter fileFilter )
```

Open file chooser dialog and get the selected file

#### Parameters

<i>initialPath</i>	Path to search the file in
<i>title</i>	Dialog title
<i>fileFilter</i>	Extension filter

**Returns**

Selected file

**4.11.3.3 setProjectInformation()**

```
ErrorCode gui.MainWindow.setProjectInformation (
    String projectName,
    String ucManufacturer,
    String ucName )
```

Set Project's name in its label

**Parameters**

<i>projectName</i>	Project's name
<i>ucManufacturer</i>	Microcontroller's manufacturer
<i>ucName</i>	Microcontroller's model

**Returns**

Error status

**4.11.3.4 setVisible()**

```
void gui.MainWindow.setVisible (
    boolean status )
```

Set visibility of About window

**Parameters**

<i>status</i>	true if visible
---------------	-----------------

**4.11.4 Member Data Documentation**

#### 4.11.4.1 FrmCodeGenerator

```
JFrame gui.MainWindow.FrmCodeGenerator
```

Frame for the main Window

The documentation for this class was generated from the following file:

- src/gui/MainWindow.java

## 4.12 gui.Messages Class Reference

### Static Public Member Functions

- static String [getString](#) (String key)

#### 4.12.1 Detailed Description

[Messages](#) window

Author

ovd

#### 4.12.2 Member Function Documentation

##### 4.12.2.1 getString()

```
static String gui.Messages.getString (  
    String key ) [static]
```

Get String

Parameters

<i>key</i>	Key
------------	-----

Returns

String

The documentation for this class was generated from the following file:

- `src/gui/Messages.java`

## 4.13 microcontroller.Microcontroller Class Reference

### Public Member Functions

- [Microcontroller](#) (Document ucDoc)
- [ErrorCode](#) `processDocument` ()
- [ErrorCode](#) `loadPinsConf` (Document confDoc)
- [Pin](#) `getPin` (int pinNum)
- String `getUc_model` ()
- String `getUc_manufacturer` ()
- int `getUc_pinNum` ()
- int `getUc_gpioNum` ()
- int `getUc_portNum` ()
- [PinConf](#) `getConfiguredPin` (String gpioName)
- boolean `isValid` ()

### Public Attributes

- String [] [Ports](#)
- String [] [Includes\\_Common](#)
- String [] [Includes\\_Gpio](#)
- [PinConf](#) [] [GpioCfgPin](#)

### Static Public Attributes

- static final int [MAX\\_NUMBER\\_OF\\_PINS\\_PER\\_PORT](#) = 16

#### 4.13.1 Detailed Description

[Microcontroller](#) related methods

##### Author

Miguel Diaz

##### Version

0.1

#### 4.13.2 Constructor & Destructor Documentation

##### 4.13.2.1 [Microcontroller](#)()

```
microcontroller.Microcontroller.Microcontroller (  
    Document ucDoc )
```

##### Constructor

## Parameters

<i>ucDoc</i>	Document obtained from XML file
--------------	---------------------------------

## 4.13.3 Member Function Documentation

## 4.13.3.1 getConfiguredPin()

```
PinConf microcontroller.Microcontroller.getConfiguredPin (
    String gpioName )
```

Get the configuration of a pin

## Parameters

<i>gpioName</i>	Name of the pin
-----------------	-----------------

## Returns

Pin configuration

## 4.13.3.2 getPin()

```
Pin microcontroller.Microcontroller.getPin (
    int pinNum )
```

Get a pin's characteristics

## Parameters

<i>pinNum</i>	Number of pin
---------------	---------------

## Returns

Pin's characteristics

#### 4.13.3.3 `getUc_gpioNum()`

```
int microcontroller.Microcontroller.getUc_gpioNum ( )
```

Get the number of GPIOs in the microcontroller

##### Returns

Number of GPIOs

#### 4.13.3.4 `getUc_manufacturer()`

```
String microcontroller.Microcontroller.getUc_manufacturer ( )
```

Get the microcontroller's manufacturer

##### Returns

[Microcontroller](#)'s manufacturer

#### 4.13.3.5 `getUc_model()`

```
String microcontroller.Microcontroller.getUc_model ( )
```

Get the microcontroller's model

##### Returns

[Microcontroller](#)'s model

#### 4.13.3.6 `getUc_pinNum()`

```
int microcontroller.Microcontroller.getUc_pinNum ( )
```

Get the microcontroller's pins number

##### Returns

Number of pins

#### 4.13.3.7 getUc\_portNum()

```
int microcontroller.Microcontroller.getUc_portNum ( )
```

Get the number of ports in the microcontroller

##### Returns

Number of ports

#### 4.13.3.8 isValid()

```
boolean microcontroller.Microcontroller.isValid ( )
```

Check if the microcontroller configuration is valid

##### Returns

true if valid

#### 4.13.3.9 loadPinsConf()

```
ErrorCode microcontroller.Microcontroller.loadPinsConf (
    Document confDoc )
```

Load pins' configuration

##### Parameters

<i>confDoc</i>	Document with pins
----------------	--------------------

##### Returns

Error Code

#### 4.13.3.10 processDocument()

```
ErrorCode microcontroller.Microcontroller.processDocument ( )
```

Process the document obtained from XML file

**Returns**

Error status

### 4.13.4 Member Data Documentation

#### 4.13.4.1 GpioCfgPin

```
PinConf [] microcontroller.Microcontroller.GpioCfgPin
```

Configured pins list

#### 4.13.4.2 Includes\_Common

```
String [] microcontroller.Microcontroller.Includes_Common
```

List of public includes

#### 4.13.4.3 Includes\_Gpio

```
String [] microcontroller.Microcontroller.Includes_Gpio
```

List of Includes for GPIO module

#### 4.13.4.4 MAX\_NUMBER\_OF\_PINS\_PER\_PORT

```
final int microcontroller.Microcontroller.MAX_NUMBER_OF_PINS_PER_PORT = 16 [static]
```

Maximum number of pins allowed in a single port

#### 4.13.4.5 Ports

```
String [] microcontroller.Microcontroller.Ports
```

Ports name list

The documentation for this class was generated from the following file:

- src/microcontroller/Microcontroller.java



## 4.14 configurator.GPIO.Mode Enum Reference

### Static Public Member Functions

- static [Mode](#) [getConfigFromString](#) (String conf)

### Public Attributes

- [MODE\\_INPUT](#)
- [MODE\\_OUTPUT](#)
- [MODE\\_ALTERNATE\\_FUNCTION](#)
- [MODE\\_ANALOG](#)
- [MODE\\_MAX\\_VALUE](#)

### Static Public Attributes

- static final String [STR\\_NAME](#) = "Mode"

#### 4.14.1 Detailed Description

GPIO modes

Author

Miguel Diaz

Version

0.1

#### 4.14.2 Member Function Documentation

##### 4.14.2.1 getConfigFromString()

```
static Mode configurator.GPIO.Mode.getConfigFromString (  
    String conf ) [static]
```

Get the corresponding mode from its name as String

**Parameters**

<i>conf</i>	Configuration name
-------------	--------------------

**Returns**

[Mode](#)

**4.14.3 Member Data Documentation****4.14.3.1 MODE\_ALTERNATE\_FUNCTION**

```
configurator.GPIO.Mode.MODE_ALTERNATE_FUNCTION
```

Alternate function

**4.14.3.2 MODE\_ANALOG**

```
configurator.GPIO.Mode.MODE_ANALOG
```

Analog

**4.14.3.3 MODE\_INPUT**

```
configurator.GPIO.Mode.MODE_INPUT
```

Input

**4.14.3.4 MODE\_MAX\_VALUE**

```
configurator.GPIO.Mode.MODE_MAX_VALUE
```

Maximum value for [Mode](#) enum

**4.14.3.5 MODE\_OUTPUT**

```
configurator.GPIO.Mode.MODE_OUTPUT
```

Output

#### 4.14.3.6 STR\_NAME

```
static final String configurator.GPIO.Mode.STR_NAME = "Mode" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- src/configurator/GPIO/Mode.java

## 4.15 configurator.GPIO.OutLevel Enum Reference

### Static Public Member Functions

- static [OutLevel](#) [getConfigFromString](#) (String conf)

### Public Attributes

- [LOW](#)
- [HIGH](#)
- [MAX\\_VALUE](#)

### Static Public Attributes

- static final String [STR\\_NAME](#) = "OutLevel"

### 4.15.1 Detailed Description

Pin's output/input level

#### Author

Miguel Diaz

#### Version

0.1

### 4.15.2 Member Function Documentation

#### 4.15.2.1 getConfigFromString()

```
static OutLevel configurator.GPIO.OutLevel.getConfigFromString (  
    String conf ) [static]
```

Get the corresponding mode from its name as String

**Parameters**

<i>conf</i>	Configuration name
-------------	--------------------

**Returns**

level

**4.15.3 Member Data Documentation****4.15.3.1 HIGH**

```
configurator.GPIO.OutLevel.HIGH
```

High, logical 1, Vcc

**4.15.3.2 LOW**

```
configurator.GPIO.OutLevel.LOW
```

Low, logical 0, Ground

**4.15.3.3 MAX\_VALUE**

```
configurator.GPIO.OutLevel.MAX_VALUE
```

Maximum value for [OutLevel](#) enum

**4.15.3.4 STR\_NAME**

```
static final String configurator.GPIO.OutLevel.STR_NAME = "OutLevel" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- `src/configurator/GPIO/OutLevel.java`

## 4.16 configurator.GPIO.OutType Enum Reference

### Static Public Member Functions

- static [OutType](#) [getConfFromString](#) (String conf)

### Public Attributes

- [OTYPE\\_PUSH\\_PULL](#)
- [OTYPE\\_OPEN\\_DRAIN](#)
- [OTYPE\\_NOT\\_AVAILABLE](#)
- [OTYPE\\_MAX\\_VALUE](#)

### Static Public Attributes

- static final String [STR\\_NAME](#) = "OutType"

#### 4.16.1 Detailed Description

Pin's output type

##### Author

Miguel Diaz

##### Version

0.1

#### 4.16.2 Member Function Documentation

##### 4.16.2.1 [getConfFromString\(\)](#)

```
static OutType configurator.GPIO.OutType.getConfFromString (  
    String conf ) [static]
```

Get the corresponding output type from its name as String

##### Parameters

<i>conf</i>	Configuration name
-------------	--------------------

**Returns**

Output type

### 4.16.3 Member Data Documentation

#### 4.16.3.1 OTYPE\_MAX\_VALUE

```
configurator.GPIO.OutType.OTYPE_MAX_VALUE
```

Maximum value for [OutType](#) enum

#### 4.16.3.2 OTYPE\_NOT\_AVAILABLE

```
configurator.GPIO.OutType.OTYPE_NOT_AVAILABLE
```

If the pin is configured as input

#### 4.16.3.3 OTYPE\_OPEN\_DRAIN

```
configurator.GPIO.OutType.OTYPE_OPEN_DRAIN
```

Open Drain

#### 4.16.3.4 OTYPE\_PUSH\_PULL

```
configurator.GPIO.OutType.OTYPE_PUSH_PULL
```

Push [Pull](#), totem

#### 4.16.3.5 STR\_NAME

```
static final String configurator.GPIO.OutType.STR_NAME = "OutType" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- `src/configurator/GPIO/OutType.java`

## 4.17 microcontroller.Pin Class Reference

### Public Member Functions

- [Pin](#) ()
- void [setFunc\\_vcc](#) (boolean funcState)
- boolean [getFunc\\_vcc](#) ()
- void [setFunc\\_gnd](#) (boolean funcState)
- boolean [getFunc\\_gnd](#) ()
- void [setFunc\\_gpio](#) (boolean funcState)
- boolean [getFunc\\_gpio](#) ()
- void [setFunc\\_reset](#) (boolean funcState)
- boolean [getFunc\\_reset](#) ()
- void [setFunc\\_misc](#) (boolean funcState)
- boolean [getFunc\\_misc](#) ()
- void [setFeat\\_int](#) (boolean featState)
- boolean [getFeat\\_int](#) ()
- void [setFeat\\_adc](#) (boolean featState)
- boolean [getFeat\\_adc](#) ()
- void [setFeat\\_uart](#) (boolean featState)
- boolean [getFeat\\_uart](#) ()
- void [setFeat\\_i2c](#) (boolean featState)
- boolean [getFeat\\_i2c](#) ()
- void [setFeat\\_spi](#) (boolean featState)
- boolean [getFeat\\_spi](#) ()
- void [setFeat\\_clock](#) (boolean featState)
- boolean [getFeat\\_clock](#) ()
- void [setFeat\\_timer](#) (boolean featState)
- boolean [getFeat\\_timer](#) ()
- void [setFeat\\_reset](#) (boolean featState)
- boolean [getFeat\\_reset](#) ()
- void [setInt](#) (String feature)
- String [getInt](#) ()
- void [setAdc](#) (String feature)
- String [getAdc](#) ()
- void [setUart](#) (String feature)
- String [getUart](#) ()
- void [setI2c](#) (String feature)
- String [getI2c](#) ()
- void [setSpi](#) (String feature)
- String [getSpi](#) ()
- void [setClock](#) (String feature)
- String [getClock](#) ()
- void [setReset](#) (String feature)
- String [getReset](#) ()
- void [setTimer](#) (String feature)
- String [getTimer](#) ()
- void [setName](#) (String pinName)
- String [getName](#) ()
- void [setNumber](#) (int pinNum)
- int [getNumber](#) ()
- void [setPort](#) (String pinPort)
- String [getPort](#) ()
- boolean [isValid](#) ()

## Static Public Attributes

- static final boolean `ENABLE` = true
- static final boolean `DISABLE` = false
- static final boolean `DEF_FUNCTION` = DEF\_BOOLEAN
- static final boolean `DEF_FEATURE_AV` = DEF\_BOOLEAN
- static final String `DEF_FEATURE` = DEF\_STRING
- static final String `DEF_NAME` = DEF\_STRING
- static final int `DEF_NUMBER` = DEF\_INT
- static final String `DEF_PORT` = DEF\_STRING

### 4.17.1 Detailed Description

Basic pin object.

- `Pin` necessary characteristics:
  - Name
  - Number
- `Pin` optional characteristics:
  - Port
- `Pin` main functions:
  - VCC
  - GND
  - GPIO
  - RESET
  - MISC
- `Pin` features:
  - Interruption
  - ADC
  - UART
  - I2C
  - SPI
  - Clock
  - Reset

#### Author

Miguel Diaz

#### Version

0.1



## 4.17.2 Constructor & Destructor Documentation

### 4.17.2.1 Pin()

```
microcontroller.Pin.Pin ( )
```

Initialize all pin's characteristics and features to their default values

## 4.17.3 Member Function Documentation

### 4.17.3.1 getAdc()

```
String microcontroller.Pin.getAdc ( )
```

Get the pin's ADC name

Returns

Pin's ADC

### 4.17.3.2 getClock()

```
String microcontroller.Pin.getClock ( )
```

Get the pin's clock name

Returns

Pin's clock

#### 4.17.3.3 `getFeat_adc()`

```
boolean microcontroller.Pin.getFeat_adc ( )
```

See if the pin has an ADC

##### Returns

Feature availability

#### 4.17.3.4 `getFeat_clock()`

```
boolean microcontroller.Pin.getFeat_clock ( )
```

See if the pin supports a clock

##### Returns

Feature availability

#### 4.17.3.5 `getFeat_i2c()`

```
boolean microcontroller.Pin.getFeat_i2c ( )
```

See if the pin has I2C

##### Returns

Feature availability

#### 4.17.3.6 `getFeat_int()`

```
boolean microcontroller.Pin.getFeat_int ( )
```

See if the pin has an interruption

##### Returns

Feature availability

#### 4.17.3.7 getFeat\_reset()

```
boolean microcontroller.Pin.getFeat_reset ( )
```

See if the pin has a reset feature

##### Returns

Feature availability

#### 4.17.3.8 getFeat\_spi()

```
boolean microcontroller.Pin.getFeat_spi ( )
```

See if the pin has SPI

##### Returns

Feature availability

#### 4.17.3.9 getFeat\_timer()

```
boolean microcontroller.Pin.getFeat_timer ( )
```

See if the pin supports a timer

##### Returns

Feature availability

#### 4.17.3.10 getFeat\_uart()

```
boolean microcontroller.Pin.getFeat_uart ( )
```

See if the pin has a UART

##### Returns

Feature availability

#### 4.17.3.11 getFunc\_gnd()

```
boolean microcontroller.Pin.getFunc_gnd ( )
```

See if the pin is GND

##### Returns

Function availability

#### 4.17.3.12 getFunc\_gpio()

```
boolean microcontroller.Pin.getFunc_gpio ( )
```

See if the pin is GPIO

##### Returns

Function availability

#### 4.17.3.13 getFunc\_misc()

```
boolean microcontroller.Pin.getFunc_misc ( )
```

See if the pin is MISC

##### Returns

Function availability

#### 4.17.3.14 getFunc\_reset()

```
boolean microcontroller.Pin.getFunc_reset ( )
```

See if the pin is RESET

##### Returns

Function availability

#### 4.17.3.15 getFunc\_vcc()

```
boolean microcontroller.Pin.getFunc_vcc ( )
```

See if the pin is Vcc

##### Returns

Function availability

#### 4.17.3.16 getI2c()

```
String microcontroller.Pin.getI2c ( )
```

Get the pin's I2C name

##### Returns

[Pin's I2C](#)

#### 4.17.3.17 getInt()

```
String microcontroller.Pin.getInt ( )
```

Get the pin's interruption name

##### Returns

[Pin's interruption](#)

#### 4.17.3.18 getName()

```
String microcontroller.Pin.getName ( )
```

Get the pin's name

##### Returns

[Pin's name](#)

#### 4.17.3.19 `getNumber()`

```
int microcontroller.Pin.getNumber ( )
```

Get the pin's number

##### Returns

Pin's number

#### 4.17.3.20 `getPort()`

```
String microcontroller.Pin.getPort ( )
```

Get the pin's port

##### Returns

Pin's port

#### 4.17.3.21 `getReset()`

```
String microcontroller.Pin.getReset ( )
```

Get the pin's reset name

##### Returns

Pin's reset

#### 4.17.3.22 `getSpi()`

```
String microcontroller.Pin.getSpi ( )
```

Get the pin's SPI name

##### Returns

Pin's SPI

#### 4.17.3.23 getTimer()

```
String microcontroller.Pin.getTimer ( )
```

Get the pin's timer name

##### Returns

Pin's timer

#### 4.17.3.24 getUart()

```
String microcontroller.Pin.getUart ( )
```

Get the pin's UART name

##### Returns

Pin's UART

#### 4.17.3.25 isValid()

```
boolean microcontroller.Pin.isValid ( )
```

Check if the pin is correctly initialized

##### Returns

True if the pin is correctly initialized

#### 4.17.3.26 setAdc()

```
void microcontroller.Pin.setAdc (
    String feature )
```

Set the pin's ADC

**Parameters**

<i>feature</i>	Pin's ADC
----------------	-----------

**4.17.3.27 setClock()**

```
void microcontroller.Pin.setClock (
    String feature )
```

Set the pin's clock

**Parameters**

<i>feature</i>	Pin's clock
----------------	-------------

**4.17.3.28 setFeat\_adc()**

```
void microcontroller.Pin.setFeat_adc (
    boolean featState )
```

Set the pin's ADC feature

**Parameters**

<i>featState</i>	Feature availability
------------------	----------------------

**4.17.3.29 setFeat\_clock()**

```
void microcontroller.Pin.setFeat_clock (
    boolean featState )
```

Set the pin's Clock feature

**Parameters**

<i>featState</i>	Feature availability
------------------	----------------------



#### 4.17.3.30 setFeat\_i2c()

```
void microcontroller.Pin.setFeat_i2c (  
    boolean featState )
```

Set the pin's I2C feature

##### Parameters

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.31 setFeat\_int()

```
void microcontroller.Pin.setFeat_int (  
    boolean featState )
```

Set the pin's interruption feature

##### Parameters

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.32 setFeat\_reset()

```
void microcontroller.Pin.setFeat_reset (  
    boolean featState )
```

Set the pin's reset feature

##### Parameters

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.33 setFeat\_spi()

```
void microcontroller.Pin.setFeat_spi (  
    boolean featState )
```

```
boolean featState )
```

Set the pin's SPI feature

**Parameters**

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.34 setFeat\_timer()

```
void microcontroller.Pin.setFeat_timer (
    boolean featState )
```

Set the pin's timer feature

**Parameters**

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.35 setFeat\_uart()

```
void microcontroller.Pin.setFeat_uart (
    boolean featState )
```

Set the pin's UART feature

**Parameters**

<i>featState</i>	Feature availability
------------------	----------------------

#### 4.17.3.36 setFunc\_gnd()

```
void microcontroller.Pin.setFunc_gnd (
    boolean funcState )
```

Set the pin to GND status

## Parameters

<i>funcState</i>	Function availability
------------------	-----------------------

## 4.17.3.37 setFunc\_gpio()

```
void microcontroller.Pin.setFunc_gpio (  
    boolean funcState )
```

Set the pin to GPIO status

## Parameters

<i>funcState</i>	Function availability
------------------	-----------------------

## 4.17.3.38 setFunc\_misc()

```
void microcontroller.Pin.setFunc_misc (  
    boolean funcState )
```

Set the pin to MISC status

## Parameters

<i>funcState</i>	Function availability
------------------	-----------------------

## 4.17.3.39 setFunc\_reset()

```
void microcontroller.Pin.setFunc_reset (  
    boolean funcState )
```

Set the pin to RESET status

## Parameters

<i>funcState</i>	Function availability
------------------	-----------------------

#### 4.17.3.40 setFunc\_vcc()

```
void microcontroller.Pin.setFunc_vcc (
    boolean funcState )
```

Set the pin to Vcc status

##### Parameters

<i>funcState</i>	Function availability
------------------	-----------------------

#### 4.17.3.41 setI2c()

```
void microcontroller.Pin.setI2c (
    String feature )
```

Set the pin's I2C

##### Parameters

<i>feature</i>	<a href="#">Pin's I2C</a>
----------------	---------------------------

#### 4.17.3.42 setInt()

```
void microcontroller.Pin.setInt (
    String feature )
```

Set the pin's interruption

##### Parameters

<i>feature</i>	<a href="#">Pin's interruption</a>
----------------	------------------------------------

#### 4.17.3.43 setName()

```
void microcontroller.Pin.setName (
```

```
String pinName )
```

Set the pin's name

#### Parameters

<i>pinName</i>	Pin's name
----------------	------------

#### 4.17.3.44 `setNumber()`

```
void microcontroller.Pin.setNumber (
    int pinNum )
```

Set the pin's number

#### Parameters

<i>pinNum</i>	Pin's number
---------------	--------------

#### 4.17.3.45 `setPort()`

```
void microcontroller.Pin.setPort (
    String pinPort )
```

Set the pin's port

#### Parameters

<i>pinPort</i>	Pin's port
----------------	------------

#### 4.17.3.46 `setReset()`

```
void microcontroller.Pin.setReset (
    String feature )
```

Set the pin's reset

**Parameters**

<i>feature</i>	<a href="#">Pin's reset</a>
----------------	-----------------------------

**4.17.3.47 setSpi()**

```
void microcontroller.Pin.setSpi (
    String feature )
```

Set the pin's SPI

**Parameters**

<i>feature</i>	<a href="#">Pin's SPI</a>
----------------	---------------------------

**4.17.3.48 setTimer()**

```
void microcontroller.Pin.setTimer (
    String feature )
```

Set the pin's timer

**Parameters**

<i>feature</i>	<a href="#">Pin's timer</a>
----------------	-----------------------------

**4.17.3.49 setUart()**

```
void microcontroller.Pin.setUart (
    String feature )
```

Set the pin's UART

**Parameters**

<i>feature</i>	<a href="#">Pin's UART</a>
----------------	----------------------------

### 4.17.4 Member Data Documentation

#### 4.17.4.1 DEF\_FEATURE

```
final String microcontroller.Pin.DEF_FEATURE = DEF_STRING [static]
```

Default value for pin's feature as not available

#### 4.17.4.2 DEF\_FEATURE\_AV

```
final boolean microcontroller.Pin.DEF_FEATURE_AV = DEF_BOOLEAN [static]
```

Default value for pin's feature availability as not available

#### 4.17.4.3 DEF\_FUNCTION

```
final boolean microcontroller.Pin.DEF_FUNCTION = DEF_BOOLEAN [static]
```

Default value for pin's function as not enabled

#### 4.17.4.4 DEF\_NAME

```
final String microcontroller.Pin.DEF_NAME = DEF_STRING [static]
```

Default value for pin's name

#### 4.17.4.5 DEF\_NUMBER

```
final int microcontroller.Pin.DEF_NUMBER = DEF_INT [static]
```

Default value for pin's number

#### 4.17.4.6 DEF\_PORT

```
final String microcontroller.Pin.DEF_PORT = DEF_STRING [static]
```

Default value for pin's port

#### 4.17.4.7 DISABLE

```
final boolean microcontroller.Pin.DISABLE = false [static]
```

Disable value for features and functions

#### 4.17.4.8 ENABLE

```
final boolean microcontroller.Pin.ENABLE = true [static]
```

Enable value for features and functions

The documentation for this class was generated from the following file:

- src/microcontroller/Pin.java

## 4.18 configurator.PinConf Class Reference

### Public Member Functions

- [PinConf](#) ([Pin](#) gpioPin)
- boolean [isValid](#) ()
- String [getPort](#) ()
- String [getPinName](#) ()
- String [getCodeName](#) ()
- void [setCodeName](#) (String name)
- [Mode](#) [getMode](#) ()
- void [setMode](#) ([Mode](#) mode)
- [OutType](#) [getOutType](#) ()
- void [setOutType](#) ([OutType](#) outType)
- [OutLevel](#) [getOutLevel](#) ()
- void [setOutLevel](#) ([OutLevel](#) level)
- [Speed](#) [getSpeed](#) ()
- void [setSpeed](#) ([Speed](#) speed)
- [Pull](#) [getPull](#) ()
- void [setPull](#) ([Pull](#) pull)
- boolean [isAv\\_Adc](#) ()
- boolean [isAv\\_altFunc](#) ()

### Static Public Attributes

- static final [Mode](#) [DF\\_MODE](#) = [Mode.MODE\\_INPUT](#)
- static final [Speed](#) [DF\\_SPEED](#) = [Speed.SPEED\\_FAST](#)
- static final [OutType](#) [DF\\_OUTTYPE](#) = [OutType.OTYPE\\_PUSH\\_PULL](#)
- static final [OutLevel](#) [DF\\_OUT\\_LEVEL](#) = [OutLevel.LOW](#)
- static final [Pull](#) [DF\\_PULL](#) = [Pull.PULL\\_NOT\\_AVAILABLE](#)
- static final String [DF\\_CODE\\_NAME](#) = ""



### 4.18.1 Detailed Description

GPIO pin configuration

Author

Miguel Diaz

Version

0.1

### 4.18.2 Constructor & Destructor Documentation

#### 4.18.2.1 PinConf()

```
configurator.PinConf.PinConf (
    Pin gpioPin )
```

Constructor

Parameters

<i>gpioPin</i>	Pin information
----------------	-----------------

### 4.18.3 Member Function Documentation

#### 4.18.3.1 getCodeName()

```
String configurator.PinConf.getCodeName ( )
```

Get the pin's user selected name

Returns

pin's name

#### 4.18.3.2 `getMode()`

```
Mode configurator.PinConf.getMode ( )
```

Get the pin's mode configuration

##### Returns

Mode

#### 4.18.3.3 `getOutLevel()`

```
OutLevel configurator.PinConf.getOutLevel ( )
```

Get the pin's output level

##### Returns

Pin's output level

#### 4.18.3.4 `getOutType()`

```
OutType configurator.PinConf.getOutType ( )
```

Get the pin's output configuration

##### Returns

Output configuration

#### 4.18.3.5 `getPinName()`

```
String configurator.PinConf.getPinName ( )
```

Get the pin's number

##### Returns

Pin's number

#### 4.18.3.6 getPort()

```
String configurator.PinConf.getPort ( )
```

Get the pin's port

##### Returns

Port

#### 4.18.3.7 getPull()

```
Pull configurator.PinConf.getPull ( )
```

Get the pin's pull resistor configuration

##### Returns

Pull Resistor configuration

#### 4.18.3.8 getSpeed()

```
Speed configurator.PinConf.getSpeed ( )
```

Get the pin's speed

##### Returns

Speed

#### 4.18.3.9 isAv\_Adc()

```
boolean configurator.PinConf.isAv_Adc ( )
```

Check availability of ADC

##### Returns

True if ADC is available

#### 4.18.3.10 isAv\_altFunc()

```
boolean configurator.PinConf.isAv_altFunc ( )
```

Check the availability of alternate function

##### Returns

True if alternate function is available

#### 4.18.3.11 isValid()

```
boolean configurator.PinConf.isValid ( )
```

Check if the GPIO pin is valid

##### Returns

True if valid

#### 4.18.3.12 setCodeName()

```
void configurator.PinConf.setCodeName (
    String name )
```

Set the pin's user selected name

##### Parameters

<i>name</i>	Pin's name
-------------	------------

#### 4.18.3.13 setMode()

```
void configurator.PinConf.setMode (
    Mode mode )
```

Set the pin's mode configuration

## Parameters

<i>mode</i>	Mode
-------------	------

## 4.18.3.14 setOutLevel()

```
void configurator.PinConf.setOutLevel (
    OutLevel level )
```

Set the pin's output level

## Parameters

<i>level</i>	Pin's output level
--------------	--------------------

## 4.18.3.15 setOutType()

```
void configurator.PinConf.setOutType (
    OutType outType )
```

Set the pin's output configuration

## Parameters

<i>outType</i>	Output configuration
----------------	----------------------

## 4.18.3.16 setPull()

```
void configurator.PinConf.setPull (
    Pull pull )
```

Set the pull resistor configuration

## Parameters

<i>pull</i>	Resistor configuration
-------------	------------------------

#### 4.18.3.17 setSpeed()

```
void configurator.PinConf.setSpeed (
    Speed speed )
```

Set the pin's speed

##### Parameters

<i>speed</i>	Speed
--------------	-------

### 4.18.4 Member Data Documentation

#### 4.18.4.1 DF\_CODE\_NAME

```
final String configurator.PinConf.DF_CODE_NAME = "" [static]
```

Default pin's code name

#### 4.18.4.2 DF\_MODE

```
final Mode configurator.PinConf.DF_MODE = Mode.MODE_INPUT [static]
```

Default Pin mode

#### 4.18.4.3 DF\_OUT\_LEVEL

```
final OutLevel configurator.PinConf.DF_OUT_LEVEL = OutLevel.LOW [static]
```

Default pin's output level

#### 4.18.4.4 DF\_OUTTYPE

```
final OutType configurator.PinConf.DF_OUTTYPE = OutType.OTYPE_PUSH_PULL [static]
```

Default pin's output type

#### 4.18.4.5 DF\_PULL

```
final Pull configurator.PinConf.DF_PULL = Pull.PULL_NOT_AVAILABLE [static]
```

Default pin's pull resistor

#### 4.18.4.6 DF\_SPEED

```
final Speed configurator.PinConf.DF_SPEED = Speed.SPEED_FAST [static]
```

Default pin's speed

The documentation for this class was generated from the following file:

- src/configurator/PinConf.java

## 4.19 projectConfiguration.ProjectSettings Class Reference

### Public Member Functions

- [ProjectSettings](#) ()
- [ErrorCode processDocument](#) ()
- [ErrorCode openProjectFile](#) (File inFile)
- File [getConfFile](#) ()
- File [getUcFile](#) ()
- String [getProjectName](#) ()
- String [getFrameworkPath](#) ()

### 4.19.1 Detailed Description

Project settings class

#### Author

Miguel Diaz

#### Version

0.2

### 4.19.2 Constructor & Destructor Documentation

#### 4.19.2.1 ProjectSettings()

```
projectConfiguration.ProjectSettings.ProjectSettings ( )
```

Constructor

### 4.19.3 Member Function Documentation

#### 4.19.3.1 getConfFile()

```
File projectConfiguration.ProjectSettings.getConfFile ( )
```

Get the project configuration file

Returns

Project configuration file

#### 4.19.3.2 getFrameworkPath()

```
String projectConfiguration.ProjectSettings.getFrameworkPath ( )
```

Get the framework folder

Returns

framework folder

#### 4.19.3.3 getProjectName()

```
String projectConfiguration.ProjectSettings.getProjectName ( )
```

Get the project's name

Returns

Project's name



#### 4.19.3.4 getUcFile()

```
File projectConfiguration.ProjectSettings.getUcFile ( )
```

Get the project microcontroller file

##### Returns

Project microcontroller file

#### 4.19.3.5 openProjectFile()

```
ErrorCode projectConfiguration.ProjectSettings.openProjectFile (
    File inFile )
```

Open the project settings file

##### Parameters

<i>inFile</i>	Project file
---------------	--------------

##### Returns

Error Status

#### 4.19.3.6 processDocument()

```
ErrorCode projectConfiguration.ProjectSettings.processDocument ( )
```

Process the document obtained from the XML file

##### Returns

Error Status

The documentation for this class was generated from the following file:

- src/projectConfiguration/ProjectSettings.java

## 4.20 configurator.GPIO.Pull Enum Reference

### Static Public Member Functions

- static [Pull](#) `getConfigFromString` (String *conf*)

### Public Attributes

- [PULL\\_UP](#)
- [PULL\\_DOWN](#)
- [PULL\\_NOT\\_AVAILABLE](#)
- [PULL\\_MAX\\_VALUE](#)

### Static Public Attributes

- static final String [STR\\_NAME](#) = "Pull"

### 4.20.1 Detailed Description

Pin's pull resistor

#### Author

Miguel Diaz

#### Version

0.1

### 4.20.2 Member Function Documentation

#### 4.20.2.1 `getConfigFromString()`

```
static Pull configurator.GPIO.Pull.getConfigFromString (  
    String conf ) [static]
```

Get the corresponding [Pull](#) configuration from its name as String

#### Parameters

<i>conf</i>	Configuration name
-------------	--------------------

#### Returns

Pull configuration

### 4.20.3 Member Data Documentation

#### 4.20.3.1 PULL\_DOWN

```
configurator.GPIO.Pull.PULL_DOWN
```

Pull Down

#### 4.20.3.2 PULL\_MAX\_VALUE

```
configurator.GPIO.Pull.PULL_MAX_VALUE
```

Maximum value for Pull enum

#### 4.20.3.3 PULL\_NOT\_AVAILABLE

```
configurator.GPIO.Pull.PULL_NOT_AVAILABLE
```

If the pin is configured as output, or there is no resistor available

#### 4.20.3.4 PULL\_UP

```
configurator.GPIO.Pull.PULL_UP
```

Pull Up

#### 4.20.3.5 STR\_NAME

```
static final String configurator.GPIO.Pull.STR_NAME = "Pull" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- src/configurator/GPIO/Pull.java

## 4.21 configurator.GPIO.Speed Enum Reference

### Static Public Member Functions

- static [Speed](#) [getConfFromString](#) (String conf)

### Public Attributes

- [SPEED\\_FAST](#)
- [SPEED\\_MEDIUM](#)
- [SPEED\\_HIGH](#)
- [SPEED\\_NOT\\_AVAILABLE](#)
- [SPEED\\_MAX\\_VALUE](#)

### Static Public Attributes

- static final String [STR\\_NAME](#) = "Speed"

### 4.21.1 Detailed Description

Pin's speed

#### Author

Miguel Diaz

#### Version

0.1

### 4.21.2 Member Function Documentation

#### 4.21.2.1 [getConfFromString\(\)](#)

```
static Speed configurator.GPIO.Speed.getConfFromString (  
    String conf ) [static]
```

Get the corresponding [Speed](#) configuration from its name as String

#### Parameters

<i>conf</i>	Configuration name
-------------	--------------------

#### Returns

[Speed](#)

### 4.21.3 Member Data Documentation

#### 4.21.3.1 SPEED\_FAST

```
configurator.GPIO.Speed.SPEED_FAST
```

Fast

#### 4.21.3.2 SPEED\_HIGH

```
configurator.GPIO.Speed.SPEED_HIGH
```

High

#### 4.21.3.3 SPEED\_MAX\_VALUE

```
configurator.GPIO.Speed.SPEED_MAX_VALUE
```

Maximum value for [Speed](#) enum

#### 4.21.3.4 SPEED\_MEDIUM

```
configurator.GPIO.Speed.SPEED_MEDIUM
```

Medium

#### 4.21.3.5 SPEED\_NOT\_AVAILABLE

```
configurator.GPIO.Speed.SPEED_NOT_AVAILABLE
```

Not all MCUs will have this setting

#### 4.21.3.6 STR\_NAME

```
static final String configurator.GPIO.Speed.STR_NAME = "Speed" [static]
```

Name as String

The documentation for this enum was generated from the following file:

- src/configurator/GPIO/Speed.java

## 4.22 xmlParser.TestMain Class Reference

### Static Public Member Functions

- static void [main](#) (String[] openOption)

#### 4.22.1 Detailed Description

Dummy main class for testing the other classes

Author

Miguel Diaz

#### 4.22.2 Member Function Documentation

##### 4.22.2.1 main()

```
static void xmlParser.TestMain.main (  
    String [] openOption ) [static]
```

Main without GUI

#### Parameters

<i>openOption</i>	Options include:
-------------------	---------------------

The documentation for this class was generated from the following file:

- src/xmlParser/TestMain.java

## 4.23 xmlParser.XmlOpener Class Reference

### Public Member Functions

- [XmlOpener](#) ()
- [ErrorCode OpenFile](#) (File inFile)
- Document [getParsedDoc](#) ()

### Static Public Member Functions

- static String [getElementInfoFromDoc](#) (Document doc, String elementName)
- static String [getElementInfo](#) (Element element, String elementName)

### 4.23.1 Detailed Description

Open and process XML files

#### Author

H112943

#### Version

0.1

### 4.23.2 Constructor & Destructor Documentation

#### 4.23.2.1 XmlOpener()

```
xmlParser.XmlOpener.XmlOpener ( )
```

Constructor

### 4.23.3 Member Function Documentation

#### 4.23.3.1 getElementInfo()

```
static String xmlParser.XmlOpener.getElementInfo (
    Element element,
    String elementName ) [static]
```

Get an XML sub element information

**Parameters**

<i>element</i>	XML main element
<i>elementName</i>	Sub element's name

**Returns**

Sub element's information

**4.23.3.2 getElementInfoFromDoc()**

```
static String xmlParser.XmlOpener.getElementInfoFromDoc (
    Document doc,
    String elementName ) [static]
```

Get an XML element information

**Parameters**

<i>doc</i>	Document from XML file
<i>elementName</i>	Element's name

**Returns**

Element's information

**4.23.3.3 getParsedDoc()**

```
Document xmlParser.XmlOpener.getParsedDoc ( )
```

Get the parsed document AFTER opening the file

**Returns**

Parsed document

**4.23.3.4 OpenFile()**

```
ErrorCode xmlParser.XmlOpener.OpenFile (
    File inFile )
```

Open the XML file



## Parameters

<i>inFile</i>	XML file
---------------	----------

## Returns

Error code

The documentation for this class was generated from the following file:

- src/xmlParser/XmlOpener.java



# Index

- AboutWindow
  - gui>AboutWindow, [9](#)
- addPin
  - xmlCreator.ConfXmlWriter, [19](#)
- CODE\_NAME
  - configurator.GPIO.CodeName, [12](#)
- CodeGenerator
  - framework.CodeGenerator, [10](#)
- common, [5](#)
- common.ErrorCode, [20](#)
  - EX\_ERROR, [20](#)
  - FILE\_CONF\_ERROR, [20](#)
  - FILE\_READ\_ERROR, [20](#)
  - FILE\_WRITE\_ERROR, [21](#)
  - NO\_ERROR, [21](#)
  - STR\_INVALID, [21](#)
- common.Features, [21](#)
  - DEBUG, [22](#)
  - DEBUG\_STR, [23](#)
  - debugPrint, [22](#)
  - SW\_VERSION, [23](#)
  - VERBOSE, [23](#)
  - VERBOSE\_STR, [23](#)
  - verbosePrint, [22](#)
  - VERSION\_NAME, [23](#)
  - VERSION\_STATUS, [23](#)
- configurator, [5](#)
- configurator.ConfigurationFile, [17](#)
  - STR\_PROJ\_CONF\_FILE, [18](#)
- configurator.GPIO.CodeName, [11](#)
  - CODE\_NAME, [12](#)
  - STR\_NAME, [12](#)
- configurator.GPIO.Mode, [37](#)
  - getConfFromString, [37](#)
  - MODE\_ALTERNATE\_FUNCTION, [38](#)
  - MODE\_ANALOG, [38](#)
  - MODE\_INPUT, [38](#)
  - MODE\_MAX\_VALUE, [38](#)
  - MODE\_OUTPUT, [38](#)
  - STR\_NAME, [38](#)
- configurator.GPIO.OutLevel, [39](#)
  - getConfFromString, [39](#)
  - HIGH, [40](#)
  - LOW, [40](#)
  - MAX\_VALUE, [40](#)
  - STR\_NAME, [40](#)
- configurator.GPIO.OutType, [41](#)
  - getConfFromString, [41](#)
  - OTYPE\_MAX\_VALUE, [42](#)
  - OTYPE\_NOT\_AVAILABLE, [42](#)
  - OTYPE\_OPEN\_DRAIN, [42](#)
  - OTYPE\_PUSH\_PULL, [42](#)
  - STR\_NAME, [42](#)
- configurator.GPIO.Pull, [70](#)
  - getConfFromString, [70](#)
  - PULL\_DOWN, [71](#)
  - PULL\_MAX\_VALUE, [71](#)
  - PULL\_NOT\_AVAILABLE, [71](#)
  - PULL\_UP, [71](#)
  - STR\_NAME, [71](#)
- configurator.GPIO.Speed, [72](#)
  - getConfFromString, [72](#)
  - SPEED\_FAST, [73](#)
  - SPEED\_HIGH, [73](#)
  - SPEED\_MAX\_VALUE, [73](#)
  - SPEED\_MEDIUM, [73](#)
  - SPEED\_NOT\_AVAILABLE, [73](#)
  - STR\_NAME, [73](#)
- configurator.PinConf, [60](#)
  - DF\_CODE\_NAME, [66](#)
  - DF\_MODE, [66](#)
  - DF\_OUT\_LEVEL, [66](#)
  - DF\_OUTTYPE, [66](#)
  - DF\_PULL, [66](#)
  - DF\_SPEED, [67](#)
  - getCodeName, [61](#)
  - getMode, [61](#)
  - getOutLevel, [62](#)
  - getOutType, [62](#)
  - getPinName, [62](#)
  - getPort, [62](#)
  - getPull, [63](#)
  - getSpeed, [63](#)
  - isAv\_Adc, [63](#)
  - isAv\_altFunc, [63](#)
  - isValid, [64](#)
  - PinConf, [61](#)
  - setCodeName, [64](#)
  - setMode, [64](#)
  - setOutLevel, [65](#)

- setOutType, [65](#)
  - setPull, [65](#)
  - setSpeed, [66](#)
- ConfXmlWriter
  - xmlCreator.ConfXmlWriter, [18](#)
- DEBUG
  - common.Features, [22](#)
- DEBUG\_STR
  - common.Features, [23](#)
- debugPrint
  - common.Features, [22](#)
- DEF\_FEATURE
  - microcontroller.Pin, [59](#)
- DEF\_FEATURE\_AV
  - microcontroller.Pin, [59](#)
- DEF\_FUNCTION
  - microcontroller.Pin, [59](#)
- DEF\_NAME
  - microcontroller.Pin, [59](#)
- DEF\_NUMBER
  - microcontroller.Pin, [59](#)
- DEF\_PORT
  - microcontroller.Pin, [59](#)
- DF\_CODE\_NAME
  - configurator.PinConf, [66](#)
- DF\_MODE
  - configurator.PinConf, [66](#)
- DF\_OUT\_LEVEL
  - configurator.PinConf, [66](#)
- DF\_OUTTYPE
  - configurator.PinConf, [66](#)
- DF\_PULL
  - configurator.PinConf, [66](#)
- DF\_SPEED
  - configurator.PinConf, [67](#)
- DISABLE
  - microcontroller.Pin, [59](#)
- ENABLE
  - microcontroller.Pin, [60](#)
- EX\_ERROR
  - common.ErrorCode, [20](#)
- FILE\_CONF\_ERROR
  - common.ErrorCode, [20](#)
- FILE\_READ\_ERROR
  - common.ErrorCode, [20](#)
- FILE\_WRITE\_ERROR
  - common.ErrorCode, [21](#)
- framework, [6](#)
- framework.CodeGenerator, [10](#)
  - CodeGenerator, [10](#)
  - Generate, [11](#)
- framework.Common, [12](#)
  - getCfgFileCPath, [13](#)
  - getCfgFileHPath, [13](#)
  - getCfgPath, [14](#)
  - getCommonIncludes, [14](#)
  - getFrameworkCommonFilePath, [14](#)
  - getFrameworkIncludesFilePath, [15](#)
  - getInstallationFwkPath, [15](#)
  - getProjectFwkPath, [15](#)
  - NL, [16](#)
  - setInstallationFwkPath, [16](#)
  - setProjectFwkPath, [16](#)
  - STR\_DEFINITION, [16](#)
  - STR\_HEADER\_EXT, [17](#)
  - STR\_INCLUDE, [17](#)
  - STR\_MODULE\_GPIO, [17](#)
- FrmCodeGenerator
  - gui.MainWindow, [30](#)
- Generate
  - framework.CodeGenerator, [11](#)
- generateCode
  - gui.MainGui, [26](#)
- getAdc
  - microcontroller.Pin, [45](#)
- getCfgFileCPath
  - framework.Common, [13](#)
- getCfgFileHPath
  - framework.Common, [13](#)
- getCfgPath
  - framework.Common, [14](#)
- getClock
  - microcontroller.Pin, [45](#)
- getCodeName
  - configurator.PinConf, [61](#)
- getCommonIncludes
  - framework.Common, [14](#)
- getConfFile
  - projectConfiguration.ProjectSettings, [68](#)
- getConfFromString
  - configurator.GPIO.Mode, [37](#)
  - configurator.GPIO.OutLevel, [39](#)
  - configurator.GPIO.OutType, [41](#)
  - configurator.GPIO.Pull, [70](#)
  - configurator.GPIO.Speed, [72](#)
- getConfiguredPin
  - microcontroller.Microcontroller, [33](#)
- getElementInfo
  - xmlParser.XmlOpener, [75](#)
- getElementInfoFromDoc
  - xmlParser.XmlOpener, [76](#)
- getFeat\_adc
  - microcontroller.Pin, [45](#)
- getFeat\_clock
  - microcontroller.Pin, [46](#)

- getFeat\_i2c
  - microcontroller.Pin, [46](#)
- getFeat\_int
  - microcontroller.Pin, [46](#)
- getFeat\_reset
  - microcontroller.Pin, [46](#)
- getFeat\_spi
  - microcontroller.Pin, [47](#)
- getFeat\_timer
  - microcontroller.Pin, [47](#)
- getFeat\_uart
  - microcontroller.Pin, [47](#)
- getFrameworkCommonFilePath
  - framework.Common, [14](#)
- getFrameworkIncludesFilePath
  - framework.Common, [15](#)
- getFrameworkPath
  - projectConfiguration.ProjectSettings, [68](#)
- getFunc\_gnd
  - microcontroller.Pin, [47](#)
- getFunc\_gpio
  - microcontroller.Pin, [48](#)
- getFunc\_misc
  - microcontroller.Pin, [48](#)
- getFunc\_reset
  - microcontroller.Pin, [48](#)
- getFunc\_vcc
  - microcontroller.Pin, [48](#)
- getI2c
  - microcontroller.Pin, [49](#)
- getInstallationFwkPath
  - framework.Common, [15](#)
- getInt
  - microcontroller.Pin, [49](#)
- getMode
  - configurator.PinConf, [61](#)
- getName
  - microcontroller.Pin, [49](#)
- getNumber
  - microcontroller.Pin, [49](#)
- getOutLevel
  - configurator.PinConf, [62](#)
- getOutType
  - configurator.PinConf, [62](#)
- getParsedDoc
  - xmlParser.XmlOpener, [76](#)
- getPin
  - microcontroller.Microcontroller, [33](#)
- getPinName
  - configurator.PinConf, [62](#)
- getPort
  - configurator.PinConf, [62](#)
  - microcontroller.Pin, [50](#)
- getProjectFwkPath
  - framework.Common, [15](#)
- getProjectName
  - projectConfiguration.ProjectSettings, [68](#)
- getPull
  - configurator.PinConf, [63](#)
- getReset
  - microcontroller.Pin, [50](#)
- getSpeed
  - configurator.PinConf, [63](#)
- getSpi
  - microcontroller.Pin, [50](#)
- getString
  - gui.Messages, [31](#)
- getTimer
  - microcontroller.Pin, [50](#)
- getUart
  - microcontroller.Pin, [51](#)
- getUc\_gpioNum
  - microcontroller.Microcontroller, [33](#)
- getUc\_manufacturer
  - microcontroller.Microcontroller, [34](#)
- getUc\_model
  - microcontroller.Microcontroller, [34](#)
- getUc\_pinNum
  - microcontroller.Microcontroller, [34](#)
- getUc\_portNum
  - microcontroller.Microcontroller, [34](#)
- getUcFile
  - projectConfiguration.ProjectSettings, [68](#)
- GpioCfgPin
  - microcontroller.Microcontroller, [36](#)
- GpioConfWindow
  - gui.GpioConfWindow, [24](#)
- gui, [6](#)
  - AboutWindow, [9](#)
  - AboutWindow, [9](#)
  - main, [10](#)
- gui.GpioConfWindow, [24](#)
  - GpioConfWindow, [24](#)
  - main, [24](#)
- gui.MainGui, [25](#)
  - generateCode, [26](#)
  - loadProjectFile, [26](#)
  - main, [26](#)
  - ProjectFile, [27](#)
  - ProjectPath, [28](#)
  - saveUc, [26](#)
  - setNewUC, [26](#)
  - showAboutWindow, [27](#)
  - showErrorDialog, [27](#)
  - showGpioConfWindow, [27](#)
- gui.MainWindow, [28](#)
  - FrmCodeGenerator, [30](#)
  - main, [29](#)

- MainWindow, [29](#)
  - OpenFileChooser, [29](#)
  - setProjectInformation, [30](#)
  - setVisible, [30](#)
- gui.Messages, [31](#)
- getString, [31](#)
- HIGH
  - configurator.GPIO.OutLevel, [40](#)
- Includes\_Common
  - microcontroller.Microcontroller, [36](#)
- Includes\_Gpio
  - microcontroller.Microcontroller, [36](#)
- isAv\_Adc
  - configurator.PinConf, [63](#)
- isAv\_altFunc
  - configurator.PinConf, [63](#)
- isValid
  - configurator.PinConf, [64](#)
  - microcontroller.Microcontroller, [35](#)
  - microcontroller.Pin, [51](#)
- loadPinsConf
  - microcontroller.Microcontroller, [35](#)
- loadProjectFile
  - gui.MainGui, [26](#)
- LOW
  - configurator.GPIO.OutLevel, [40](#)
- main
  - gui.AboutWindow, [10](#)
  - gui.GpioConfWindow, [24](#)
  - gui.MainGui, [26](#)
  - gui.MainWindow, [29](#)
  - xmlParser.TestMain, [74](#)
- MainWindow
  - gui.MainWindow, [29](#)
- MAX\_NUMBER\_OF\_PINS\_PER\_PORT
  - microcontroller.Microcontroller, [36](#)
- MAX\_VALUE
  - configurator.GPIO.OutLevel, [40](#)
- Microcontroller
  - microcontroller.Microcontroller, [32](#)
- microcontroller, [7](#)
- microcontroller.Microcontroller, [32](#)
  - getConfiguredPin, [33](#)
  - getPin, [33](#)
  - getUc\_gpioNum, [33](#)
  - getUc\_manufacturer, [34](#)
  - getUc\_model, [34](#)
  - getUc\_pinNum, [34](#)
  - getUc\_portNum, [34](#)
  - GpioCfgPin, [36](#)
  - Includes\_Common, [36](#)
  - Includes\_Gpio, [36](#)
  - isValid, [35](#)
  - loadPinsConf, [35](#)
  - MAX\_NUMBER\_OF\_PINS\_PER\_PORT, [36](#)
  - Microcontroller, [32](#)
  - Ports, [36](#)
  - processDocument, [35](#)
- microcontroller.Pin, [43](#)
  - DEF\_FEATURE, [59](#)
  - DEF\_FEATURE\_AV, [59](#)
  - DEF\_FUNCTION, [59](#)
  - DEF\_NAME, [59](#)
  - DEF\_NUMBER, [59](#)
  - DEF\_PORT, [59](#)
  - DISABLE, [59](#)
  - ENABLE, [60](#)
  - getAdc, [45](#)
  - getClock, [45](#)
  - getFeat\_adc, [45](#)
  - getFeat\_clock, [46](#)
  - getFeat\_i2c, [46](#)
  - getFeat\_int, [46](#)
  - getFeat\_reset, [46](#)
  - getFeat\_spi, [47](#)
  - getFeat\_timer, [47](#)
  - getFeat\_uart, [47](#)
  - getFunc\_gnd, [47](#)
  - getFunc\_gpio, [48](#)
  - getFunc\_misc, [48](#)
  - getFunc\_reset, [48](#)
  - getFunc\_vcc, [48](#)
  - getI2c, [49](#)
  - getInt, [49](#)
  - getName, [49](#)
  - getNumber, [49](#)
  - getPort, [50](#)
  - getReset, [50](#)
  - getSpi, [50](#)
  - getTimer, [50](#)
  - getUart, [51](#)
  - isValid, [51](#)
  - Pin, [45](#)
  - setAdc, [51](#)
  - setClock, [52](#)
  - setFeat\_adc, [52](#)
  - setFeat\_clock, [52](#)
  - setFeat\_i2c, [53](#)
  - setFeat\_int, [53](#)
  - setFeat\_reset, [53](#)
  - setFeat\_spi, [53](#)
  - setFeat\_timer, [54](#)
  - setFeat\_uart, [54](#)
  - setFunc\_gnd, [54](#)
  - setFunc\_gpio, [55](#)

- setFunc\_misc, 55
- setFunc\_reset, 55
- setFunc\_vcc, 56
- setI2c, 56
- setInt, 56
- setName, 56
- setNumber, 57
- setPort, 57
- setReset, 57
- setSpi, 58
- setTimer, 58
- setUart, 58
- MODE\_ALTERNATE\_FUNCTION
  - configurator.GPIO.Mode, 38
- MODE\_ANALOG
  - configurator.GPIO.Mode, 38
- MODE\_INPUT
  - configurator.GPIO.Mode, 38
- MODE\_MAX\_VALUE
  - configurator.GPIO.Mode, 38
- MODE\_OUTPUT
  - configurator.GPIO.Mode, 38
- NL
  - framework.Common, 16
- NO\_ERROR
  - common.ErrorCode, 21
- OpenFile
  - xmlParser.XmlOpener, 76
- OpenFileChooser
  - gui.MainWindow, 29
- openProjectFile
  - projectConfiguration.ProjectSettings, 69
- OTYPE\_MAX\_VALUE
  - configurator.GPIO.OutType, 42
- OTYPE\_NOT\_AVAILABLE
  - configurator.GPIO.OutType, 42
- OTYPE\_OPEN\_DRAIN
  - configurator.GPIO.OutType, 42
- OTYPE\_PUSH\_PULL
  - configurator.GPIO.OutType, 42
- Pin
  - microcontroller.Pin, 45
- PinConf
  - configurator.PinConf, 61
- Ports
  - microcontroller.Microcontroller, 36
- processDocument
  - microcontroller.Microcontroller, 35
  - projectConfiguration.ProjectSettings, 69
- projectConfiguration, 7
  - projectConfiguration.ProjectSettings, 67
  - getConfFile, 68
  - getFrameworkPath, 68
  - getProjectName, 68
  - getUcFile, 68
  - openProjectFile, 69
  - processDocument, 69
  - ProjectSettings, 67
- ProjectFile
  - gui.MainGui, 27
- ProjectPath
  - gui.MainGui, 28
- ProjectSettings
  - projectConfiguration.ProjectSettings, 67
- PULL\_DOWN
  - configurator.GPIO.Pull, 71
- PULL\_MAX\_VALUE
  - configurator.GPIO.Pull, 71
- PULL\_NOT\_AVAILABLE
  - configurator.GPIO.Pull, 71
- PULL\_UP
  - configurator.GPIO.Pull, 71
- saveUc
  - gui.MainGui, 26
- setAdc
  - microcontroller.Pin, 51
- setClock
  - microcontroller.Pin, 52
- setCodeName
  - configurator.PinConf, 64
- setFeat\_adc
  - microcontroller.Pin, 52
- setFeat\_clock
  - microcontroller.Pin, 52
- setFeat\_i2c
  - microcontroller.Pin, 53
- setFeat\_int
  - microcontroller.Pin, 53
- setFeat\_reset
  - microcontroller.Pin, 53
- setFeat\_spi
  - microcontroller.Pin, 53
- setFeat\_timer
  - microcontroller.Pin, 54
- setFeat\_uart
  - microcontroller.Pin, 54
- setFunc\_gnd
  - microcontroller.Pin, 54
- setFunc\_gpio
  - microcontroller.Pin, 55
- setFunc\_misc
  - microcontroller.Pin, 55
- setFunc\_reset
  - microcontroller.Pin, 55
- setFunc\_vcc

- microcontroller.Pin, 56
- setI2c
  - microcontroller.Pin, 56
- setInstallationFwkPath
  - framework.Common, 16
- setInt
  - microcontroller.Pin, 56
- setMode
  - configurator.PinConf, 64
- setName
  - microcontroller.Pin, 56
- setNewUC
  - gui.MainGui, 26
- setNumber
  - microcontroller.Pin, 57
- setOutLevel
  - configurator.PinConf, 65
- setOutType
  - configurator.PinConf, 65
- setPort
  - microcontroller.Pin, 57
- setProjectFwkPath
  - framework.Common, 16
- setProjectInformation
  - gui.MainWindow, 30
- setPull
  - configurator.PinConf, 65
- setReset
  - microcontroller.Pin, 57
- setSpeed
  - configurator.PinConf, 66
- setSpi
  - microcontroller.Pin, 58
- setTimer
  - microcontroller.Pin, 58
- setUart
  - microcontroller.Pin, 58
- setVisible
  - gui.MainWindow, 30
- showAboutWindow
  - gui.MainGui, 27
- showErrorDialog
  - gui.MainGui, 27
- showGpioConfWindow
  - gui.MainGui, 27
- SPEED\_FAST
  - configurator.GPIO.Speed, 73
- SPEED\_HIGH
  - configurator.GPIO.Speed, 73
- SPEED\_MAX\_VALUE
  - configurator.GPIO.Speed, 73
- SPEED\_MEDIUM
  - configurator.GPIO.Speed, 73
- SPEED\_NOT\_AVAILABLE
  - configurator.GPIO.Speed, 73
- STR\_DEFINITION
  - framework.Common, 16
- STR\_HEADER\_EXT
  - framework.Common, 17
- STR\_INCLUDE
  - framework.Common, 17
- STR\_INVALID
  - common.ErrorCode, 21
- STR\_MODULE\_GPIO
  - framework.Common, 17
- STR\_NAME
  - configurator.GPIO.CodeName, 12
  - configurator.GPIO.Mode, 38
  - configurator.GPIO.OutLevel, 40
  - configurator.GPIO.OutType, 42
  - configurator.GPIO.Pull, 71
  - configurator.GPIO.Speed, 73
- STR\_PROJ\_CONF\_FILE
  - configurator.ConfigurationFile, 18
- SW\_VERSION
  - common.Features, 23
- VERBOSE
  - common.Features, 23
- VERBOSE\_STR
  - common.Features, 23
- verbosePrint
  - common.Features, 22
- VERSION\_NAME
  - common.Features, 23
- VERSION\_STATUS
  - common.Features, 23
- writeXml
  - xmlCreator.ConfXmlWriter, 19
- xmlCreator, 8
- xmlCreator.ConfXmlWriter, 18
  - addPin, 19
  - ConfXmlWriter, 18
  - writeXml, 19
- XmlOpener
  - xmlParser.XmlOpener, 75
- xmlParser, 8
- xmlParser.TestMain, 74
  - main, 74
- xmlParser.XmlOpener, 75
  - getElementInfo, 75
  - getElementInfoFromDoc, 76
  - getParsedDoc, 76
  - OpenFile, 76
  - XmlOpener, 75