

Code\_generator

Generated by Doxygen 1.8.18



---

<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 microcontroller.Adc Class Reference	10
4.2.1 Constructor & Destructor Documentation	11
4.2.1.1 Adc()	11
4.2.2 Member Function Documentation	11
4.2.2.1 addChannel()	11
4.2.2.2 addClock()	11
4.2.2.3 addJustification()	11
4.2.2.4 addPrescaler()	12
4.2.2.5 addReference()	12

---

4.2.2.6 addResolution()	12
4.2.2.7 addSample()	13
4.2.2.8 getChannel()	13
4.2.2.9 getChannelNum()	13
4.2.2.10 getClock()	14
4.2.2.11 getClockNum()	14
4.2.2.12 getJustification()	14
4.2.2.13 getJustificationNum()	15
4.2.2.14 getName()	15
4.2.2.15 getPrescaler()	15
4.2.2.16 getPrescalerNum()	15
4.2.2.17 getReference()	16
4.2.2.18 getReferenceNum()	16
4.2.2.19 getResolution()	16
4.2.2.20 getResolutionNum()	17
4.2.2.21 getSample()	17
4.2.2.22 getSampleNum()	17
4.2.2.23 isValid()	18
4.2.2.24 setName()	18
4.3 configurator.ADC.AdcChannel Class Reference	18
4.3.1 Member Function Documentation	19
4.3.1.1 getCodeName()	19
4.3.1.2 getName()	19
4.3.1.3 getPinIndex()	19
4.3.1.4 getSelected()	20
4.3.1.5 isValid()	20
4.3.1.6 setCodeName()	20
4.3.1.7 setSelected()	20
4.3.2 Member Data Documentation	21
4.3.2.1 DF_SELECTED	21
4.4 configurator.AdcConf Class Reference	21
4.4.1 Constructor & Destructor Documentation	22
4.4.1.1 AdcConf()	22
4.4.2 Member Function Documentation	22
4.4.2.1 getChannel()	22
4.4.2.2 getChannelsNum()	23
4.4.2.3 getClock()	23
4.4.2.4 getCodeName()	23
4.4.2.5 getJustification()	24

4.4.2.6 getPrescaler()	24
4.4.2.7 getReference()	24
4.4.2.8 getResolution()	24
4.4.2.9 getSample()	25
4.4.2.10 getSelected()	25
4.4.2.11 setChannels()	25
4.4.2.12 setClock()	25
4.4.2.13 setCodeName()	26
4.4.2.14 setJustification()	26
4.4.2.15 setPrescaler()	26
4.4.2.16 setReference()	27
4.4.2.17 setResolution()	27
4.4.2.18 setSample()	27
4.4.2.19 setSelected()	27
4.4.3 Member Data Documentation	28
4.4.3.1 DF_SELECTED	28
4.5 gui.AdcConfWindow Class Reference	28
4.5.1 Constructor & Destructor Documentation	28
4.5.1.1 AdcConfWindow()	28
4.5.2 Member Function Documentation	29
4.5.2.1 main()	29
4.6 framework.AdcGenerator Class Reference	29
4.6.1 Member Function Documentation	30
4.6.1.1 getEIDefs()	30
4.6.1.2 getElements()	30
4.6.1.3 getIncludes()	30
4.7 configurator.GPIO.AltMode Enum Reference	31
4.7.1 Detailed Description	31
4.7.2 Member Function Documentation	31
4.7.2.1 getConfFromString()	31
4.7.3 Member Data Documentation	32
4.7.3.1 ALT_MODE_ANALOG	32
4.7.3.2 ALT_MODE_I2C	32
4.7.3.3 ALT_MODE_MAX_VALUE	32
4.7.3.4 ALT_MODE_NONE	32
4.7.3.5 ALT_MODE_SPI	32
4.7.3.6 ALT_MODE_UART	33
4.7.3.7 STR_NAME	33
4.8 framework.CodeGenerator Class Reference	33

4.8.1 Detailed Description . . . . .	33
4.8.2 Constructor & Destructor Documentation . . . . .	33
4.8.2.1 CodeGenerator() . . . . .	33
4.8.3 Member Function Documentation . . . . .	34
4.8.3.1 Generate() . . . . .	34
4.9 configurator.GPIO.CodeName Enum Reference . . . . .	34
4.9.1 Detailed Description . . . . .	34
4.9.2 Member Data Documentation . . . . .	35
4.9.2.1 CODE_NAME . . . . .	35
4.9.2.2 STR_NAME . . . . .	35
4.10 framework.Common Class Reference . . . . .	35
4.10.1 Detailed Description . . . . .	36
4.10.2 Member Function Documentation . . . . .	36
4.10.2.1 getCfgFileCPath() . . . . .	36
4.10.2.2 getCfgFileHPath() . . . . .	36
4.10.2.3 getCfgPath() . . . . .	37
4.10.2.4 getCommonCfgDefinitions() . . . . .	37
4.10.2.5 getCommonIncludes() . . . . .	38
4.10.2.6 getFrameworkCommonFilePath() . . . . .	38
4.10.2.7 getFrameworkIncludesFilePath() . . . . .	38
4.10.2.8 getInstallationFwkPath() . . . . .	39
4.10.2.9 getProjectFwkPath() . . . . .	39
4.10.2.10 setInstallationFwkPath() . . . . .	39
4.10.2.11 setProjectFwkPath() . . . . .	40
4.10.3 Member Data Documentation . . . . .	40
4.10.3.1 NL . . . . .	40
4.10.3.2 STR_DEFINITION . . . . .	40
4.10.3.3 STR_GEN_CODE_NOTICE_FOOTER . . . . .	40
4.10.3.4 STR_GEN_CODE_NOTICE_HEADER . . . . .	41
4.10.3.5 STR_HEADER_EXT . . . . .	41
4.10.3.6 STR_INCLUDE . . . . .	41
4.10.3.7 STR_MODULE_ADC . . . . .	41
4.10.3.8 STR_MODULE_GPIO . . . . .	41
4.10.3.9 STR_MODULE_UART . . . . .	41
4.11 configurator.ConfigurationFile Class Reference . . . . .	42
4.11.1 Detailed Description . . . . .	42
4.11.2 Member Data Documentation . . . . .	42
4.11.2.1 STR_PROJ_CONF_FILE . . . . .	42
4.12 xmlCreator.ConfXmlWriter Class Reference . . . . .	42

4.12.1 Detailed Description . . . . .	43
4.12.2 Constructor & Destructor Documentation . . . . .	43
4.12.2.1 ConfXmlWriter() . . . . .	43
4.12.3 Member Function Documentation . . . . .	43
4.12.3.1 addPin() . . . . .	43
4.12.3.2 writeXml() . . . . .	44
4.13 common.ErrorCode Enum Reference . . . . .	44
4.13.1 Detailed Description . . . . .	44
4.13.2 Member Data Documentation . . . . .	45
4.13.2.1 EX_ERROR . . . . .	45
4.13.2.2 FILE_CONF_ERROR . . . . .	45
4.13.2.3 FILE_READ_ERROR . . . . .	45
4.13.2.4 FILE_WRITE_ERROR . . . . .	45
4.13.2.5 INT_INVALID_INDEX . . . . .	45
4.13.2.6 NO_ERROR . . . . .	45
4.13.2.7 STR_INVALID . . . . .	46
4.14 common.Features Class Reference . . . . .	46
4.14.1 Detailed Description . . . . .	46
4.14.2 Member Function Documentation . . . . .	46
4.14.2.1 debugPrint() . . . . .	46
4.14.2.2 verbosePrint() . . . . .	47
4.14.3 Member Data Documentation . . . . .	47
4.14.3.1 DEBUG . . . . .	47
4.14.3.2 DEBUG_STR . . . . .	47
4.14.3.3 SW_VERSION . . . . .	47
4.14.3.4 VERBOSE . . . . .	48
4.14.3.5 VERBOSE_STR . . . . .	48
4.14.3.6 VERSION_NAME . . . . .	48
4.14.3.7 VERSION_STATUS . . . . .	48
4.15 common.GeneralSettings Class Reference . . . . .	48
4.15.1 Member Data Documentation . . . . .	49
4.15.1.1 logFilePath . . . . .	49
4.16 gui.GeneralSettingsWindow Class Reference . . . . .	49
4.16.1 Constructor & Destructor Documentation . . . . .	49
4.16.1.1 GeneralSettingsWindow() . . . . .	49
4.16.2 Member Function Documentation . . . . .	49
4.16.2.1 main() . . . . .	50
4.17 gui.GpioConfWindow Class Reference . . . . .	50
4.17.1 Detailed Description . . . . .	50

---

4.17.2 Constructor & Destructor Documentation	50
4.17.2.1 GpioConfWindow()	50
4.17.3 Member Function Documentation	51
4.17.3.1 main()	51
4.18 gui.MainGui Class Reference	51
4.18.1 Detailed Description	52
4.18.2 Member Function Documentation	52
4.18.2.1 generateCode()	52
4.18.2.2 loadProjectFile()	52
4.18.2.3 main()	53
4.18.2.4 saveGeneralSettings()	53
4.18.2.5 saveProjectPreferences()	53
4.18.2.6 saveUc()	54
4.18.2.7 setNewUC()	54
4.18.2.8 showAboutWindow()	54
4.18.2.9 showAdcConfWindow()	54
4.18.2.10 showErrorDialog()	54
4.18.2.11 showGeneralSettingsWindow()	55
4.18.2.12 showGpioConfWindow()	55
4.18.2.13 showProjectPreferencesWindow()	55
4.18.2.14 showUartConfWindow()	55
4.18.3 Member Data Documentation	55
4.18.3.1 ProjectFile	55
4.18.3.2 ProjectPath	55
4.19 gui.MainWindow Class Reference	56
4.19.1 Detailed Description	56
4.19.2 Constructor & Destructor Documentation	56
4.19.2.1 MainWindow()	56
4.19.3 Member Function Documentation	56
4.19.3.1 main()	56
4.19.3.2 OpenFileChooser()	57
4.19.3.3 setProjectInformation()	57
4.19.3.4 setVisible()	58
4.19.4 Member Data Documentation	58
4.19.4.1 FrmCodeGenerator	58
4.20 gui.Messages Class Reference	58
4.20.1 Detailed Description	58
4.20.2 Member Function Documentation	58
4.20.2.1 getString()	58

---



4.21 microcontroller.Microcontroller Class Reference . . . . .	59
4.21.1 Detailed Description . . . . .	60
4.21.2 Constructor & Destructor Documentation . . . . .	60
4.21.2.1 Microcontroller() . . . . .	60
4.21.3 Member Function Documentation . . . . .	60
4.21.3.1 getConfiguredPin() . . . . .	60
4.21.3.2 getPin() . . . . .	61
4.21.3.3 getUc_adcNum() . . . . .	61
4.21.3.4 getUc_gpioNum() . . . . .	61
4.21.3.5 getUc_manufacturer() . . . . .	62
4.21.3.6 getUc_model() . . . . .	62
4.21.3.7 getUc_pinNum() . . . . .	62
4.21.3.8 getUc_portNum() . . . . .	62
4.21.3.9 getUc_selectedAdcsNum() . . . . .	63
4.21.3.10 getUc_selectedPinsNum() . . . . .	63
4.21.3.11 getUc_selectedUartsNum() . . . . .	63
4.21.3.12 getUc_uartNum() . . . . .	63
4.21.3.13 isValid() . . . . .	64
4.21.3.14 loadAdcChannelsConf() . . . . .	64
4.21.3.15 loadPinsConf() . . . . .	64
4.21.3.16 processDocument() . . . . .	65
4.21.4 Member Data Documentation . . . . .	65
4.21.4.1 AdcCfg . . . . .	65
4.21.4.2 Adcs . . . . .	65
4.21.4.3 Definitions_Adc . . . . .	65
4.21.4.4 Definitions_Common . . . . .	65
4.21.4.5 Definitions_Gpio . . . . .	65
4.21.4.6 Definitions_Uart . . . . .	66
4.21.4.7 GpioCfgPin . . . . .	66
4.21.4.8 Includes_Adc . . . . .	66
4.21.4.9 Includes_Common . . . . .	66
4.21.4.10 Includes_Gpio . . . . .	66
4.21.4.11 Includes_Uart . . . . .	66
4.21.4.12 MAX_NUMBER_OF_ADCS . . . . .	66
4.21.4.13 MAX_NUMBER_OF_PINS_PER_PORT . . . . .	67
4.21.4.14 MAX_NUMBER_OF_UARTS . . . . .	67
4.21.4.15 Ports . . . . .	67
4.21.4.16 UartCfg . . . . .	67
4.21.4.17 Uarts . . . . .	67

4.22 configurator.GPIO.Mode Enum Reference . . . . .	67
4.22.1 Detailed Description . . . . .	68
4.22.2 Member Function Documentation . . . . .	68
4.22.2.1 getConfFromString() . . . . .	68
4.22.3 Member Data Documentation . . . . .	69
4.22.3.1 MODE_ALTERNATE_FUNCTION . . . . .	69
4.22.3.2 MODE_INPUT . . . . .	69
4.22.3.3 MODE_MAX_VALUE . . . . .	69
4.22.3.4 MODE_OUTPUT . . . . .	69
4.22.3.5 STR_NAME . . . . .	69
4.23 configurator.GPIO.OutLevel Enum Reference . . . . .	70
4.23.1 Detailed Description . . . . .	70
4.23.2 Member Function Documentation . . . . .	70
4.23.2.1 getConfFromString() . . . . .	70
4.23.3 Member Data Documentation . . . . .	71
4.23.3.1 HIGH . . . . .	71
4.23.3.2 LOW . . . . .	71
4.23.3.3 MAX_VALUE . . . . .	71
4.23.3.4 STR_NAME . . . . .	71
4.24 configurator.GPIO.OutType Enum Reference . . . . .	71
4.24.1 Detailed Description . . . . .	72
4.24.2 Member Function Documentation . . . . .	72
4.24.2.1 getConfFromString() . . . . .	72
4.24.3 Member Data Documentation . . . . .	73
4.24.3.1 OTYPE_MAX_VALUE . . . . .	73
4.24.3.2 OTYPE_NOT_AVAILABLE . . . . .	73
4.24.3.3 OTYPE_OPEN_DRAIN . . . . .	73
4.24.3.4 OTYPE_PUSH_PULL . . . . .	73
4.24.3.5 STR_NAME . . . . .	73
4.25 microcontroller.Pin Class Reference . . . . .	74
4.25.1 Detailed Description . . . . .	75
4.25.2 Constructor & Destructor Documentation . . . . .	76
4.25.2.1 Pin() . . . . .	76
4.25.3 Member Function Documentation . . . . .	76
4.25.3.1 getAdc() . . . . .	76
4.25.3.2 getAdcChannel() . . . . .	76
4.25.3.3 getClock() . . . . .	77
4.25.3.4 getFeat_adc() . . . . .	77
4.25.3.5 getFeat_clock() . . . . .	77

---

4.25.3.6 getFeat_i2c()	77
4.25.3.7 getFeat_int()	78
4.25.3.8 getFeat_reset()	78
4.25.3.9 getFeat_spi()	78
4.25.3.10 getFeat_timer()	78
4.25.3.11 getFeat_uart()	79
4.25.3.12 getFunc_gnd()	79
4.25.3.13 getFunc_gpio()	79
4.25.3.14 getFunc_misc()	79
4.25.3.15 getFunc_reset()	80
4.25.3.16 getFunc_vcc()	80
4.25.3.17 getI2c()	80
4.25.3.18 getInt()	80
4.25.3.19 getName()	81
4.25.3.20 getNumber()	81
4.25.3.21 getPort()	81
4.25.3.22 getPortPin()	81
4.25.3.23 getReset()	82
4.25.3.24 getSpi()	82
4.25.3.25 getTimer()	82
4.25.3.26 getUart()	82
4.25.3.27 isValid()	83
4.25.3.28 setAdc()	83
4.25.3.29 setClock()	83
4.25.3.30 setFeat_adc()	83
4.25.3.31 setFeat_clock()	84
4.25.3.32 setFeat_i2c()	84
4.25.3.33 setFeat_int()	84
4.25.3.34 setFeat_reset()	85
4.25.3.35 setFeat_spi()	85
4.25.3.36 setFeat_timer()	85
4.25.3.37 setFeat_uart()	85
4.25.3.38 setFunc_gnd()	86
4.25.3.39 setFunc_gpio()	86
4.25.3.40 setFunc_misc()	86
4.25.3.41 setFunc_reset()	87
4.25.3.42 setFunc_vcc()	87
4.25.3.43 setI2c()	87
4.25.3.44 setInt()	88

---

4.25.3.45 setName()	88
4.25.3.46 setNumber()	88
4.25.3.47 setPort()	88
4.25.3.48 setPortPin()	89
4.25.3.49 setReset()	89
4.25.3.50 setSpi()	89
4.25.3.51 setTimer()	90
4.25.3.52 setUart()	90
4.25.4 Member Data Documentation	90
4.25.4.1 DEF_FEATURE	90
4.25.4.2 DEF_FEATURE_AV	91
4.25.4.3 DEF_FUNCTION	91
4.25.4.4 DEF_NAME	91
4.25.4.5 DEF_NUMBER	91
4.25.4.6 DEF_PORT	91
4.25.4.7 DISABLE	91
4.25.4.8 ENABLE	91
4.26 configurator.PinConf Class Reference	92
4.26.1 Detailed Description	92
4.26.2 Constructor & Destructor Documentation	93
4.26.2.1 PinConf()	93
4.26.3 Member Function Documentation	93
4.26.3.1 getAltMode()	93
4.26.3.2 getCodeName()	93
4.26.3.3 getMode()	94
4.26.3.4 getOutLevel()	94
4.26.3.5 getOutType()	94
4.26.3.6 getPinName()	94
4.26.3.7 getPort()	95
4.26.3.8 getPortPin()	95
4.26.3.9 getPull()	95
4.26.3.10 getSelected()	95
4.26.3.11 getSpeed()	96
4.26.3.12 isAv_Adc()	96
4.26.3.13 isAv_altFunc()	96
4.26.3.14 isAv_I2c()	96
4.26.3.15 isAv_Spi()	97
4.26.3.16 isAv_Uart()	97
4.26.3.17 isValid()	97

---

4.26.3.18 setAltMode()	97
4.26.3.19 setCodeName()	98
4.26.3.20 setMode()	98
4.26.3.21 setOutLevel()	98
4.26.3.22 setOutType()	99
4.26.3.23 setPull()	99
4.26.3.24 setSelected()	99
4.26.3.25 setSpeed()	99
4.26.4 Member Data Documentation	100
4.26.4.1 DF_ALT_MODE	100
4.26.4.2 DF_CODE_NAME	100
4.26.4.3 DF_MODE	100
4.26.4.4 DF_OUT_LEVEL	100
4.26.4.5 DF_OUTTYPE	100
4.26.4.6 DF_PULL	101
4.26.4.7 DF_SELECTED	101
4.26.4.8 DF_SPEED	101
4.27 projectConfiguration.ProjectSettings Class Reference	101
4.27.1 Detailed Description	101
4.27.2 Constructor & Destructor Documentation	102
4.27.2.1 ProjectSettings()	102
4.27.3 Member Function Documentation	102
4.27.3.1 getConfFile()	102
4.27.3.2 getFrameworkPath()	102
4.27.3.3 getProjectName()	103
4.27.3.4 getUcFile()	103
4.27.3.5 openProjectFile()	103
4.27.3.6 processDocument()	103
4.27.3.7 setFrameworkPath()	104
4.28 gui.ProjectSettingsWindow Class Reference	104
4.28.1 Constructor & Destructor Documentation	104
4.28.1.1 ProjectSettingsWindow()	104
4.28.2 Member Function Documentation	105
4.28.2.1 main()	105
4.29 configurator.GPIO.Pull Enum Reference	105
4.29.1 Detailed Description	105
4.29.2 Member Function Documentation	105
4.29.2.1 getConfFromString()	105
4.29.3 Member Data Documentation	106

4.29.3.1 PULL_DOWN	106
4.29.3.2 PULL_MAX_VALUE	106
4.29.3.3 PULL_NOT_AVAILABLE	106
4.29.3.4 PULL_UP	106
4.29.3.5 STR_NAME	106
4.30 configurator.Selected Enum Reference	107
4.30.1 Detailed Description	107
4.30.2 Member Function Documentation	107
4.30.2.1 getBoolean()	107
4.30.2.2 getConfFromBoolean()	107
4.30.2.3 getConfFromString()	108
4.30.3 Member Data Documentation	108
4.30.3.1 NOT	108
4.30.3.2 STR_NAME	108
4.30.3.3 YES	109
4.31 configurator.GPIO.Speed Enum Reference	109
4.31.1 Detailed Description	109
4.31.2 Member Function Documentation	109
4.31.2.1 getConfFromString()	109
4.31.3 Member Data Documentation	110
4.31.3.1 SPEED_FAST	110
4.31.3.2 SPEED_HIGH	110
4.31.3.3 SPEED_MAX_VALUE	110
4.31.3.4 SPEED_MEDIUM	110
4.31.3.5 SPEED_NOT_AVAILABLE	110
4.31.3.6 STR_NAME	111
4.32 microcontroller.Uart Class Reference	111
4.32.1 Constructor & Destructor Documentation	111
4.32.1.1 Uart()	111
4.32.2 Member Function Documentation	112
4.32.2.1 addBaudRate()	112
4.32.2.2 addClock()	112
4.32.2.3 addDataBits()	112
4.32.2.4 addParity()	113
4.32.2.5 addPrescaler()	113
4.32.2.6 addStopBits()	113
4.32.2.7 getBaudRate()	113
4.32.2.8 getBaudRateNum()	114
4.32.2.9 getClock()	114

4.32.2.10 getClockNum()	114
4.32.2.11 getDataBits()	115
4.32.2.12 getDataBitsNum()	115
4.32.2.13 getName()	115
4.32.2.14 getParity()	115
4.32.2.15 getParityNum()	116
4.32.2.16 getPrescaler()	116
4.32.2.17 getPrescalerNum()	116
4.32.2.18 getStopBits()	117
4.32.2.19 getStopBitsNum()	117
4.32.2.20 isValid()	117
4.32.2.21 setName()	117
4.33 configurator.UartConf Class Reference	118
4.33.1 Constructor & Destructor Documentation	119
4.33.1.1 UartConf()	119
4.33.2 Member Function Documentation	119
4.33.2.1 getBaudRate()	119
4.33.2.2 getClock()	119
4.33.2.3 getCodeName()	120
4.33.2.4 getDataBits()	120
4.33.2.5 getParity()	120
4.33.2.6 getPrescaler()	120
4.33.2.7 getSelected()	121
4.33.2.8 getStopBits()	121
4.33.2.9 setBaudRate()	121
4.33.2.10 setClock()	121
4.33.2.11 setCodeName()	122
4.33.2.12 setDataBits()	122
4.33.2.13 setParity()	122
4.33.2.14 setPrescaler()	123
4.33.2.15 setSelected()	123
4.33.2.16 setStopBits()	123
4.33.3 Member Data Documentation	123
4.33.3.1 DF_SELECTED	124
4.34 gui.UartConfWindow Class Reference	124
4.34.1 Constructor & Destructor Documentation	124
4.34.1.1 UartConfWindow()	124
4.34.2 Member Function Documentation	124
4.34.2.1 main()	125

4.35 framework.UartGenerator Class Reference . . . . .	125
4.35.1 Member Function Documentation . . . . .	125
4.35.1.1 getElDefs() . . . . .	125
4.35.1.2 getElements() . . . . .	126
4.35.1.3 getIncludes() . . . . .	126
4.36 xmlParser.XmlOpener Class Reference . . . . .	127
4.36.1 Detailed Description . . . . .	127
4.36.2 Constructor & Destructor Documentation . . . . .	127
4.36.2.1 XmlOpener() . . . . .	127
4.36.3 Member Function Documentation . . . . .	127
4.36.3.1 getElementInfo() . . . . .	127
4.36.3.2 getElementInfoFromDoc() . . . . .	128
4.36.3.3 getParsedDoc() . . . . .	128
4.36.3.4 OpenFile() . . . . .	128

<b>Index</b>	<b>131</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:

gui.AboutWindow	9
microcontroller.Adc	10
configurator.ADC.AdcChannel	18
configurator.AdcConf	21
gui.AdcConfWindow	28
framework.AdcGenerator	29
configurator.GPIO.AltMode	31
framework.CodeGenerator	33
configurator.GPIO.CodeName	34
framework.Common	35
configurator.ConfigurationFile	42
xmlCreator.ConfXmlWriter	42
common.ErrorCode	44
common.Features	46
common.GeneralSettings	48
gui.GeneralSettingsWindow	49
gui.GpioConfWindow	50
gui.MainGui	51
gui.MainWindow	56
gui.Messages	58
microcontroller.Microcontroller	59
configurator.GPIO.Mode	67
configurator.GPIO.OutLevel	70
configurator.GPIO.OutType	71
microcontroller.Pin	74
configurator.PinConf	92
projectConfiguration.ProjectSettings	101
gui.ProjectSettingsWindow	104
configurator.GPIO.Pull	105
configurator.Selected	107
configurator.GPIO.Speed	109

<a href="#">microcontroller.Uart</a>	<a href="#">111</a>
<a href="#">configurator.UartConf</a>	<a href="#">118</a>
<a href="#">gui.UartConfWindow</a>	<a href="#">124</a>
<a href="#">framework.UartGenerator</a>	<a href="#">125</a>
<a href="#">xmlParser.XmlOpener</a>	<a href="#">127</a>

## Chapter 3

# Namespace Documentation

### 3.1 Package common

#### Classes

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

#### 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 [AdcConf](#)
- class [ConfigurationFile](#)
- class [PinConf](#)
- enum [Selected](#)
- class [UartConf](#)

### 3.2.1 Detailed Description

Configuration classes

Author

Miguel Diaz

Version

0.1

## 3.3 Package framework

### Classes

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

### 3.3.1 Detailed Description

Framework information

Author

H112943

Version

0.1

## 3.4 Package gui

### Classes

- class [AboutWindow](#)
- class [AdcConfWindow](#)
- class [GeneralSettingsWindow](#)
- class [GpioConfWindow](#)
- class [MainGui](#)
- class [MainWindow](#)
- class [Messages](#)
- class [ProjectSettingsWindow](#)
- class [UartConfWindow](#)

### 3.4.1 Detailed Description

Author

Miguel Diaz

Version

0.1

## 3.5 Package microcontroller

### Classes

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

### 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 [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 microcontroller.Adc Class Reference

### Public Member Functions

- [Adc](#) ()
- void [setName](#) (String name)
- String [getName](#) ()
- void [addSample](#) (String sample)
- int [getSampleNum](#) ()
- String [getSample](#) (int index)
- void [addClock](#) (String clock)
- int [getClockNum](#) ()
- String [getClock](#) (int index)
- void [addJustification](#) (String justification)
- int [getJustificationNum](#) ()
- String [getJustification](#) (int index)
- void [addPrescaler](#) (String prescaler)
- int [getPrescalerNum](#) ()
- String [getPrescaler](#) (int index)
- void [addResolution](#) (String resolution)
- int [getResolutionNum](#) ()
- String [getResolution](#) (int index)
- void [addReference](#) (String reference)
- int [getReferenceNum](#) ()
- String [getReference](#) (int index)
- void [addChannel](#) ([AdcChannel](#) channel)
- int [getChannelNum](#) ()
- [AdcChannel](#) [getChannel](#) (int index)
- boolean [isValid](#) ()

## 4.2.1 Constructor & Destructor Documentation

### 4.2.1.1 Adc()

```
microcontroller.Adc.Adc ( )
```

ADC instance constructor

## 4.2.2 Member Function Documentation

### 4.2.2.1 addChannel()

```
void microcontroller.Adc.addChannel (
    AdcChannel channel )
```

Add ADC's channel

#### Parameters

<i>channel</i>	Channel
----------------	---------

### 4.2.2.2 addClock()

```
void microcontroller.Adc.addClock (
    String clock )
```

Add ADC supported clock source

#### Parameters

<i>clock</i>	Clock source
--------------	--------------

### 4.2.2.3 addJustification()

```
void microcontroller.Adc.addJustification (
```

```
String justification )
```

Add ADC's supported bits justification

#### Parameters

<i>justification</i>	Bits justification
----------------------	--------------------

#### 4.2.2.4 addPrescaler()

```
void microcontroller.Adc.addPrescaler (
    String prescaler )
```

Add ADC's supported clock prescaler

#### Parameters

<i>prescaler</i>	Clock prescaler
------------------	-----------------

#### 4.2.2.5 addReference()

```
void microcontroller.Adc.addReference (
    String reference )
```

Add ADC's supported voltage references

#### Parameters

<i>reference</i>	Voltage references
------------------	--------------------

#### 4.2.2.6 addResolution()

```
void microcontroller.Adc.addResolution (
    String resolution )
```

Add ADC's supported bits resolution

## Parameters

<i>resolution</i>	bits resolution
-------------------	-----------------

**4.2.2.7 addSample()**

```
void microcontroller.Adc.addSample (  
    String sample )
```

Add ADC supported samples

## Parameters

<i>sample</i>	Sample definition
---------------	-------------------

**4.2.2.8 getChannel()**

```
AdcChannel microcontroller.Adc.getChannel (  
    int index )
```

Get ADC's channel

## Parameters

<i>index</i>	Channel index
--------------	---------------

## Returns

Channel

**4.2.2.9 getChannelNum()**

```
int microcontroller.Adc.getChannelNum ( )
```

Get ADC's number of channels

## Returns

Number of channels

#### 4.2.2.10 **getClock()**

```
String microcontroller.Adc.getClock (
    int index )
```

Get ADC's clock source

##### Parameters

<i>index</i>	Clock source index
--------------	--------------------

##### Returns

Clock source

#### 4.2.2.11 **getClockNum()**

```
int microcontroller.Adc.getClockNum ( )
```

Get ADCs number of clock sources

##### Returns

Number of clock sources

#### 4.2.2.12 **getJustification()**

```
String microcontroller.Adc.getJustification (
    int index )
```

Get ADC's bits justification

##### Parameters

<i>index</i>	bits justification index
--------------	--------------------------

##### Returns

Bits justification

#### 4.2.2.13 getJustificationNum()

```
int microcontroller.Adc.getJustificationNum ( )
```

Get ADC's number of supported justifications

##### Returns

Number of supported justifications

#### 4.2.2.14 getName()

```
String microcontroller.Adc.getName ( )
```

Get ADCs instance name

##### Returns

Instance name

#### 4.2.2.15 getPrescaler()

```
String microcontroller.Adc.getPrescaler (
    int index )
```

Get ADC's clock prescaler

##### Parameters

<i>index</i>	Clock prescaler index
--------------	-----------------------

##### Returns

Clock prescaler

#### 4.2.2.16 getPrescalerNum()

```
int microcontroller.Adc.getPrescalerNum ( )
```

Get ADC's number of supported prescalers

**Returns**

Number of supported prescalers

**4.2.2.17 getReference()**

```
String microcontroller.Adc.getReference (
    int index )
```

Get ADC's voltage references

**Parameters**

<i>index</i>	Voltage references index
--------------	--------------------------

**Returns**

Voltage references

**4.2.2.18 getReferenceNum()**

```
int microcontroller.Adc.getReferenceNum ( )
```

Get ADC's number of supported voltage references

**Returns**

Number of supported voltage references

**4.2.2.19 getResolution()**

```
String microcontroller.Adc.getResolution (
    int index )
```

Get ADC's bits resolution

**Parameters**

<i>index</i>	bits resolution index
--------------	-----------------------



**Returns**

bits resolution

**4.2.2.20 getResolutionNum()**

```
int microcontroller.Adc.getResolutionNum ( )
```

Get ADC's number of supported bits resolutions

**Returns**

Number of supported bits resolutions

**4.2.2.21 getSample()**

```
String microcontroller.Adc.getSample (
    int index )
```

Get ADC's Sample definition

**Parameters**

<i>index</i>	sample definition index
--------------	-------------------------

**Returns**

Sample definition

**4.2.2.22 getSampleNum()**

```
int microcontroller.Adc.getSampleNum ( )
```

Get ADCs number of samples definitions

**Returns**

Number of samples definitions

#### 4.2.2.23 isValid()

```
boolean microcontroller.Adc.isValid ( )
```

Check validity of ADC

##### Returns

True if valid

#### 4.2.2.24 setName()

```
void microcontroller.Adc.setName (
    String name )
```

Set ADCs instance name

##### Parameters

<i>name</i>	Instance name
-------------	---------------

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

- src/microcontroller/Adc.java

## 4.3 configurator.ADC.AdcChannel Class Reference

### Public Member Functions

- **AdcChannel** (String name, int pinIndex)
- String [getName](#) ()
- [Selected](#) [getSelected](#) ()
- void [setSelected](#) ([Selected](#) selection)
- String [getCodeName](#) ()
- void [setCodeName](#) (String codeName)
- int [getPinIndex](#) ()
- boolean [isValid](#) ()

### Static Public Attributes

- static final String **INVALID\_NAME** = [ErrorCode.STR\\_INVALID](#)
- static final int **INVALID\_INDEX** = [ErrorCode.INT\\_INVALID\\_INDEX](#)
- static final String **STR\_NAME** = "name"
- static final String **STR\_CODE\_NAME** = "codeName"
- static final String **STR\_PIN\_INDEX** = "pinIndex"
- static final [Selected](#) **DF\_SELECTED** = [configurator.Selected.NOT](#)

### 4.3.1 Member Function Documentation

#### 4.3.1.1 getCodeName()

```
String configurator.ADC.AdcChannel.getCodeName ( )
```

Get ADC channel's code name

##### Returns

ADC channel's code name

#### 4.3.1.2 getName()

```
String configurator.ADC.AdcChannel.getName ( )
```

Get ADC channel's name

##### Returns

ADC channel's name

#### 4.3.1.3 getPinIndex()

```
int configurator.ADC.AdcChannel.getPinIndex ( )
```

Get ADC channel's pin index

##### Returns

#### 4.3.1.4 `getSelected()`

```
Selected configurator.ADC.AdcChannel.getSelected ( )
```

Get channel's selection

##### Returns

Channel's selection

#### 4.3.1.5 `isValid()`

```
boolean configurator.ADC.AdcChannel.isValid ( )
```

Check channel validity

##### Returns

True if valid

#### 4.3.1.6 `setCodeName()`

```
void configurator.ADC.AdcChannel.setCodeName (
    String codeName )
```

Set ADC channel's code name

##### Parameters

<i>codeName</i>	ADC channel's code name
-----------------	-------------------------

#### 4.3.1.7 `setSelected()`

```
void configurator.ADC.AdcChannel.setSelected (
    Selected selection )
```

Set channel's selection

## Parameters

<i>selection</i>	Channel's selection
------------------	---------------------

## 4.3.2 Member Data Documentation

### 4.3.2.1 DF\_SELECTED

```
final Selected configurator.ADC.AdcChannel.DF_SELECTED = configurator.Selected.NOT [static]
```

Default Pin's selection

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

- src/configurator/ADC/AdcChannel.java

## 4.4 configurator.AdcConf Class Reference

### Public Member Functions

- [AdcConf](#) ([Adc](#) adc)
- [Selected](#) [getSelected](#) ()
- void [setSelected](#) ([Selected](#) selection)
- String [getCodeName](#) ()
- void [setCodeName](#) (String codeName)
- String [getSample](#) ()
- void [setSample](#) (String sample)
- String [getClock](#) ()
- void [setClock](#) (String clock)
- String [getJustification](#) ()
- void [setJustification](#) (String justification)
- String [getPrescaler](#) ()
- void [setPrescaler](#) (String prescaler)
- String [getResolution](#) ()
- void [setResolution](#) (String resolution)
- String [getReference](#) ()
- void [setReference](#) (String reference)
- void [setChannels](#) ([Adc](#) adc)
- int [getChannelsNum](#) ()
- [AdcChannel](#) [getChannel](#) (int index)
- int [getChannelIndexFromName](#) (String name)

## Public Attributes

- [Adc](#) **AdcFeatures**

## Static Public Attributes

- static final [Selected](#) **DF\_SELECTED** = [Selected](#).NOT
- static final String **STR\_NAME** = "name"
- static final String **STR\_CODE\_NAME** = "codeName"
- static final String **STR\_SAMPLE** = "sample"
- static final String **STR\_CLOCK** = "clock"
- static final String **STR\_JUSTIFICATION** = "justification"
- static final String **STR\_PRESCALER** = "prescaler"
- static final String **STR\_RESOLUTION** = "resolution"
- static final String **STR\_REFERENCE** = "reference"
- static final String **STR\_CHANNEL** = "adcChannel"

## 4.4.1 Constructor & Destructor Documentation

### 4.4.1.1 AdcConf()

```
configurator.AdcConf.AdcConf (
    Adc adc )
```

ADC configuration constructor

#### Parameters

<i>adc</i>	ADC instance
------------	--------------

## 4.4.2 Member Function Documentation

### 4.4.2.1 getChannel()

```
AdcChannel configurator.AdcConf.getChannel (
    int index )
```

Get ADC channel

**Parameters**

<i>index</i>	ADC channel index
--------------	-------------------

**Returns**

Channel

**4.4.2.2 getChannelsNum()**

```
int configurator.AdcConf.getChannelsNum ( )
```

Get the total of channels in the ADC

**Returns**

Total of channels in the ADC

**4.4.2.3 getClock()**

```
String configurator.AdcConf.getClock ( )
```

Get ADC's configured clock

**Returns**

ADC's configured clock

**4.4.2.4 getCodeName()**

```
String configurator.AdcConf.getCodeName ( )
```

Get ADC's code name

**Returns**

ADC's code name

#### 4.4.2.5 getJustification()

```
String configurator.AdcConf.getJustification ( )
```

Get ADC's configured justification

##### Returns

ADC's configured justification

#### 4.4.2.6 getPrescaler()

```
String configurator.AdcConf.getPrescaler ( )
```

Get ADC's prescaler

##### Returns

ADC's prescaler

#### 4.4.2.7 getReference()

```
String configurator.AdcConf.getReference ( )
```

Get ADC's configured reference

##### Returns

ADC's configured reference

#### 4.4.2.8 getResolution()

```
String configurator.AdcConf.getResolution ( )
```

Get ADC's configured resolution

##### Returns

ADC's configured resolution



#### 4.4.2.9 getSample()

```
String configurator.AdcConf.getSample ( )
```

Get ADC's configured samples

##### Returns

ADC's configured samples

#### 4.4.2.10 getSelected()

```
Selected configurator.AdcConf.getSelected ( )
```

Get the ADC's selection

##### Returns

Selection

#### 4.4.2.11 setChannels()

```
void configurator.AdcConf.setChannels (
    Adc adc )
```

Set ADC channels

##### Parameters

<i>adc</i>	ADC instance
------------	--------------

#### 4.4.2.12 setClock()

```
void configurator.AdcConf.setClock (
    String clock )
```

Set ADC's configured clock

**Parameters**

<i>clock</i>	ADC's configured clock
--------------	------------------------

**4.4.2.13 setCodeName()**

```
void configurator.AdcConf.setCodeName (
    String codeName )
```

Set Get ADC's code name

**Parameters**

<i>codeName</i>	ADC's code name
-----------------	-----------------

**4.4.2.14 setJustification()**

```
void configurator.AdcConf.setJustification (
    String justification )
```

Set ADC's configured justification

**Parameters**

<i>justification</i>	ADC's configured justification
----------------------	--------------------------------

**4.4.2.15 setPrescaler()**

```
void configurator.AdcConf.setPrescaler (
    String prescaler )
```

Set ADC's prescaler

**Parameters**

<i>prescaler</i>	ADC's prescaler
------------------	-----------------

#### 4.4.2.16 setReference()

```
void configurator.AdcConf.setReference (
    String reference )
```

Set ADC's configured reference

##### Parameters

<i>reference</i>	ADC's configured reference
------------------	----------------------------

#### 4.4.2.17 setResolution()

```
void configurator.AdcConf.setResolution (
    String resolution )
```

Set ADC's configured resolution

##### Parameters

<i>resolution</i>	ADC's configured resolution
-------------------	-----------------------------

#### 4.4.2.18 setSample()

```
void configurator.AdcConf.setSample (
    String sample )
```

Get ADC's configured samples

##### Parameters

<i>sample</i>	ADC's configured samples
---------------	--------------------------

#### 4.4.2.19 setSelected()

```
void configurator.AdcConf.setSelected (
```

```
Selected selection )
```

Set the ADC's selection

Parameters

<i>selection</i>	Selection
------------------	-----------

### 4.4.3 Member Data Documentation

#### 4.4.3.1 DF\_SELECTED

```
final Selected configurator.AdcConf.DF_SELECTED = Selected.NOT [static]
```

Default Pin's selection

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

- src/configurator/AdcConf.java

## 4.5 gui.AdcConfWindow Class Reference

### Public Member Functions

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

### Static Public Member Functions

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

### 4.5.1 Constructor & Destructor Documentation

#### 4.5.1.1 AdcConfWindow()

```
gui.AdcConfWindow.AdcConfWindow (
    Microcontroller uCtrl )
```

Create the application.

## Parameters

<i>uCtrl</i>	Microcontroller
--------------	-----------------

## 4.5.2 Member Function Documentation

### 4.5.2.1 main()

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

Launch the application.

## Parameters

<i>args</i>	General arguments
-------------	-------------------

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

- src/gui/AdcConfWindow.java

## 4.6 framework.AdcGenerator Class Reference

### Static Public Member Functions

- static String **getCfgArray** ([Microcontroller](#) uC)
- static String **getElDefs** ([Microcontroller](#) uC)
- static String **getElements** ([Microcontroller](#) uC)
- static String **getIncludes** ([Microcontroller](#) uC)
- static String **getCfgDefinitions** ([Microcontroller](#) uC)

### Static Public Attributes

- static final String **STR\_TKN\_CFG\_ARRAY** = "FWK\_ADC\_CFG\_ARRAY"
- static final String **STR\_TKN\_ELEMENTS** = "FWK\_ADC\_ELEMENTS"
- static final String **STR\_TKN\_INC** = "FWK\_ADC\_INCLUDES"
- static final String **STR\_TKN\_CFG\_DEFS** = "FWK\_ADC\_CFG\_DEFINITIONS"
- static final String **STR\_TKN\_EL\_DEFS** = "FWK\_ADC\_ELEMENTS\_DEFINITIONS"

## 4.6.1 Member Function Documentation

### 4.6.1.1 getElDefs()

```
static String framework.AdcGenerator.getElDefs (  
    Microcontroller uC ) [static]
```

#### Parameters

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

#### Returns

Elements definitions as String

### 4.6.1.2 getElements()

```
static String framework.AdcGenerator.getElements (  
    Microcontroller uC ) [static]
```

#### Parameters

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

#### Returns

Elements list as String

### 4.6.1.3 getIncludes()

```
static String framework.AdcGenerator.getIncludes (  
    Microcontroller uC ) [static]
```

#### Parameters

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

#### Returns

Headers needed for GPIO module

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

- `src/framework/AdcGenerator.java`

## 4.7 configurator.GPIO.AltMode Enum Reference

### Static Public Member Functions

- static [AltMode](#) `getConfigFromString` (String conf)

### Public Attributes

- [ALT\\_MODE\\_ANALOG](#)
- [ALT\\_MODE\\_UART](#)
- [ALT\\_MODE\\_I2C](#)
- [ALT\\_MODE\\_SPI](#)
- [ALT\\_MODE\\_NONE](#)
- [ALT\\_MODE\\_MAX\\_VALUE](#)

### Static Public Attributes

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

### 4.7.1 Detailed Description

GPIO modes

#### Author

Miguel Diaz

#### Version

0.1

### 4.7.2 Member Function Documentation

#### 4.7.2.1 `getConfigFromString()`

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

Get the corresponding mode from its name as String

**Parameters**

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

**Returns**

[Mode](#)

### 4.7.3 Member Data Documentation

#### 4.7.3.1 ALT\_MODE\_ANALOG

```
configurator.GPIO.AltMode.ALT_MODE_ANALOG
```

Analog

#### 4.7.3.2 ALT\_MODE\_I2C

```
configurator.GPIO.AltMode.ALT_MODE_I2C
```

I2C

#### 4.7.3.3 ALT\_MODE\_MAX\_VALUE

```
configurator.GPIO.AltMode.ALT_MODE_MAX_VALUE
```

Maximum value for [Mode](#) enum

#### 4.7.3.4 ALT\_MODE\_NONE

```
configurator.GPIO.AltMode.ALT_MODE_NONE
```

No alternate mode

#### 4.7.3.5 ALT\_MODE\_SPI

```
configurator.GPIO.AltMode.ALT_MODE_SPI
```

SPI



#### 4.7.3.6 ALT\_MODE\_UART

```
configurator.GPIO.AltMode.ALT_MODE_UART
```

UART

#### 4.7.3.7 STR\_NAME

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

Name as String

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

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

## 4.8 framework.CodeGenerator Class Reference

### Public Member Functions

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

### Static Public Attributes

- static final String **STR\_TKN\_CFG\_DEFS\_COMMON** = "FWK\_GPIO\_COMMON\_DEFINITIONS"
- static final String **STR\_TKN\_CFG\_DEFS\_GPIO** = "FWK\_GPIO\_CFG\_DEFINITIONS"

#### 4.8.1 Detailed Description

Author

ovd

#### 4.8.2 Constructor & Destructor Documentation

##### 4.8.2.1 CodeGenerator()

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

Constructor

**Parameters**

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

### 4.8.3 Member Function Documentation

#### 4.8.3.1 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

## 4.9 configurator.GPIO.CodeName Enum Reference

**Public Attributes**

- [CODE\\_NAME](#)

**Static Public Attributes**

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

#### 4.9.1 Detailed Description

**Author**

Miguel Diaz

**Version**

0.1

## 4.9.2 Member Data Documentation

### 4.9.2.1 CODE\_NAME

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

Code name for pin

### 4.9.2.2 STR\_NAME

```
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

## 4.10 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 String [getCommonCfgDefinitions](#) (Microcontroller uC)

### Static Public Attributes

- static final String [NL](#) = "\r\n"
- static final String [STR\\_GEN\\_CODE\\_NOTICE\\_HEADER](#)
- static final String [STR\\_GEN\\_CODE\\_NOTICE\\_FOOTER](#)
- static final String [STR\\_MODULE\\_GPIO](#) = "gpio"
- static final String [STR\\_MODULE\\_ADC](#) = "adc"
- static final String [STR\\_MODULE\\_UART](#) = "uart"
- static final String [STR\\_DEFINITION](#) = "#define "
- static final String [STR\\_INCLUDE](#) = "#include "
- static final String [STR\\_HEADER\\_EXT](#) = ".h"

### 4.10.1 Detailed Description

Framework common fields and methods

#### Author

Miguel Diaz

#### Version

0.1

### 4.10.2 Member Function Documentation

#### 4.10.2.1 `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

#### 4.10.2.2 `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

**4.10.2.3 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

**4.10.2.4 getCommonCfgDefinitions()**

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

Get Framework [Common](#) definitions

## Parameters

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

## Returns

[Common](#) definitions needed for framework

#### 4.10.2.5 `getCommonIncludes()`

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

Get Framework common headers

##### Parameters

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

##### Returns

*Common* headers needed for framework

#### 4.10.2.6 `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

#### 4.10.2.7 `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

**4.10.2.8 getInstallationFwkPath()**

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

Get installation framework path

**Returns**

installation framework path

**4.10.2.9 getProjectFwkPath()**

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

Get project's framework path

**Returns**

project's framework path

**4.10.2.10 setInstallationFwkPath()**

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

Set installation framework path

**Parameters**

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

#### 4.10.2.11 setProjectFwkPath()

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

Set project's framework path

##### Parameters

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

### 4.10.3 Member Data Documentation

#### 4.10.3.1 NL

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

[Common](#) implementation of New Line

#### 4.10.3.2 STR\_DEFINITION

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

Macro definition String

#### 4.10.3.3 STR\_GEN\_CODE\_NOTICE\_FOOTER

```
final String framework.Common.STR_GEN_CODE_NOTICE_FOOTER [static]
```

##### Initial value:

```
= "// ##### " + Features.GENERATOR_NAME
    + " generator v" + common.Features.SW\_VERSION + ": Generated code! #####" + NL
    + "// ##### Do NOT modify code between this footer and the header above #####"
```

Footer for indicating generated code



#### 4.10.3.4 STR\_GEN\_CODE\_NOTICE\_HEADER

```
final String framework.Common.STR_GEN_CODE_NOTICE_HEADER [static]
```

##### Initial value:

```
= "// ##### " + Features.GENERATOR_NAME  
  + " generator v" + common.Features.SW_VERSION + ": Generated code! #####" + NL  
  + "// ##### Do NOT modify code between this header and the footer below #####"
```

Header for indicating generated code

#### 4.10.3.5 STR\_HEADER\_EXT

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

Header file extension

#### 4.10.3.6 STR\_INCLUDE

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

Include header file string

#### 4.10.3.7 STR\_MODULE\_ADC

```
final String framework.Common.STR_MODULE_ADC = "adc" [static]
```

GPIO module name

#### 4.10.3.8 STR\_MODULE\_GPIO

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

GPIO module name

#### 4.10.3.9 STR\_MODULE\_UART

```
final String framework.Common.STR_MODULE_UART = "uart" [static]
```

GPIO module name

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

- src/framework/Common.java

## 4.11 configurator.ConfigurationFile Class Reference

### Static Public Attributes

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

### 4.11.1 Detailed Description

Configuration files properties

Author

Miguel Diaz

Version

0.1

### 4.11.2 Member Data Documentation

#### 4.11.2.1 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

## 4.12 xmlCreator.ConfXmlWriter Class Reference

### Public Member Functions

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

### 4.12.1 Detailed Description

Write a XML file

Author

Miguel Diaz

Version

0.1

### 4.12.2 Constructor & Destructor Documentation

#### 4.12.2.1 ConfXmlWriter()

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

Constructor

Parameters

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

### 4.12.3 Member Function Documentation

#### 4.12.3.1 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

#### 4.12.3.2 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

### 4.13 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"
- static final int [INT\\_INVALID\\_INDEX](#) = -1

#### 4.13.1 Detailed Description

Error codes enum

##### Author

Miguel Diaz

##### Version

0.1

## 4.13.2 Member Data Documentation

### 4.13.2.1 EX\_ERROR

```
common.ErrorCode.EX_ERROR
```

Error during execution

### 4.13.2.2 FILE\_CONF\_ERROR

```
common.ErrorCode.FILE_CONF_ERROR
```

File configuration error

### 4.13.2.3 FILE\_READ\_ERROR

```
common.ErrorCode.FILE_READ_ERROR
```

File reading error

### 4.13.2.4 FILE\_WRITE\_ERROR

```
common.ErrorCode.FILE_WRITE_ERROR
```

File writing error

### 4.13.2.5 INT\_INVALID\_INDEX

```
final int common.ErrorCode.INT_INVALID_INDEX = -1 [static]
```

Invalid index

### 4.13.2.6 NO\_ERROR

```
common.ErrorCode.NO_ERROR
```

No error message

#### 4.13.2.7 STR\_INVALID

```
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

## 4.14 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](#) = "Felucia"
- static final String [GENERATOR\\_NAME](#) = "Kamino"

#### 4.14.1 Detailed Description

Class that includes all project features

Author

Miguel Diaz

Version

0.1

#### 4.14.2 Member Function Documentation

##### 4.14.2.1 debugPrint()

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

Print Debug message to console

## Parameters

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

**4.14.2.2 verbosePrint()**

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

Print Verbose message to console

## Parameters

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

**4.14.3 Member Data Documentation****4.14.3.1 DEBUG**

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

Enables debug functions

**4.14.3.2 DEBUG\_STR**

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

Debug messages indicator on system console

**4.14.3.3 SW\_VERSION**

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

Complete Software version

#### 4.14.3.4 VERBOSE

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

Enables console messages

#### 4.14.3.5 VERBOSE\_STR

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

Verbose messages indicator on system console

#### 4.14.3.6 VERSION\_NAME

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

Code name of the software version

#### 4.14.3.7 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

## 4.15 common.GeneralSettings Class Reference

### Static Public Member Functions

- static void **initLog** ()

### Static Public Attributes

- static final String **LOG\_NAME\_SUFFIX** = "\_log.log"
- static final String **logFilePath**
- static File **logFile**
- static BufferedWriter **logWriter**
- static boolean **LOG\_FILE** = true



### 4.15.1 Member Data Documentation

#### 4.15.1.1 logFilePath

```
final String common.GeneralSettings.logFilePath [static]
```

**Initial value:**

```
= System.getProperty("user.dir") + System.getProperty("file.separator")  
  + "logs"
```

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

- src/common/GeneralSettings.java

## 4.16 gui.GeneralSettingsWindow Class Reference

### Public Member Functions

- [GeneralSettingsWindow](#) ([GeneralSettings](#) settings)

### Static Public Member Functions

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

### 4.16.1 Constructor & Destructor Documentation

#### 4.16.1.1 GeneralSettingsWindow()

```
gui.GeneralSettingsWindow.GeneralSettingsWindow (  
    GeneralSettings settings )
```

Create the application.

### 4.16.2 Member Function Documentation

#### 4.16.2.1 main()

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

Launch the application.

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

- src/gui/GeneralSettingsWindow.java

## 4.17 gui.GpioConfWindow Class Reference

### Public Member Functions

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

### Static Public Member Functions

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

#### 4.17.1 Detailed Description

Window for configuring GPIO pins

Author

Miguel Diaz

Version

0.1

#### 4.17.2 Constructor & Destructor Documentation

##### 4.17.2.1 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
--------------	---

### 4.17.3 Member Function Documentation

#### 4.17.3.1 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

## 4.18 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 [showAdcConfWindow](#) ()
- static void [showUartConfWindow](#) ()
- static void [showProjectPreferencesWindow](#) ()
- static void [showGeneralSettingsWindow](#) ()
- static void [setNewUC](#) (Microcontroller uC)
- static void [saveProjectPreferences](#) (ProjectSettings preferences)
- static void [saveGeneralSettings](#) (GeneralSettings settings)
- static void [saveUc](#) ()
- static [ErrorCode](#) [generateCode](#) ()

## Static Public Attributes

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

### 4.18.1 Detailed Description

Main GUI state machine

Author

Miguel Diaz

Version

0.1

### 4.18.2 Member Function Documentation

#### 4.18.2.1 generateCode()

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

Generate source code files

Returns

Error code

#### 4.18.2.2 loadProjectFile()

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

Load the project settings file

Parameters

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

**Returns**

Error status

**4.18.2.3 main()**

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

**Parameters**

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

**4.18.2.4 saveGeneralSettings()**

```
static void gui.MainGui.saveGeneralSettings (
    GeneralSettings settings ) [static]
```

Save the General Settings

**Parameters**

<i>settings</i>	General Settings
-----------------	------------------

**4.18.2.5 saveProjectPreferences()**

```
static void gui.MainGui.saveProjectPreferences (
    ProjectSettings preferences ) [static]
```

Save the project's preferences

**Parameters**

<i>preferences</i>	Project's preferences
--------------------	-----------------------

#### 4.18.2.6 saveUc()

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

Save the microcontroller's configuration to disk

#### 4.18.2.7 setNewUC()

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

Set the project's microcontroller configuration

Parameters

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

#### 4.18.2.8 showAboutWindow()

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

Show about information window

#### 4.18.2.9 showAdcConfWindow()

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

Show the ADCs configuration window

#### 4.18.2.10 showErrorDialog()

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

Show an error dialog

Parameters

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

#### 4.18.2.11 showGeneralSettingsWindow()

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

Show the General Settings window

#### 4.18.2.12 showGpioConfWindow()

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

Show the GPIOs configuration window

#### 4.18.2.13 showProjectPreferencesWindow()

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

Show the Project Preferences window

#### 4.18.2.14 showUartConfWindow()

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

Show the UARTs configuration window

### 4.18.3 Member Data Documentation

#### 4.18.3.1 ProjectFile

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

Project configuration file

#### 4.18.3.2 ProjectPath

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

Project's location

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

- src/gui/MainGui.java

## 4.19 gui.MainWindow Class Reference

### Public Member Functions

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

### Static Public Member Functions

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

### Public Attributes

- JFrame [FrmCodeGenerator](#)

### 4.19.1 Detailed Description

Main application window

Author

Miguel Diaz

Version

0.1

### 4.19.2 Constructor & Destructor Documentation

#### 4.19.2.1 MainWindow()

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

Create the application.

### 4.19.3 Member Function Documentation

#### 4.19.3.1 main()

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

Open main window



## Parameters

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

### 4.19.3.2 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

### 4.19.3.3 setProjectInformation()

```
ErrorCode gui.MainWindow.setProjectInformation (
    Microcontroller uC,
    String projectName )
```

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

#### 4.19.3.4 setVisible()

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

Set visibility of About window

##### Parameters

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

### 4.19.4 Member Data Documentation

#### 4.19.4.1 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

## 4.20 gui.Messages Class Reference

### Static Public Member Functions

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

#### 4.20.1 Detailed Description

[Messages](#) window

Author

ovd

### 4.20.2 Member Function Documentation

#### 4.20.2.1 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

## 4.21 microcontroller.Microcontroller Class Reference

### Public Member Functions

- [Microcontroller](#) (Document ucDoc)
- [ErrorCode](#) [processDocument](#) ()
- [ErrorCode](#) [loadPinsConf](#) (Document confDoc)
- [ErrorCode](#) [loadAdcsConf](#) (Document confDoc)
- [ErrorCode](#) [loadAdcChannelsConf](#) (Document confDoc)
- [ErrorCode](#) [loadUartsConf](#) (Document confDoc)
- [Pin](#) [getPin](#) (int pinNum)
- String [getUc\\_model](#) ()
- String [getUc\\_manufacturer](#) ()
- int [getUc\\_pinNum](#) ()
- int [getUc\\_gpioNum](#) ()
- int [getUc\\_portNum](#) ()
- int [getUc\\_adcNum](#) ()
- int [getUc\\_uartNum](#) ()
- int [getUc\\_selectedPinsNum](#) ()
- int [getUc\\_selectedAdcsNum](#) ()
- int [getUc\\_selectedUartsNum](#) ()
- [PinConf](#) [getConfiguredPin](#) (String gpioName)
- boolean [isValid](#) ()

### Public Attributes

- String[] [Ports](#)
- String[] [Includes\\_Common](#)
- String[] [Includes\\_Gpio](#)
- String[] [Includes\\_Adc](#)
- String[] [Includes\\_Uart](#)
- String[] [Definitions\\_Common](#)
- String[] [Definitions\\_Gpio](#)
- String[] [Definitions\\_Adc](#)
- String[] [Definitions\\_Uart](#)
- [PinConf](#)[] [GpioCfgPin](#)
- String[] [Adcs](#)
- [AdcConf](#)[] [AdcCfg](#)
- String[] [Uarts](#)
- [UartConf](#)[] [UartCfg](#)

## Static Public Attributes

- static final int [MAX\\_NUMBER\\_OF\\_PINS\\_PER\\_PORT](#) = 32
- static final int [MAX\\_NUMBER\\_OF\\_ADCS](#) = 16
- static final int [MAX\\_NUMBER\\_OF\\_UARTS](#) = 16

### 4.21.1 Detailed Description

[Microcontroller](#) related methods

Author

Miguel Diaz

Version

0.1

### 4.21.2 Constructor & Destructor Documentation

#### 4.21.2.1 Microcontroller()

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

Constructor

Parameters

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

### 4.21.3 Member Function Documentation

#### 4.21.3.1 getConfiguredPin()

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

Get the configuration of a pin

## Parameters

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

## Returns

Pin configuration

**4.21.3.2 getPin()**

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

Get a pin's characteristics

## Parameters

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

## Returns

Pin's characteristics

**4.21.3.3 getUc\_adcNum()**

```
int microcontroller.Microcontroller.getUc_adcNum ( )
```

Get the number of ADCs in the microcontroller

## Returns

Number of ADCs

**4.21.3.4 getUc\_gpioNum()**

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

Get the number of GPIOs in the microcontroller

## Returns

Number of GPIOs

#### 4.21.3.5 `getUc_manufacturer()`

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

Get the microcontroller's manufacturer

##### Returns

[Microcontroller](#)'s manufacturer

#### 4.21.3.6 `getUc_model()`

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

Get the microcontroller's model

##### Returns

[Microcontroller](#)'s model

#### 4.21.3.7 `getUc_pinNum()`

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

Get the microcontroller's pins number

##### Returns

Number of pins

#### 4.21.3.8 `getUc_portNum()`

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

Get the number of ports in the microcontroller

##### Returns

Number of ports

#### 4.21.3.9 getUc\_selectedAdcsNum()

```
int microcontroller.Microcontroller.getUc_selectedAdcsNum ( )
```

Get the total ADCs selected

##### Returns

Total of ADCs selected

#### 4.21.3.10 getUc\_selectedPinsNum()

```
int microcontroller.Microcontroller.getUc_selectedPinsNum ( )
```

Get the total pins selected

##### Returns

Total of pins selected

#### 4.21.3.11 getUc\_selectedUartsNum()

```
int microcontroller.Microcontroller.getUc_selectedUartsNum ( )
```

Get the total UARTs selected

##### Returns

Total of UARTs selected

#### 4.21.3.12 getUc\_uartNum()

```
int microcontroller.Microcontroller.getUc_uartNum ( )
```

Get the number of UARTs in the microcontroller

##### Returns

Number of UARTs

#### 4.21.3.13 isValid()

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

Check if the microcontroller configuration is valid

##### Returns

true if valid

#### 4.21.3.14 loadAdcChannelsConf()

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

Load ADC channels

##### Parameters

<i>confDoc</i>	Configuration document
----------------	------------------------

##### Returns

Error code

#### 4.21.3.15 loadPinsConf()

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

Load pins' configuration

##### Parameters

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

##### Returns

Error Code



#### 4.21.3.16 processDocument()

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

Process the document obtained from XML file

Returns

Error status

### 4.21.4 Member Data Documentation

#### 4.21.4.1 AdcCfg

```
AdcConf [ ] microcontroller.Microcontroller.AdcCfg
```

Configured ADCs list

#### 4.21.4.2 Adcs

```
String [ ] microcontroller.Microcontroller.Adcs
```

List of ADCs

#### 4.21.4.3 Definitions\_Adc

```
String [ ] microcontroller.Microcontroller.Definitions_Adc
```

List of definitions for ADC module

#### 4.21.4.4 Definitions\_Common

```
String [ ] microcontroller.Microcontroller.Definitions_Common
```

List of common definitions that will be available for all framework

#### 4.21.4.5 Definitions\_Gpio

```
String [ ] microcontroller.Microcontroller.Definitions_Gpio
```

List of definitions for GPIO module

#### 4.21.4.6 Definitions\_Uart

```
String [] microcontroller.Microcontroller.Definitions_Uart
```

List of definitions for UART module

#### 4.21.4.7 GpioCfgPin

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

Configured pins list

#### 4.21.4.8 Includes\_Adc

```
String [] microcontroller.Microcontroller.Includes_Adc
```

List of Includes for ADC module

#### 4.21.4.9 Includes\_Common

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

List of common includes that will be available for all framework

#### 4.21.4.10 Includes\_Gpio

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

List of Includes for GPIO module

#### 4.21.4.11 Includes\_Uart

```
String [] microcontroller.Microcontroller.Includes_Uart
```

List of Includes for UART module

#### 4.21.4.12 MAX\_NUMBER\_OF\_ADCS

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

Maximum number of ADCs allowed

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

```
final int microcontroller.Microcontroller.MAX_NUMBER_OF_PINS_PER_PORT = 32 [static]
```

Maximum number of pins allowed in a single port

#### 4.21.4.14 MAX\_NUMBER\_OF\_UARTS

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

Maximum number of ADCs allowed

#### 4.21.4.15 Ports

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

Ports name list

#### 4.21.4.16 UartCfg

```
UartConf [] microcontroller.Microcontroller.UartCfg
```

Configured UARTs list

#### 4.21.4.17 Uarts

```
String [] microcontroller.Microcontroller.Uarts
```

List of UARTs

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

- src/microcontroller/Microcontroller.java

## 4.22 configurator.GPIO.Mode Enum Reference

### Static Public Member Functions

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

## Public Attributes

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

## Static Public Attributes

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

### 4.22.1 Detailed Description

GPIO modes

Author

Miguel Diaz

Version

0.1

### 4.22.2 Member Function Documentation

#### 4.22.2.1 getConfFromString()

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

Get the corresponding mode from its name as String

Parameters

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

Returns

[Mode](#)

### 4.22.3 Member Data Documentation

#### 4.22.3.1 MODE\_ALTERNATE\_FUNCTION

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

Alternate function

#### 4.22.3.2 MODE\_INPUT

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

Input

#### 4.22.3.3 MODE\_MAX\_VALUE

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

Maximum value for [Mode](#) enum

#### 4.22.3.4 MODE\_OUTPUT

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

Output

#### 4.22.3.5 STR\_NAME

```
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

## 4.23 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"

#### 4.23.1 Detailed Description

Pin's output/input level

Author

Miguel Diaz

Version

0.1

#### 4.23.2 Member Function Documentation

##### 4.23.2.1 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

### 4.23.3 Member Data Documentation

#### 4.23.3.1 HIGH

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

High, logical 1, Vcc

#### 4.23.3.2 LOW

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

Low, logical 0, Ground

#### 4.23.3.3 MAX\_VALUE

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

Maximum value for [OutLevel](#) enum

#### 4.23.3.4 STR\_NAME

```
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`

## 4.24 configurator.GPIO.OutType Enum Reference

### Static Public Member Functions

- static [OutType](#) `getConfigFromString` (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"

### 4.24.1 Detailed Description

Pin's output type

Author

Miguel Diaz

Version

0.1

### 4.24.2 Member Function Documentation

#### 4.24.2.1 `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

### 4.24.3 Member Data Documentation

#### 4.24.3.1 OTYPE\_MAX\_VALUE

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

Maximum value for [OutType](#) enum

#### 4.24.3.2 OTYPE\_NOT\_AVAILABLE

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

If the pin is configured as input

#### 4.24.3.3 OTYPE\_OPEN\_DRAIN

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

Open Drain

#### 4.24.3.4 OTYPE\_PUSH\_PULL

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

Push [Pull](#), totem

#### 4.24.3.5 STR\_NAME

```
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

## 4.25 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 instance, String channel)
- String [getAdc](#) ()
- String [getAdcChannel](#) ()
- 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](#) ()
- String [getPortPin](#) ()
- void [setPortPin](#) (String portPin)
- 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

### 4.25.1 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

## 4.25.2 Constructor & Destructor Documentation

### 4.25.2.1 Pin()

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

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

## 4.25.3 Member Function Documentation

### 4.25.3.1 getAdc()

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

Get the pin's ADC name

**Returns**

[Pin's ADC](#)

### 4.25.3.2 getAdcChannel()

```
String microcontroller.Pin.getAdcChannel ( )
```

Get the pin's ADC channel

**Returns**

[Pin's ADC channel](#)

#### 4.25.3.3 getClock()

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

Get the pin's clock name

Returns

Pin's clock

#### 4.25.3.4 getFeat\_adc()

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

See if the pin has an ADC

Returns

Feature availability

#### 4.25.3.5 getFeat\_clock()

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

See if the pin supports a clock

Returns

Feature availability

#### 4.25.3.6 getFeat\_i2c()

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

See if the pin has I2C

Returns

Feature availability

#### 4.25.3.7 `getFeat_int()`

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

See if the pin has an interruption

##### Returns

Feature availability

#### 4.25.3.8 `getFeat_reset()`

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

See if the pin has a reset feature

##### Returns

Feature availability

#### 4.25.3.9 `getFeat_spi()`

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

See if the pin has SPI

##### Returns

Feature availability

#### 4.25.3.10 `getFeat_timer()`

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

See if the pin supports a timer

##### Returns

Feature availability

#### 4.25.3.11 getFeat\_uart()

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

See if the pin has a UART

##### Returns

Feature availability

#### 4.25.3.12 getFunc\_gnd()

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

See if the pin is GND

##### Returns

Function availability

#### 4.25.3.13 getFunc\_gpio()

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

See if the pin is GPIO

##### Returns

Function availability

#### 4.25.3.14 getFunc\_misc()

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

See if the pin is MISC

##### Returns

Function availability

#### 4.25.3.15 getFunc\_reset()

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

See if the pin is RESET

##### Returns

Function availability

#### 4.25.3.16 getFunc\_vcc()

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

See if the pin is Vcc

##### Returns

Function availability

#### 4.25.3.17 getI2c()

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

Get the pin's I2C name

##### Returns

[Pin's I2C](#)

#### 4.25.3.18 getInt()

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

Get the pin's interruption name

##### Returns

[Pin's interruption](#)



#### 4.25.3.19 getName()

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

Get the pin's name

Returns

Pin's name

#### 4.25.3.20 getNumber()

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

Get the pin's number

Returns

Pin's number

#### 4.25.3.21 getPort()

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

Get the pin's port

Returns

Pin's port

#### 4.25.3.22 getPortPin()

```
String microcontroller.Pin.getPortPin ( )
```

Get port pin number

Returns

port pin number

#### 4.25.3.23 `getReset()`

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

Get the pin's reset name

Returns

[Pin's reset](#)

#### 4.25.3.24 `getSpi()`

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

Get the pin's SPI name

Returns

[Pin's SPI](#)

#### 4.25.3.25 `getTimer()`

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

Get the pin's timer name

Returns

[Pin's timer](#)

#### 4.25.3.26 `getUart()`

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

Get the pin's UART name

Returns

[Pin's UART](#)

#### 4.25.3.27 isValid()

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

Check if the pin is correctly initialized

##### Returns

True if the pin is correctly initialized

#### 4.25.3.28 setAdc()

```
void microcontroller.Pin.setAdc (
    String instance,
    String channel )
```

Set the pin's ADC

##### Parameters

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

#### 4.25.3.29 setClock()

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

Set the pin's clock

##### Parameters

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

#### 4.25.3.30 setFeat\_adc()

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

Set the pin's ADC feature

**Parameters**

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

**4.25.3.31 setFeat\_clock()**

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

Set the pin's Clock feature

**Parameters**

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

**4.25.3.32 setFeat\_i2c()**

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

Set the pin's I2C feature

**Parameters**

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

**4.25.3.33 setFeat\_int()**

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

Set the pin's interruption feature

**Parameters**

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

#### 4.25.3.34 setFeat\_reset()

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

Set the pin's reset feature

##### Parameters

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

#### 4.25.3.35 setFeat\_spi()

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

Set the pin's SPI feature

##### Parameters

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

#### 4.25.3.36 setFeat\_timer()

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

Set the pin's timer feature

##### Parameters

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

#### 4.25.3.37 setFeat\_uart()

```
void microcontroller.Pin.setFeat_uart (
```

```
boolean featState )
```

Set the pin's UART feature

#### Parameters

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

#### 4.25.3.38 setFunc\_gnd()

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

Set the pin to GND status

#### Parameters

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

#### 4.25.3.39 setFunc\_gpio()

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

Set the pin to GPIO status

#### Parameters

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

#### 4.25.3.40 setFunc\_misc()

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

Set the pin to MISC status

## Parameters

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

**4.25.3.41 setFunc\_reset()**

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

Set the pin to RESET status

## Parameters

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

**4.25.3.42 setFunc\_vcc()**

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

Set the pin to Vcc status

## Parameters

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

**4.25.3.43 setI2c()**

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

Set the pin's I2C

## Parameters

<i>feature</i>	Pin's I2C
----------------	-----------

#### 4.25.3.44 `setInt()`

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

Set the pin's interruption

##### Parameters

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

#### 4.25.3.45 `setName()`

```
void microcontroller.Pin.setName (
    String pinName )
```

Set the pin's name

##### Parameters

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

#### 4.25.3.46 `setNumber()`

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

Set the pin's number

##### Parameters

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

#### 4.25.3.47 `setPort()`

```
void microcontroller.Pin.setPort (
```

---



```
String pinPort )
```

Set the pin's port

#### Parameters

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

#### 4.25.3.48 setPortPin()

```
void microcontroller.Pin.setPortPin (
    String portPin )
```

Set port pin number

#### Parameters

<i>portPin</i>	Port pin number
----------------	-----------------

#### 4.25.3.49 setReset()

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

Set the pin's reset

#### Parameters

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

#### 4.25.3.50 setSpi()

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

Set the pin's SPI

**Parameters**

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

**4.25.3.51 setTimer()**

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

Set the pin's timer

**Parameters**

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

**4.25.3.52 setUart()**

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

Set the pin's UART

**Parameters**

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

**4.25.4 Member Data Documentation****4.25.4.1 DEF\_FEATURE**

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

Default value for pin's feature as not available

#### 4.25.4.2 DEF\_FEATURE\_AV

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

Default value for pin's feature availability as not available

#### 4.25.4.3 DEF\_FUNCTION

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

Default value for pin's function as not enabled

#### 4.25.4.4 DEF\_NAME

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

Default value for pin's name

#### 4.25.4.5 DEF\_NUMBER

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

Default value for pin's number

#### 4.25.4.6 DEF\_PORT

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

Default value for pin's port

#### 4.25.4.7 DISABLE

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

Disable value for features and functions

#### 4.25.4.8 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

## 4.26 configurator.PinConf Class Reference

### Public Member Functions

- [PinConf](#) ([Pin](#) gpioPin)
- boolean [isValid](#) ()
- String [getPort](#) ()
- String [getPortPin](#) ()
- String [getPinName](#) ()
- String [getCodeName](#) ()
- void [setCodeName](#) (String name)
- [Selected](#) [getSelected](#) ()
- void [setSelected](#) ([Selected](#) selection)
- [Mode](#) [getMode](#) ()
- void [setMode](#) ([Mode](#) mode)
- [AltMode](#) [getAltMode](#) ()
- void [setAltMode](#) ([AltMode](#) altMode)
- [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\\_Uart](#) ()
- boolean [isAv\\_I2c](#) ()
- boolean [isAv\\_Spi](#) ()
- boolean [isAv\\_altFunc](#) ()

### Static Public Attributes

- static final [Selected](#) [DF\\_SELECTED](#) = [Selected](#).NOT
- static final [Mode](#) [DF\\_MODE](#) = [Mode](#).MODE\_INPUT
- static final [AltMode](#) [DF\\_ALT\\_MODE](#) = [AltMode](#).ALT\_MODE\_NONE
- 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](#) = ""

### 4.26.1 Detailed Description

GPIO pin configuration

Author

Miguel Diaz

Version

0.1

## 4.26.2 Constructor & Destructor Documentation

### 4.26.2.1 PinConf()

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

Constructor

Parameters

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

## 4.26.3 Member Function Documentation

### 4.26.3.1 getAltMode()

```
AltMode configurator.PinConf.getAltMode ( )
```

Get pin's alternative mode

Returns

Alternative mode

### 4.26.3.2 getCodeName()

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

Get the pin's user selected name

Returns

pin's name

#### 4.26.3.3 `getMode()`

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

Get the pin's mode configuration

##### Returns

Mode

#### 4.26.3.4 `getOutLevel()`

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

Get the pin's output level

##### Returns

Pin's output level

#### 4.26.3.5 `getOutType()`

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

Get the pin's output configuration

##### Returns

Output configuration

#### 4.26.3.6 `getPinName()`

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

Get the pin's number

##### Returns

Pin's number

#### 4.26.3.7 getPort()

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

Get the pin's port

##### Returns

Port

#### 4.26.3.8 getPortPin()

```
String configurator.PinConf.getPortPin ( )
```

Get the port pin number

##### Returns

Port pin number

#### 4.26.3.9 getPull()

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

Get the pin's pull resistor configuration

##### Returns

Pull Resistor configuration

#### 4.26.3.10 getSelected()

```
Selected configurator.PinConf.getSelected ( )
```

Get the pin's selection

##### Returns

Selection

#### 4.26.3.11 `getSpeed()`

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

Get the pin's speed

##### Returns

Speed

#### 4.26.3.12 `isAv_Adc()`

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

Check availability of ADC

##### Returns

True if ADC is available

#### 4.26.3.13 `isAv_altFunc()`

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

Check the availability of alternate function

##### Returns

True if alternate function is available

#### 4.26.3.14 `isAv_I2c()`

```
boolean configurator.PinConf.isAv_I2c ( )
```

Check availability of I2C

##### Returns

True if I2C is available



#### 4.26.3.15 isAv\_Spi()

```
boolean configurator.PinConf.isAv_Spi ( )
```

Check availability of SPI

##### Returns

True if SPI is available

#### 4.26.3.16 isAv\_Uart()

```
boolean configurator.PinConf.isAv_Uart ( )
```

Check availability of UART

##### Returns

True id UART is available

#### 4.26.3.17 isValid()

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

Check if the GPIO pin is valid

##### Returns

True if valid

#### 4.26.3.18 setAltMode()

```
void configurator.PinConf.setAltMode (
    AltMode altMode )
```

Set pin's alternative mode

**Parameters**

<i>altMode</i>	Alternative mode
----------------	------------------

**4.26.3.19 setCodeName()**

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

Set the pin's user selected name

**Parameters**

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

**4.26.3.20 setMode()**

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

Set the pin's mode configuration

**Parameters**

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

**4.26.3.21 setOutLevel()**

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

Set the pin's output level

**Parameters**

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

#### 4.26.3.22 setOutType()

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

Set the pin's output configuration

##### Parameters

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

#### 4.26.3.23 setPull()

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

Set the pull resistor configuration

##### Parameters

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

#### 4.26.3.24 setSelected()

```
void configurator.PinConf.setSelected (
    Selected selection )
```

Set the pin's selection

##### Parameters

<i>selection</i>	Selection
------------------	-----------

#### 4.26.3.25 setSpeed()

```
void configurator.PinConf.setSpeed (
```

```
Speed speed )
```

Set the pin's speed

#### Parameters

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

## 4.26.4 Member Data Documentation

### 4.26.4.1 DF\_ALT\_MODE

```
final AltMode configurator.PinConf.DF_ALT_MODE = AltMode.ALT_MODE_NONE [static]
```

Default Pin alternative mode

### 4.26.4.2 DF\_CODE\_NAME

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

Default pin's code name

### 4.26.4.3 DF\_MODE

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

Default Pin mode

### 4.26.4.4 DF\_OUT\_LEVEL

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

Default pin's output level

### 4.26.4.5 DF\_OUTTYPE

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

Default pin's output type

#### 4.26.4.6 DF\_PULL

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

Default pin's pull resistor

#### 4.26.4.7 DF\_SELECTED

```
final Selected configurator.PinConf.DF_SELECTED = Selected.NOT [static]
```

Default Pin's selection

#### 4.26.4.8 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

## 4.27 projectConfiguration.ProjectSettings Class Reference

### Public Member Functions

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

### 4.27.1 Detailed Description

Project settings class

Author

Miguel Diaz

Version

0.2

## 4.27.2 Constructor & Destructor Documentation

### 4.27.2.1 ProjectSettings()

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

Constructor

## 4.27.3 Member Function Documentation

### 4.27.3.1 getConfFile()

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

Get the project configuration file

Returns

Project configuration file

### 4.27.3.2 getFrameworkPath()

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

Get the framework folder

Returns

framework folder

#### 4.27.3.3 `getProjectName()`

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

Get the project's name

##### Returns

Project's name

#### 4.27.3.4 `getUcFile()`

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

Get the project microcontroller file

##### Returns

Project microcontroller file

#### 4.27.3.5 `openProjectFile()`

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

Open the project settings file

##### Parameters

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

##### Returns

Error Status

#### 4.27.3.6 `processDocument()`

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

Process the document obtained from the XML file

**Returns**

Error Status

**4.27.3.7 setFrameworkPath()**

```
void projectConfiguration.ProjectSettings.setFrameworkPath (
    String path )
```

Set the framework folder

**Parameters**

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

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

- src/projectConfiguration/ProjectSettings.java

## 4.28 gui.ProjectSettingsWindow Class Reference

**Public Member Functions**

- [ProjectSettingsWindow](#) ([ProjectSettings](#) settings)

**Static Public Member Functions**

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

**4.28.1 Constructor & Destructor Documentation****4.28.1.1 ProjectSettingsWindow()**

```
gui.ProjectSettingsWindow.ProjectSettingsWindow (
    ProjectSettings settings )
```

Create the application.



## 4.28.2 Member Function Documentation

### 4.28.2.1 main()

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

Launch the application.

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

- src/gui/ProjectSettingsWindow.java

## 4.29 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"

### 4.29.1 Detailed Description

Pin's pull resistor

Author

Miguel Diaz

Version

0.1

### 4.29.2 Member Function Documentation

#### 4.29.2.1 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

### 4.29.3 Member Data Documentation

#### 4.29.3.1 PULL\_DOWN

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

[Pull](#) Down

#### 4.29.3.2 PULL\_MAX\_VALUE

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

Maximum value for [Pull](#) enum

#### 4.29.3.3 PULL\_NOT\_AVAILABLE

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

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

#### 4.29.3.4 PULL\_UP

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

[Pull](#) Up

#### 4.29.3.5 STR\_NAME

```
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

## 4.30 configurator.Selected Enum Reference

### Public Member Functions

- boolean `getBoolean` ()

### Static Public Member Functions

- static `Selected` `getConfigFromString` (String *conf*)
- static `Selected` `getConfigFromBoolean` (Boolean *conf*)

### Public Attributes

- `NOT`
- `YES`

### Static Public Attributes

- static final String `STR_NAME` = "selected"

#### 4.30.1 Detailed Description

Pin's selection

Author

Miguel Díaz

Version

0.1

#### 4.30.2 Member Function Documentation

##### 4.30.2.1 `getBoolean()`

```
boolean configurator.Selected.getBoolean ( )
```

Get the corresponding boolean from its selection

Returns

`Selected` pin state

##### 4.30.2.2 `getConfigFromBoolean()`

```
static Selected configurator.Selected.getConfigFromBoolean (
    Boolean conf ) [static]
```

Get the corresponding mode from a boolean

**Parameters**

<i>conf</i>	Configuration value
-------------	---------------------

**Returns**

[Selected](#)

**4.30.2.3 getConfFromString()**

```
static Selected configurator.Selected.getConfFromString (  
    String conf ) [static]
```

Get the corresponding mode from its name as String

**Parameters**

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

**Returns**

[Selected](#)

**4.30.3 Member Data Documentation****4.30.3.1 NOT**

```
configurator.Selected.NOT
```

Pin not selected

**4.30.3.2 STR\_NAME**

```
final String configurator.Selected.STR_NAME = "selected" [static]
```

Name as String

#### 4.30.3.3 YES

```
configurator.Selected.YES
```

Pin selected

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

- src/configurator/Selected.java

## 4.31 configurator.GPIO.Speed Enum Reference

### Static Public Member Functions

- static [Speed](#) [getConfigFromString](#) (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"

### 4.31.1 Detailed Description

Pin's speed

Author

Miguel Diaz

Version

0.1

### 4.31.2 Member Function Documentation

#### 4.31.2.1 getConfigFromString()

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

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

**Parameters**

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

**Returns**

[Speed](#)

### 4.31.3 Member Data Documentation

#### 4.31.3.1 SPEED\_FAST

`configurator.GPIO.Speed.SPEED_FAST`

Fast

#### 4.31.3.2 SPEED\_HIGH

`configurator.GPIO.Speed.SPEED_HIGH`

High

#### 4.31.3.3 SPEED\_MAX\_VALUE

`configurator.GPIO.Speed.SPEED_MAX_VALUE`

Maximum value for [Speed](#) enum

#### 4.31.3.4 SPEED\_MEDIUM

`configurator.GPIO.Speed.SPEED_MEDIUM`

Medium

#### 4.31.3.5 SPEED\_NOT\_AVAILABLE

`configurator.GPIO.Speed.SPEED_NOT_AVAILABLE`

Not all MCUs will have this setting

#### 4.31.3.6 STR\_NAME

```
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

## 4.32 microcontroller.Uart Class Reference

### Public Member Functions

- [Uart](#) ()
- void [setName](#) (String name)
- String [getName](#) ()
- void [addClock](#) (String clock)
- int [getClockNum](#) ()
- String [getClock](#) (int index)
- void [addPrescaler](#) (String prescaler)
- int [getPrescalerNum](#) ()
- String [getPrescaler](#) (int index)
- void [addBaudRate](#) (String baudRate)
- int [getBaudRateNum](#) ()
- String [getBaudRate](#) (int index)
- void [addDataBits](#) (String dataBits)
- int [getDataBitsNum](#) ()
- String [getDataBits](#) (int index)
- void [addStopBits](#) (String stopBits)
- int [getStopBitsNum](#) ()
- String [getStopBits](#) (int index)
- void [addParity](#) (String parity)
- int [getParityNum](#) ()
- String [getParity](#) (int index)
- boolean [isValid](#) ()

### 4.32.1 Constructor & Destructor Documentation

#### 4.32.1.1 Uart()

```
microcontroller.Uart.Uart ( )
```

UART instance constructor

## 4.32.2 Member Function Documentation

### 4.32.2.1 addBaudRate()

```
void microcontroller.Uart.addBaudRate (
    String baudRate )
```

Add UART's supported clock BaudRate

#### Parameters

<i>BaudRate</i>	Clock BaudRate
-----------------	----------------

### 4.32.2.2 addClock()

```
void microcontroller.Uart.addClock (
    String clock )
```

Add ADC supported clock source

#### Parameters

<i>clock</i>	Clock source
--------------	--------------

### 4.32.2.3 addDataBits()

```
void microcontroller.Uart.addDataBits (
    String dataBits )
```

Add UART's supported clock DataBits

#### Parameters

<i>DataBits</i>	Clock DataBits
-----------------	----------------



#### 4.32.2.4 addParity()

```
void microcontroller.Uart.addParity (
    String parity )
```

Add UART's supported clock Parity

##### Parameters

<i>Parity</i>	Clock Parity
---------------	--------------

#### 4.32.2.5 addPrescaler()

```
void microcontroller.Uart.addPrescaler (
    String prescaler )
```

Add UART's supported clock prescaler

##### Parameters

<i>prescaler</i>	Clock prescaler
------------------	-----------------

#### 4.32.2.6 addStopBits()

```
void microcontroller.Uart.addStopBits (
    String stopBits )
```

Add UART's supported clock StopBits

##### Parameters

<i>StopBits</i>	Clock StopBits
-----------------	----------------

#### 4.32.2.7 getBaudRate()

```
String microcontroller.Uart.getBaudRate (
    int index )
```

Get UART's clock BaudRate

**Parameters**

<i>index</i>	Clock BaudRate index
--------------	----------------------

**Returns**

Clock BaudRate

**4.32.2.8 getBaudRateNum()**

```
int microcontroller.Uart.getBaudRateNum ( )
```

Get UART's number of supported BaudRates

**Returns**

Number of supported BaudRates

**4.32.2.9 getClock()**

```
String microcontroller.Uart.getClock (
    int index )
```

Get UART's clock source

**Parameters**

<i>index</i>	Clock source index
--------------	--------------------

**Returns**

Clock source

**4.32.2.10 getClockNum()**

```
int microcontroller.Uart.getClockNum ( )
```

Get UARTs number of clock sources

**Returns**

Number of clock sources

**4.32.2.11 getDataBits()**

```
String microcontroller.Uart.getDataBits (
    int index )
```

Get UART's clock DataBits

**Parameters**

<i>index</i>	Clock DataBits index
--------------	----------------------

**Returns**

Clock DataBits

**4.32.2.12 getDataBitsNum()**

```
int microcontroller.Uart.getDataBitsNum ( )
```

Get UART's number of supported DataBits

**Returns**

Number of supported DataBits

**4.32.2.13 getName()**

```
String microcontroller.Uart.getName ( )
```

Get UARTs instance name

**Returns**

Instance name

**4.32.2.14 getParity()**

```
String microcontroller.Uart.getParity (
    int index )
```

Get UART's clock Parity

**Parameters**

<i>index</i>	Clock Parity index
--------------	--------------------

**Returns**

Clock Parity

**4.32.2.15 getParityNum()**

```
int microcontroller.Uart.getParityNum ( )
```

Get UART's number of supported Paritys

**Returns**

Number of supported Paritys

**4.32.2.16 getPrescaler()**

```
String microcontroller.Uart.getPrescaler (
    int index )
```

Get UART's clock prescaler

**Parameters**

<i>index</i>	Clock prescaler index
--------------	-----------------------

**Returns**

Clock prescaler

**4.32.2.17 getPrescalerNum()**

```
int microcontroller.Uart.getPrescalerNum ( )
```

Get UART's number of supported prescalers

**Returns**

Number of supported prescalers

**4.32.2.18 getStopBits()**

```
String microcontroller.Uart.getStopBits (
    int index )
```

Get UART's clock StopBits

**Parameters**

<i>index</i>	Clock StopBits index
--------------	----------------------

**Returns**

Clock StopBits

**4.32.2.19 getStopBitsNum()**

```
int microcontroller.Uart.getStopBitsNum ( )
```

Get UART's number of supported StopBits

**Returns**

Number of supported StopBits

**4.32.2.20 isValid()**

```
boolean microcontroller.Uart.isValid ( )
```

Check validity of ADC

**Returns**

True if valid

**4.32.2.21 setName()**

```
void microcontroller.Uart.setName (
    String name )
```

Set UARTs instance name

## Parameters

<i>name</i>	Instance name
-------------	---------------

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

- src/microcontroller/Uart.java

## 4.33 configurator.UartConf Class Reference

### Public Member Functions

- [UartConf](#) ([Uart](#) uart)
- [Selected](#) [getSelected](#) ()
- void [setSelected](#) ([Selected](#) selection)
- String [getCodeName](#) ()
- void [setCodeName](#) (String codeName)
- String [getClock](#) ()
- void [setClock](#) (String clock)
- String [getPrescaler](#) ()
- void [setPrescaler](#) (String prescaler)
- String [getBaudRate](#) ()
- void [setBaudRate](#) (String baudRate)
- String [getDataBits](#) ()
- void [setDataBits](#) (String dataBits)
- String [getStopBits](#) ()
- void [setStopBits](#) (String stopBits)
- String [getParity](#) ()
- void [setParity](#) (String parity)

### Public Attributes

- [Uart](#) [UartFeatures](#)

### Static Public Attributes

- static final [Selected](#) [DF\\_SELECTED](#) = [Selected](#).NOT
- static final String [STR\\_NAME](#) = "name"
- static final String [STR\\_CODE\\_NAME](#) = "codeName"
- static final String [STR\\_CLOCK](#) = "clock"
- static final String [STR\\_PRESCALER](#) = "prescaler"
- static final String [STR\\_BAUD\\_RATE](#) = "baudRate"
- static final String [STR\\_DATA\\_BITS](#) = "dataBits"
- static final String [STR\\_STOP\\_BITS](#) = "stopBits"
- static final String [STR\\_PARITY](#) = "parity"

### 4.33.1 Constructor & Destructor Documentation

#### 4.33.1.1 UartConf()

```
configurator.UartConf.UartConf (
    Uart uart )
```

UART configuration constructor

##### Parameters

<i>uart</i>	UART instance
-------------	---------------

### 4.33.2 Member Function Documentation

#### 4.33.2.1 getBaudRate()

```
String configurator.UartConf.getBaudRate ( )
```

Get UART's configured BaudRate

##### Returns

UART's configured BaudRate

#### 4.33.2.2 getClock()

```
String configurator.UartConf.getClock ( )
```

Get UART's configured Clock

##### Returns

UART's configured Clock

#### 4.33.2.3 getCodeName()

```
String configurator.UartConf.getCodeName ( )
```

Get UART's code name

##### Returns

UART's code name

#### 4.33.2.4 getDataBits()

```
String configurator.UartConf.getDataBits ( )
```

Get UART's configured DataBits

##### Returns

UART's configured DataBits

#### 4.33.2.5 getParity()

```
String configurator.UartConf.getParity ( )
```

Get UART's configured Parity

##### Returns

UART's configured Parity

#### 4.33.2.6 getPrescaler()

```
String configurator.UartConf.getPrescaler ( )
```

Get UART's configured Prescaler

##### Returns

UART's configured Prescaler



#### 4.33.2.7 getSelected()

```
Selected configurator.UartConf.getSelected ( )
```

Get the UART's selection

##### Returns

Selection

#### 4.33.2.8 getStopBits()

```
String configurator.UartConf.getStopBits ( )
```

Get UART's configured StopBits

##### Returns

UART's configured StopBits

#### 4.33.2.9 setBaudRate()

```
void configurator.UartConf.setBaudRate (
    String baudRate )
```

Get UART's configured BaudRate

##### Parameters

<i>baudRate</i>	UART's configured BaudRate
-----------------	----------------------------

#### 4.33.2.10 setClock()

```
void configurator.UartConf.setClock (
    String clock )
```

Get UART's configured Clock

**Parameters**

<i>clock</i>	UART's configured Clock
--------------	-------------------------

**4.33.2.11 setCodeName()**

```
void configurator.UartConf.setCodeName (  
    String codeName )
```

Set Get UART's code name

**Parameters**

<i>codeName</i>	UART's code name
-----------------	------------------

**4.33.2.12 setDataBits()**

```
void configurator.UartConf.setDataBits (  
    String dataBits )
```

Get UART's configured DataBits

**Parameters**

<i>dataBits</i>	UART's configured DataBits
-----------------	----------------------------

**4.33.2.13 setParity()**

```
void configurator.UartConf.setParity (  
    String parity )
```

Get UART's configured Parity

**Parameters**

<i>parity</i>	UART's configured Parity
---------------	--------------------------

#### 4.33.2.14 setPrescaler()

```
void configurator.UartConf.setPrescaler (
    String prescaler )
```

Get UART's configured Prescaler

##### Parameters

<i>prescaler</i>	UART's configured Prescaler
------------------	-----------------------------

#### 4.33.2.15 setSelected()

```
void configurator.UartConf.setSelected (
    Selected selection )
```

Set the UART's selection

##### Parameters

<i>selection</i>	Selection
------------------	-----------

#### 4.33.2.16 setStopBits()

```
void configurator.UartConf.setStopBits (
    String stopBits )
```

Get UART's configured StopBits

##### Parameters

<i>stopBits</i>	UART's configured StopBits
-----------------	----------------------------

### 4.33.3 Member Data Documentation

#### 4.33.3.1 DF\_SELECTED

```
final Selected configurator.UartConf.DF_SELECTED = Selected.NOT [static]
```

Default Pin's selection

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

- src/configurator/UartConf.java

### 4.34 gui.UartConfWindow Class Reference

#### Public Member Functions

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

#### Static Public Member Functions

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

#### 4.34.1 Constructor & Destructor Documentation

##### 4.34.1.1 UartConfWindow()

```
gui.UartConfWindow.UartConfWindow (  
    Microcontroller uCtrl )
```

Create the application.

#### Parameters

<i>uCtrl</i>	Microcontroller
--------------	-----------------

#### 4.34.2 Member Function Documentation

#### 4.34.2.1 main()

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

Launch the application.

##### Parameters

<i>args</i>	General arguments
-------------	-------------------

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

- src/gui/UartConfWindow.java

## 4.35 framework.UartGenerator Class Reference

### Static Public Member Functions

- static String **getCfgArray** ([Microcontroller](#) uC)
- static String **getElDefs** ([Microcontroller](#) uC)
- static String **getElements** ([Microcontroller](#) uC)
- static String **getIncludes** ([Microcontroller](#) uC)
- static String **getCfgDefinitions** ([Microcontroller](#) uC)

### Static Public Attributes

- static final String **STR\_TKN\_CFG\_ARRAY** = "FWK\_UART\_CFG\_ARRAY"
- static final String **STR\_TKN\_ELEMENTS** = "FWK\_UART\_ELEMENTS"
- static final String **STR\_TKN\_INC** = "FWK\_UART\_INCLUDES"
- static final String **STR\_TKN\_CFG\_DEFS** = "FWK\_UART\_CFG\_DEFINITIONS"
- static final String **STR\_TKN\_EL\_DEFS** = "FWK\_UART\_ELEMENTS\_DEFINITIONS"

### 4.35.1 Member Function Documentation

#### 4.35.1.1 getElDefs()

```
static String framework.UartGenerator.getElDefs (
    Microcontroller uC ) [static]
```

**Parameters**

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

**Returns**

Elements definitions as String

**4.35.1.2 getElements()**

```
static String framework.UartGenerator.getElements (  
    Microcontroller uC ) [static]
```

**Parameters**

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

**Returns**

Elements list as String

**4.35.1.3 getIncludes()**

```
static String framework.UartGenerator.getIncludes (  
    Microcontroller uC ) [static]
```

**Parameters**

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

**Returns**

Headers needed for GPIO module

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

- src/framework/UartGenerator.java

## 4.36 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)

### 4.36.1 Detailed Description

Open and process XML files

Author

H112943

Version

0.1

### 4.36.2 Constructor & Destructor Documentation

#### 4.36.2.1 XmlOpener()

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

Constructor

### 4.36.3 Member Function Documentation

#### 4.36.3.1 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

**4.36.3.2 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

**4.36.3.3 getParsedDoc()**

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

Get the parsed document AFTER opening the file

**Returns**

Parsed document

**4.36.3.4 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](#)
- Adc
  - microcontroller.Adc, [11](#)
- AdcCfg
  - microcontroller.Microcontroller, [65](#)
- AdcConf
  - configurator.AdcConf, [22](#)
- AdcConfWindow
  - gui.AdcConfWindow, [28](#)
- Adcs
  - microcontroller.Microcontroller, [65](#)
- addBaudRate
  - microcontroller.Uart, [112](#)
- addChannel
  - microcontroller.Adc, [11](#)
- addClock
  - microcontroller.Adc, [11](#)
  - microcontroller.Uart, [112](#)
- addDataBits
  - microcontroller.Uart, [112](#)
- addJustification
  - microcontroller.Adc, [11](#)
- addParity
  - microcontroller.Uart, [112](#)
- addPin
  - xmlCreator.ConfXmlWriter, [43](#)
- addPrescaler
  - microcontroller.Adc, [12](#)
  - microcontroller.Uart, [113](#)
- addReference
  - microcontroller.Adc, [12](#)
- addResolution
  - microcontroller.Adc, [12](#)
- addSample
  - microcontroller.Adc, [13](#)
- addStopBits
  - microcontroller.Uart, [113](#)
- ALT\_MODE\_ANALOG
  - configurator.GPIO.AltMode, [32](#)
- ALT\_MODE\_I2C
  - configurator.GPIO.AltMode, [32](#)
- ALT\_MODE\_MAX\_VALUE
  - configurator.GPIO.AltMode, [32](#)
- ALT\_MODE\_NONE
  - configurator.GPIO.AltMode, [32](#)
- ALT\_MODE\_SPI
  - configurator.GPIO.AltMode, [32](#)
- ALT\_MODE\_UART
  - configurator.GPIO.AltMode, [32](#)
- CODE\_NAME
  - configurator.GPIO.CodeName, [35](#)
- CodeGenerator
  - framework.CodeGenerator, [33](#)
- common, [5](#)
- common.ErrorCode, [44](#)
  - EX\_ERROR, [45](#)
  - FILE\_CONF\_ERROR, [45](#)
  - FILE\_READ\_ERROR, [45](#)
  - FILE\_WRITE\_ERROR, [45](#)
  - INT\_INVALID\_INDEX, [45](#)
  - NO\_ERROR, [45](#)
  - STR\_INVALID, [45](#)
- common.Features, [46](#)
  - DEBUG, [47](#)
  - DEBUG\_STR, [47](#)
  - debugPrint, [46](#)
  - SW\_VERSION, [47](#)
  - VERBOSE, [47](#)
  - VERBOSE\_STR, [48](#)
  - verbosePrint, [47](#)
  - VERSION\_NAME, [48](#)
  - VERSION\_STATUS, [48](#)
- common.GeneralSettings, [48](#)
  - logFilePath, [49](#)
- configurator, [5](#)
- configurator.ADC.AdcChannel, [18](#)
  - DF\_SELECTED, [21](#)
  - getCodeName, [19](#)
  - getName, [19](#)
  - getPinIndex, [19](#)
  - getSelected, [19](#)
  - isValid, [20](#)
  - setCodeName, [20](#)
  - setSelected, [20](#)
- configurator.AdcConf, [21](#)
  - AdcConf, [22](#)
  - DF\_SELECTED, [28](#)
  - getChannel, [22](#)
  - getChannelsNum, [23](#)

- getClock, [23](#)
- getCodeName, [23](#)
- getJustification, [23](#)
- getPrescaler, [24](#)
- getReference, [24](#)
- getResolution, [24](#)
- getSample, [24](#)
- getSelected, [25](#)
- setChannels, [25](#)
- setClock, [25](#)
- setCodeName, [26](#)
- setJustification, [26](#)
- setPrescaler, [26](#)
- setReference, [27](#)
- setResolution, [27](#)
- setSample, [27](#)
- setSelected, [27](#)
- configurator.ConfigurationFile, [42](#)
  - STR\_PROJ\_CONF\_FILE, [42](#)
- configurator.GPIO.AltMode, [31](#)
  - ALT\_MODE\_ANALOG, [32](#)
  - ALT\_MODE\_I2C, [32](#)
  - ALT\_MODE\_MAX\_VALUE, [32](#)
  - ALT\_MODE\_NONE, [32](#)
  - ALT\_MODE\_SPI, [32](#)
  - ALT\_MODE\_UART, [32](#)
  - getConfFromString, [31](#)
  - STR\_NAME, [33](#)
- configurator.GPIO.CodeName, [34](#)
  - CODE\_NAME, [35](#)
  - STR\_NAME, [35](#)
- configurator.GPIO.Mode, [67](#)
  - getConfFromString, [68](#)
  - MODE\_ALTERNATE\_FUNCTION, [69](#)
  - MODE\_INPUT, [69](#)
  - MODE\_MAX\_VALUE, [69](#)
  - MODE\_OUTPUT, [69](#)
  - STR\_NAME, [69](#)
- configurator.GPIO.OutLevel, [70](#)
  - getConfFromString, [70](#)
  - HIGH, [71](#)
  - LOW, [71](#)
  - MAX\_VALUE, [71](#)
  - STR\_NAME, [71](#)
- configurator.GPIO.OutType, [71](#)
  - getConfFromString, [72](#)
  - OTYPE\_MAX\_VALUE, [73](#)
  - OTYPE\_NOT\_AVAILABLE, [73](#)
  - OTYPE\_OPEN\_DRAIN, [73](#)
  - OTYPE\_PUSH\_PULL, [73](#)
  - STR\_NAME, [73](#)
- configurator.GPIO.Pull, [105](#)
  - getConfFromString, [105](#)
  - PULL\_DOWN, [106](#)
  - PULL\_MAX\_VALUE, [106](#)
  - PULL\_NOT\_AVAILABLE, [106](#)
  - PULL\_UP, [106](#)
  - STR\_NAME, [106](#)
- configurator.GPIO.Speed, [109](#)
  - getConfFromString, [109](#)
  - SPEED\_FAST, [110](#)
  - SPEED\_HIGH, [110](#)
  - SPEED\_MAX\_VALUE, [110](#)
  - SPEED\_MEDIUM, [110](#)
  - SPEED\_NOT\_AVAILABLE, [110](#)
  - STR\_NAME, [110](#)
- configurator.PinConf, [92](#)
  - DF\_ALT\_MODE, [100](#)
  - DF\_CODE\_NAME, [100](#)
  - DF\_MODE, [100](#)
  - DF\_OUT\_LEVEL, [100](#)
  - DF\_OUTTYPE, [100](#)
  - DF\_PULL, [100](#)
  - DF\_SELECTED, [101](#)
  - DF\_SPEED, [101](#)
  - getAltMode, [93](#)
  - getCodeName, [93](#)
  - getMode, [93](#)
  - getOutLevel, [94](#)
  - getOutType, [94](#)
  - getPinName, [94](#)
  - getPort, [94](#)
  - getPortPin, [95](#)
  - getPull, [95](#)
  - getSelected, [95](#)
  - getSpeed, [95](#)
  - isAv\_Adc, [96](#)
  - isAv\_altFunc, [96](#)
  - isAv\_I2c, [96](#)
  - isAv\_Spi, [96](#)
  - isAv\_Uart, [97](#)
  - isValid, [97](#)
  - PinConf, [93](#)
  - setAltMode, [97](#)
  - setCodeName, [98](#)
  - setMode, [98](#)
  - setOutLevel, [98](#)
  - setOutType, [99](#)
  - setPull, [99](#)
  - setSelected, [99](#)
  - setSpeed, [99](#)
- configurator.Selected, [107](#)
  - getBoolean, [107](#)
  - getConfFromBoolean, [107](#)
  - getConfFromString, [108](#)
  - NOT, [108](#)
  - STR\_NAME, [108](#)
  - YES, [108](#)

- configurator.UartConf, 118
  - DF\_SELECTED, 123
  - getBaudRate, 119
  - getClock, 119
  - getCodeName, 119
  - getDataBits, 120
  - getParity, 120
  - getPrescaler, 120
  - getSelected, 120
  - getStopBits, 121
  - setBaudRate, 121
  - setClock, 121
  - setCodeName, 122
  - setDataBits, 122
  - setParity, 122
  - setPrescaler, 123
  - setSelected, 123
  - setStopBits, 123
  - UartConf, 119
- ConfXmlWriter
  - xmlCreator.ConfXmlWriter, 43
- DEBUG
  - common.Features, 47
- DEBUG\_STR
  - common.Features, 47
- debugPrint
  - common.Features, 46
- DEF\_FEATURE
  - microcontroller.Pin, 90
- DEF\_FEATURE\_AV
  - microcontroller.Pin, 90
- DEF\_FUNCTION
  - microcontroller.Pin, 91
- DEF\_NAME
  - microcontroller.Pin, 91
- DEF\_NUMBER
  - microcontroller.Pin, 91
- DEF\_PORT
  - microcontroller.Pin, 91
- Definitions\_Adc
  - microcontroller.Microcontroller, 65
- Definitions\_Common
  - microcontroller.Microcontroller, 65
- Definitions\_Gpio
  - microcontroller.Microcontroller, 65
- Definitions\_Uart
  - microcontroller.Microcontroller, 65
- DF\_ALT\_MODE
  - configurator.PinConf, 100
- DF\_CODE\_NAME
  - configurator.PinConf, 100
- DF\_MODE
  - configurator.PinConf, 100
- DF\_OUT\_LEVEL
  - configurator.PinConf, 100
- DF\_OUTTYPE
  - configurator.PinConf, 100
- DF\_PULL
  - configurator.PinConf, 100
- DF\_SELECTED
  - configurator.ADC.AdcChannel, 21
  - configurator.AdcConf, 28
  - configurator.PinConf, 101
  - configurator.UartConf, 123
- DF\_SPEED
  - configurator.PinConf, 101
- DISABLE
  - microcontroller.Pin, 91
- ENABLE
  - microcontroller.Pin, 91
- EX\_ERROR
  - common.ErrorCode, 45
- FILE\_CONF\_ERROR
  - common.ErrorCode, 45
- FILE\_READ\_ERROR
  - common.ErrorCode, 45
- FILE\_WRITE\_ERROR
  - common.ErrorCode, 45
- framework, 6
- framework.AdcGenerator, 29
  - getElDefs, 30
  - getElements, 30
  - getIncludes, 30
- framework.CodeGenerator, 33
  - CodeGenerator, 33
  - Generate, 34
- framework.Common, 35
  - getCfgFileCPath, 36
  - getCfgFileHPATH, 36
  - getCfgPath, 37
  - getCommonCfgDefinitions, 37
  - getCommonIncludes, 38
  - getFrameworkCommonFilePath, 38
  - getFrameworkIncludesFilePath, 38
  - getInstallationFwkPath, 39
  - getProjectFwkPath, 39
  - NL, 40
  - setInstallationFwkPath, 39
  - setProjectFwkPath, 40
  - STR\_DEFINITION, 40
  - STR\_GEN\_CODE\_NOTICE\_FOOTER, 40
  - STR\_GEN\_CODE\_NOTICE\_HEADER, 40
  - STR\_HEADER\_EXT, 41
  - STR\_INCLUDE, 41
  - STR\_MODULE\_ADC, 41
  - STR\_MODULE\_GPIO, 41

- STR\_MODULE\_UART, [41](#)
- framework.UartGenerator, [125](#)
  - getEIDefs, [125](#)
  - getElements, [126](#)
  - getIncludes, [126](#)
- FrmCodeGenerator
  - gui.MainWindow, [58](#)
- GeneralSettingsWindow
  - gui.GeneralSettingsWindow, [49](#)
- Generate
  - framework.CodeGenerator, [34](#)
- generateCode
  - gui.MainGui, [52](#)
- getAdc
  - microcontroller.Pin, [76](#)
- getAdcChannel
  - microcontroller.Pin, [76](#)
- getAltMode
  - configurator.PinConf, [93](#)
- getBaudRate
  - configurator.UartConf, [119](#)
  - microcontroller.Uart, [113](#)
- getBaudRateNum
  - microcontroller.Uart, [114](#)
- getBoolean
  - configurator.Selected, [107](#)
- getCfgFileCPath
  - framework.Common, [36](#)
- getCfgFileHPath
  - framework.Common, [36](#)
- getCfgPath
  - framework.Common, [37](#)
- getChannel
  - configurator.AdcConf, [22](#)
  - microcontroller.Adc, [13](#)
- getChannelNum
  - microcontroller.Adc, [13](#)
- getChannelsNum
  - configurator.AdcConf, [23](#)
- getClock
  - configurator.AdcConf, [23](#)
  - configurator.UartConf, [119](#)
  - microcontroller.Adc, [13](#)
  - microcontroller.Pin, [76](#)
  - microcontroller.Uart, [114](#)
- getClockNum
  - microcontroller.Adc, [14](#)
  - microcontroller.Uart, [114](#)
- getCodeName
  - configurator.ADC.AdcChannel, [19](#)
  - configurator.AdcConf, [23](#)
  - configurator.PinConf, [93](#)
  - configurator.UartConf, [119](#)
- getCommonCfgDefinitions
  - framework.Common, [37](#)
- getCommonIncludes
  - framework.Common, [38](#)
- getConfFile
  - projectConfiguration.ProjectSettings, [102](#)
- getConfFromBoolean
  - configurator.Selected, [107](#)
- getConfFromString
  - configurator.GPIO.AltMode, [31](#)
  - configurator.GPIO.Mode, [68](#)
  - configurator.GPIO.OutLevel, [70](#)
  - configurator.GPIO.OutType, [72](#)
  - configurator.GPIO.Pull, [105](#)
  - configurator.GPIO.Speed, [109](#)
  - configurator.Selected, [108](#)
- getConfiguredPin
  - microcontroller.Microcontroller, [60](#)
- getDataBits
  - configurator.UartConf, [120](#)
  - microcontroller.Uart, [115](#)
- getDataBitsNum
  - microcontroller.Uart, [115](#)
- getEIDefs
  - framework.AdcGenerator, [30](#)
  - framework.UartGenerator, [125](#)
- getElementInfo
  - xmlParser.XmlOpener, [127](#)
- getElementInfoFromDoc
  - xmlParser.XmlOpener, [128](#)
- getElements
  - framework.AdcGenerator, [30](#)
  - framework.UartGenerator, [126](#)
- getFeat\_adc
  - microcontroller.Pin, [77](#)
- getFeat\_clock
  - microcontroller.Pin, [77](#)
- getFeat\_i2c
  - microcontroller.Pin, [77](#)
- getFeat\_int
  - microcontroller.Pin, [77](#)
- getFeat\_reset
  - microcontroller.Pin, [78](#)
- getFeat\_spi
  - microcontroller.Pin, [78](#)
- getFeat\_timer
  - microcontroller.Pin, [78](#)
- getFeat\_uart
  - microcontroller.Pin, [78](#)
- getFrameworkCommonFilePath
  - framework.Common, [38](#)
- getFrameworkIncludesFilePath
  - framework.Common, [38](#)
- getFrameworkPath

- projectConfiguration.ProjectSettings, 102
- getFunc\_gnd
  - microcontroller.Pin, 79
- getFunc\_gpio
  - microcontroller.Pin, 79
- getFunc\_misc
  - microcontroller.Pin, 79
- getFunc\_reset
  - microcontroller.Pin, 79
- getFunc\_vcc
  - microcontroller.Pin, 80
- getI2c
  - microcontroller.Pin, 80
- getIncludes
  - framework.AdcGenerator, 30
  - framework.UartGenerator, 126
- getInstallationFwkPath
  - framework.Common, 39
- getInt
  - microcontroller.Pin, 80
- getJustification
  - configurator.AdcConf, 23
  - microcontroller.Adc, 14
- getJustificationNum
  - microcontroller.Adc, 14
- getMode
  - configurator.PinConf, 93
- getName
  - configurator.ADC.AdcChannel, 19
  - microcontroller.Adc, 15
  - microcontroller.Pin, 80
  - microcontroller.Uart, 115
- getNumber
  - microcontroller.Pin, 81
- getOutLevel
  - configurator.PinConf, 94
- getOutType
  - configurator.PinConf, 94
- getParity
  - configurator.UartConf, 120
  - microcontroller.Uart, 115
- getParityNum
  - microcontroller.Uart, 116
- getParsedDoc
  - xmlParser.XmlOpener, 128
- getPin
  - microcontroller.Microcontroller, 61
- getPinIndex
  - configurator.ADC.AdcChannel, 19
- getPinName
  - configurator.PinConf, 94
- getPort
  - configurator.PinConf, 94
  - microcontroller.Pin, 81
- getPortPin
  - configurator.PinConf, 95
  - microcontroller.Pin, 81
- getPrescaler
  - configurator.AdcConf, 24
  - configurator.UartConf, 120
  - microcontroller.Adc, 15
  - microcontroller.Uart, 116
- getPrescalerNum
  - microcontroller.Adc, 15
  - microcontroller.Uart, 116
- getProjectFwkPath
  - framework.Common, 39
- getProjectName
  - projectConfiguration.ProjectSettings, 102
- getPull
  - configurator.PinConf, 95
- getReference
  - configurator.AdcConf, 24
  - microcontroller.Adc, 16
- getReferenceNum
  - microcontroller.Adc, 16
- getReset
  - microcontroller.Pin, 81
- getResolution
  - configurator.AdcConf, 24
  - microcontroller.Adc, 16
- getResolutionNum
  - microcontroller.Adc, 17
- getSample
  - configurator.AdcConf, 24
  - microcontroller.Adc, 17
- getSampleNum
  - microcontroller.Adc, 17
- getSelected
  - configurator.ADC.AdcChannel, 19
  - configurator.AdcConf, 25
  - configurator.PinConf, 95
  - configurator.UartConf, 120
- getSpeed
  - configurator.PinConf, 95
- getSpi
  - microcontroller.Pin, 82
- getStopBits
  - configurator.UartConf, 121
  - microcontroller.Uart, 117
- getStopBitsNum
  - microcontroller.Uart, 117
- getString
  - gui.Messages, 58
- getTimer
  - microcontroller.Pin, 82
- getUart
  - microcontroller.Pin, 82

- getUc\_adcNum
  - microcontroller.Microcontroller, 61
- getUc\_gpioNum
  - microcontroller.Microcontroller, 61
- getUc\_manufacturer
  - microcontroller.Microcontroller, 61
- getUc\_model
  - microcontroller.Microcontroller, 62
- getUc\_pinNum
  - microcontroller.Microcontroller, 62
- getUc\_portNum
  - microcontroller.Microcontroller, 62
- getUc\_selectedAdcsNum
  - microcontroller.Microcontroller, 62
- getUc\_selectedPinsNum
  - microcontroller.Microcontroller, 63
- getUc\_selectedUartsNum
  - microcontroller.Microcontroller, 63
- getUc\_uartNum
  - microcontroller.Microcontroller, 63
- getUcFile
  - projectConfiguration.ProjectSettings, 103
- GpioCfgPin
  - microcontroller.Microcontroller, 66
- GpioConfWindow
  - gui.GpioConfWindow, 50
- gui, 6
- gui.AboutWindow, 9
  - AboutWindow, 9
  - main, 10
- gui.AdcConfWindow, 28
  - AdcConfWindow, 28
  - main, 29
- gui.GeneralSettingsWindow, 49
  - GeneralSettingsWindow, 49
  - main, 49
- gui.GpioConfWindow, 50
  - GpioConfWindow, 50
  - main, 51
- gui.MainGui, 51
  - generateCode, 52
  - loadProjectFile, 52
  - main, 53
  - ProjectFile, 55
  - ProjectPath, 55
  - saveGeneralSettings, 53
  - saveProjectPreferences, 53
  - saveUc, 53
  - setNewUC, 54
  - showAboutWindow, 54
  - showAdcConfWindow, 54
  - showErrorDialog, 54
  - showGeneralSettingsWindow, 54
  - showGpioConfWindow, 55
  - showProjectPreferencesWindow, 55
  - showUartConfWindow, 55
- gui.MainWindow, 56
  - FrmCodeGenerator, 58
  - main, 56
  - MainWindow, 56
  - OpenFileChooser, 57
  - setProjectInformation, 57
  - setVisible, 57
- gui.Messages, 58
  - getString, 58
- gui.ProjectSettingsWindow, 104
  - main, 105
  - ProjectSettingsWindow, 104
- gui.UartConfWindow, 124
  - main, 124
  - UartConfWindow, 124
- HIGH
  - configurator.GPIO.OutLevel, 71
- Includes\_Adc
  - microcontroller.Microcontroller, 66
- Includes\_Common
  - microcontroller.Microcontroller, 66
- Includes\_Gpio
  - microcontroller.Microcontroller, 66
- Includes\_Uart
  - microcontroller.Microcontroller, 66
- INT\_INVALID\_INDEX
  - common.ErrorCode, 45
- isAv\_Adc
  - configurator.PinConf, 96
- isAv\_altFunc
  - configurator.PinConf, 96
- isAv\_I2c
  - configurator.PinConf, 96
- isAv\_Spi
  - configurator.PinConf, 96
- isAv\_Uart
  - configurator.PinConf, 97
- isValid
  - configurator.ADC.AdcChannel, 20
  - configurator.PinConf, 97
  - microcontroller.Adc, 17
  - microcontroller.Microcontroller, 63
  - microcontroller.Pin, 82
  - microcontroller.Uart, 117
- loadAdcChannelsConf
  - microcontroller.Microcontroller, 64
- loadPinsConf
  - microcontroller.Microcontroller, 64
- loadProjectFile
  - gui.MainGui, 52



- logFilePath
  - common.GeneralSettings, 49
- LOW
  - configurator.GPIO.OutLevel, 71
- main
  - gui.AboutWindow, 10
  - gui.AdcConfWindow, 29
  - gui.GeneralSettingsWindow, 49
  - gui.GpioConfWindow, 51
  - gui.MainGui, 53
  - gui.MainWindow, 56
  - gui.ProjectSettingsWindow, 105
  - gui.UartConfWindow, 124
- MainWindow
  - gui.MainWindow, 56
- MAX\_NUMBER\_OF\_ADCS
  - microcontroller.Microcontroller, 66
- MAX\_NUMBER\_OF\_PINS\_PER\_PORT
  - microcontroller.Microcontroller, 66
- MAX\_NUMBER\_OF\_UARTS
  - microcontroller.Microcontroller, 67
- MAX\_VALUE
  - configurator.GPIO.OutLevel, 71
- Microcontroller
  - microcontroller.Microcontroller, 60
- microcontroller, 7
- microcontroller.Adc, 10
  - Adc, 11
  - addChannel, 11
  - addClock, 11
  - addJustification, 11
  - addPrescaler, 12
  - addReference, 12
  - addResolution, 12
  - addSample, 13
  - getChannel, 13
  - getChannelNum, 13
  - getClock, 13
  - getClockNum, 14
  - getJustification, 14
  - getJustificationNum, 14
  - getName, 15
  - getPrescaler, 15
  - getPrescalerNum, 15
  - getReference, 16
  - getReferenceNum, 16
  - getResolution, 16
  - getResolutionNum, 17
  - getSample, 17
  - getSampleNum, 17
  - isValid, 17
  - setName, 18
- microcontroller.Microcontroller, 59
- AdcCfg, 65
- Adcs, 65
- Definitions\_Adc, 65
- Definitions\_Common, 65
- Definitions\_Gpio, 65
- Definitions\_Uart, 65
- getConfiguredPin, 60
- getPin, 61
- getUc\_adcNum, 61
- getUc\_gpioNum, 61
- getUc\_manufacturer, 61
- getUc\_model, 62
- getUc\_pinNum, 62
- getUc\_portNum, 62
- getUc\_selectedAdcsNum, 62
- getUc\_selectedPinsNum, 63
- getUc\_selectedUartsNum, 63
- getUc\_uartNum, 63
- GpioCfgPin, 66
- Includes\_Adc, 66
- Includes\_Common, 66
- Includes\_Gpio, 66
- Includes\_Uart, 66
- isValid, 63
- loadAdcChannelsConf, 64
- loadPinsConf, 64
- MAX\_NUMBER\_OF\_ADCS, 66
- MAX\_NUMBER\_OF\_PINS\_PER\_PORT, 66
- MAX\_NUMBER\_OF\_UARTS, 67
- Microcontroller, 60
- Ports, 67
- processDocument, 64
- UartCfg, 67
- Uarts, 67
- microcontroller.Pin, 74
  - DEF\_FEATURE, 90
  - DEF\_FEATURE\_AV, 90
  - DEF\_FUNCTION, 91
  - DEF\_NAME, 91
  - DEF\_NUMBER, 91
  - DEF\_PORT, 91
  - DISABLE, 91
  - ENABLE, 91
  - getAdc, 76
  - getAdcChannel, 76
  - getClock, 76
  - getFeat\_adc, 77
  - getFeat\_clock, 77
  - getFeat\_i2c, 77
  - getFeat\_int, 77
  - getFeat\_reset, 78
  - getFeat\_spi, 78
  - getFeat\_timer, 78
  - getFeat\_uart, 78

- getFunc\_gnd, [79](#)
- getFunc\_gpio, [79](#)
- getFunc\_misc, [79](#)
- getFunc\_reset, [79](#)
- getFunc\_vcc, [80](#)
- getI2c, [80](#)
- getInt, [80](#)
- getName, [80](#)
- getNumber, [81](#)
- getPort, [81](#)
- getPortPin, [81](#)
- getReset, [81](#)
- getSpi, [82](#)
- getTimer, [82](#)
- getUart, [82](#)
- isValid, [82](#)
- Pin, [76](#)
- setAdc, [83](#)
- setClock, [83](#)
- setFeat\_adc, [83](#)
- setFeat\_clock, [84](#)
- setFeat\_i2c, [84](#)
- setFeat\_int, [84](#)
- setFeat\_reset, [85](#)
- setFeat\_spi, [85](#)
- setFeat\_timer, [85](#)
- setFeat\_uart, [85](#)
- setFunc\_gnd, [86](#)
- setFunc\_gpio, [86](#)
- setFunc\_misc, [86](#)
- setFunc\_reset, [87](#)
- setFunc\_vcc, [87](#)
- setI2c, [87](#)
- setInt, [88](#)
- setName, [88](#)
- setNumber, [88](#)
- setPort, [88](#)
- setPortPin, [89](#)
- setReset, [89](#)
- setSpi, [89](#)
- setTimer, [90](#)
- setUart, [90](#)
- microcontroller.Uart, [111](#)
  - addBaudRate, [112](#)
  - addClock, [112](#)
  - addDataBits, [112](#)
  - addParity, [112](#)
  - addPrescaler, [113](#)
  - addStopBits, [113](#)
  - getBaudRate, [113](#)
  - getBaudRateNum, [114](#)
  - getClock, [114](#)
  - getClockNum, [114](#)
  - getDataBits, [115](#)
  - getDataBitsNum, [115](#)
  - getName, [115](#)
  - getParity, [115](#)
  - getParityNum, [116](#)
  - getPrescaler, [116](#)
  - getPrescalerNum, [116](#)
  - getStopBits, [117](#)
  - getStopBitsNum, [117](#)
  - isValid, [117](#)
  - setName, [117](#)
  - Uart, [111](#)
- MODE\_ALTERNATE\_FUNCTION
  - configurator.GPIO.Mode, [69](#)
- MODE\_INPUT
  - configurator.GPIO.Mode, [69](#)
- MODE\_MAX\_VALUE
  - configurator.GPIO.Mode, [69](#)
- MODE\_OUTPUT
  - configurator.GPIO.Mode, [69](#)
- NL
  - framework.Common, [40](#)
- NO\_ERROR
  - common.ErrorCode, [45](#)
- NOT
  - configurator.Selected, [108](#)
- OpenFile
  - xmlParser.XmlOpener, [128](#)
- OpenFileChooser
  - gui.MainWindow, [57](#)
- openProjectFile
  - projectConfiguration.ProjectSettings, [103](#)
- OTYPE\_MAX\_VALUE
  - configurator.GPIO.OutType, [73](#)
- OTYPE\_NOT\_AVAILABLE
  - configurator.GPIO.OutType, [73](#)
- OTYPE\_OPEN\_DRAIN
  - configurator.GPIO.OutType, [73](#)
- OTYPE\_PUSH\_PULL
  - configurator.GPIO.OutType, [73](#)
- Pin
  - microcontroller.Pin, [76](#)
- PinConf
  - configurator.PinConf, [93](#)
- Ports
  - microcontroller.Microcontroller, [67](#)
- processDocument
  - microcontroller.Microcontroller, [64](#)
  - projectConfiguration.ProjectSettings, [103](#)
- projectConfiguration, [7](#)
  - projectConfiguration.ProjectSettings, [101](#)
  - getConfFile, [102](#)
  - getFrameworkPath, [102](#)

- getProjectName, [102](#)
- getUcFile, [103](#)
- openProjectFile, [103](#)
- processDocument, [103](#)
- ProjectSettings, [102](#)
- setFrameworkPath, [104](#)
- ProjectFile
  - gui.MainGui, [55](#)
- ProjectPath
  - gui.MainGui, [55](#)
- ProjectSettings
  - projectConfiguration.ProjectSettings, [102](#)
- ProjectSettingsWindow
  - gui.ProjectSettingsWindow, [104](#)
- PULL\_DOWN
  - configurator.GPIO.Pull, [106](#)
- PULL\_MAX\_VALUE
  - configurator.GPIO.Pull, [106](#)
- PULL\_NOT\_AVAILABLE
  - configurator.GPIO.Pull, [106](#)
- PULL\_UP
  - configurator.GPIO.Pull, [106](#)
- saveGeneralSettings
  - gui.MainGui, [53](#)
- saveProjectPreferences
  - gui.MainGui, [53](#)
- saveUc
  - gui.MainGui, [53](#)
- setAdc
  - microcontroller.Pin, [83](#)
- setAltMode
  - configurator.PinConf, [97](#)
- setBaudRate
  - configurator.UartConf, [121](#)
- setChannels
  - configurator.AdcConf, [25](#)
- setClock
  - configurator.AdcConf, [25](#)
  - configurator.UartConf, [121](#)
  - microcontroller.Pin, [83](#)
- setCodeName
  - configurator.ADC.AdcChannel, [20](#)
  - configurator.AdcConf, [26](#)
  - configurator.PinConf, [98](#)
  - configurator.UartConf, [122](#)
- setDataBits
  - configurator.UartConf, [122](#)
- setFeat\_adc
  - microcontroller.Pin, [83](#)
- setFeat\_clock
  - microcontroller.Pin, [84](#)
- setFeat\_i2c
  - microcontroller.Pin, [84](#)
- setFeat\_int
  - microcontroller.Pin, [84](#)
- setFeat\_reset
  - microcontroller.Pin, [85](#)
- setFeat\_spi
  - microcontroller.Pin, [85](#)
- setFeat\_timer
  - microcontroller.Pin, [85](#)
- setFeat\_uart
  - microcontroller.Pin, [85](#)
- setFrameworkPath
  - projectConfiguration.ProjectSettings, [104](#)
- setFunc\_gnd
  - microcontroller.Pin, [86](#)
- setFunc\_gpio
  - microcontroller.Pin, [86](#)
- setFunc\_misc
  - microcontroller.Pin, [86](#)
- setFunc\_reset
  - microcontroller.Pin, [87](#)
- setFunc\_vcc
  - microcontroller.Pin, [87](#)
- setI2c
  - microcontroller.Pin, [87](#)
- setInstallationFwkPath
  - framework.Common, [39](#)
- setInt
  - microcontroller.Pin, [88](#)
- setJustification
  - configurator.AdcConf, [26](#)
- setMode
  - configurator.PinConf, [98](#)
- setName
  - microcontroller.Adc, [18](#)
  - microcontroller.Pin, [88](#)
  - microcontroller.Uart, [117](#)
- setNewUC
  - gui.MainGui, [54](#)
- setNumber
  - microcontroller.Pin, [88](#)
- setOutLevel
  - configurator.PinConf, [98](#)
- setOutType
  - configurator.PinConf, [99](#)
- setParity
  - configurator.UartConf, [122](#)
- setPort
  - microcontroller.Pin, [88](#)
- setPortPin
  - microcontroller.Pin, [89](#)
- setPrescaler
  - configurator.AdcConf, [26](#)
  - configurator.UartConf, [123](#)
- setProjectFwkPath

- framework.Common, 40
- setProjectInformation
  - gui.MainWindow, 57
- setPull
  - configurator.PinConf, 99
- setReference
  - configurator.AdcConf, 27
- setReset
  - microcontroller.Pin, 89
- setResolution
  - configurator.AdcConf, 27
- setSample
  - configurator.AdcConf, 27
- setSelected
  - configurator.ADC.AdcChannel, 20
  - configurator.AdcConf, 27
  - configurator.PinConf, 99
  - configurator.UartConf, 123
- setSpeed
  - configurator.PinConf, 99
- setSpi
  - microcontroller.Pin, 89
- setStopBits
  - configurator.UartConf, 123
- setTimer
  - microcontroller.Pin, 90
- setUart
  - microcontroller.Pin, 90
- setVisible
  - gui.MainWindow, 57
- showAboutWindow
  - gui.MainGui, 54
- showAdcConfWindow
  - gui.MainGui, 54
- showErrorDialog
  - gui.MainGui, 54
- showGeneralSettingsWindow
  - gui.MainGui, 54
- showGpioConfWindow
  - gui.MainGui, 55
- showProjectPreferencesWindow
  - gui.MainGui, 55
- showUartConfWindow
  - gui.MainGui, 55
- SPEED\_FAST
  - configurator.GPIO.Speed, 110
- SPEED\_HIGH
  - configurator.GPIO.Speed, 110
- SPEED\_MAX\_VALUE
  - configurator.GPIO.Speed, 110
- SPEED\_MEDIUM
  - configurator.GPIO.Speed, 110
- SPEED\_NOT\_AVAILABLE
  - configurator.GPIO.Speed, 110
- STR\_DEFINITION
  - framework.Common, 40
- STR\_GEN\_CODE\_NOTICE\_FOOTER
  - framework.Common, 40
- STR\_GEN\_CODE\_NOTICE\_HEADER
  - framework.Common, 40
- STR\_HEADER\_EXT
  - framework.Common, 41
- STR\_INCLUDE
  - framework.Common, 41
- STR\_INVALID
  - common.ErrorCode, 45
- STR\_MODULE\_ADC
  - framework.Common, 41
- STR\_MODULE\_GPIO
  - framework.Common, 41
- STR\_MODULE\_UART
  - framework.Common, 41
- STR\_NAME
  - configurator.GPIO.AltMode, 33
  - configurator.GPIO.CodeName, 35
  - configurator.GPIO.Mode, 69
  - configurator.GPIO.OutLevel, 71
  - configurator.GPIO.OutType, 73
  - configurator.GPIO.Pull, 106
  - configurator.GPIO.Speed, 110
  - configurator.Selected, 108
- STR\_PROJ\_CONF\_FILE
  - configurator.ConfigurationFile, 42
- SW\_VERSION
  - common.Features, 47
- Uart
  - microcontroller.Uart, 111
- UartCfg
  - microcontroller.Microcontroller, 67
- UartConf
  - configurator.UartConf, 119
- UartConfWindow
  - gui.UartConfWindow, 124
- Uarts
  - microcontroller.Microcontroller, 67
- VERBOSE
  - common.Features, 47
- VERBOSE\_STR
  - common.Features, 48
- verbosePrint
  - common.Features, 47
- VERSION\_NAME
  - common.Features, 48
- VERSION\_STATUS
  - common.Features, 48
- writeXml

xmlCreator.ConfXmlWriter, [44](#)

xmlCreator, [8](#)

xmlCreator.ConfXmlWriter, [42](#)

addPin, [43](#)

ConfXmlWriter, [43](#)

writeXml, [44](#)

XmlOpener

xmlParser.XmlOpener, [127](#)

xmlParser, [8](#)

xmlParser.XmlOpener, [127](#)

getElementInfo, [127](#)

getElementInfoFromDoc, [128](#)

getParsedDoc, [128](#)

OpenFile, [128](#)

XmlOpener, [127](#)

YES

configurator.Selected, [108](#)