

Calculatrice NPI

1.0

Généré par Doxygen 1.8.1.1

Dimanche Juin 17 2012 20 :18 :38

Table des matières

1	Index des classes	1
1.1	Hiérarchie des classes	1
2	Index des classes	5
2.1	Liste des classes	5
3	Documentation des classes	9
3.1	Référence de la classe calculator : :Add	9
3.1.1	Description détaillée	9
3.1.2	Documentation des fonctions membres	9
3.1.2.1	clone	9
3.2	Référence de la classe calculator : :AllParametersCommand	10
3.2.1	Description détaillée	10
3.2.2	Documentation des fonctions membres	11
3.2.2.1	apply	11
3.2.2.2	clone	11
3.2.2.3	createMemento	11
3.2.2.4	restoreFromMemento	11
3.3	Référence de la classe calculator : :AngleUnitParameterMemento	12
3.3.1	Description détaillée	12
3.4	Référence de la classe calculator : :AngularUnaryOperator	12
3.4.1	Description détaillée	13
3.4.2	Documentation des fonctions membres	13
3.4.2.1	clone	13
3.4.2.2	createMemento	13
3.4.2.3	isExecutable	13
3.5	Référence de la classe calculator : :ArithmeticException	14
3.5.1	Description détaillée	14
3.6	Référence de la classe calculator : :BinaryOperator	14
3.6.1	Description détaillée	15
3.6.2	Documentation des fonctions membres	15
3.6.2.1	clone	15

3.6.2.2	createMemento	15
3.6.2.3	isExecutable	15
3.7	Référence de la classe calculator : :BooleanParameterMemento	16
3.7.1	Description détaillée	16
3.8	Référence de la classe Ui : :CalculatorMainWindow	16
3.8.1	Description détaillée	16
3.9	Référence de la classe calculator : :CalculatorMainWindow	17
3.9.1	Description détaillée	18
3.9.2	Documentation des constructeurs et destructeur	18
3.9.2.1	CalculatorMainWindow	18
3.9.3	Documentation des fonctions membres	18
3.9.3.1	getEngine	18
3.9.3.2	init	18
3.9.3.3	prepareForShutdown	19
3.9.3.4	setEngine	19
3.10	Référence de la classe calculator : :CalculatorParser	19
3.10.1	Description détaillée	19
3.10.2	Documentation des constructeurs et destructeur	20
3.10.2.1	CalculatorParser	20
3.10.2.2	~CalculatorParser	20
3.10.3	Documentation des fonctions membres	20
3.10.3.1	isCommand	20
3.10.3.2	isComplex	20
3.10.3.3	isExpression	20
3.10.3.4	isInteger	21
3.10.3.5	isRationnal	21
3.10.3.6	isReal	21
3.10.3.7	isSimpleNumber	21
3.10.3.8	parse	22
3.11	Référence de la classe calculator : :Clear	22
3.11.1	Description détaillée	23
3.11.2	Documentation des fonctions membres	23
3.11.2.1	clone	23
3.11.2.2	createMemento	23
3.11.2.3	isExecutable	23
3.11.2.4	restoreFromMemento	23
3.12	Référence de la classe calculator : :Command	24
3.12.1	Description détaillée	24
3.12.2	Documentation des constructeurs et destructeur	25
3.12.2.1	Command	25

3.12.3 Documentation des fonctions membres	25
3.12.3.1 clone	25
3.12.3.2 execute	25
3.12.3.3 isExecutable	25
3.12.3.4 toString	26
3.13 Référence de la classe calculator : :CommandException	26
3.13.1 Description détaillée	27
3.14 Référence de la classe calculator : :CommandManager	27
3.14.1 Description détaillée	27
3.14.2 Documentation des constructeurs et destructeur	27
3.14.2.1 CommandManager	27
3.14.2.2 ~CommandManager	28
3.14.3 Documentation des fonctions membres	28
3.14.3.1 executeCommand	28
3.15 Référence de la classe calculator : :CommandMap	28
3.15.1 Description détaillée	28
3.15.2 Documentation des fonctions membres	28
3.15.2.1 deleteInstance	28
3.15.2.2 getCommand	29
3.15.2.3 getInstance	29
3.15.2.4 isCommandName	29
3.16 Référence de la classe calculator : :Complex	29
3.16.1 Description détaillée	30
3.16.2 Documentation des fonctions membres	30
3.16.2.1 clone	30
3.16.2.2 isExecutable	31
3.16.2.3 toString	31
3.17 Référence de la classe calculator : :ComplexConverter	31
3.17.1 Description détaillée	32
3.17.2 Documentation des fonctions membres	32
3.17.2.1 clone	32
3.17.2.2 isExecutable	32
3.18 Référence de la classe calculator : :ComplexParameterCommand	32
3.18.1 Description détaillée	33
3.18.2 Documentation des fonctions membres	33
3.18.2.1 apply	33
3.18.2.2 clone	33
3.18.2.3 createMemento	33
3.18.2.4 restoreFromMemento	34
3.19 Référence de la classe calculator : :Constant	34

3.19.1	Description détaillée	35
3.19.2	Documentation des fonctions membres	35
3.19.2.1	clone	35
3.20	Référence de la classe calculator : :ConstantTypeParameterMemento	35
3.20.1	Description détaillée	35
3.21	Référence de la classe calculator : :Context	36
3.21.1	Description détaillée	36
3.21.2	Documentation des fonctions membres	36
3.21.2.1	deleteInstance	36
3.21.2.2	getInstance	37
3.21.2.3	saveContextToXML	37
3.22	Référence de la classe calculator : :ContextException	37
3.22.1	Description détaillée	37
3.23	Référence de la classe calculator : :ContextMemento	38
3.23.1	Description détaillée	38
3.24	Référence de la classe calculator : :Cos	38
3.24.1	Description détaillée	39
3.24.2	Documentation des fonctions membres	39
3.24.2.1	clone	39
3.25	Référence de la classe calculator : :Cosh	39
3.25.1	Description détaillée	40
3.25.2	Documentation des fonctions membres	40
3.25.2.1	clone	40
3.26	Référence de la classe calculator : :Cube	40
3.26.1	Description détaillée	41
3.26.2	Documentation des fonctions membres	41
3.26.2.1	clone	41
3.26.2.2	isExecutable	41
3.27	Référence de la classe calculator : :CurrentStackMemento	42
3.27.1	Description détaillée	42
3.28	Référence de la classe calculator : :DegreeParameterCommand	42
3.28.1	Description détaillée	43
3.28.2	Documentation des fonctions membres	43
3.28.2.1	apply	43
3.28.2.2	clone	43
3.28.2.3	createMemento	43
3.28.2.4	restoreFromMemento	43
3.29	Référence de la classe calculator : :Div	44
3.29.1	Description détaillée	44
3.29.2	Documentation des fonctions membres	44

3.29.2.1	clone	44
3.30	Référence de la classe calculator : :Drop	45
3.30.1	Description détaillée	45
3.30.2	Documentation des fonctions membres	45
3.30.2.1	clone	45
3.30.2.2	createMemento	46
3.30.2.3	isExecutable	46
3.30.2.4	restoreFromMemento	46
3.31	Référence de la classe calculator : :Dup	47
3.31.1	Description détaillée	47
3.31.2	Documentation des fonctions membres	47
3.31.2.1	clone	47
3.31.2.2	createMemento	48
3.31.2.3	isExecutable	48
3.31.2.4	restoreFromMemento	48
3.32	Référence de la classe calculator : :DuplicateStackCommand	48
3.32.1	Description détaillée	49
3.32.2	Documentation des fonctions membres	49
3.32.2.1	clone	49
3.32.2.2	createMemento	49
3.32.2.3	isExecutable	50
3.32.2.4	restoreFromMemento	50
3.33	Référence de la classe calculator : :EmptyMemento	50
3.33.1	Description détaillée	51
3.34	Référence de la classe calculator : :Engine	51
3.34.1	Description détaillée	51
3.34.2	Documentation des constructeurs et destructeur	52
3.34.2.1	Engine	52
3.34.2.2	~Engine	52
3.34.3	Documentation des fonctions membres	52
3.34.3.1	drop	52
3.34.3.2	dupStack	52
3.34.3.3	getContext	52
3.34.3.4	newStack	53
3.34.3.5	redo	53
3.34.3.6	removeCurrentStack	53
3.34.3.7	run	53
3.34.3.8	saveAllParameters	53
3.34.3.9	setAngleUnitTo	54
3.34.3.10	setConstantTypeTo	54

3.34.3.11	setCurrentStack	54
3.34.3.12	setInputText	54
3.34.3.13	undo	54
3.35	Référence de la classe calculator : :Eval	55
3.35.1	Description détaillée	55
3.35.2	Documentation des fonctions membres	55
3.35.2.1	clone	55
3.35.2.2	createMemento	56
3.35.2.3	execute	56
3.35.2.4	isExecutable	56
3.35.2.5	restoreFromMemento	56
3.36	Référence de la classe calculator : :Expression	57
3.36.1	Description détaillée	57
3.36.2	Documentation des constructeurs et destructeur	58
3.36.2.1	Expression	58
3.36.2.2	Expression	58
3.36.2.3	Expression	58
3.36.2.4	Expression	58
3.36.2.5	~Expression	59
3.36.3	Documentation des fonctions membres	59
3.36.3.1	append	59
3.36.3.2	clone	59
3.36.3.3	empty	59
3.36.3.4	getFirstCommand	59
3.36.3.5	toString	59
3.37	Référence de la classe calculator : :Fact	60
3.37.1	Description détaillée	60
3.37.2	Documentation des fonctions membres	60
3.37.2.1	clone	60
3.37.2.2	isExecutable	61
3.38	Référence de la classe calculator : :InstantComputeParametersCommand	61
3.38.1	Description détaillée	62
3.38.2	Documentation des fonctions membres	62
3.38.2.1	apply	62
3.38.2.2	clone	62
3.38.2.3	createMemento	62
3.38.2.4	restoreFromMemento	62
3.39	Référence de la classe calculator : :Integer	63
3.39.1	Description détaillée	64
3.39.2	Documentation des fonctions membres	64

3.39.2.1	clone	64
3.39.2.2	toString	64
3.40	Référence de la classe calculator : :IntegerConverter	64
3.40.1	Description détaillée	65
3.40.2	Documentation des fonctions membres	65
3.40.2.1	clone	65
3.40.2.2	isExecutable	65
3.41	Référence de la classe calculator : :IntegerDivisionParameterCommand	65
3.41.1	Description détaillée	66
3.41.2	Documentation des fonctions membres	66
3.41.2.1	apply	66
3.41.2.2	clone	66
3.41.2.3	createMemento	67
3.41.2.4	restoreFromMemento	67
3.42	Référence de la classe calculator : :IntegerParameterCommand	67
3.42.1	Description détaillée	68
3.42.2	Documentation des fonctions membres	68
3.42.2.1	apply	68
3.42.2.2	clone	68
3.42.2.3	createMemento	68
3.42.2.4	restoreFromMemento	69
3.43	Référence de la classe calculator : :Inv	69
3.43.1	Description détaillée	69
3.43.2	Documentation des fonctions membres	69
3.43.2.1	clone	69
3.44	Référence de la classe calculator : :KeyboardParameterCommand	70
3.44.1	Description détaillée	70
3.44.2	Documentation des fonctions membres	70
3.44.2.1	apply	71
3.44.2.2	clone	71
3.44.2.3	createMemento	71
3.44.2.4	restoreFromMemento	71
3.45	Référence de la classe calculator : :Literal	71
3.45.1	Description détaillée	72
3.45.2	Documentation des constructeurs et destructeur	72
3.45.2.1	Literal	72
3.45.3	Documentation des fonctions membres	72
3.45.3.1	clone	72
3.45.3.2	execute	73
3.45.3.3	isExecutable	73

3.45.3.4	toString	73
3.46	Référence de la classe calculator : :Ln	73
3.46.1	Description détaillée	74
3.46.2	Documentation des fonctions membres	74
3.46.2.1	clone	74
3.46.2.2	isExecutable	74
3.47	Référence de la classe calculator : :Log	75
3.47.1	Description détaillée	75
3.47.2	Documentation des fonctions membres	75
3.47.2.1	clone	75
3.47.2.2	isExecutable	76
3.48	Référence de la classe calculator : :Logger	76
3.48.1	Description détaillée	76
3.49	Référence de la classe calculator : :LoggerManager	77
3.49.1	Description détaillée	77
3.49.2	Documentation des fonctions membres	77
3.49.2.1	deleteInstance	77
3.49.2.2	getInstance	77
3.49.2.3	getLogger	77
3.50	Référence de la classe calculator : :LogMessage	78
3.50.1	Description détaillée	78
3.50.2	Documentation des constructeurs et destructeur	78
3.50.2.1	LogMessage	78
3.50.2.2	~LogMessage	79
3.50.3	Documentation des fonctions membres	79
3.50.3.1	getClassname	79
3.50.3.2	getLevel	79
3.50.3.3	getMessage	79
3.51	Référence de la classe calculator : :LogSystem	79
3.51.1	Description détaillée	80
3.51.2	Documentation des fonctions membres	80
3.51.2.1	log	80
3.52	Référence de la classe calculator : :Mean	80
3.52.1	Description détaillée	81
3.52.2	Documentation des fonctions membres	81
3.52.2.1	clone	81
3.53	Référence de la classe calculator : :Memento	81
3.53.1	Description détaillée	83
3.53.2	Documentation des constructeurs et destructeur	83
3.53.2.1	Memento	83

3.53.3	Documentation des fonctions membres	83
3.53.3.1	getInputString	83
3.53.3.2	getUndoableRedoableCommand	83
3.54	Référence de la classe calculator : :MementoCaretaker	84
3.54.1	Description détaillée	84
3.55	Référence de la classe calculator : :MementoException	84
3.55.1	Description détaillée	84
3.56	Référence de la classe calculator : :Mod	84
3.56.1	Description détaillée	85
3.56.2	Documentation des fonctions membres	85
3.56.2.1	clone	85
3.56.2.2	isExecutable	85
3.57	Référence de la classe calculator : :Mul	86
3.57.1	Description détaillée	86
3.57.2	Documentation des fonctions membres	86
3.57.2.1	clone	86
3.58	Référence de la classe calculator : :NaryOperator	87
3.58.1	Description détaillée	87
3.58.2	Documentation des fonctions membres	87
3.58.2.1	clone	87
3.58.2.2	createMemento	88
3.58.2.3	isExecutable	88
3.59	Référence de la classe calculator : :NewStackCommand	88
3.59.1	Description détaillée	89
3.59.2	Documentation des fonctions membres	89
3.59.2.1	clone	89
3.59.2.2	createMemento	89
3.59.2.3	isExecutable	89
3.59.2.4	restoreFromMemento	90
3.60	Référence de la classe calculator : :NullaryOperator	90
3.60.1	Description détaillée	91
3.60.2	Documentation des fonctions membres	91
3.60.2.1	clone	91
3.60.2.2	createMemento	91
3.60.2.3	isExecutable	91
3.60.2.4	restoreFromMemento	92
3.61	Référence de la classe calculator : :Number	92
3.61.1	Description détaillée	92
3.61.2	Documentation des fonctions membres	92
3.61.2.1	clone	92

3.62	Référence de la classe calculator : :Observable	93
3.62.1	Description détaillée	93
3.62.2	Documentation des fonctions membres	93
3.62.2.1	addObserver	93
3.62.2.2	removeObserver	94
3.63	Référence de la classe calculator : :Observer	94
3.63.1	Description détaillée	94
3.64	Référence de la classe calculator : :Operator	94
3.64.1	Description détaillée	95
3.64.2	Documentation des constructeurs et destructeur	95
3.64.2.1	Operator	95
3.64.3	Documentation des fonctions membres	95
3.64.3.1	clone	95
3.64.3.2	createMemento	96
3.64.3.3	execute	96
3.64.3.4	isExecutable	96
3.64.3.5	restoreFromMemento	97
3.65	Référence de la classe calculator : :Parameters	97
3.65.1	Description détaillée	98
3.65.2	Documentation des énumérations membres	98
3.65.2.1	AngleUnit	98
3.65.2.2	ConstantType	98
3.65.3	Documentation des constructeurs et destructeur	98
3.65.3.1	Parameters	98
3.65.4	Documentation des fonctions membres	99
3.65.4.1	clone	99
3.65.4.2	deleteInstance	99
3.65.4.3	getInstance	99
3.66	Référence de la classe calculator : :ParametersCommand	99
3.66.1	Description détaillée	100
3.66.2	Documentation des constructeurs et destructeur	100
3.66.2.1	ParametersCommand	101
3.66.3	Documentation des fonctions membres	101
3.66.3.1	apply	101
3.66.3.2	clone	101
3.66.3.3	createMemento	101
3.66.3.4	execute	102
3.66.3.5	isExecutable	102
3.66.3.6	restoreFromMemento	102
3.67	Référence de la classe calculator : :ParametersDialog	103

3.67.1	Description détaillée	103
3.67.2	Documentation des constructeurs et destructeur	103
3.67.2.1	ParametersDialog	103
3.68	Référence de la classe Ui : :ParametersDialog	104
3.68.1	Description détaillée	104
3.69	Référence de la classe calculator : :ParametersMemento	104
3.69.1	Description détaillée	104
3.70	Référence de la classe calculator : :ParseException	105
3.70.1	Description détaillée	105
3.71	Référence de la classe calculator : :Pow	105
3.71.1	Description détaillée	106
3.71.2	Documentation des fonctions membres	106
3.71.2.1	clone	106
3.71.2.2	isExecutable	106
3.72	Référence de la classe calculator : :RadianParameterCommand	106
3.72.1	Description détaillée	107
3.72.2	Documentation des fonctions membres	107
3.72.2.1	apply	107
3.72.2.2	clone	107
3.72.2.3	createMemento	107
3.72.2.4	restoreFromMemento	108
3.73	Référence de la classe calculator : :Rational	108
3.73.1	Description détaillée	109
3.73.2	Documentation des fonctions membres	109
3.73.2.1	clone	109
3.73.2.2	toString	109
3.74	Référence de la classe calculator : :RationalConverter	109
3.74.1	Description détaillée	110
3.74.2	Documentation des fonctions membres	110
3.74.2.1	clone	110
3.74.2.2	isExecutable	110
3.75	Référence de la classe calculator : :RationalParameterCommand	111
3.75.1	Description détaillée	111
3.75.2	Documentation des fonctions membres	111
3.75.2.1	apply	111
3.75.2.2	clone	112
3.75.2.3	createMemento	112
3.75.2.4	restoreFromMemento	112
3.76	Référence de la classe calculator : :Real	112
3.76.1	Description détaillée	113

3.76.2	Documentation des fonctions membres	113
3.76.2.1	clone	113
3.76.2.2	toString	113
3.77	Référence de la classe calculator : :RealConverter	114
3.77.1	Description détaillée	114
3.77.2	Documentation des fonctions membres	114
3.77.2.1	clone	114
3.77.2.2	isExecutable	115
3.78	Référence de la classe calculator : :RealParameterCommand	115
3.78.1	Description détaillée	116
3.78.2	Documentation des fonctions membres	116
3.78.2.1	apply	116
3.78.2.2	clone	116
3.78.2.3	createMemento	116
3.78.2.4	restoreFromMemento	116
3.79	Référence de la classe calculator : :RedoCommand	117
3.79.1	Description détaillée	117
3.79.2	Documentation des fonctions membres	117
3.79.2.1	clone	117
3.79.2.2	execute	118
3.79.2.3	isExecutable	118
3.80	Référence de la classe calculator : :RemoveStackCommand	118
3.80.1	Description détaillée	119
3.80.2	Documentation des fonctions membres	119
3.80.2.1	clone	119
3.80.2.2	createMemento	119
3.80.2.3	isExecutable	119
3.80.2.4	restoreFromMemento	120
3.81	Référence de la classe calculator : :RemoveStackMemento	120
3.81.1	Description détaillée	120
3.82	Référence de la classe calculator : :SaveOnExitCommand	120
3.82.1	Description détaillée	121
3.82.2	Documentation des fonctions membres	121
3.82.2.1	apply	121
3.82.2.2	clone	121
3.82.2.3	createMemento	122
3.82.2.4	restoreFromMemento	122
3.83	Référence de la classe calculator : :Sign	122
3.83.1	Description détaillée	123
3.83.2	Documentation des fonctions membres	123

3.83.2.1	clone	123
3.83.2.2	isExecutable	123
3.84	Référence de la classe calculator : :SimpleNumber	123
3.84.1	Description détaillée	124
3.84.2	Documentation des fonctions membres	124
3.84.2.1	clone	124
3.84.2.2	operator double	124
3.84.2.3	toString	124
3.85	Référence de la classe calculator : :Sin	125
3.85.1	Description détaillée	125
3.85.2	Documentation des fonctions membres	125
3.85.2.1	clone	125
3.86	Référence de la classe calculator : :Sinh	126
3.86.1	Description détaillée	126
3.86.2	Documentation des fonctions membres	126
3.86.2.1	clone	126
3.87	Référence de la classe calculator : :Sqr	127
3.87.1	Description détaillée	127
3.87.2	Documentation des fonctions membres	127
3.87.2.1	clone	127
3.87.2.2	isExecutable	128
3.88	Référence de la classe calculator : :Sqrt	128
3.88.1	Description détaillée	129
3.88.2	Documentation des fonctions membres	129
3.88.2.1	clone	129
3.88.2.2	isExecutable	129
3.89	Référence de la classe calculator : :Stack	129
3.89.1	Description détaillée	130
3.89.2	Documentation des constructeurs et destructeur	130
3.89.2.1	Stack	130
3.89.2.2	~Stack	130
3.89.3	Documentation des fonctions membres	131
3.89.3.1	clone	131
3.89.3.2	empty	131
3.89.3.3	getConstant	131
3.89.3.4	push	131
3.89.3.5	size	131
3.89.3.6	swapAnyConstant	132
3.89.3.7	top	132
3.89.4	Documentation des fonctions amies et associées	132

3.89.4.1	XMLParser	132
3.90	Référence de la classe calculator : :StackList	132
3.90.1	Description détaillée	133
3.90.2	Documentation des fonctions membres	133
3.90.2.1	clone	133
3.90.2.2	deleteInstance	133
3.90.2.3	duplicateCurrentStack	133
3.90.2.4	empty	133
3.90.2.5	getCurrentStack	134
3.90.2.6	getCurrentStackIndex	134
3.90.2.7	getInstance	134
3.90.2.8	getStack	134
3.90.2.9	insertStackAtIndex	134
3.90.2.10	newStack	135
3.90.2.11	removeCurrentStack	135
3.90.2.12	setCurrentStack	135
3.90.2.13	size	135
3.90.3	Documentation des fonctions amies et associées	135
3.90.3.1	XMLParser	135
3.91	Référence de la classe calculator : :StackListCommand	136
3.91.1	Description détaillée	136
3.91.2	Documentation des constructeurs et destructeur	136
3.91.2.1	StackListCommand	136
3.91.3	Documentation des fonctions membres	137
3.91.3.1	apply	137
3.91.3.2	clone	137
3.91.3.3	createMemento	137
3.91.3.4	execute	137
3.91.3.5	isExecutable	137
3.91.3.6	restoreFromMemento	138
3.92	Référence de la classe calculator : :StackMemento	138
3.92.1	Description détaillée	138
3.93	Référence de la classe Ui : :StackTab	139
3.93.1	Description détaillée	139
3.94	Référence de la classe calculator : :StackTabList	139
3.94.1	Description détaillée	139
3.94.2	Documentation des constructeurs et destructeur	140
3.94.2.1	StackTabList	140
3.94.2.2	~StackTabList	140
3.94.3	Documentation des fonctions membres	140

3.94.3.1	duplicateCurrentTab	140
3.94.3.2	getCurrentTab	140
3.94.3.3	getCurrentTabIndex	140
3.94.3.4	getTab	140
3.94.3.5	newTab	141
3.94.3.6	removeCurrentTab	141
3.94.3.7	setCurrentTab	141
3.94.3.8	size	141
3.95	Référence de la classe calculator : :StackTabWidget	141
3.95.1	Description détaillée	142
3.95.2	Documentation des constructeurs et destructeur	142
3.95.2.1	StackTabWidget	142
3.95.2.2	~StackTabWidget	142
3.96	Référence de la classe calculator : :Sub	142
3.96.1	Description détaillée	143
3.96.2	Documentation des fonctions membres	143
3.96.2.1	clone	143
3.97	Référence de la classe calculator : :Sum	143
3.97.1	Description détaillée	144
3.97.2	Documentation des fonctions membres	144
3.97.2.1	clone	144
3.98	Référence de la classe calculator : :Swap	144
3.98.1	Description détaillée	145
3.98.2	Documentation des fonctions membres	145
3.98.2.1	clone	145
3.98.2.2	createMemento	145
3.98.2.3	isExecutable	146
3.99	Référence de la classe calculator : :SwitchCurrentStackCommand	146
3.99.1	Description détaillée	146
3.99.2	Documentation des fonctions membres	147
3.99.2.1	clone	147
3.99.2.2	createMemento	147
3.99.2.3	isExecutable	147
3.99.2.4	restoreFromMemento	147
3.100	Référence de la classe calculator : :SwitchCurrentStackMemento	148
3.100.1	Description détaillée	148
3.101	Référence de la classe calculator : :Tan	148
3.101.1	Description détaillée	149
3.101.2	Documentation des fonctions membres	149
3.101.2.1	clone	149

3.102	Référence de la classe calculator : :Tanh	149
3.102.1	Description détaillée	150
3.102.2	Documentation des fonctions membres	150
3.102.2.1	clone	150
3.103	Référence de la structure calculator : :toupper	150
3.103.1	Description détaillée	150
3.103.2	Documentation des fonctions membres	151
3.103.2.1	operator()	151
3.104	Référence de la classe Ui_CalculatorMainWindow	151
3.104.1	Description détaillée	152
3.105	Référence de la classe Ui_ParametersDialog	153
3.105.1	Description détaillée	153
3.106	Référence de la classe Ui_StackTab	153
3.106.1	Description détaillée	154
3.107	Référence de la classe calculator : :UnaryOperator	154
3.107.1	Description détaillée	155
3.107.2	Documentation des fonctions membres	155
3.107.2.1	clone	155
3.107.2.2	createMemento	155
3.107.2.3	isExecutable	155
3.108	Référence de la classe calculator : :UndoableRedoableCommand	156
3.108.1	Description détaillée	156
3.108.2	Documentation des constructeurs et destructeur	157
3.108.2.1	UndoableRedoableCommand	157
3.108.3	Documentation des fonctions membres	157
3.108.3.1	clone	157
3.108.3.2	createMemento	157
3.108.3.3	execute	158
3.108.3.4	isExecutable	158
3.108.3.5	redo	158
3.108.3.6	restoreFromMemento	159
3.109	Référence de la classe calculator : :UndoCommand	159
3.109.1	Description détaillée	159
3.109.2	Documentation des fonctions membres	159
3.109.2.1	clone	160
3.109.2.2	execute	160
3.109.2.3	isExecutable	160
3.110	Référence de la classe calculator : :ViewException	160
3.110.1	Description détaillée	161
3.111	Référence de la classe calculator : :VisibleStackSizeParameterCommand	161

3.111.1 Description détaillée	161
3.111.2 Documentation des fonctions membres	162
3.111.2.1 clone	162
3.111.2.2 createMemento	162
3.111.2.3 execute	162
3.111.2.4 isExecutable	162
3.111.2.5 restoreFromMemento	163
3.112Référence de la classe calculator : :VisibleStackSizeParameterMemento	163
3.112.1 Description détaillée	163
3.113Référence de la classe calculator : :XMLParser	163
3.113.1 Description détaillée	164
3.113.2 Documentation des fonctions membres	164
3.113.2.1 deleteInstance	164
3.113.2.2 getInstance	164
3.113.2.3 loadContextFromFile	164
3.113.2.4 saveContextToFile	164

Chapitre 1

Index des classes

1.1 Hiérarchie des classes

Cette liste d'héritage est classée approximativement par ordre alphabétique :

calculator : :CalculatorMainWindow	17
calculator : :CalculatorParser	19
calculator : :Command	24
calculator : :RedoCommand	117
calculator : :UndoableRedoableCommand	156
calculator : :Eval	55
calculator : :Literal	71
calculator : :Constant	34
calculator : :Expression	57
calculator : :Number	92
calculator : :Complex	29
calculator : :SimpleNumber	123
calculator : :Integer	63
calculator : :Rationnal	108
calculator : :Real	112
calculator : :Operator	94
calculator : :AngularUnaryOperator	12
calculator : :Cos	38
calculator : :Cosh	39
calculator : :Sin	125
calculator : :Sinh	126
calculator : :Tan	148
calculator : :Tanh	149
calculator : :BinaryOperator	14
calculator : :Add	9
calculator : :Div	44
calculator : :Mod	84
calculator : :Mul	86
calculator : :Pow	105
calculator : :Sub	142
calculator : :Dup	47
calculator : :NaryOperator	87
calculator : :Mean	80
calculator : :Sum	143
calculator : :NullaryOperator	90
calculator : :Clear	22

calculator : :Drop	45
calculator : :Swap	144
calculator : :UnaryOperator	154
calculator : :ComplexConverter	31
calculator : :Cube	40
calculator : :Fact	60
calculator : :IntegerConverter	64
calculator : :Inv	69
calculator : :Ln	73
calculator : :Log	75
calculator : :RationalConverter	109
calculator : :RealConverter	114
calculator : :Sign	122
calculator : :Sqr	127
calculator : :Sqrt	128
calculator : :ParametersCommand	99
calculator : :AllParametersCommand	10
calculator : :ComplexParameterCommand	32
calculator : :DegreeParameterCommand	42
calculator : :InstantComputeParametersCommand	61
calculator : :IntegerDivisionParameterCommand	65
calculator : :IntegerParameterCommand	67
calculator : :KeyboardParameterCommand	70
calculator : :RadianParameterCommand	106
calculator : :RationalParameterCommand	111
calculator : :RealParameterCommand	115
calculator : :SaveOnExitCommand	120
calculator : :VisibleStackSizeParameterCommand	161
calculator : :StackListCommand	136
calculator : :DuplicateStackCommand	48
calculator : :NewStackCommand	88
calculator : :RemoveStackCommand	118
calculator : :SwitchCurrentStackCommand	146
calculator : :UndoCommand	159
calculator : :CommandException	26
calculator : :ArithmeticException	14
calculator : :CommandManager	27
calculator : :CommandMap	28
calculator : :Context	36
calculator : :ContextException	37
calculator : :Engine	51
calculator : :Logger	76
calculator : :LoggerManager	77
calculator : :LogMessage	78
calculator : :LogSystem	79
calculator : :Memento	81
calculator : :AngleUnitParameterMemento	12
calculator : :BooleanParameterMemento	16
calculator : :ConstantTypeParameterMemento	35
calculator : :ContextMemento	38
calculator : :CurrentStackMemento	42
calculator : :EmptyMemento	50
calculator : :ParametersMemento	104
calculator : :RemoveStackMemento	120
calculator : :StackMemento	138
calculator : :SwitchCurrentStackMemento	148
calculator : :VisibleStackSizeParameterMemento	163

calculator : :MementoCaretaker	84
calculator : :MementoException	84
calculator : :Observable	93
calculator : :Stack	129
calculator : :Observer	94
calculator : :StackTabWidget	141
calculator : :Parameters	97
calculator : :ParametersDialog	103
calculator : :ParseException	105
calculator : :StackList	132
calculator : :StackTabList	139
calculator : :toupper	150
Ui_CalculatorMainWindow	151
Ui : :CalculatorMainWindow	16
Ui_ParametersDialog	153
Ui : :ParametersDialog	104
Ui_StackTab	153
Ui : :StackTab	139
calculator : :ViewException	160
calculator : :XMLParser	163

Chapitre 2

Index des classes

2.1 Liste des classes

Liste des classes, structures, unions et interfaces avec une brève description :

calculator : :Add	9
calculator : :AllParametersCommand	10
calculator : :AngleUnitParameterMemento	12
calculator : :AngularUnaryOperator	12
calculator : :ArithmeticException	
Classe d'exception arithmétique	14
calculator : :BinaryOperator	14
calculator : :BooleanParameterMemento	16
Ui : :CalculatorMainWindow	16
calculator : :CalculatorMainWindow	
La fenêtre principale de l'application	17
calculator : :CalculatorParser	
Parseur de l'application CalculatriceNPI	19
calculator : :Clear	22
calculator : :Command	
Classe de base des commandes	24
calculator : :CommandException	
Classe d'exception pour les Command	26
calculator : :CommandManager	
Le gestionnaire de commande de l'application	27
calculator : :CommandMap	
Classe conteneur des Command fixes de l'application	28
calculator : :Complex	
Nombre Complexe	29
calculator : :ComplexConverter	31
calculator : :ComplexParameterCommand	32
calculator : :Constant	
Element de base des Stack	34
calculator : :ConstantTypeParameterMemento	35
calculator : :Context	
Classe de base du modèle dans le schéma MVC	36
calculator : :ContextException	
Classe d'exception pour la couche Model	37
calculator : :ContextMemento	38
calculator : :Cos	38
calculator : :Cosh	39
calculator : :Cube	40
calculator : :CurrentStackMemento	42

calculator : :DegreeParameterCommand	42
calculator : :Div	44
calculator : :Drop	45
calculator : :Dup	47
calculator : :DuplicateStackCommand	48
calculator : :EmptyMemento	50
calculator : :Engine	
Moteur principal de l'application	51
calculator : :Eval	55
calculator : :Expression	
Classe Expression , composée de Command	57
calculator : :Fact	60
calculator : :InstantComputeParametersCommand	61
calculator : :Integer	63
calculator : :IntegerConverter	64
calculator : :IntegerDivisionParameterCommand	65
calculator : :IntegerParameterCommand	67
calculator : :Inv	69
calculator : :KeyboardParameterCommand	70
calculator : :Literal	
Classe mère des constantes	71
calculator : :Ln	73
calculator : :Log	75
calculator : :Logger	
Logger utilisé par le système de journalisation	76
calculator : :LoggerManager	
Classe conteneur de Logger	77
calculator : :LogMessage	
Un LogMessage est une entrée de journalisation	78
calculator : :LogSystem	
Classe centrale du système de journalisation	79
calculator : :Mean	80
calculator : :Memento	
Classe de base des Memento	81
calculator : :MementoCaretaker	84
calculator : :MementoException	
Classe d'exception pour les Memento	84
calculator : :Mod	84
calculator : :Mul	86
calculator : :NaryOperator	87
calculator : :NewStackCommand	88
calculator : :NullaryOperator	90
calculator : :Number	92
calculator : :Observable	
Design Pattern Observer	93
calculator : :Observer	
Design Pattern Observer	94
calculator : :Operator	94
calculator : :Parameters	97
calculator : :ParametersCommand	99
calculator : :ParametersDialog	
Fenêtre affichant l'ensemble des paramètres de l'application	103
Ui : :ParametersDialog	104
calculator : :ParametersMemento	104
calculator : :ParseException	
Classe d'exception de parsing Exception lancée si CalculatorParser ne parvient pas à traduire une chaîne de caractères	105
calculator : :Pow	105

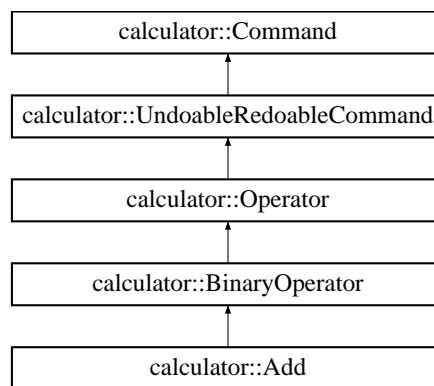
calculator : :RadianParameterCommand	106
calculator : :Rationnal	108
calculator : :RationnalConverter	109
calculator : :RationnalParameterCommand	111
calculator : :Real	112
calculator : :RealConverter	114
calculator : :RealParameterCommand	115
calculator : :RedoCommand	117
calculator : :RemoveStackCommand	118
calculator : :RemoveStackMemento	120
calculator : :SaveOnExitCommand	120
calculator : :Sign	122
calculator : :SimpleNumber	123
calculator : :Sin	125
calculator : :Sinh	126
calculator : :Sqr	127
calculator : :Sqrt	128
calculator : :Stack	
Classe conteneur de Constant	129
calculator : :StackList	
Classe conteneur de Stack	132
calculator : :StackListCommand	136
calculator : :StackMemento	138
Ui : :StackTab	139
calculator : :StackTabList	
Classe conteneur de StackTabWidget	139
calculator : :StackTabWidget	
Elément du TabWidget spécialisé pour afficher les Stack	141
calculator : :Sub	142
calculator : :Sum	143
calculator : :Swap	144
calculator : :SwitchCurrentStackCommand	146
calculator : :SwitchCurrentStackMemento	148
calculator : :Tan	148
calculator : :Tanh	149
calculator : :toupper	
Foncteur permettant de transformer une chaîne de caractère en majuscules	150
Ui_CalculatorMainWindow	151
Ui_ParametersDialog	153
Ui_StackTab	153
calculator : :UnaryOperator	154
calculator : :UndoableRedoableCommand	
Classe de base des commandes annulables/réexécutables	156
calculator : :UndoCommand	159
calculator : :ViewException	
Classe d'exception pour la couche View	160
calculator : :VisibleStackSizeParameterCommand	161
calculator : :VisibleStackSizeParameterMemento	163
calculator : :XMLParser	
Parseur XML/CalculatriceNPI	163

Chapitre 3

Documentation des classes

3.1 Référence de la classe calculator : :Add

Graphe d'héritage de calculator : :Add :



Fonctions membres publiques

- Add * clone () const
Clone l'opérateur.
- const Number * apply (const Number *n1, const Number *n2) const throw (ArithmeticException)

Amis

- class CommandMap

Additional Inherited Members

3.1.1 Description détaillée

Définition à la ligne 15 du fichier Add.h.

3.1.2 Documentation des fonctions membres

3.1.2.1 Add * calculator : :Add : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :BinaryOperator](#).

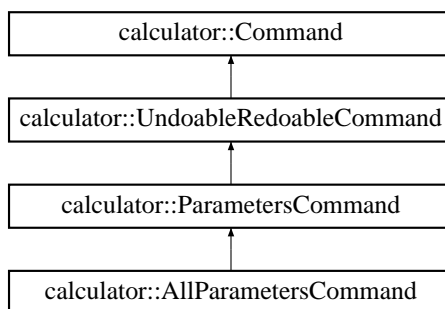
Définition à la ligne 24 du fichier Add.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Add.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Add.cpp

3.2 Référence de la classe calculator : :AllParametersCommand

Graphe d'héritage de calculator : :AllParametersCommand :



Fonctions membres publiques

- **AllParametersCommand** (const [Parameters](#) *parameters)
- **AllParametersCommand** * [clone](#) () const
Clone l'opérateur.

Fonctions membres protégées

- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un Memento propre à chaque Command contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- **AllParametersCommand** (const std : :string name)
- **AllParametersCommand** (const [AllParametersCommand](#) &orig)

Amis

- class **XMLParser**

3.2.1 Description détaillée

Définition à la ligne 16 du fichier AllParametersCommand.h.

3.2.2 Documentation des fonctions membres

3.2.2.1 `void calculator : :AllParametersCommand : :apply (Parameters * parameters) const [protected], [virtual]`

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 32 du fichier AllParametersCommand.cpp.

3.2.2.2 `AllParametersCommand * calculator : :AllParametersCommand : :clone () const [virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 27 du fichier AllParametersCommand.cpp.

3.2.2.3 `const Memento * calculator : :AllParametersCommand : :createMemento () const throw (CommandException) [protected], [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 36 du fichier AllParametersCommand.cpp.

3.2.2.4 `void calculator : :AllParametersCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [protected], [virtual]`

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

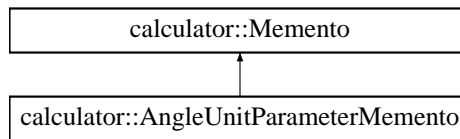
Définition à la ligne 40 du fichier AllParametersCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/AllParametersCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/AllParametersCommand.cpp

3.3 Référence de la classe calculator : :AngleUnitParameterMemento

Graphe d'héritage de calculator : :AngleUnitParameterMemento :



Fonctions membres publiques

- **AngleUnitParameterMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, [Parameters](#) : :-
[AngleUnit](#) angleUnit)

Amis

- class **DegreeParameterCommand**
- class **RadianParameterCommand**

Additional Inherited Members

3.3.1 Description détaillée

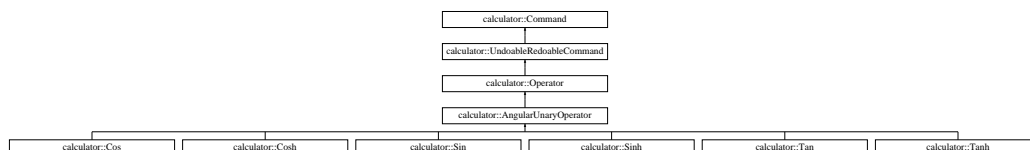
Définition à la ligne 17 du fichier AngleUnitParameterMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/AngleUnitParameterMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/AngleUnitParameterMemento.cpp

3.4 Référence de la classe calculator : :AngularUnaryOperator

Graphe d'héritage de calculator : :AngularUnaryOperator :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [AngularUnaryOperator](#) * [clone](#) () const =0
Clone l'opérateur.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Fonctions membres protégées

- virtual const [Number](#) * [apply](#) (const [SimpleNumber](#) *number) const =0 throw (ArithmeticException)
- const [Number](#) * [apply](#) (const [Number](#) *number) const throw (ArithmeticException)
- **AngularUnaryOperator** (const std : :string name)

3.4.1 Description détaillée

Définition à la ligne 19 du fichier AngularUnaryOperator.h.

3.4.2 Documentation des fonctions membres

3.4.2.1 `virtual AngularUnaryOperator* calculator : :AngularUnaryOperator : :clone () const [pure virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :Cos](#), [calculator : :Cosh](#), [calculator : :Sin](#), [calculator : :Sinh](#), [calculator : :Tan](#), et [calculator : :Tanh](#).

3.4.2.2 `const Memento * calculator : :AngularUnaryOperator : :createMemento () const throw (CommandException) [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 34 du fichier AngularUnaryOperator.cpp.

3.4.2.3 `std : :string calculator : :AngularUnaryOperator : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

Définition à la ligne 28 du fichier AngularUnaryOperator.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

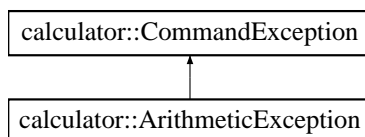
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/AngularUnaryOperator.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/AngularUnaryOperator.cpp

3.5 Référence de la classe calculator : :ArithmeticException

Classe d'exception arithmétique.

```
#include <ArithmeticException.h>
```

Graphe d'héritage de calculator : :ArithmeticException :



Fonctions membres publiques

- **ArithmeticException** (const std : :string &e)

3.5.1 Description détaillée

Classe d'exception arithmétique.

Utilisée pour les exceptions dans les opérations mathématiques.

Paramètres

e	la cause de l'exception.
---	--------------------------

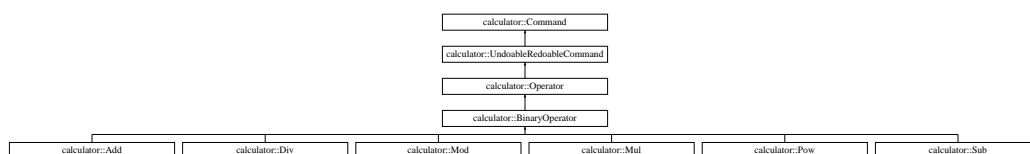
Définition à la ligne 20 du fichier ArithmeticException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/exception/ArithmeticException.h

3.6 Référence de la classe calculator : :BinaryOperator

Graphe d'héritage de calculator : :BinaryOperator :



Fonctions membres publiques

- virtual std : :string **isExecutable** () const
Vérifie si la commande est exécutable.
- virtual **BinaryOperator** * **clone** () const =0
Clone l'opérateur.
- const **Memento** * **createMemento** () const throw (CommandException)
*Crée un **Memento** propre à chaque **Command** contenant une copie des données qui vont être modifiées.*

Fonctions membres protégées

- virtual const **Number** * **apply** (const **Number** *n1, const **Number** *n2) const =0 throw (ArithmeticException)
- **BinaryOperator** (const std : :string name)

3.6.1 Description détaillée

Définition à la ligne 16 du fichier BinaryOperator.h.

3.6.2 Documentation des fonctions membres

3.6.2.1 virtual BinaryOperator* calculator : :BinaryOperator : :clone () const [pure virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :Div](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Add](#), [calculator : :Mul](#), et [calculator : :Sub](#).

3.6.2.2 const Memento * calculator : :BinaryOperator : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 38 du fichier BinaryOperator.cpp.

3.6.2.3 std : :string calculator : :BinaryOperator : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

Réimplémentée dans [calculator : :Mod](#), et [calculator : :Pow](#).

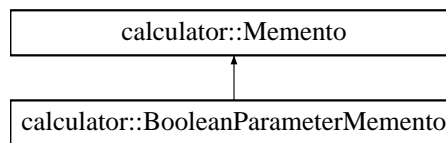
Définition à la ligne 27 du fichier BinaryOperator.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/BinaryOperator.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/BinaryOperator.cpp

3.7 Référence de la classe calculator : :BooleanParameterMemento

Graphe d'héritage de calculator : :BooleanParameterMemento :



Fonctions membres publiques

- **BooleanParameterMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, bool boolean)

Amis

- class **ComplexParameterCommand**
- class **InstantComputeParametersCommand**
- class **IntegerDivisionParameterCommand**
- class **KeyboardParameterCommand**
- class **SaveOnExitCommand**

Additional Inherited Members

3.7.1 Description détaillée

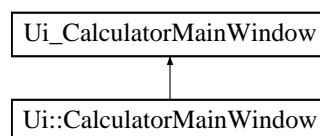
Définition à la ligne 16 du fichier BooleanParameterMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/BooleanParameterMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/BooleanParameterMemento.cpp

3.8 Référence de la classe Ui : :CalculatorMainWindow

Graphe d'héritage de Ui : :CalculatorMainWindow :



Additional Inherited Members

3.8.1 Description détaillée

Définition à la ligne 721 du fichier ui_CalculatorMainWindow.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/ui_CalculatorMainWindow.h

3.9 Référence de la classe calculator : :CalculatorMainWindow

La fenêtre principale de l'application.

```
#include <CalculatorMainWindow.h>
```

Connecteurs publics

- void **sRun** ()
- void **sUndo** ()
- void **sRedo** ()
- void **sEval** ()
- void **sShowKeyboard** ()
- void **sHideKeyboard** ()
- void **sShowParametersWidget** ()
- void **sHelp** ()
- void **sAbout** ()
- void **sSave** ()
- void **sInputLineEdited** ()
- void **sNumeralButton** (const std : :string str)
- void **sTextButton** (const std : :string str)
- void **s0** ()
- void **s1** ()
- void **s2** ()
- void **s3** ()
- void **s4** ()
- void **s5** ()
- void **s6** ()
- void **s7** ()
- void **s8** ()
- void **s9** ()
- void **sAdd** ()
- void **sSub** ()
- void **sMul** ()
- void **sDiv** ()
- void **sDot** ()
- void **sDollar** ()
- void **sSwap** ()
- void **sDup** ()
- void **sDrop** ()
- void **sClear** ()
- void **sMod** ()
- void **sLog** ()
- void **sFact** ()
- void **sCube** ()
- void **sPow** ()
- void **sSqrt** ()
- void **sSqr** ()
- void **sLn** ()
- void **sSign** ()
- void **sSin** ()
- void **sCos** ()
- void **sTan** ()
- void **sSinh** ()
- void **sCosh** ()
- void **sTanh** ()
- void **sInv** ()
- void **sExpression** ()
- void **sSum** ()
- void **sMean** ()
- void **sUserCommand** ()
- void **sSpace** ()
- void **sBackSpace** ()
- void **sRaz** ()
- void **sIntegerMode** ()
- void **sRationnalMode** ()
- void **sRealMode** ()
- void **sComplexMode** ()
- void **sDegree** ()
- void **sRadian** ()
- void **sNewStack** ()
- void **sDupStack** ()
- void **sRemStack** ()
- void **sSwitchStack** (int index)

Fonctions membres publiques

- [CalculatorMainWindow](#) (QWidget *parent=0)
Constructeur de l'interface principale.
- virtual ~[CalculatorMainWindow](#) ()
Destructeur de l'interface.
- const [Engine](#) *const [getEngine](#) () const
Getter vers le moteur utilisé par l'application.
- void [setEngine](#) (const [Engine](#) *engine)
Setter pour le moteur de l'interface.
- void [init](#) () throw (ViewException)
Getter pour.
- void [updateView](#) ()
Met à jour les différents composants de la vue.
- void [prepareForShutdown](#) ()
Désactive la connexion entre le signal émis par un changement.

3.9.1 Description détaillée

La fenêtre principale de l'application.

Obligatoirement ssociée à un moteur pour accéder au [Context](#).

Définition à la ligne 31 du fichier CalculatorMainWindow.h.

3.9.2 Documentation des constructeurs et destructeur

3.9.2.1 calculator : :CalculatorMainWindow : :CalculatorMainWindow (QWidget * parent = 0)

Constructeur de l'interface principale.

Paramètres

<i>parent</i>	laissé vide, aucun parent.
---------------	----------------------------

Définition à la ligne 19 du fichier CalculatorMainWindow.cpp.

3.9.3 Documentation des fonctions membres

3.9.3.1 const Engine* const calculator : :CalculatorMainWindow : :getEngine () const [inline]

Getter vers le moteur utilisé par l'application.

Renvoie

le moteur utilisé.

Définition à la ligne 51 du fichier CalculatorMainWindow.h.

3.9.3.2 void calculator : :CalculatorMainWindow : :init () throw (ViewException)

Getter pour.

```
@return
```

Initialise les composants de la vue avec les paramètres.

Définition à la ligne 56 du fichier CalculatorMainWindow.cpp.

3.9.3.3 void calculator : :CalculatorMainWindow : :prepareForShutdown ()

Désactive la connexion entre le signal émis par un changement.

de [StackTabWidget](#) pour éviter les problèmes lors de la suppression . de ces derniers

Définition à la ligne 85 du fichier CalculatorMainWindow.cpp.

3.9.3.4 void calculator : :CalculatorMainWindow : :setEngine (const Engine * engine) [inline]

Setter pour le moteur de l'interface.

Passage obligatoire car impossibilité de passer des paramètres supplémentaires dans le constructeur de l'interface. (Qt interdit la copie d'un QMainWindow).

Paramètres

<i>engine</i>	le moteru à utiliser.
---------------	-----------------------

Définition à la ligne 62 du fichier CalculatorMainWindow.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/view/CalculatorMainWindow.h
- F :/NetBeansProjects/LO21/Calculatrice/view/CalculatorMainWindow.cpp

3.10 Référence de la classe calculator : :CalculatorParser

Parseur de l'application CalculatriceNPI.

```
#include <CalculatorParser.h>
```

Fonctions membres publiques

- [CalculatorParser](#) (const [Context](#) *const context)
Constructeur de [CalculatorParser](#).
- virtual ~[CalculatorParser](#) ()
Destructeur de [CalculatorParser](#).
- const [Command](#) * [parse](#) (const std : :string &str) const throw (ParseException)
Fonction de parsing d'une chaîne de caractère en expression.
- bool [isInteger](#) (const std : :string &str) const
Teste si une chaîne est parsable en [Integer](#).
- bool [isRational](#) (const std : :string &str, int *pos=0) const
Teste si une chaîne est parsable en [Rational](#).
- bool [isReal](#) (const std : :string &str, int *pos=0) const
Teste si une chaîne est parsable en [Real](#).
- bool [isComplex](#) (const std : :string &str, int *pos=0) const
Teste si une chaîne est parsable en [Complex](#).
- bool [isSimpleNumber](#) (const std : :string &str) const
Teste si une chaîne est parsable soit en un [Integer](#), soit un [Rational](#), soit un [Real](#).
- bool [isCommand](#) (const std : :string &str) const
Teste si un chaîne de caractère correspond à une [Command](#) du gestionnaire de commande.
- bool [isExpression](#) (const std : :string &str) const
Teste si une chaîne de caractères correspond à une expression.

3.10.1 Description détaillée

Parseur de l'application CalculatriceNPI.

Permet de passer d'une chaîne de caractères à une commande. On obtient en général soit une commande, soit un nombre soit une expression. Rappel : les nombres et expressions sont aussi des commandes.

Définition à la ligne 29 du fichier CalculatorParser.h.

3.10.2 Documentation des constructeurs et destructeur

3.10.2.1 calculator : :CalculatorParser : :CalculatorParser (const Context *const context)

Constructeur de [CalculatorParser](#).

Paramètres

<i>context</i>	le contexte courant dont on utilise le jeu de commandes fixes ainsi que les commandes utilisateurs créées.
----------------	--

Définition à la ligne 15 du fichier CalculatorParser.cpp.

3.10.2.2 calculator : :CalculatorParser : :~CalculatorParser () [virtual]

Destructeur de [CalculatorParser](#).

Ne détruit pas le moteur de l'application

Définition à la ligne 19 du fichier CalculatorParser.cpp.

3.10.3 Documentation des fonctions membres

3.10.3.1 bool calculator : :CalculatorParser : :isCommand (const std : :string & str) const

Teste si un chaîne de caractère correspond à une [Command](#) du gestionnaire de commande.

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si une occurrence est trouvée, false sinon.

Définition à la ligne 93 du fichier CalculatorParser.cpp.

3.10.3.2 bool calculator : :CalculatorParser : :isComplex (const std : :string & str, int * pos = 0) const

Teste si une chaîne est parsable en [Complex](#).

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si str peut être un [Complex](#), false sinon.

Définition à la ligne 62 du fichier CalculatorParser.cpp.

3.10.3.3 bool calculator : :CalculatorParser : :isExpression (const std : :string & str) const

Teste si une chaîne de caractères correspond à une expression.

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si la chaîne est une expression, false sinon. Une [Expression](#) commence par un ' et termine par un '. Il ne peut y avoir d'[Expression](#) à l'intérieur d'une [Expression](#), une [Expression](#) est composée d'au moins une [Command](#).

Définition à la ligne 98 du fichier CalculatorParser.cpp.

3.10.3.4 bool calculator : :CalculatorParser : :isInteger (const std : :string & *str*) const

Teste si une chaîne est parsable en [Integer](#).

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si *str* peut être un [Integer](#), false sinon.

Définition à la ligne 22 du fichier CalculatorParser.cpp.

3.10.3.5 bool calculator : :CalculatorParser : :isRationnal (const std : :string & *str*, int * *pos* = 0) const

Teste si une chaîne est parsable en [Rationnal](#).

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si *str* peut être un [Rationnal](#), false sinon.

Définition à la ligne 31 du fichier CalculatorParser.cpp.

3.10.3.6 bool calculator : :CalculatorParser : :isReal (const std : :string & *str*, int * *pos* = 0) const

Teste si une chaîne est parsable en [Real](#).

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si *str* peut être un [Real](#), false sinon.

Définition à la ligne 47 du fichier CalculatorParser.cpp.

3.10.3.7 bool calculator : :CalculatorParser : :isSimpleNumber (const std : :string & *str*) const

Teste si une chaîne est parsable soit en un [Integer](#), soit un [Rationnal](#), soit un [Real](#).

Paramètres

<i>str</i>	la chaîne à tester.
------------	---------------------

Renvoie

true si str peut être un [Integer](#), un [Rationnal](#) ou un [Real](#), false sinon.

Définition à la ligne 83 du fichier CalculatorParser.cpp.

3.10.3.8 const Command * calculator : :CalculatorParser : :parse (const std : :string & str) const throw (ParseException)

Fonction de parsing d'une chaîne de caractère en expression.

Paramètres

<i>str</i>	la chaîne de caractère à convertir.
------------	-------------------------------------

Renvoie

une commande, qui peut aussi être un nombre ou une expression.

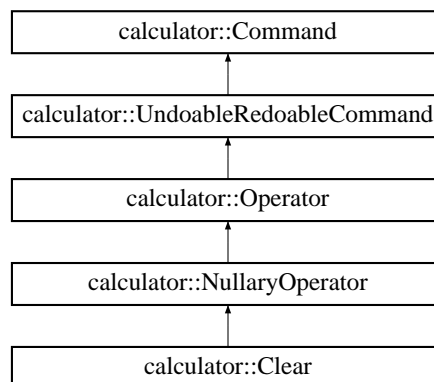
Définition à la ligne 115 du fichier CalculatorParser.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/controler/CalculatorParser.h
- F :/NetBeansProjects/LO21/Calculatrice/controler/CalculatorParser.cpp

3.11 Référence de la classe calculator : :Clear

Graphe d'héritage de calculator : :Clear :

**Fonctions membres publiques**

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- [Clear](#) * [clone](#) () const
Clone l'opérateur.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

- class [CommandMap](#)

Additional Inherited Members

3.11.1 Description détaillée

Définition à la ligne 15 du fichier Clear.h.

3.11.2 Documentation des fonctions membres

3.11.2.1 `Clear * calculator : :Clear : :clone () const [virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :NullaryOperator](#).

Définition à la ligne 24 du fichier Clear.cpp.

3.11.2.2 `const Memento * calculator : :Clear : :createMemento () const throw (CommandException) [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :NullaryOperator](#).

Définition à la ligne 33 du fichier Clear.cpp.

3.11.2.3 `std : :string calculator : :Clear : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :NullaryOperator](#).

Définition à la ligne 28 du fichier Clear.cpp.

3.11.2.4 `void calculator : :Clear : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

3.12.2 Documentation des constructeurs et destructeur

3.12.2.1 calculator : :Command : :Command (const std : :string name) [protected]

Constructeur de [Command](#).

Chaque commande est défini par son nom.

Paramètres

<i>name</i>	le nom de la commande.
-------------	------------------------

Définition à la ligne 14 du fichier Command.cpp.

3.12.3 Documentation des fonctions membres

3.12.3.1 virtual Command* calculator : :Command : :clone () const [pure virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémenté dans [calculator : :Expression](#), [calculator : :UndoableRedoableCommand](#), [calculator : :Complex](#), [calculator : :Operator](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :Constant](#), [calculator : :Literal](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), [calculator : :Eval](#), [calculator : :Div](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :Swap](#), [calculator : :AngularUnaryOperator](#), [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationnalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), [calculator : :AllParametersCommand](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :Add](#), [calculator : :Mul](#), [calculator : :Sub](#), [calculator : :Cos](#), [calculator : :Cosh](#), [calculator : :Inv](#), [calculator : :Sin](#), [calculator : :Sinh](#), [calculator : :Tan](#), [calculator : :Tanh](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :SimpleNumber](#), [calculator : :NaryOperator](#), [calculator : :RedoCommand](#), [calculator : :UndoCommand](#), [calculator : :Number](#), [calculator : :BinaryOperator](#), [calculator : :Mean](#), [calculator : :Sum](#), [calculator : :Dup](#), [calculator : :UnaryOperator](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), et [calculator : :NullaryOperator](#).

3.12.3.2 virtual void calculator : :Command : :execute () const throw (CommandException) [pure virtual]

Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, la méthode appelée est précisée par les filles.

Implémenté dans [calculator : :Operator](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :UndoableRedoableCommand](#), [calculator : :Literal](#), [calculator : :Constant](#), [calculator : :Eval](#), [calculator : :RedoCommand](#), [calculator : :UndoCommand](#), et [calculator : :VisibleStackSizeParameterCommand](#).

3.12.3.3 virtual std : :string calculator : :Command : :isExecutable () const [pure virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std::string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas, ou dans le cas d'une [Expression](#) renvoie le restant non exécuté de l'expression.

Implémenté dans [calculator : :Complex](#), [calculator : :UndoableRedoableCommand](#), [calculator : :Operator](#), [calculator : :Eval](#), [calculator : :Literal](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :RedoCommand](#), [calculator : :UndoCommand](#), [calculator : :AngularUnaryOperator](#), [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationnalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :NaryOperator](#), [calculator : :Swap](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :BinaryOperator](#), [calculator : :Dup](#), [calculator : :UnaryOperator](#), et [calculator : :NullaryOperator](#).

3.12.3.4 `std::string calculator : :Command : :toString () const` `[virtual]`

Renvoie la [Command](#) sous la forme d'une `std::string`.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une `std::string`.

Réimplémentée dans [calculator : :Expression](#), [calculator : :Complex](#), [calculator : :Literal](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), et [calculator : :SimpleNumber](#).

Définition à la ligne 25 du fichier `Command.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

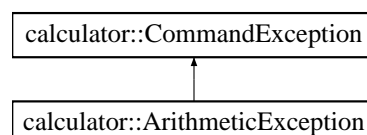
- `F : /NetBeansProjects/LO21/Calculatrice/model/command/Command.h`
- `F : /NetBeansProjects/LO21/Calculatrice/model/command/Command.cpp`

3.13 Référence de la classe calculator : :CommandException

Classe d'exception pour les [Command](#).

```
#include <CommandException.h>
```

Graphe d'héritage de `calculator : :CommandException` :



Fonctions membres publiques

- **CommandException** (`const std::string &str`)
- `const char * what () const throw ()`

3.13.1 Description détaillée

Classe d'exception pour les [Command](#).

Exception de base de l'application, lancée pour la plupart des erreurs.

Paramètres

<i>e</i>	la cause de l'exception.
----------	--------------------------

Définition à la ligne 21 du fichier CommandException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/exception/CommandException.h

3.14 Référence de la classe calculator : :CommandManager

Le gestionnaire de commande de l'application.

```
#include <CommandManager.h>
```

Fonctions membres publiques

- [CommandManager](#) ([Engine](#) *const engine, [MementoCaretaker](#) *const mementoCaretaker)
Constructeur de [CommandManager](#).
- virtual ~[CommandManager](#) ()
Destructeur de [CommandManager](#).
- std::string [executeCommand](#) (const [Command](#) *command) throw (CommandException)
- std::string [getPreviousInputString](#) () const

3.14.1 Description détaillée

Le gestionnaire de commande de l'application.

Cette classe a la charge d'exécuter proprement les commandes et manipule les mementos pour les fonctions UNDO/REDO.

Paramètres

<i>parentEngine</i>	Le moteur de l'application qui l'utilise.
<i>memento-Caretaker</i>	Le conteneur intelligent de mementos utilisé pour une exécution de l'application.

Définition à la ligne 28 du fichier CommandManager.h.

3.14.2 Documentation des constructeurs et destructeur

3.14.2.1 calculator : :CommandManager : :CommandManager ([Engine](#) *const engine, [MementoCaretaker](#) *const mementoCaretaker)

Constructeur de [CommandManager](#).

Paramètres

<i>engine</i>	Le moteur qui utilise ce gestionnaire de commande.
<i>memento-Caretaker</i>	Le gestionnaire de mementos utilisé pour cette application.

Définition à la ligne 18 du fichier CommandManager.cpp.

3.14.2.2 calculator : :CommandManager : ~CommandManager () [virtual]

Destructeur de Commandanager.

Ne détruit pas le moteur et le gestionnaire de mementos.

Définition à la ligne 23 du fichier CommandManager.cpp.

3.14.3 Documentation des fonctions membres

3.14.3.1 std : :string calculator : :CommandManager : executeCommand (const Command * command) throw (CommandException)

on ne sauvegardera le memento que si la commande s'est exécutée correctement

sauvegarde du memento si UndoRedoCommand

Définition à la ligne 30 du fichier CommandManager.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/controler/CommandManager.h
- F :/NetBeansProjects/LO21/Calculatrice/controler/CommandManager.cpp

3.15 Référence de la classe calculator : :CommandMap

Classe conteneur des [Command](#) fixes de l'application.

```
#include <CommandMap.h>
```

Fonctions membres publiques

- bool [isCommandName](#) (const std : :string &str) const
Parcours la liste de commande pour déterminer si la commande demandée existe.
- const [Command](#) * [getCommand](#) (const std : :string &str) const throw (CommandException)
Parcours la liste et renvoie la commande si le nom correspond.

Fonctions membres publiques statiques

- static [CommandMap](#) * [getInstance](#) ()
Singleton [CommandMap](#).
- static void [deleteInstance](#) ()
Singleton [CommandMap](#).

3.15.1 Description détaillée

Classe conteneur des [Command](#) fixes de l'application.

et des commandes utilisateurs Utilisation du Design Pattern Prototype.

Définition à la ligne 24 du fichier CommandMap.h.

3.15.2 Documentation des fonctions membres

3.15.2.1 void calculator : :CommandMap : deleteInstance () [static]

Singleton [CommandMap](#).

Détruit l'unique instance de [CommandMap](#). Libère la mémoire prise par les commandes stockées.

Définition à la ligne 132 du fichier CommandMap.cpp.

3.15.2.2 `const Command * calculator : :CommandMap : :getCommand (const std : :string & str) const throw (CommandException)`

Parcours la liste et renvoie la commande si le nom correspond.

Lance une exception si elle n'est pas trouvée.

Paramètres

<i>str</i>	le nom de la commande demandée.
------------	---------------------------------

Renvoie

une copie de la commande demandée.

Définition à la ligne 182 du fichier CommandMap.cpp.

3.15.2.3 `CommandMap * calculator : :CommandMap : :getInstance () [static]`

Singleton [CommandMap](#).

Renvoie

crée et/ou renvoie l'unique instance de [CommandMap](#).

Définition à la ligne 125 du fichier CommandMap.cpp.

3.15.2.4 `bool calculator : :CommandMap : :isCommandName (const std : :string & str) const`

Parcours la liste de commande pour déterminer si la commande demandée existe.

Paramètres

<i>str</i>	le nom de la commande recherchée.
------------	-----------------------------------

Renvoie

true si la commande existe.

Remarque : on aurait pu renvoyer un itérateur et éviter de lancer éventuellement une exception pour la méthode `getCommand(const std : :string& str)`.

Définition à la ligne 177 du fichier CommandMap.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

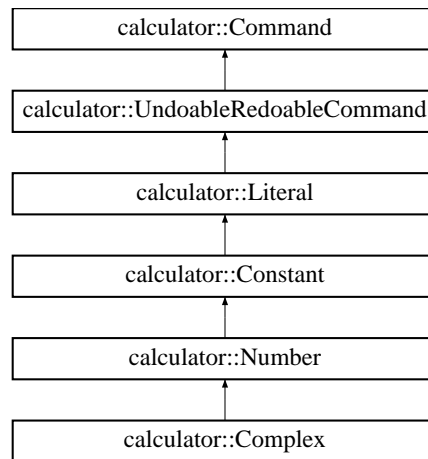
- F :/NetBeansProjects/LO21/Calculatrice/model/command/CommandMap.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/CommandMap.cpp

3.16 Référence de la classe calculator : :Complex

Nombre Complexe.

```
#include <Complex.h>
```

Graphe d'héritage de calculator : :Complex :



Fonctions membres publiques

- **Complex** (const [Complex](#) &orig)
- **Complex** (const [SimpleNumber](#) &x)
- **Complex** (const [SimpleNumber](#) *x)
- **Complex** (const [SimpleNumber](#) &x1, const [SimpleNumber](#) &x2)
- **Complex** (const [SimpleNumber](#) *x1, const [SimpleNumber](#) *x2)
- [SimpleNumber](#) * **getRe** () const
- [SimpleNumber](#) * **getIm** () const
- std::string **isExecutable** () const
- [Complex](#) * **clone** () const
Clone la [Constant](#).
- std::string **toString** () const
Renvoie la [Command](#) sous la forme d'une std::string.
- [Complex](#) & **operator=** (const [Complex](#) &c)
- [Complex](#) & **operator+=** (const [Complex](#) &c)
- [Complex](#) & **operator-=** (const [Complex](#) &c)
- [Complex](#) & **operator*=** (const [Complex](#) &c)
- [Complex](#) & **operator/=** (const [Complex](#) &c)
- [Complex](#) & **pow** (const [SimpleNumber](#) &sn)

Additional Inherited Members

3.16.1 Description détaillée

Nombre Complexe.

Composé de 2 [SimpleNumber](#).

Définition à la ligne 21 du fichier Complex.h.

3.16.2 Documentation des fonctions membres

3.16.2.1 [Complex](#) * calculator::Complex::clone () const [virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator::Number](#).

Définition à la ligne 47 du fichier Complex.cpp.

3.16.2.2 `std::string calculator::Complex::isExecutable() const [virtual]`

Renvoie

`std::string("")` si le mode complex est activé, une chaîne non vide sinon

Réimplémentée à partir de [calculator::Literal](#).

Définition à la ligne 51 du fichier Complex.cpp.

3.16.2.3 `std::string calculator::Complex::toString() const [virtual]`

Renvoie la [Command](#) sous la forme d'une `std::string`.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une `std::string`.

Implémente [calculator::Literal](#).

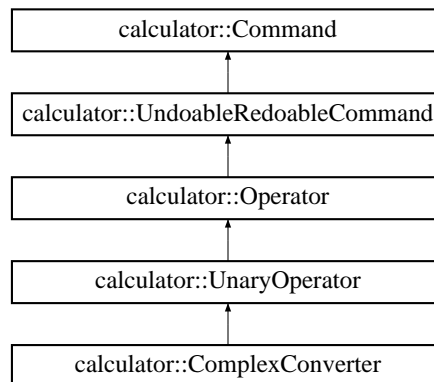
Définition à la ligne 56 du fichier Complex.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Complex.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Complex.cpp

3.17 Référence de la classe calculator : :ComplexConverter

Graphe d'héritage de calculator : :ComplexConverter :



Fonctions membres publiques

- `std::string isExecutable() const`
Vérifie si la commande est exécutable.
- `ComplexConverter * clone() const`
Clone l'opérateur.
- `const Number * apply(const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.17.1 Description détaillée

Définition à la ligne 15 du fichier ComplexConverter.h.

3.17.2 Documentation des fonctions membres

3.17.2.1 `ComplexConverter * calculator : :ComplexConverter : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 29 du fichier ComplexConverter.cpp.

3.17.2.2 `std : :string calculator : :ComplexConverter : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

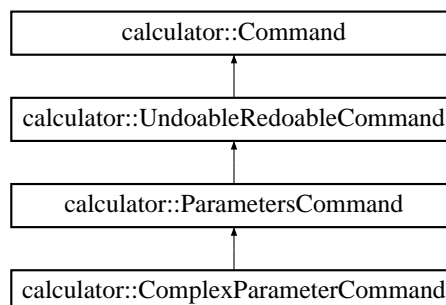
Définition à la ligne 24 du fichier ComplexConverter.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/ComplexConverter.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/ComplexConverter.cpp

3.18 Référence de la classe calculator : :ComplexParameterCommand

Grappe d'héritage de calculator : :ComplexParameterCommand :



Fonctions membres publiques

- virtual [ComplexParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètre dans leur état précédent.

Fonctions membres protégées

- [ComplexParameterCommand](#) (const std : :string name)
- [ComplexParameterCommand](#) (const [ComplexParameterCommand](#) &orig)

Amis

- class [CommandMap](#)

3.18.1 Description détaillée

Définition à la ligne 15 du fichier ComplexParameterCommand.h.

3.18.2 Documentation des fonctions membres

3.18.2.1 void calculator : :ComplexParameterCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 27 du fichier ComplexParameterCommand.cpp.

3.18.2.2 [ComplexParameterCommand](#) * calculator : :ComplexParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 23 du fichier ComplexParameterCommand.cpp.

3.18.2.3 const [Memento](#) * calculator : :ComplexParameterCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 31 du fichier ComplexParameterCommand.cpp.

3.18.2.4 void calculator : :ComplexParameterCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 35 du fichier ComplexParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

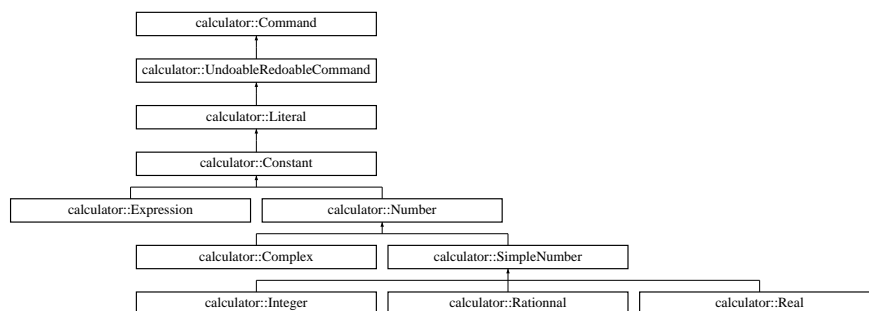
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/ComplexParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/ComplexParameterCommand.cpp

3.19 Référence de la classe calculator : :Constant

Element de base des [Stack](#).

```
#include <Constant.h>
```

Graphe d'héritage de calculator : :Constant :

**Fonctions membres publiques**

- virtual void [execute](#) () const throw (CommandException)
Exécute une [Constant](#) Consiste à empiler la [Constant](#) sur la [Stack](#) courante.
- virtual [Constant](#) * [clone](#) () const =0
Clone la [Constant](#).
- virtual ~[Constant](#) ()
Destructeur de [Constant](#).

Fonctions membres protégées

- [Constant](#) ()
Constructeur de [Constant](#).

3.19.1 Description détaillée

Element de base des [Stack](#).

Classe non instanciable, il s'agit soit d'un [Number](#), soit d'une [Expression](#).

Définition à la ligne 19 du fichier Constant.h.

3.19.2 Documentation des fonctions membres

3.19.2.1 virtual Constant* calculator : :Constant : :clone () const [pure virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator : :Literal](#).

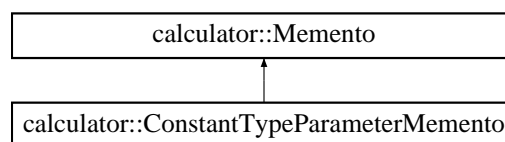
Implémenté dans [calculator : :Expression](#), [calculator : :Complex](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), [calculator : :SimpleNumber](#), et [calculator : :Number](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Constant.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Constant.cpp

3.20 Référence de la classe calculator : :ConstantTypeParameterMemento

Graphe d'héritage de calculator : :ConstantTypeParameterMemento :



Fonctions membres publiques

- **ConstantTypeParameterMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, [Parameters : :ConstantType](#) constantType)

Amis

- class **IntegerParameterCommand**
- class **RationnalParameterCommand**
- class **RealParameterCommand**

Additional Inherited Members

3.20.1 Description détaillée

Définition à la ligne 17 du fichier ConstantTypeParameterMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/ConstantTypeParameterMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/ConstantTypeParameterMemento.cpp

3.21 Référence de la classe calculator : :Context

Classe de base du modèle dans le schéma MVC.

```
#include <Context.h>
```

Fonctions membres publiques

- void **setInputString** (const std : :string &str)
 - std : :string **getInputString** () const
 - [Parameters](#) * **getParameters** () const
 - void **setParameters** (const [Parameters](#) *parameters)
 - [MementoCaretaker](#) * **getMementoCaretaker** () const
 - [CommandMap](#) * **getCommandMap** () const
 - [Stack](#) * **getCurrentStack** () const
 - void **setCurrentStack** (const unsigned int i) throw (ContextException)
 - [StackList](#) * **getStackList** () const
 - void **setStackList** (const [StackList](#) *stackList)
 - bool **getError** () const
 - void **setError** (bool b=true)
 - void **saveContextToXML** () const
- Sauvegarde du contexte directe.*

Fonctions membres publiques statiques

- static [Context](#) * **getInstance** ()
- Singleton [Context](#).*
- static void **deleteInstance** ()
- Singleton [Context](#).*

3.21.1 Description détaillée

Classe de base du modèle dans le schéma MVC.

Le contexte contient :

- la saisie courante
- le gestionnaire de [Memento](#)
- les paramètres

Voir également

[Parameters](#)

- les piles

Voir également

[StackList](#), [Stack](#)

- la liste de commandes

Voir également

[CommandMap](#)

Définition à la ligne 32 du fichier Context.h.

3.21.2 Documentation des fonctions membres

3.21.2.1 static void calculator : :Context : :deleteInstance () [inline],[static]

Singleton [Context](#).

Détruit l'unique instance de [Context](#). Libère la mémoire utilisée par :

- le conteneur de [Memento](#)
- la liste de piles

Voir également

[StackList](#)

- la liste de commandes

Voir également

[CommandMap](#)

- les paramètres

Voir également

[Parameters](#)

Définition à la ligne 59 du fichier Context.h.

3.21.2.2 `static Context* calculator : :Context : :getInstance () [inline],[static]`

Singleton [Context](#).

Renvoie

crée et/ou renvoie l'unique instance de [Context](#). Initialise la ligne de saisie à vide et le statut `_error` du contexte à false Met en place les [Parameters](#), la liste de commandes, le conteneur de [Memento](#) et initialise une [StackList](#) avec une [Stack](#) vide

Définition à la ligne 43 du fichier Context.h.

3.21.2.3 `void calculator : :Context : :saveContextToXML () const`

Sauvegarde du contexte directe.

Entorse au schéma MVC, le contexte est capable de s'enregistrer sans passer par le moteur.

Définition à la ligne 32 du fichier Context.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/Context.h
- F :/NetBeansProjects/LO21/Calculatrice/model/Context.cpp

3.22 Référence de la classe calculator : :ContextException

Classe d'exception pour la couche Model.

```
#include <ContextException.h>
```

Fonctions membres publiques

- **ContextException** (const std : :string &i)
- const char * **what** () const throw ()

3.22.1 Description détaillée

Classe d'exception pour la couche Model.

Exception lancée si le contexte se retrouve dans un état inconsistant.

Paramètres

e	la cause de l'exception.
---	--------------------------

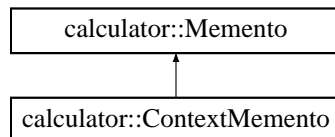
Définition à la ligne 21 du fichier ContextException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/exception/ContextException.h

3.23 Référence de la classe calculator : :ContextMemento

Graphe d'héritage de calculator : :ContextMemento :



Fonctions membres publiques

- **ContextMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const [Parameters](#) *parameters, const [StackList](#) *stackList, const unsigned int currentStackIndex)

Amis

- class **Eval**

Additional Inherited Members

3.23.1 Description détaillée

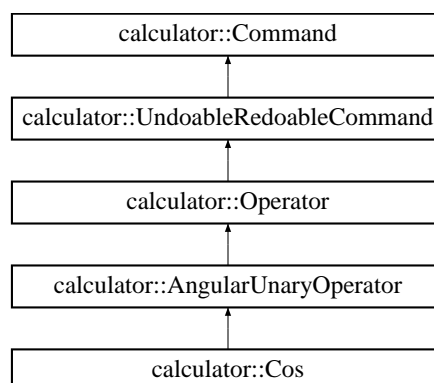
Définition à la ligne 21 du fichier ContextMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/ContextMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/ContextMemento.cpp

3.24 Référence de la classe calculator : :Cos

Graphe d'héritage de calculator : :Cos :



Fonctions membres publiques

- `Cos * clone () const`
Clone l'opérateur.
- `const Number * apply (const SimpleNumber *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.24.1 Description détaillée

Définition à la ligne 15 du fichier Cos.h.

3.24.2 Documentation des fonctions membres

3.24.2.1 `Cos * calculator : :Cos : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :AngularUnaryOperator](#).

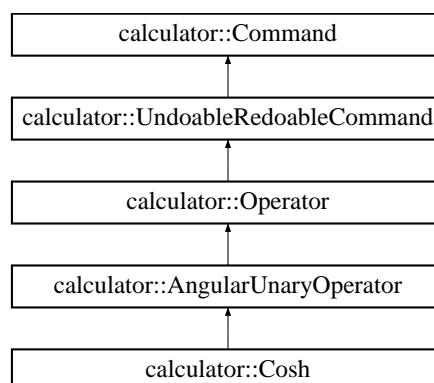
Définition à la ligne 25 du fichier Cos.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cos.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cos.cpp

3.25 Référence de la classe calculator : :Cosh

Graphe d'héritage de calculator : :Cosh :



Fonctions membres publiques

- `Cosh * clone () const`
Clone l'opérateur.
- `const Number * apply (const SimpleNumber *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.25.1 Description détaillée

Définition à la ligne 15 du fichier Cosh.h.

3.25.2 Documentation des fonctions membres

3.25.2.1 `Cosh * calculator : :Cosh : :clone () const [virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :AngularUnaryOperator`.

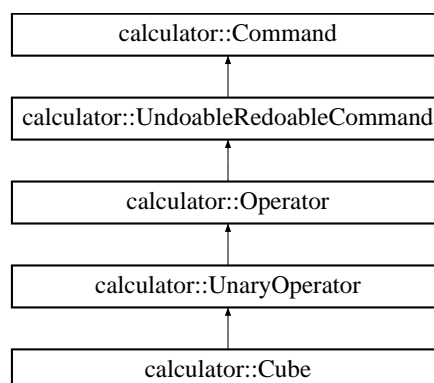
Définition à la ligne 25 du fichier Cosh.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cosh.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cosh.cpp

3.26 Référence de la classe calculator : :Cube

Graphe d'héritage de calculator : :Cube :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- [Cube](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * **apply** (const [Number](#) *n) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.26.1 Description détaillée

Définition à la ligne 15 du fichier Cube.h.

3.26.2 Documentation des fonctions membres

3.26.2.1 [Cube](#) * calculator : :Cube : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 28 du fichier Cube.cpp.

3.26.2.2 std : :string calculator : :Cube : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

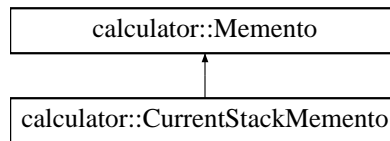
Définition à la ligne 23 du fichier Cube.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cube.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Cube.cpp

3.27 Référence de la classe calculator : :CurrentStackMemento

Graphe d'héritage de calculator : :CurrentStackMemento :



Fonctions membres publiques

- **CurrentStackMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const unsigned int stack-Index)

Amis

- class **NewStackCommand**

Additional Inherited Members

3.27.1 Description détaillée

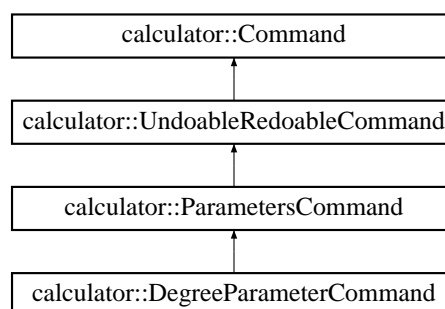
Définition à la ligne 16 du fichier CurrentStackMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/CurrentStackMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/CurrentStackMemento.cpp

3.28 Référence de la classe calculator : :DegreeParameterCommand

Graphe d'héritage de calculator : :DegreeParameterCommand :



Fonctions membres publiques

- virtual [DegreeParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètre dans leur état précédent.

Fonctions membres protégées

- **DegreeParameterCommand** (const std : :string name)
- **DegreeParameterCommand** (const [DegreeParameterCommand](#) &orig)

Amis

- class **CommandMap**

3.28.1 Description détaillée

Définition à la ligne 15 du fichier DegreeParameterCommand.h.

3.28.2 Documentation des fonctions membres

3.28.2.1 void calculator : :DegreeParameterCommand : :apply (**Parameters** * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 27 du fichier DegreeParameterCommand.cpp.

3.28.2.2 **DegreeParameterCommand** * calculator : :DegreeParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 23 du fichier DegreeParameterCommand.cpp.

3.28.2.3 const **Memento** * calculator : :DegreeParameterCommand : :createMemento () const throw
(**CommandException**) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 31 du fichier DegreeParameterCommand.cpp.

3.28.2.4 void calculator : :DegreeParameterCommand : :restoreFromMemento (const **Memento** * *memento*) const throw
(**MementoException**) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

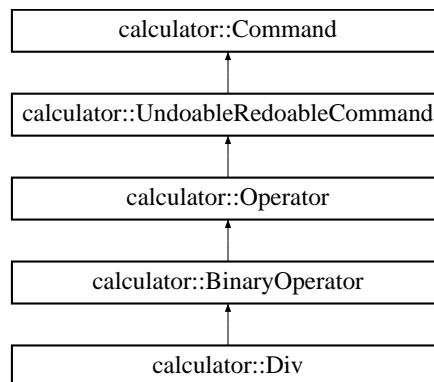
Définition à la ligne 35 du fichier DegreeParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/DegreeParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/DegreeParameterCommand.cpp

3.29 Référence de la classe calculator : :Div

Graphe d'héritage de calculator : :Div :



Fonctions membres publiques

- [Div](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * **apply** (const [Number](#) *n1, const [Number](#) *n2) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.29.1 Description détaillée

Définition à la ligne 16 du fichier Div.h.

3.29.2 Documentation des fonctions membres

3.29.2.1 [Div](#) * calculator : :Div : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :BinaryOperator`.

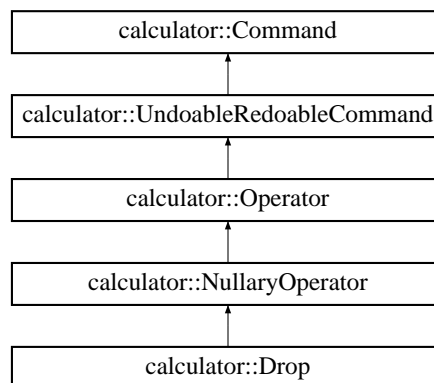
Définition à la ligne 27 du fichier Div.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Div.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Div.cpp

3.30 Référence de la classe calculator : :Drop

Graphe d'héritage de calculator : :Drop :



Fonctions membres publiques

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `Drop * clone () const`
Clone l'opérateur.
- `const Memento * createMemento () const throw (CommandException)`
Crée un Memento propre à chaque Command contenant une copie des données qui vont être modifiées.
- `void restoreFromMemento (const Memento *memento) const throw (MementoException)`
Restaure le Context dans l'état précédent l'exécution de la commande.

Amis

- class **CommandMap**

Additional Inherited Members

3.30.1 Description détaillée

Définition à la ligne 15 du fichier Drop.h.

3.30.2 Documentation des fonctions membres

3.30.2.1 Drop * calculator : :Drop : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :NullaryOperator](#).

Définition à la ligne 24 du fichier Drop.cpp.

3.30.2.2 `const Memento * calculator : :Drop : :createMemento () const throw (CommandException) [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :NullaryOperator](#).

Définition à la ligne 33 du fichier Drop.cpp.

3.30.2.3 `std : :string calculator : :Drop : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :NullaryOperator](#).

Définition à la ligne 28 du fichier Drop.cpp.

3.30.2.4 `void calculator : :Drop : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Implémente [calculator : :NullaryOperator](#).

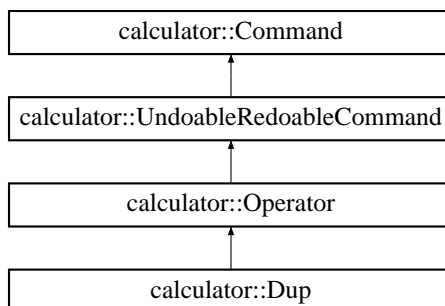
Définition à la ligne 40 du fichier Drop.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nullary/Drop.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nullary/Drop.cpp

3.31 Référence de la classe calculator : :Dup

Graphe d'héritage de calculator : :Dup :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [Dup](#) * [clone](#) () const
Clone l'opérateur.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Fonctions membres protégées

- **Dup** (const std : :string name)

Amis

- class **CommandMap**

3.31.1 Description détaillée

Définition à la ligne 15 du fichier Dup.h.

3.31.2 Documentation des fonctions membres

3.31.2.1 **Dup** * calculator : :Dup : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Définition à la ligne 24 du fichier Dup.cpp.

3.31.2.2 `const Memento * calculator : :Dup : :createMemento () const throw (CommandException) [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 33 du fichier Dup.cpp.

3.31.2.3 `std : :string calculator : :Dup : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

Définition à la ligne 28 du fichier Dup.cpp.

3.31.2.4 `void calculator : :Dup : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Réimplémentée à partir de [calculator : :Operator](#).

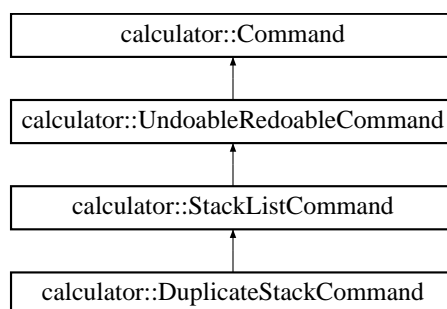
Définition à la ligne 38 du fichier Dup.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F : /NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Dup.h
- F : /NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Dup.cpp

3.32 Référence de la classe calculator : :DuplicateStackCommand

Graphe d'héritage de calculator : :DuplicateStackCommand :



Fonctions membres publiques

- std::string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [DuplicateStackCommand](#) * [clone](#) () const
Clone la commande.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

- class **CommandMap**

Additional Inherited Members

3.32.1 Description détaillée

Définition à la ligne 15 du fichier DuplicateStackCommand.h.

3.32.2 Documentation des fonctions membres

3.32.2.1 [DuplicateStackCommand](#) * calculator : :DuplicateStackCommand : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 25 du fichier DuplicateStackCommand.cpp.

3.32.2.2 const [Memento](#) * calculator : :DuplicateStackCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 34 du fichier DuplicateStackCommand.cpp.

3.32.2.3 `std : :string calculator : :DuplicateStackCommand : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 29 du fichier DuplicateStackCommand.cpp.

3.32.2.4 `void calculator : :DuplicateStackCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Implémente [calculator : :StackListCommand](#).

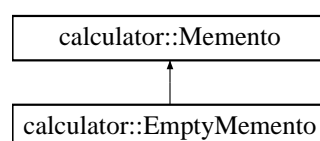
Définition à la ligne 38 du fichier DuplicateStackCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/DuplicateStackCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/DuplicateStackCommand.cpp

3.33 Référence de la classe calculator : :EmptyMemento

Graphe d'héritage de calculator : :EmptyMemento :



Amis

- class **Literal**
- class **DuplicateStackCommand**
- class **Dup**

Additional Inherited Members

3.33.1 Description détaillée

Définition à la ligne 17 du fichier EmptyMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/EmptyMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/EmptyMemento.cpp

3.34 Référence de la classe calculator : :Engine

Moteur principal de l'application.

```
#include <Engine.h>
```

Fonctions membres publiques

- **Engine** (**Context** *const context)
Constructeur de Engine.
- virtual **~Engine** ()
Destructeur de Engine.
- const **Context** *const **getContext** () const
getter Context.
- void **run** (std : :string byPassCommand="") const
Méthode centrale de l'application.
- void **drop** () const
Appel direct de la commande DROP.
- void **undo** () const
Appel direct de la commande UNDO.
- void **redo** () const
Appel direct de la commande REDO.
- void **newStack** () const
Appel direct de la commande SNEW.
- void **dupStack** () const
Appel direct de la commande SDUP.
- void **removeCurrentStack** () const
Appel direct de la commande SREM.
- void **setConstantTypeTo** (const std : :string &constantType) const
Modifie le type de constante utilisé par l'application.
- void **setAngleUnitTo** (const std : :string &angleUnit) const
Modifie l'unité de l'angle utilisée par l'application.
- void **switchComplexMode** () const
Active ou désactive le mode complexe.
- void **saveAllParameters** (const **Parameters** *parameters) const
Utilisation via l'interface des paramètres, permet de sauvegarder les paramètres après validation.
- void **setCurrentStack** (int index) const
Utilisation lors d'un clic sur un onglet.
- void **setInputText** (const std : :string &str) const
Utilisation par l'interface pour mettre à jour la ligne de saisie du contexte.

3.34.1 Description détaillée

Moteur principal de l'application.

Initialisé à partir du contexte, il fait le lien entre la vue et le modèle. Aucun paramètre n'est enregistré dans cette couche de l'application. Les méthodes d'appel direct de commandes de cette classe permettent d'éviter l'affichage du nom de la commande dans la ligne de saisie après son exécution.

Définition à la ligne 28 du fichier Engine.h.

3.34.2 Documentation des constructeurs et destructeur

3.34.2.1 calculator : :Engine : :Engine (Context *const context)

Constructeur de [Engine](#).

Référence un unique contexte, et initialise le parseur et le gestionnaire de commandes de l'application.

Paramètres

<i>context</i>	le contexte courant de l'application, il n'est pas possible d'en changer pendant l'exécution du programme.
----------------	--

Définition à la ligne 21 du fichier Engine.cpp.

3.34.2.2 calculator : :Engine : :~Engine () [virtual]

Destructeur de [Engine](#).

Libère le parseur et le gestionnaire de commandes.

Définition à la ligne 31 du fichier Engine.cpp.

3.34.3 Documentation des fonctions membres

3.34.3.1 void calculator : :Engine : :drop () const

Appel direct de la commande DROP.

Voir également

[run\(\)](#)

Définition à la ligne 106 du fichier Engine.cpp.

3.34.3.2 void calculator : :Engine : :dupStack () const

Appel direct de la commande SDUP.

Voir également

[run\(\)](#)

Définition à la ligne 72 du fichier Engine.cpp.

3.34.3.3 const Context *const calculator : :Engine : :getContext () const

getter [Context](#).

Renvoie

Le contexte courant, il peut être modifié, mais ne peut pas être remplacé.

Définition à la ligne 27 du fichier Engine.cpp.

3.34.3.4 void calculator : :Engine : :newStack () const

Appel direct de la commande SNEW.

Voir également

[run\(\)](#)

Définition à la ligne 68 du fichier Engine.cpp.

3.34.3.5 void calculator : :Engine : :redo () const

Appel direct de la commande REDO.

Voir également

[run\(\)](#)

Définition à la ligne 48 du fichier Engine.cpp.

3.34.3.6 void calculator : :Engine : :removeCurrentStack () const

Appel direct de la commande SREM.

Voir également

[run\(\)](#)

Définition à la ligne 86 du fichier Engine.cpp.

3.34.3.7 void calculator : :Engine : :run (std : :string *byPassCommand* = " ") const

Méthode centrale de l'application.

Lit le texte dans la ligne de saisie par défaut et lance l'exécution du moteur, si aucun argument, une chaîne vide est utilisée par défaut et conduit à la Commande DUP.

Paramètres

<i>byPass-Command</i>	Une commande passée directement depuis un bouton de l'interface.
-----------------------	--

resultExpressionExecution = _commandManager->executeCommand(command) ; resultExpressionExecution = ""
si la commande s'est bien déroulée resultExpressionExecution = "contenu non évalué d'une expression " si suite à un eval avec problème

Définition à la ligne 110 du fichier Engine.cpp.

3.34.3.8 void calculator : :Engine : :saveAllParameters (const Parameters * *parameters*) const

Utilisation via l'interface des paramètres, permet de sauvegarder les paramètres après validation.

Paramètres

<i>parameters</i>	les paramètres à sauvegarder.
-------------------	-------------------------------

Définition à la ligne 56 du fichier Engine.cpp.

3.34.3.9 void calculator : :Engine : :setAngleUnitTo (const std : :string & *angleUnit*) const

Modifie l'unité de l'angle utilisée par l'application.

Paramètres

<i>angleUnit</i>	le type de constante souhaité sous la forme d'une string.
------------------	---

Voir également

AngleUnit

Définition à la ligne 98 du fichier Engine.cpp.

3.34.3.10 void calculator : :Engine : :setConstantTypeTo (const std : :string & *constantType*) const

Modifie le type de constante utilisé par l'application.

Paramètres

<i>constantType</i>	le type de constante souhaité sous la forme d'une string.
---------------------	---

Voir également

ConstantType

Définition à la ligne 94 du fichier Engine.cpp.

3.34.3.11 void calculator : :Engine : :setCurrentStack (int *index*) const

Utilisation lors d'un clic sur un onglet.

Modifie l'itérateur pointant vers la pile courante du contexte.

Paramètres

<i>index</i>	désigne la nouvelle pile courante
--------------	-----------------------------------

Définition à la ligne 76 du fichier Engine.cpp.

3.34.3.12 void calculator : :Engine : :setInputText (const std : :string & *str*) const

Utilisation par l'interface pour mettre à jour la ligne de saisie du contexte.

[Engine](#) travaille avec cette copie et non directement avec la QLineEdit de l'interface.

Paramètres

<i>str</i>	le texte à placer dans la ligne de saisie du contexte.
------------	--

Définition à la ligne 36 du fichier Engine.cpp.

3.34.3.13 void calculator : :Engine : :undo () const

Appel direct de la commande UNDO.

Voir également

[run\(\)](#)

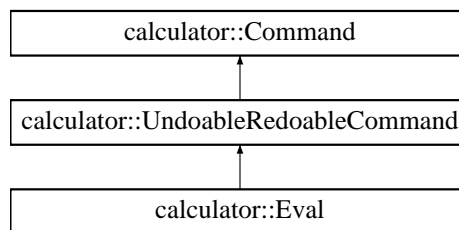
Définition à la ligne 40 du fichier Engine.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/controler/Engine.h
- F :/NetBeansProjects/LO21/Calculatrice/controler/Engine.cpp

3.35 Référence de la classe calculator : :Eval

Graphe d'héritage de calculator : :Eval :



Fonctions membres publiques

- virtual [Eval](#) * [clone](#) () const
Clone la commande.
- void [execute](#) () const throw (CommandException)
Exécute le premier élément de la pile [Number](#) -> copie (number.execute()) [Expression](#) -> boucle qui exécute le contenu de l'expression.
- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

- class **CommandMap**

Additional Inherited Members

3.35.1 Description détaillée

Définition à la ligne 18 du fichier Eval.h.

3.35.2 Documentation des fonctions membres

3.35.2.1 [Eval](#) * calculator : :Eval : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 32 du fichier Eval.cpp.

3.35.2.2 `const Memento * calculator : :Eval : :createMemento () const throw (CommandException) [virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 36 du fichier Eval.cpp.

3.35.2.3 `void calculator : :Eval : :execute () const throw (CommandException) [virtual]`

Exécute le premier élément de la pile [Number](#) -> copie (number.execute()) [Expression](#) -> boucle qui exécute le contenu de l'expression.

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 62 du fichier Eval.cpp.

3.35.2.4 `std : :string calculator : :Eval : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas, ou dans le cas d'une [Expression](#) renvoie le restant non exécuté de l'expression.

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 28 du fichier Eval.cpp.

3.35.2.5 `void calculator : :Eval : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Paramètres

<i>memento</i>	le Memento contenant les données à restituer
----------------	--

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 44 du fichier Eval.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

– F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/Eval.h

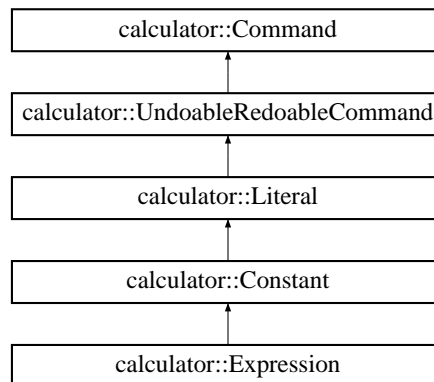
– F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/Eval.cpp

3.36 Référence de la classe calculator : :Expression

Classe [Expression](#), composée de [Command](#).

```
#include <Expression.h>
```

Graphe d'héritage de calculator : :Expression :



Fonctions membres publiques

- [Expression](#) (const [Command](#) *command)
Constructeur de [Expression](#).
- [Expression](#) (const [Command](#) *command1, const [Command](#) *command2)
Constructeur de [Expression](#).
- [Expression](#) (const [Command](#) *command1, const [Command](#) *command2, const [Command](#) *command3)
Constructeur de [Expression](#).
- [Expression](#) (const [Expression](#) *exp1, const [Expression](#) *exp2, const [Command](#) *command)
Constructeur de [Expression](#).
- virtual ~[Expression](#) ()
Destructeur de [Expression](#).
- [Expression](#) * clone () const
Clone la [Constant](#).
- std : :string toString () const
Traduit l'expression en std : :string.
- void append (const [Command](#) *command)
Ajoute une commande à la fin de l'expression.
- const [Command](#) * getFirstCommand () const
getter vers la première [Command](#) contenue dans l'expression
- void removeFirstCommand ()
supprime la première [Command](#) contenue dans l'expression
- bool empty () const
Permet de savoir si l'expression contient au moins une [Command](#).

Additional Inherited Members

3.36.1 Description détaillée

Classe [Expression](#), composée de [Command](#).

Une expression se construit avec au minimum une commande, on peut lui en ajouter et en retirer

Définition à la ligne 21 du fichier Expression.h.

3.36.2 Documentation des constructeurs et destructeur

3.36.2.1 `calculator : :Expression : :Expression (const Command * command)`

Constructeur de [Expression](#).

Copie la commande dans l'expression.

Paramètres

<i>command</i>	
----------------	--

Définition à la ligne 18 du fichier Expression.cpp.

3.36.2.2 `calculator : :Expression : :Expression (const Command * command1, const Command * command2)`

Constructeur de [Expression](#).

Concatène les commandes en une expression, tient compte de la présence d'expressions dans la première commande, la dernière ne devant pas être une nouvelle expression.

Paramètres

<i>command1</i>	une Command .
<i>command2</i>	une Command hors Expression .

Définition à la ligne 64 du fichier Expression.cpp.

3.36.2.3 `calculator : :Expression : :Expression (const Command * command1, const Command * command2, const Command * command3)`

Constructeur de [Expression](#).

Concatène les commandes en une expression, tient compte de la présence d'expressions dans les deux premières commandes, la dernière ne devant pas être une expression.

Paramètres

<i>command1</i>	une Command .
<i>command2</i>	une Command .
<i>command3</i>	une Command hors Expression .

Définition à la ligne 39 du fichier Expression.cpp.

3.36.2.4 `calculator : :Expression : :Expression (const Expression * exp1, const Expression * exp2, const Command * command)`

Constructeur de [Expression](#).

Concatène les deux expressions et la commande en une nouvelle expression.

Paramètres

<i>exp1</i>	une Expression .
<i>exp2</i>	une Expression .
<i>command</i>	une Command hors Expression .

Définition à la ligne 25 du fichier Expression.cpp.

3.36.2.5 calculator : :Expression : :~Expression () [virtual]

Destructeur de [Expression](#).

Libère la mémoire utilisée par les [Command](#) qui la composent

Définition à la ligne 78 du fichier Expression.cpp.

3.36.3 Documentation des fonctions membres

3.36.3.1 void calculator : :Expression : :append (const [Command](#) * *command*)

Ajoute une commande à la fin de l'expression.

Paramètres

<i>command</i>	la commande à ajouter
----------------	-----------------------

Définition à la ligne 108 du fichier Expression.cpp.

3.36.3.2 [Expression](#) * calculator : :Expression : :clone () const [virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator : :Constant](#).

Définition à la ligne 87 du fichier Expression.cpp.

3.36.3.3 bool calculator : :Expression : :empty () const

Permet de savoir si l'expression contient au moins une [Command](#).

Renvoie

true s'il n'y a aucune [Constant](#), faux sinon

Définition à la ligne 121 du fichier Expression.cpp.

3.36.3.4 const [Command](#) * calculator : :Expression : :getFirstCommand () const

getter vers la première [Command](#) contenue dans l'expression

Renvoie

la première [Command](#) contenue dans l'expression

Définition à la ligne 112 du fichier Expression.cpp.

3.36.3.5 std : :string calculator : :Expression : :toString () const [virtual]

Traduit l'expression en std : :string.

Renvoie

l'ensemble des commandes séparées par un espace, le tout est encadré du début à la fin par le caractère '.

Implémente `calculator : :Literal`.

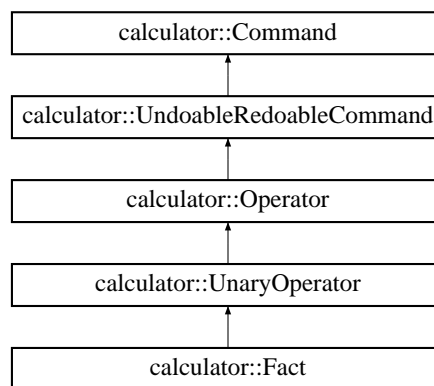
Définition à la ligne 97 du fichier `Expression.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Expression.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Expression.cpp

3.37 Référence de la classe calculator : :Fact

Graphe d'héritage de `calculator : :Fact` :



Fonctions membres publiques

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `Fact * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class `CommandMap`

Additional Inherited Members

3.37.1 Description détaillée

Définition à la ligne 15 du fichier `Fact.h`.

3.37.2 Documentation des fonctions membres

3.37.2.1 `Fact * calculator : :Fact : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 31 du fichier Fact.cpp.

3.37.2.2 std : :string calculator : :Fact : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

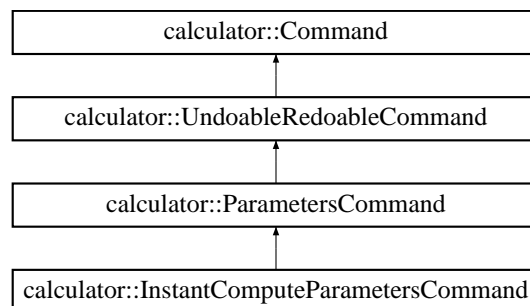
Définition à la ligne 24 du fichier Fact.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Fact.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Fact.cpp

3.38 Référence de la classe calculator : :InstantComputeParametersCommand

Graphes d'héritage de calculator : :InstantComputeParametersCommand :

**Fonctions membres publiques**

- virtual
[InstantComputeParametersCommand * clone \(\) const](#)
Clone l'opérateur.
- void [apply \(Parameters *parameters\) const](#)
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento * createMemento \(\) const throw \(CommandException\)](#)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento \(const Memento *memento\) const throw \(MementoException\)](#)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- **InstantComputeParametersCommand** (const std : :string name)
- **InstantComputeParametersCommand** (const [InstantComputeParametersCommand](#) &orig)

Amis

- class **CommandMap**

3.38.1 Description détaillée

Définition à la ligne 15 du fichier InstantComputeParameterCommand.h.

3.38.2 Documentation des fonctions membres

3.38.2.1 void calculator : :InstantComputeParametersCommand : :apply (**Parameters** * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 28 du fichier InstantComputeParameterCommand.cpp.

3.38.2.2 **InstantComputeParametersCommand** * calculator : :InstantComputeParametersCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 24 du fichier InstantComputeParameterCommand.cpp.

3.38.2.3 const **Memento** * calculator : :InstantComputeParametersCommand : :createMemento () const throw (**CommandException**) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 32 du fichier InstantComputeParameterCommand.cpp.

3.38.2.4 void calculator : :InstantComputeParametersCommand : :restoreFromMemento (const **Memento** * *memento*) const throw (**MementoException**) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

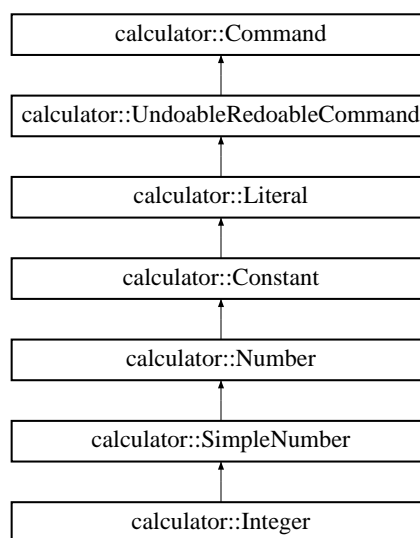
Définition à la ligne 36 du fichier InstantComputeParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/InstantComputeParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/InstantComputeParameterCommand.cpp

3.39 Référence de la classe calculator : :Integer

Graphe d'héritage de calculator : :Integer :



Fonctions membres publiques

- **Integer** (int x)
- **Integer** (const [Integer](#) &orig)
- [Integer](#) * [clone](#) () const
Clone la [Constant](#).
- std : :string [toString](#) () const
Renvoie la [Command](#) sous la forme d'une std : :string.
- double [value](#) () const
- [Integer](#) & [operator=](#) (const [Integer](#) &integer)
- [Integer](#) & [operator+=](#) (const [Integer](#) &integer)
- [Integer](#) & [operator-=](#) (const [Integer](#) &integer)
- [Integer](#) & [operator*=](#) (const [Integer](#) &integer)
- [Integer](#) & [operator/=](#) (const [Integer](#) &integer)
- [Integer](#) & [operator%=](#) (const [Integer](#) &integer)
- [Integer](#) & [pow](#) (const unsigned int &n)

Amis

- class **Rationnal**
- class **Real**

Additional Inherited Members

3.39.1 Description détaillée

Définition à la ligne 17 du fichier Integer.h.

3.39.2 Documentation des fonctions membres

3.39.2.1 `Integer * calculator : :Integer : :clone () const` [virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator : :SimpleNumber](#).

Définition à la ligne 23 du fichier Integer.cpp.

3.39.2.2 `std : :string calculator : :Integer : :toString () const` [virtual]

Renvoie la [Command](#) sous la forme d'une `std : :string`.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme. d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une `std : :string`.

Implémente [calculator : :SimpleNumber](#).

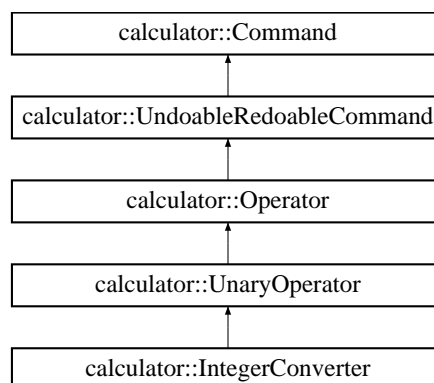
Définition à la ligne 27 du fichier Integer.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Integer.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Integer.cpp

3.40 Référence de la classe calculator : :IntegerConverter

Graphe d'héritage de calculator : :IntegerConverter :



Fonctions membres publiques

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.

- `IntegerConverter * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.40.1 Description détaillée

Définition à la ligne 15 du fichier IntegerConverter.h.

3.40.2 Documentation des fonctions membres

3.40.2.1 IntegerConverter * calculator : :IntegerConverter : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 30 du fichier IntegerConverter.cpp.

3.40.2.2 std : :string calculator : :IntegerConverter : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

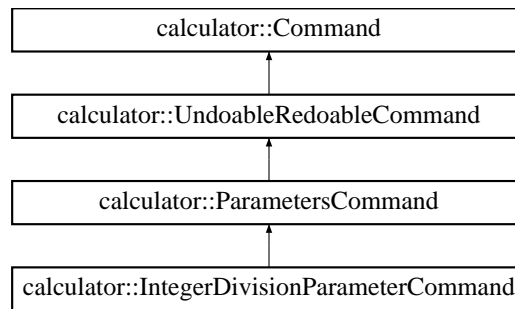
Définition à la ligne 25 du fichier IntegerConverter.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/IntegerConverter.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/IntegerConverter.cpp

3.41 Référence de la classe calculator : :IntegerDivisionParameterCommand

Graphe d'héritage de calculator : :IntegerDivisionParameterCommand :



Fonctions membres publiques

- virtual
[IntegerDivisionParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- [IntegerDivisionParameterCommand](#) (const std : :string name)
- [IntegerDivisionParameterCommand](#) (const [IntegerDivisionParameterCommand](#) &orig)

Amis

- class **CommandMap**

3.41.1 Description détaillée

Définition à la ligne 15 du fichier IntegerDivisionParameterCommand.h.

3.41.2 Documentation des fonctions membres

3.41.2.1 void calculator : :IntegerDivisionParameterCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 28 du fichier IntegerDivisionParameterCommand.cpp.

3.41.2.2 [IntegerDivisionParameterCommand](#) * calculator : :IntegerDivisionParameterCommand : :clone () const
 [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente `calculator : :ParametersCommand`.

Définition à la ligne 24 du fichier IntegerDivisionParameterCommand.cpp.

3.41.2.3 `const Memento * calculator : :IntegerDivisionParameterCommand : :createMemento () const throw (CommandException) [virtual]`

Crée un `Memento` propre à chaque `Command` contenant une copie des données qui vont être modifiées.

Renvoie

un `Memento` spécialisé.

Implémente `calculator : :ParametersCommand`.

Définition à la ligne 32 du fichier IntegerDivisionParameterCommand.cpp.

3.41.2.4 `void calculator : :IntegerDivisionParameterCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure les paramètre dans leur état précédent.

Paramètres

<code>memento</code>	le <code>Memento</code> contenant les paramètres à restituer.
----------------------	---

Implémente `calculator : :ParametersCommand`.

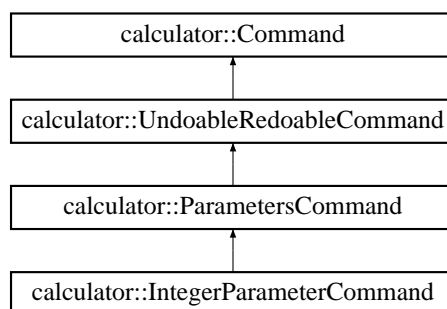
Définition à la ligne 36 du fichier IntegerDivisionParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/IntegerDivisionParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/IntegerDivisionParameterCommand.cpp

3.42 Référence de la classe calculator : :IntegerParameterCommand

Graphe d'héritage de calculator : :IntegerParameterCommand :

**Fonctions membres publiques**

- virtual `IntegerParameterCommand * clone () const`
Clone l'opérateur.
- void `apply (Parameters *parameters) const`
Méthode appelée par `execute()` via un Template Method.

- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- [IntegerParameterCommand](#) (const std : :string name)
- [IntegerParameterCommand](#) (const [IntegerParameterCommand](#) &orig)

Amis

- class [CommandMap](#)

3.42.1 Description détaillée

Définition à la ligne 15 du fichier IntegerParameterCommand.h.

3.42.2 Documentation des fonctions membres

3.42.2.1 void [calculator](#) : :IntegerParameterCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 27 du fichier IntegerParameterCommand.cpp.

3.42.2.2 [IntegerParameterCommand](#) * [calculator](#) : :IntegerParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 23 du fichier IntegerParameterCommand.cpp.

3.42.2.3 const [Memento](#) * [calculator](#) : :IntegerParameterCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator](#) : :ParametersCommand.

Définition à la ligne 31 du fichier IntegerParameterCommand.cpp.

3.42.2.4 void calculator : :IntegerParameterCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

memento	le Memento contenant les paramètres à restituer.
---------	--

Implémente [calculator : :ParametersCommand](#).

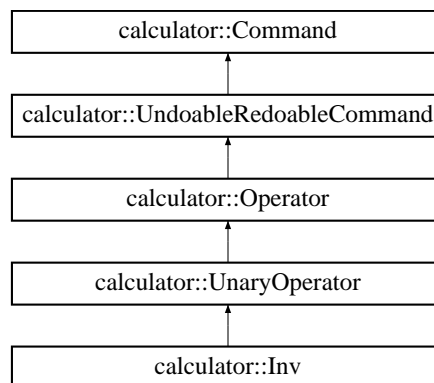
Définition à la ligne 35 du fichier IntegerParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/IntegerParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/IntegerParameterCommand.cpp

3.43 Référence de la classe calculator : :Inv

Graphe d'héritage de calculator : :Inv :



Fonctions membres publiques

- Inv * clone () const
Clone l'opérateur.
- const Number * apply (const Number *n) const throw (ArithmeticException)

Amis

- class CommandMap

Additional Inherited Members

3.43.1 Description détaillée

Définition à la ligne 15 du fichier Inv.h.

3.43.2 Documentation des fonctions membres

3.43.2.1 Inv * calculator : :Inv : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

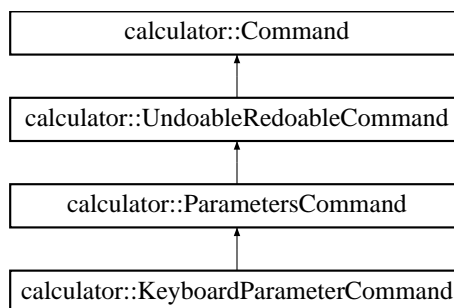
Définition à la ligne 26 du fichier Inv.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Inv.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Inv.cpp

3.44 Référence de la classe calculator : :KeyboardParameterCommand

Graphe d'héritage de calculator : :KeyboardParameterCommand :



Fonctions membres publiques

- virtual [KeyboardParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- [KeyboardParameterCommand](#) (const std : :string name)
- [KeyboardParameterCommand](#) (const [KeyboardParameterCommand](#) &orig)

Amis

- class [CommandMap](#)

3.44.1 Description détaillée

Définition à la ligne 15 du fichier KeyboardParameterCommand.h.

3.44.2 Documentation des fonctions membres

3.44.2.1 void calculator : :KeyboardParameterCommand : :apply (Parameters * parameters) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 27 du fichier KeyboardParameterCommand.cpp.

3.44.2.2 KeyboardParameterCommand * calculator : :KeyboardParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 23 du fichier KeyboardParameterCommand.cpp.

3.44.2.3 const Memento * calculator : :KeyboardParameterCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 31 du fichier KeyboardParameterCommand.cpp.

3.44.2.4 void calculator : :KeyboardParameterCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 35 du fichier KeyboardParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

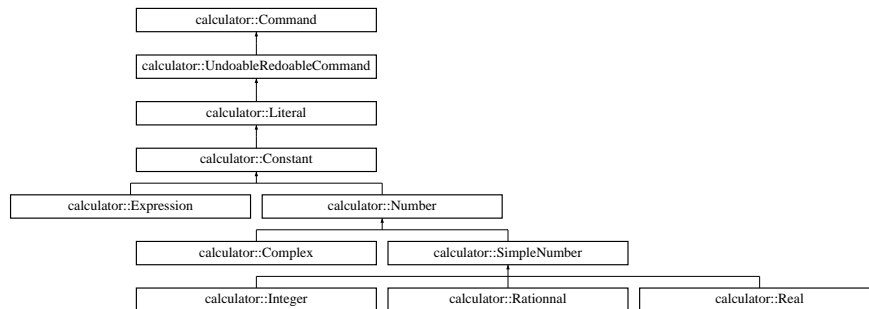
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/KeyboardParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/KeyboardParameterCommand.cpp

3.45 Référence de la classe calculator : :Literal

Classe mère des constantes.

```
#include <Literal.h>
```

Graphe d'héritage de calculator : :Literal :



Fonctions membres publiques

- virtual std : :string [isExecutable](#) () const
Un [Literal](#) est toujours exécutable.
- virtual void [execute](#) () const =0 throw (CommandException)
Méthode virtuelle pure d'entrée du Design Pattern [Command](#).
- virtual std : :string [toString](#) () const =0
Renvoie la [Command](#) sous la forme d'une std : :string.
- virtual [Literal](#) * [clone](#) () const =0
Clone la commande.

Fonctions membres protégées

- [Literal](#) ()
Constructeur de [Literal](#).
- virtual ~[Literal](#) ()
Destructeur de [Literal](#).

3.45.1 Description détaillée

Classe mère des constantes.

En vue d'étendre l'application aux variables pour les commandes utilisateurs.

Définition à la ligne 19 du fichier Literal.h.

3.45.2 Documentation des constructeurs et destructeur

3.45.2.1 calculator : :Literal : :Literal () [protected]

Constructeur de [Literal](#).

On ne passe pas de nom en argument, la méthode toString utilise la valeur du [Literal](#).

Définition à la ligne 18 du fichier Literal.cpp.

3.45.3 Documentation des fonctions membres

3.45.3.1 virtual [Literal](#)* calculator : :Literal : :clone () const [pure virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :Expression](#), [calculator : :Complex](#), [calculator : :Constant](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), [calculator : :SimpleNumber](#), et [calculator : :Number](#).

3.45.3.2 `virtual void calculator : :Literal : :execute () const throw (CommandException) [pure virtual]`

Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, la méthode appelée est précisée par les filles.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :Constant](#).

3.45.3.3 `std : :string calculator : :Literal : :isExecutable () const [virtual]`

Un [Literal](#) est toujours exécutable.

Renvoie

`std : :string("")`.

Implémente [calculator : :UndoableRedoableCommand](#).

Réimplémentée dans [calculator : :Complex](#).

Définition à la ligne 24 du fichier Literal.cpp.

3.45.3.4 `virtual std : :string calculator : :Literal : :toString () const [pure virtual]`

Renvoie la [Command](#) sous la forme d'une `std : :string`.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une `std : :string`.

Réimplémentée à partir de [calculator : :Command](#).

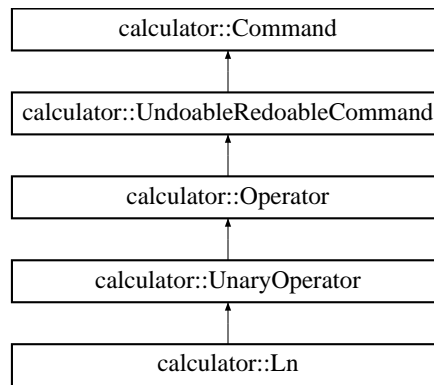
Implémenté dans [calculator : :Expression](#), [calculator : :Complex](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), et [calculator : :SimpleNumber](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Literal.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Literal.cpp

3.46 Référence de la classe calculator : :Ln

Graphe d'héritage de calculator : :Ln :



Fonctions membres publiques

- `std::string isExecutable () const`
Vérifie si la commande est exécutable.
- `Ln * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.46.1 Description détaillée

Définition à la ligne 15 du fichier Ln.h.

3.46.2 Documentation des fonctions membres

3.46.2.1 `Ln * calculator::Ln::clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator::UnaryOperator`.

Définition à la ligne 31 du fichier Ln.cpp.

3.46.2.2 `std::string calculator::Ln::isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant `execute()` afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

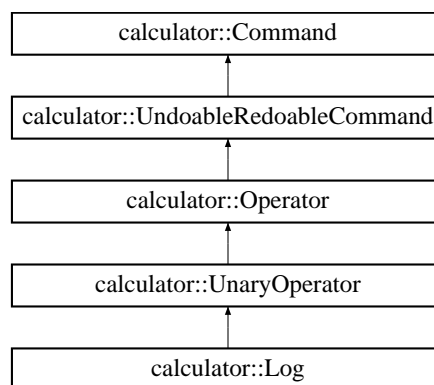
Définition à la ligne 25 du fichier Ln.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Ln.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Ln.cpp

3.47 Référence de la classe calculator : :Log

Graphe d'héritage de calculator : :Log :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- [Log](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * **apply** (const [Number](#) *n) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.47.1 Description détaillée

Définition à la ligne 15 du fichier Log.h.

3.47.2 Documentation des fonctions membres

3.47.2.1 **Log** * calculator : :Log : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 31 du fichier Log.cpp.

3.47.2.2 std : :string calculator : :Log : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

Définition à la ligne 25 du fichier Log.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Log.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Log.cpp

3.48 Référence de la classe calculator : :Logger

[Logger](#) utilisé par le système de journalisation.

```
#include <Logger.h>
```

Fonctions membres publiques

- void **trace** (const std : :string &message)
- void **debug** (const std : :string &message)
- void **error** (const std : :string &message)
- const std : :string **getClassname** () const

Amis

- class **LoggerManager**

3.48.1 Description détaillée

[Logger](#) utilisé par le système de journalisation.

Il existe au maximum un [Logger](#) par classe.

Définition à la ligne 18 du fichier Logger.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/log/Logger.h
- F :/NetBeansProjects/LO21/Calculatrice/log/Logger.cpp

3.49 Référence de la classe calculator : :LoggerManager

Classe conteneur de [Logger](#).

```
#include <LoggerManager.h>
```

Fonctions membres publiques

- [Logger](#) * [getLogger](#) (const std : :string &classname)
Génère ou renvoie un [Logger](#) pour une classe donnée.

Fonctions membres publiques statiques

- static [LoggerManager](#) * [getInstance](#) ()
Singleton [LoggerManager](#).
- static void [deleteInstance](#) ()
Singleton [LoggerManager](#).

3.49.1 Description détaillée

Classe conteneur de [Logger](#).

Chaque classe utilisant un logger stocke effectivement son [Logger](#) dans ce conteneur. Permet d'assurer l'existence d'un unique [Logger](#) par classe et d'identifier la classe à l'origine du log généré.

Définition à la ligne 22 du fichier LoggerManager.h.

3.49.2 Documentation des fonctions membres

3.49.2.1 void calculator : :LoggerManager : :deleteInstance () [static]

Singleton [LoggerManager](#).

Détruit l'unique instance de [LoggerManager](#). Permet de libérer la mémoire prise pour stocker les logger, doit ainsi être utilisé en tout dernier dans l'application.

Définition à la ligne 23 du fichier LoggerManager.cpp.

3.49.2.2 LoggerManager * calculator : :LoggerManager : :getInstance () [static]

Singleton [LoggerManager](#).

Renvoie

crée et/ou renvoie l'unique instance de [LoggerManager](#).

Définition à la ligne 16 du fichier LoggerManager.cpp.

3.49.2.3 Logger * calculator : :LoggerManager : :getLogger (const std : :string & classname)

Génère ou renvoie un [Logger](#) pour une classe donnée.

Paramètres

<i>classname</i>	le nom de la classe qui l'utilise (en Java, on passerait directement une classe pour en avoir ses métadonnées, ce la n'existe pas en C++).
------------------	--

Renvoie

le logger de la classe indiquée, le crée si besoin.

Définition à la ligne 42 du fichier LogManager.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/log/LogManager.h
- F :/NetBeansProjects/LO21/Calculatrice/log/LogManager.cpp

3.50 Référence de la classe calculator : :LogMessage

Un [LogMessage](#) est une entrée de journalisation.

```
#include <LogMessage.h>
```

Fonctions membres publiques

- [LogMessage](#) (int level, const std : :string &message, const std : :string &classname)
Constructeur de [LogMessage](#).
- int [getLevel](#) () const
getter Level.
- std : :string [getClassname](#) () const
getter Classname.
- const std : :string & [getMessage](#) () const
getter Message.
- virtual [~LogMessage](#) ()
Destructeur de [LogMessage](#).

3.50.1 Description détaillée

Un [LogMessage](#) est une entrée de journalisation.

Il contient l'ensemble des informations nécessaires pour être affiché.

Définition à la ligne 19 du fichier LogMessage.h.

3.50.2 Documentation des constructeurs et destructeur

3.50.2.1 calculator : :LogMessage : :LogMessage (int level, const std : :string & message, const std : :string & classname)
[inline]

Constructeur de [LogMessage](#).

Paramètres

<i>level</i>	le niveau d'alerte du LogMessage , {0, 1 ,2} = {TRACE, DEBUG, ERROR}
--------------	--

Voir également

[LogSystem](#).

Paramètres

<i>message</i>	le message à afficher.
<i>classname</i>	le nom de la classe qui publie un LogMessage .

Définition à la ligne 28 du fichier LogMessage.h.

3.50.2.2 virtual calculator : :LogMessage : ~LogMessage () [inline], [virtual]

Destructeur de [LogMessage](#).

Aucun objet à détruire

Définition à la ligne 62 du fichier LogMessage.h.

3.50.3 Documentation des fonctions membres

3.50.3.1 std : :string calculator : :LogMessage : :getClassname () const [inline]

getter Classname.

Renvoie

le nom de la classe à l'origine du [LogMessage](#).

Définition à la ligne 46 du fichier LogMessage.h.

3.50.3.2 int calculator : :LogMessage : :getLevel () const [inline]

getter Level.

Renvoie

Le niveau d'alerte du message.

Définition à la ligne 38 du fichier LogMessage.h.

3.50.3.3 const std : :string& calculator : :LogMessage : :getMessage () const [inline]

getter Message.

Renvoie

le contenu du message.

Définition à la ligne 54 du fichier LogMessage.h.

La documentation de cette classe a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/log/LogMessage.h

3.51 Référence de la classe calculator : :LogSystem

Classe centrale du système de journalisation.

```
#include <LogSystem.h>
```

Fonctions membres publiques statiques

– static void [log](#) (const [LogMessage](#) &message)
Transmet un [LogMessage](#) aux différentes sorties.

Attributs publics statiques

- static int `TRACE` = 0
Niveau de log le plus bas, simple information.
- static int `DEBUG` = 1
Niveau de log intermédiaire, pour les points un peu compliqués du code.
- static int `ERROR` = 2
Niveau de log le plus haut, pour les erreurs d'exécution.

3.51.1 Description détaillée

Classe centrale du système de journalisation.

Définit les niveaux possible de `logMessage`. Cette classe est chargée de publier les `LogMessage` dans la console et un fichier `calculatrice.log` (voir `.cpp`).

Définition à la ligne 23 du fichier `LogSystem.h`.

3.51.2 Documentation des fonctions membres

3.51.2.1 void calculator : :LogSystem : :log (const LogMessage & message) [static]

Transmet un `LogMessage` aux différentes sorties.

Paramètres

<code>message</code>	ue message à publier.
----------------------	-----------------------

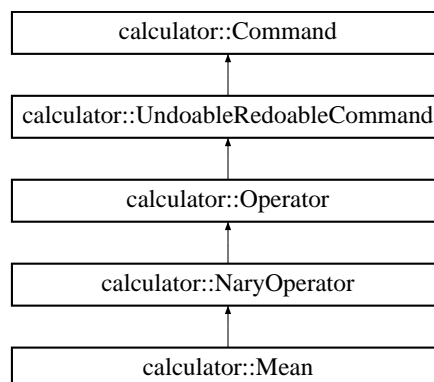
Définition à la ligne 32 du fichier `LogSystem.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/log/LogSystem.h
- F :/NetBeansProjects/LO21/Calculatrice/log/LogSystem.cpp

3.52 Référence de la classe calculator : :Mean

Graphe d'héritage de calculator : :Mean :



Fonctions membres publiques

- virtual `Mean * clone ()` const
Clone l'opérateur.
- const `Number * apply (const Stack *stack)` const throw (`ArithmeticException`)

Amis

– class **CommandMap**

Additional Inherited Members

3.52.1 Description détaillée

Définition à la ligne 15 du fichier Mean.h.

3.52.2 Documentation des fonctions membres

3.52.2.1 **Mean** * calculator : :Mean : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :NaryOperator](#).

Définition à la ligne 23 du fichier Mean.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

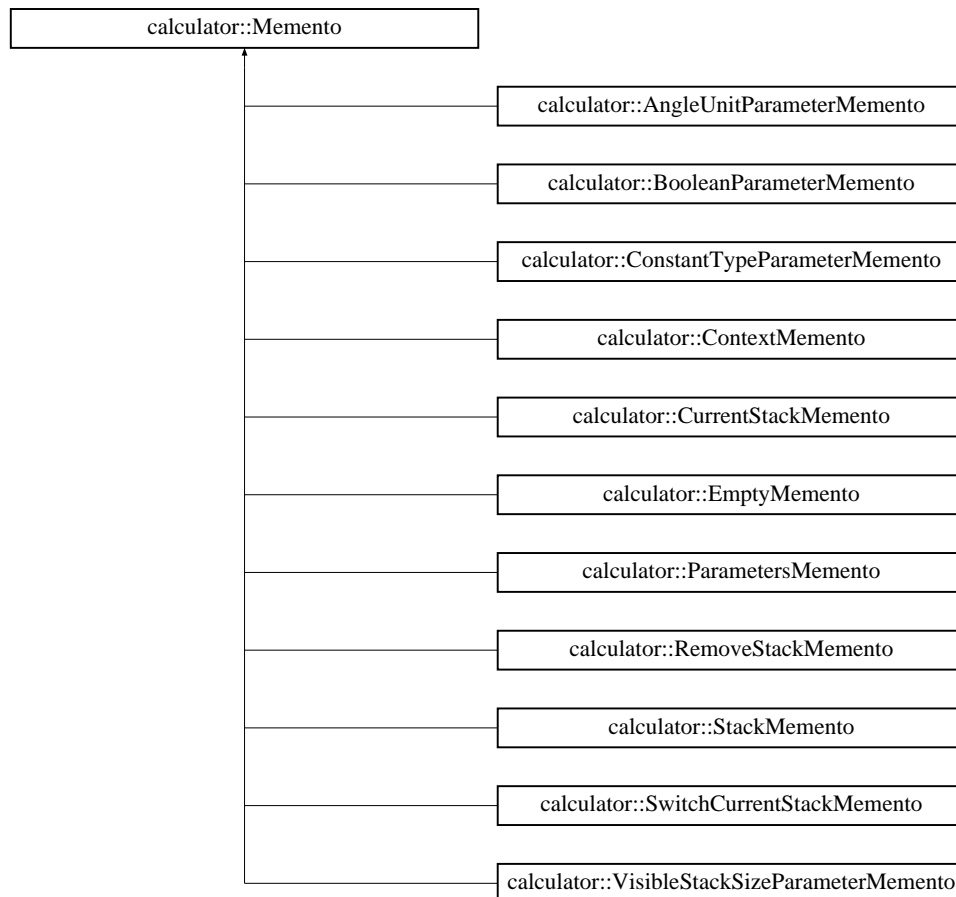
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/Mean.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/Mean.cpp

3.53 Référence de la classe calculator : :Memento

Classe de base des [Memento](#).

```
#include <Memento.h>
```

Graphe d'héritage de calculator : :Memento :



Fonctions membres publiques

- std : :string `getInputString ()` const
getter InputString.
- const `UndoableRedoableCommand *` `getUndoableRedoableCommand ()` const
getter UndoableRedoableCommand.

Fonctions membres protégées

- `Memento (UndoableRedoableCommand *undoableRedoableCommand)`
Constructeur de Memento (protégé).
- virtual `~Memento ()`
Destructeur de Memento Détruit la copie de la commande Seul le CommandManager et le MementoCaretaker peuvent l'utiliser.

Attributs protégés

- const std : :string `_inputString`
la std : :string présente avant l'exécution d'une commande.
- `UndoableRedoableCommand *` `_undoableRedoableCommand`
la commande exécutée à l'origine du Memento.

Amis

- class **CommandManager**
- class **MementoCaretaker**

3.53.1 Description détaillée

Classe de base des [Memento](#).

Non instanciable. Les commandes ne l'utilisent pas directement mais passent par ses filles. Les mementos ont en commun l'enregistrement de la std : :string avant le lancement de l'exécution et d'une copie de la commande utilisée. Les classes filles spécialisent les [Memento](#) en ajoutant les données à mémoriser en fonction des commandes utilisées.

Les mementos ne peuvent pas être modifiés au cours de l'exécution du programme, les données supplémentaires qu'ils contiennent ne possèdent pas de getter/setter, elles sont précisées dans les constructeurs spécialisés des classes filles. On utilise le principe des classes amies pour n'autoriser que les commandes concernées à en lire le contenu.

Définition à la ligne 31 du fichier Memento.h.

3.53.2 Documentation des constructeurs et destructeur

3.53.2.1 calculator : :Memento : :Memento (UndoableRedoableCommand * undoableRedoableCommand)
[protected]

Constructeur de [Memento](#) (protégé).

Paramètres

<i>undoable-Redoable-Command</i>	la commande exécutée est copiée
----------------------------------	---------------------------------

Définition à la ligne 14 du fichier Memento.cpp.

3.53.3 Documentation des fonctions membres

3.53.3.1 std : :string calculator : :Memento : :getInputString () const [inline]

getter InputString.

Renvoie

la std : :string présente avant l'exécution d'une commande.

Définition à la ligne 41 du fichier Memento.h.

3.53.3.2 const UndoableRedoableCommand* calculator : :Memento : :getUndoableRedoableCommand () const
[inline]

getter [UndoableRedoableCommand](#).

Renvoie

la commande exécutée et à l'origine du [Memento](#).

Définition à la ligne 49 du fichier Memento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/Memento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/Memento.cpp

3.54 Référence de la classe calculator : :MementoCaretaker

Fonctions membres publiques

- void **addMemento** (const [Memento](#) *memento)
- unsigned int **getCurrentMementoIndex** () const
- std::string **getPreviousInputString** () const
- bool **hasPrevious** () const
- bool **hasNext** () const
- void **goPrevious** () throw (MementoException)
- void **goNext** () throw (MementoException)
- const [Memento](#) * **getPreviousMemento** () const throw (MementoException)
- const [Memento](#) * **getNextMemento** () const throw (MementoException)

Fonctions membres publiques statiques

- static [MementoCaretaker](#) * **getInstance** ()
- static void **deleteInstance** ()

3.54.1 Description détaillée

Définition à la ligne 18 du fichier MementoCaretaker.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/MementoCaretaker.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/MementoCaretaker.cpp

3.55 Référence de la classe calculator : :MementoException

Classe d'exception pour les [Memento](#).

```
#include <MementoException.h>
```

Fonctions membres publiques

- **MementoException** (const std::string &i)
- const char * **what** () const throw ()

3.55.1 Description détaillée

Classe d'exception pour les [Memento](#).

Exception lancée en cas de mauvaise utilisation d'un [Memento](#) lors d'une action de type ANNULER (UNDO).

Paramètres

<i>e</i>	la cause de l'exception.
----------	--------------------------

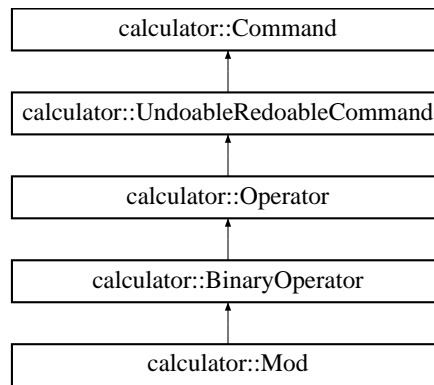
Définition à la ligne 21 du fichier MementoException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/exception/MementoException.h

3.56 Référence de la classe calculator : :Mod

Graphe d'héritage de calculator : :Mod :



Fonctions membres publiques

- `std::string isExecutable () const`
Vérifie si la commande est exécutable.
- `Mod * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n1, const Number *n2) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.56.1 Description détaillée

Définition à la ligne 15 du fichier Mod.h.

3.56.2 Documentation des fonctions membres

3.56.2.1 `Mod * calculator : :Mod : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :BinaryOperator](#).

Définition à la ligne 33 du fichier Mod.cpp.

3.56.2.2 `std::string calculator : :Mod : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std::string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de `calculator::BinaryOperator`.

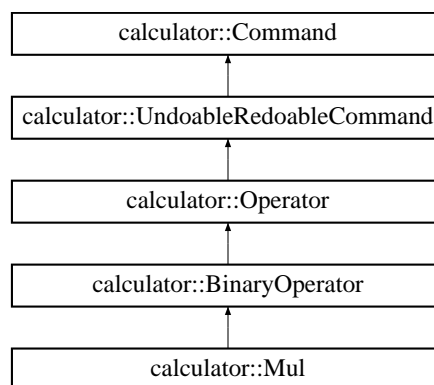
Définition à la ligne 25 du fichier `Mod.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Mod.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Mod.cpp

3.57 Référence de la classe calculator : :Mul

Graphe d'héritage de `calculator::Mul` :



Fonctions membres publiques

- `Mul * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n1, const Number *n2) const throw (ArithmeticException)`

Amis

- class `CommandMap`

Additional Inherited Members

3.57.1 Description détaillée

Définition à la ligne 15 du fichier `Mul.h`.

3.57.2 Documentation des fonctions membres

3.57.2.1 `Mul * calculator::Mul::clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :BinaryOperator`.

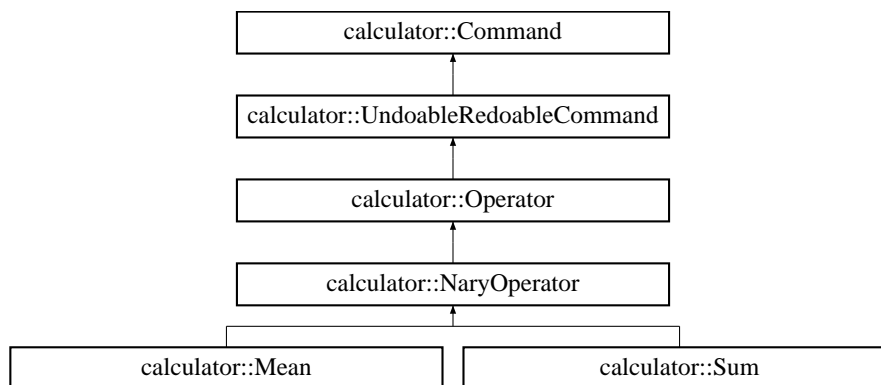
Définition à la ligne 24 du fichier Mul.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Mul.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Mul.cpp

3.58 Référence de la classe calculator : :NaryOperator

Graphe d'héritage de calculator : :NaryOperator :

**Fonctions membres publiques**

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `virtual NaryOperator * clone () const =0`
Clone l'opérateur.
- `const Memento * createMemento () const throw (CommandException)`
Crée un Memento propre à chaque Command contenant une copie des données qui vont être modifiées.

Fonctions membres protégées

- `virtual const Number * apply (const Stack *stack) const =0 throw (ArithmeticException)`
- `NaryOperator (const std : :string name)`

3.58.1 Description détaillée

Définition à la ligne 17 du fichier NaryOperator.h.

3.58.2 Documentation des fonctions membres**3.58.2.1 virtual NaryOperator* calculator : :NaryOperator : :clone () const [pure virtual]**

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :Mean](#), et [calculator : :Sum](#).

3.58.2.2 `const Memento * calculator : :NaryOperator : :createMemento () const throw (CommandException)`
[virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 51 du fichier NaryOperator.cpp.

3.58.2.3 `std : :string calculator : :NaryOperator : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

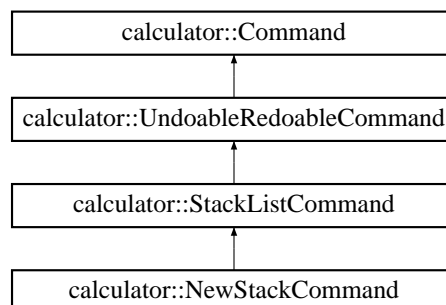
Définition à la ligne 28 du fichier NaryOperator.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/NaryOperator.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/NaryOperator.cpp

3.59 Référence de la classe calculator : :NewStackCommand

Graphe d'héritage de calculator : :NewStackCommand :



Fonctions membres publiques

- std::string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [NewStackCommand](#) * [clone](#) () const
Clone la commande.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

- class [CommandMap](#)

Additional Inherited Members

3.59.1 Description détaillée

Définition à la ligne 15 du fichier NewStackCommand.h.

3.59.2 Documentation des fonctions membres

3.59.2.1 [NewStackCommand](#) * calculator : :NewStackCommand : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 25 du fichier NewStackCommand.cpp.

3.59.2.2 const [Memento](#) * calculator : :NewStackCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 33 du fichier NewStackCommand.cpp.

3.59.2.3 std::string calculator : :NewStackCommand : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std::string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente `calculator::StackListCommand`.

Définition à la ligne 29 du fichier `NewStackCommand.cpp`.

3.59.2.4 `void calculator::NewStackCommand::restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le `Context` dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le `Memento` utilisé est un `StackMemento`, sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une `MementoException`.

Paramètres

<code>memento</code>	le <code>Memento</code> contenant les données à restituer.
----------------------	--

Implémente `calculator::StackListCommand`.

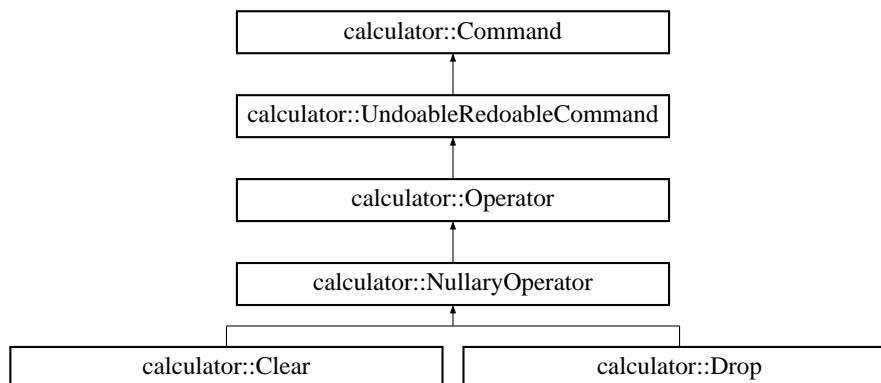
Définition à la ligne 37 du fichier `NewStackCommand.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- `F:/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/NewStackCommand.h`
- `F:/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/NewStackCommand.cpp`

3.60 Référence de la classe `calculator::NullaryOperator`

Graphe d'héritage de `calculator::NullaryOperator` :

**Fonctions membres publiques**

- `std::string isExecutable () const`
Vérifie si la commande est exécutable.
- `virtual NullaryOperator * clone () const =0`
Clone l'opérateur.

Fonctions membres protégées

- `virtual const Memento * createMemento () const =0 throw (CommandException)`
Crée un `Memento` propre à chaque `Command` contenant une copie des données qui vont être modifiées.
- `virtual void restoreFromMemento (const Memento *memento) const =0 throw (MementoException)`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

– **NullaryOperator** (const std : :string name)

3.60.1 Description détaillée

Définition à la ligne 15 du fichier NullaryOperator.h.

3.60.2 Documentation des fonctions membres

3.60.2.1 virtual NullaryOperator* calculator : :NullaryOperator : :clone () const [pure virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :Clear](#), et [calculator : :Drop](#).

3.60.2.2 virtual const Memento* calculator : :NullaryOperator : :createMemento () const throw (CommandException) [protected], [pure virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :Clear](#), et [calculator : :Drop](#).

3.60.2.3 std : :string calculator : :NullaryOperator : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

Réimplémentée dans [calculator : :Clear](#), et [calculator : :Drop](#).

Définition à la ligne 23 du fichier NullaryOperator.cpp.

3.60.2.4 `virtual void calculator : :NullaryOperator : :restoreFromMemento (const Memento * memento) const throw (MementoException) [protected], [pure virtual]`

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Réimplémentée à partir de [calculator : :Operator](#).

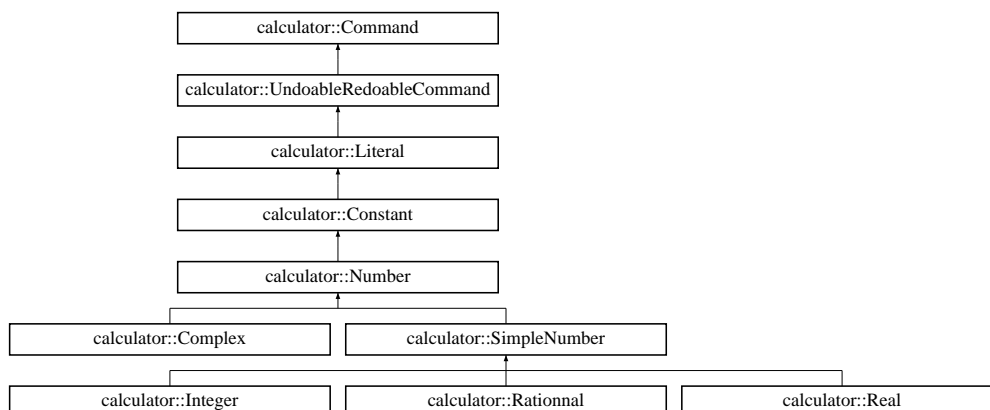
Implémenté dans [calculator : :Clear](#), et [calculator : :Drop](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nullary/NullaryOperator.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nullary/NullaryOperator.cpp

3.61 Référence de la classe calculator : :Number

Graphe d'héritage de calculator : :Number :



Fonctions membres publiques

- virtual [Number](#) * [clone](#) () const =0
Clone la [Constant](#).

Additional Inherited Members

3.61.1 Description détaillée

Définition à la ligne 16 du fichier Number.h.

3.61.2 Documentation des fonctions membres

3.61.2.1 `virtual Number* calculator : :Number : :clone () const [pure virtual]`

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator : :Constant](#).

Implémenté dans [calculator : :Complex](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rational](#), et [calculator : :SimpleNumber](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

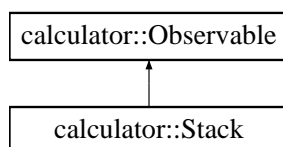
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Number.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Number.cpp

3.62 Référence de la classe calculator : :Observable

Desgin Pattern [Observer](#).

```
#include <Observable.h>
```

Graphe d'héritage de calculator : :Observable :



Fonctions membres publiques

- void [addObserver](#) ([Observer](#) *observer)
Ajoute un observateur à cet objet.
- void [removeObserver](#) ([Observer](#) *observer)
Retire un observateur de cet objet.
- virtual [~Observable](#) ()
Destructeur de [Observable](#).

Fonctions membres protégées

- [Observable](#) ()
Constructeur de [Observable](#).
- void [notify](#) ()
Informe les observateurs que l'objet qu'ils observent a changé.

3.62.1 Description détaillée

Desgin Pattern [Observer](#).

Définition à la ligne 19 du fichier Observable.h.

3.62.2 Documentation des fonctions membres

3.62.2.1 void calculator : :Observable : :addObserver ([Observer](#) * *observer*) [inline]

Ajoute un observateur à cet objet.

Paramètres

<i>observer</i>	un objet héritant de Observer
-----------------	---

Définition à la ligne 26 du fichier Observable.h.

3.62.2.2 `void calculator::Observable::removeObserver (Observer * observer) [inline]`

Retire un observateur de cet objet.

Paramètres

<i>o</i>	l'observateur à retirer.
----------	--------------------------

Définition à la ligne 34 du fichier Observable.h.

La documentation de cette classe a été générée à partir du fichier suivant :

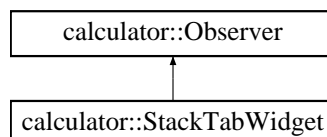
– F :/NetBeansProjects/LO21/Calculatrice/tool/Observable.h

3.63 Référence de la classe calculator::Observer

Desgin Pattern [Observer](#).

```
#include <Observer.h>
```

Grappe d'héritage de calculator::Observer :



Fonctions membres publiques

- virtual void [updateObserver](#) ()=0
Méthode virtuelle pure pour mettre à jour les classes héritant de celle-ci.
- virtual [~Observer](#) ()
Destructeur de [Observer](#).

Fonctions membres protégées

- [Observer](#) ()
Constructeur de [Observer](#).

3.63.1 Description détaillée

Desgin Pattern [Observer](#).

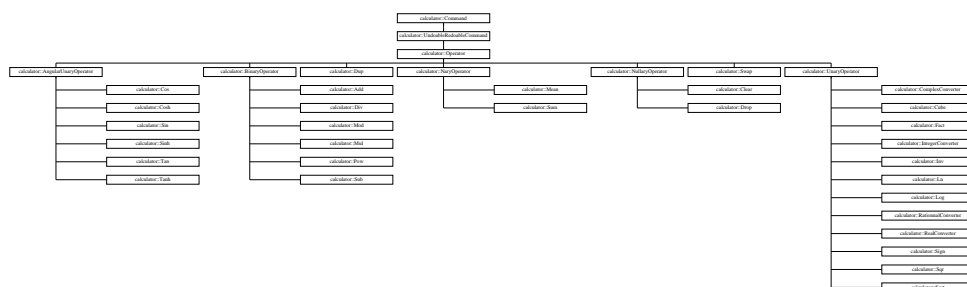
Définition à la ligne 16 du fichier Observer.h.

La documentation de cette classe a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/tool/Observer.h

3.64 Référence de la classe calculator::Operator

Grappe d'héritage de calculator::Operator :



Fonctions membres publiques

- virtual std::string **isExecutable** () const =0
Vérifie si la commande est exécutable.
- void **execute** () const throw (CommandException)
*Méthode d'entrée du Design Pattern **Command**.*
- virtual **Operator** * **clone** () const =0
Clone l'opérateur.
- virtual const **Memento** * **createMemento** () const =0 throw (CommandException)
*Crée un **Memento** propre à chaque **Command** contenant une copie des données qui vont être modifiées.*
- virtual void **restoreFromMemento** (const **Memento** *memento) const throw (MementoException)
*Restaure le **Context** dans l'état précédent l'exécution de la commande.*
- virtual **~Operator** ()
*Destructeur de **Operator**.*

Fonctions membres protégées

- **Operator** (const std::string name)
*Constructeur de **Operator**.*

3.64.1 Description détaillée

Définition à la ligne 18 du fichier Operator.h.

3.64.2 Documentation des constructeurs et destructeur

3.64.2.1 calculator::Operator::Operator (const std::string name) [protected]

Constructeur de **Operator**.

Chaque opérateur est défini par son nom.

Voir également

UndoableRedoable

Paramètres

<i>name</i>	le nom de l'opérateur.
-------------	------------------------

Définition à la ligne 16 du fichier Operator.cpp.

3.64.3 Documentation des fonctions membres

3.64.3.1 virtual **Operator*** calculator::Operator::clone () const [pure virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :Div](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :Swap](#), [calculator : :AngularUnaryOperator](#), [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), [calculator : :Add](#), [calculator : :-Mul](#), [calculator : :Sub](#), [calculator : :Cos](#), [calculator : :Cosh](#), [calculator : :Inv](#), [calculator : :Sin](#), [calculator : :Sinh](#), [calculator : :Tan](#), [calculator : :Tanh](#), [calculator : :NaryOperator](#), [calculator : :BinaryOperator](#), [calculator : :Mean](#), [calculator : :Sum](#), [calculator : :Dup](#), [calculator : :UnaryOperator](#), et [calculator : :NullaryOperator](#).

3.64.3.2 `virtual const Memento* calculator : :Operator : :createMemento () const throw (CommandException)`
[pure virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :Swap](#), [calculator : :AngularUnaryOperator](#), [calculator : :NaryOperator](#), [calculator : :BinaryOperator](#), [calculator : :NullaryOperator](#), [calculator : :Dup](#), et [calculator : :UnaryOperator](#).

3.64.3.3 `void calculator : :Operator : :execute () const throw (CommandException)` [virtual]

Méthode d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, appelle la méthode `eval(Stack* stack)`, un opérateur s'appliquant toujours à une pile

Implémente [calculator : :UndoableRedoableCommand](#).

Définition à la ligne 22 du fichier `Operator.cpp`.

3.64.3.4 `virtual std : :string calculator : :Operator : :isExecutable () const` [pure virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans `calculator : :AngularUnaryOperator`, `calculator : :ComplexConverter`, `calculator : :Cube`, `calculator : :Fact`, `calculator : :IntegerConverter`, `calculator : :Ln`, `calculator : :Log`, `calculator : :RationalConverter`, `calculator : :RealConverter`, `calculator : :Sign`, `calculator : :Sqr`, `calculator : :Sqrt`, `calculator : :Mod`, `calculator : :Pow`, `calculator : :Clear`, `calculator : :Drop`, `calculator : :NaryOperator`, `calculator : :Swap`, `calculator : :BinaryOperator`, `calculator : :Dup`, `calculator : :UnaryOperator`, et `calculator : :NullaryOperator`.

3.64.3.5 `void calculator : :Operator : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]`

Restaure le `Context` dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le `Memento` utilisé est un `StackMemento`, sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une `MementoException`.

Paramètres

<code>memento</code>	le <code>Memento</code> contenant les données à restituer.
----------------------	--

Implémente `calculator : :UndoableRedoableCommand`.

Réimplémentée dans `calculator : :Clear`, `calculator : :Drop`, `calculator : :NullaryOperator`, et `calculator : :Dup`.

Définition à la ligne 30 du fichier `Operator.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- `F : /NetBeansProjects/LO21/Calculatrice/model/command/operator/Operator.h`
- `F : /NetBeansProjects/LO21/Calculatrice/model/command/operator/Operator.cpp`

3.65 Référence de la classe calculator : :Parameters

Types publics

- enum `AngleUnit` { `DEGREE` = 0, `RADIAN` = 1 }
Unité de l'angle pour les opérateurs angulaires.
- enum `ConstantType` { `INTEGER` = 0, `RATIONNAL` = 1, `REAL` = 2 }
Type de constante utilisé dans les calculs.

Fonctions membres publiques

- `Parameters` (const `Parameters` &orig)
Constructeur de copie de `Parameters`.
- `Parameters` (`ConstantType` constantType, `AngleUnit` angleUnit, bool complexMode, bool displayKeyboard, int visibleStackSize, bool instantCompute, bool integerDivision, bool saveOnExit)
Constructeur de `Parameters` détaillé
- `Parameters * clone () const`
Clone l'instance de `Parameters` en vue de la sauvegarder.
- `ConstantType getConstantType () const`
- void `setConstantType` (`ConstantType` constantType)
- `AngleUnit getAngleUnit () const`
- void `setAngleUnit` (`AngleUnit` angleUnit)
- bool `getComplexMode () const`
- void `setComplexMode` (bool complexMode)
- bool `getDisplayKeyboard () const`
- void `setDisplayKeyboard` (bool displayKeyboard)
- unsigned int `getVisibleStackSize () const`
- void `setVisibleStackSize` (int visibleStackSize)
- bool `getInstantCompute () const`
- void `setInstantCompute` (bool instantCompute)
- bool `getIntegerDivision () const`
- void `setIntegerDivision` (bool integerDivision)
- bool `getSaveOnExit () const`
- void `setSaveOnExit` (bool saveOnExit)

Fonctions membres publiques statiques

- static [Parameters](#) * [getInstance](#) ()
 Singleton [Parameters](#).
- static void [deleteInstance](#) ()
 Singleton [Parameters](#).

3.65.1 Description détaillée

Définition à la ligne 15 du fichier Parameters.h.

3.65.2 Documentation des énumérations membres

3.65.2.1 enum calculator : :Parameters : :AngleUnit

Unité de l'angle pour les opérateurs angulaires.

2 unités disponibles, DEGREE est choisi par défaut.

Valeurs énumérées :

DEGREE Angles en degrés.

RADIAN Angles en radians.

Définition à la ligne 22 du fichier Parameters.h.

3.65.2.2 enum calculator : :Parameters : :ConstantType

Type de constante utilisé dans les calculs.

Essentiellement utilisé lors des divisions pour déterminer le type de résultat attendu. 3 types disponibles, INTEGER est choisi par défaut.

Valeurs énumérées :

INTEGER mode entier.

RATIONNAL mode rationnel.

REAL mode réel.

Définition à la ligne 32 du fichier Parameters.h.

3.65.3 Documentation des constructeurs et destructeur

3.65.3.1 calculator : :Parameters : :Parameters (**ConstantType** *constantType*, **AngleUnit** *angleUnit*, bool *complexMode*, bool *displayKeyboard*, int *visibleStackSize*, bool *instantCompute*, bool *integerDivision*, bool *saveOnExit*)

Constructeur de [Parameters](#) détaillé

Paramètres

<i>constantType</i>	
<i>angleUnit</i>	
<i>complexMode</i>	
<i>displayKeyboard</i>	
<i>visibleStackSize</i>	
<i>instantCompute</i>	
<i>integerDivision</i>	
<i>saveOnExit</i>	

Définition à la ligne 47 du fichier Parameters.cpp.

3.65.4 Documentation des fonctions membres

3.65.4.1 Parameters * calculator : :Parameters : :clone () const

Clone l'instance de [Parameters](#) en vue de la sauvegarder.

Renvoie

une copie des paramètres dans un objet [Parameters](#)

Définition à la ligne 68 du fichier Parameters.cpp.

3.65.4.2 void calculator : :Parameters : :deleteInstance () [static]

Singleton [Parameters](#).

Détruit l'instance de [Parameters](#).

Définition à la ligne 21 du fichier Parameters.cpp.

3.65.4.3 Parameters * calculator : :Parameters : :getInstance () [static]

Singleton [Parameters](#).

Renvoie

crée et/ou renvoie l'instance de [Parameters](#).

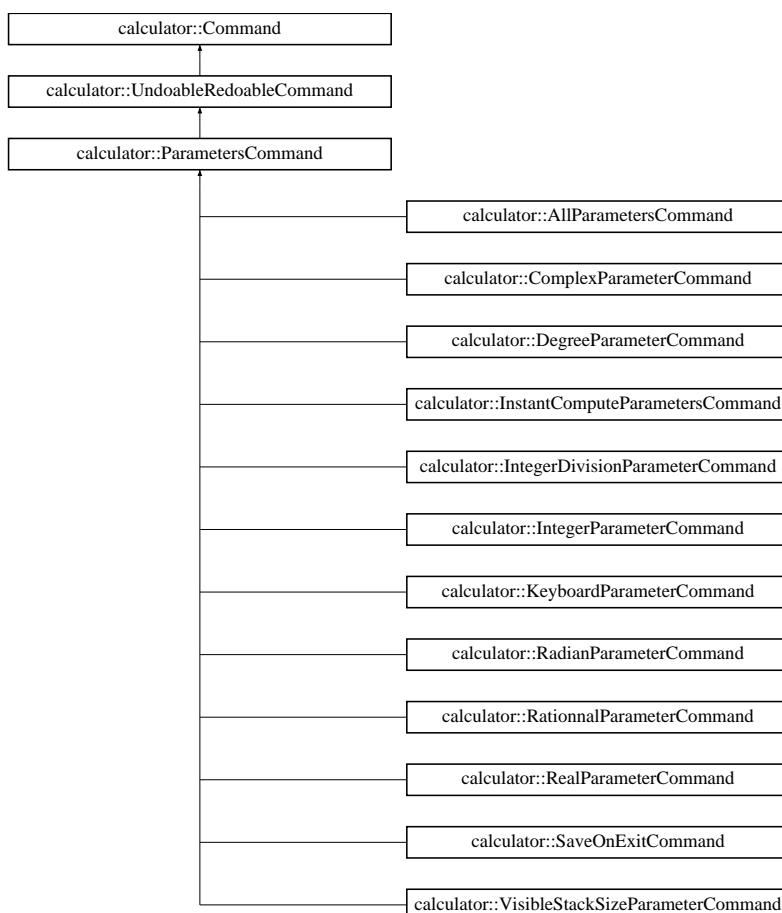
Définition à la ligne 14 du fichier Parameters.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/parameters/Parameters.h
- F :/NetBeansProjects/LO21/Calculatrice/model/parameters/Parameters.cpp

3.66 Référence de la classe calculator : :ParametersCommand

Graphe d'héritage de calculator : :ParametersCommand :



Fonctions membres publiques

- virtual std : :string **isExecutable** () const
Vérifie si la commande est exécutable.
- virtual void **execute** () const throw (CommandException)
Méthode d'entrée du Design Pattern [Command](#).
- virtual [ParametersCommand](#) * **clone** () const =0
Clone l'opérateur.

Fonctions membres protégées

- virtual const [Memento](#) * **createMemento** () const =0 throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- virtual void **restoreFromMemento** (const [Memento](#) *memento) const =0 throw (MementoException)
Restaure les paramètre dans leur état précédent.
- virtual void **apply** ([Parameters](#) *parameters) const =0
Méthode appelée par [execute\(\)](#) via un Template Method.
- [ParametersCommand](#) (const std : :string name)
Constructeur de [ParametersCommand](#).
- virtual ~[ParametersCommand](#) ()
Destructeur de [ParametersCommand](#).

3.66.1 Description détaillée

Définition à la ligne 16 du fichier ParametersCommand.h.

3.66.2 Documentation des constructeurs et destructeur

3.66.2.1 calculator : :ParametersCommand : :ParametersCommand (const std : :string *name*) [protected]

Constructeur de [ParametersCommand](#).

Chaque [ParametersCommand](#) est défini par son nom.

Voir également

[UndoableRedoableCommand](#)

Paramètres

<i>name</i>	le nom de la commande.
-------------	------------------------

Définition à la ligne 16 du fichier ParametersCommand.cpp.

3.66.3 Documentation des fonctions membres

3.66.3.1 virtual void calculator : :ParametersCommand : :apply (Parameters * *parameters*) const [protected],
[pure virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémenté dans [calculator : :AllParametersCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), et [calculator : :SaveOnExitCommand](#).

3.66.3.2 virtual ParametersCommand* calculator : :ParametersCommand : :clone () const [pure virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :AllParametersCommand](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), et [calculator : :SaveOnExitCommand](#).

3.66.3.3 virtual const Memento* calculator : :ParametersCommand : :createMemento () const throw
(CommandException) [protected], [pure virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :AllParametersCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), et [calculator : :VisibleStackSizeParameterCommand](#).

3.66.3.4 `void calculator : :ParametersCommand : :execute () const throw (CommandException) [virtual]`

Méthode d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, appelle la méthode `apply(Parameters* parameters)`, une [ParametersCommand](#) ayant pour but de modifier un ou plusieurs paramètres

Implémente [calculator : :UndoableRedoableCommand](#).

Réimplémentée dans [calculator : :VisibleStackSizeParameterCommand](#).

Définition à la ligne 26 du fichier `ParametersCommand.cpp`.

3.66.3.5 `std : :string calculator : :ParametersCommand : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Une commande modifiant les paramètres est toujours exécutable Si un paramètre d'entrée est incorrect, il sera réglé sur les limites acceptables

Voir également

[Parameters](#)

Renvoie

`std : :string("")` (toujours exécutable)

Voir également

[Command](#)

Implémente [calculator : :UndoableRedoableCommand](#).

Réimplémentée dans [calculator : :VisibleStackSizeParameterCommand](#).

Définition à la ligne 22 du fichier `ParametersCommand.cpp`.

3.66.3.6 `virtual void calculator : :ParametersCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [protected],[pure virtual]`

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :AllParametersCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#),

[calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), et [calculator : :VisibleStackSizeParameterCommand](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/ParametersCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/ParametersCommand.cpp

3.67 Référence de la classe calculator : :ParametersDialog

Fenêtre affichant l'ensemble des paramètres de l'application.

```
#include <ParametersDialog.h>
```

Connecteurs publics

- void [sSave](#) ()
Bouton OK.
- void [sCancel](#) ()
Bouton Annuler.

Fonctions membres publiques

- [ParametersDialog](#) (const [Engine](#) *engine, QWidget *parent=0)
Constructeur de la fenêtre.
- virtual [~ParametersDialog](#) ()
Destructeur de la fenêtre.

3.67.1 Description détaillée

Fenêtre affichant l'ensemble des paramètres de l'application.

Voir également

[Parameters](#).

Définition à la ligne 25 du fichier ParametersDialog.h.

3.67.2 Documentation des constructeurs et destructeur

3.67.2.1 calculator : :ParametersDialog : :ParametersDialog (const Engine * engine, QWidget * parent = 0)

Constructeur de la fenêtre.

Paramètres

<i>engine</i>	le moteur associé à l'application.
<i>parent</i>	le QWidget parent, peut être 0.

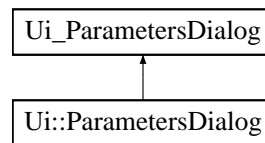
Définition à la ligne 14 du fichier ParametersDialog.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/view/ParametersDialog.h
- F :/NetBeansProjects/LO21/Calculatrice/view/ParametersDialog.cpp

3.68 Référence de la classe Ui : :ParametersDialog

Graphe d'héritage de Ui : :ParametersDialog :



Additional Inherited Members

3.68.1 Description détaillée

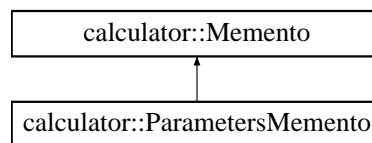
Définition à la ligne 222 du fichier ui_ParametersDialog.h.

La documentation de cette classe a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/ui_ParametersDialog.h

3.69 Référence de la classe calculator : :ParametersMemento

Graphe d'héritage de calculator : :ParametersMemento :



Fonctions membres publiques

– **ParametersMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const [Parameters](#) *parameters)

Amis

– class **AllParametersCommand**

Additional Inherited Members

3.69.1 Description détaillée

Définition à la ligne 17 du fichier ParametersMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

– F :/NetBeansProjects/LO21/Calculatrice/model/memento/ParametersMemento.h
 – F :/NetBeansProjects/LO21/Calculatrice/model/memento/ParametersMemento.cpp

3.70 Référence de la classe calculator : :ParseException

Classe d'exception de parsing Exception lancée si [CalculatorParser](#) ne parvient pas à traduire une chaîne de caractères.

```
#include <ParseException.h>
```

Fonctions membres publiques

- **ParseException** (const std : :string &i)
- const char * **what** () const throw ()

3.70.1 Description détaillée

Classe d'exception de parsing Exception lancée si [CalculatorParser](#) ne parvient pas à traduire une chaîne de caractères.

Paramètres

<i>e</i>	la cause de l'exception.
----------	--------------------------

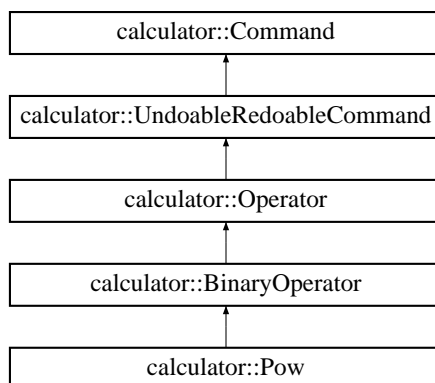
Définition à la ligne 21 du fichier ParseException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/exception/ParseException.h

3.71 Référence de la classe calculator : :Pow

Graphe d'héritage de calculator : :Pow :



Fonctions membres publiques

- std : :string **isExecutable** () const
Vérifie si la commande est exécutable.
- **Pow** * **clone** () const
Clone l'opérateur.
- const **Number** * **apply** (const **Number** *n1, const **Number** *n2) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.71.1 Description détaillée

Définition à la ligne 15 du fichier Pow.h.

3.71.2 Documentation des fonctions membres

3.71.2.1 `Pow * calculator : :Pow : :clone () const` `[virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :BinaryOperator](#).

Définition à la ligne 34 du fichier Pow.cpp.

3.71.2.2 `std : :string calculator : :Pow : :isExecutable () const` `[virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :BinaryOperator](#).

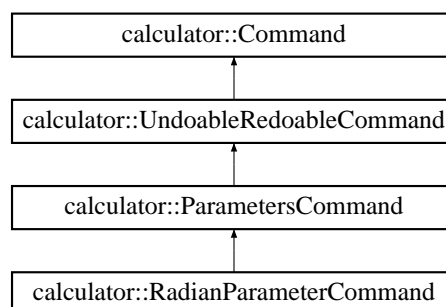
Définition à la ligne 24 du fichier Pow.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Pow.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Pow.cpp

3.72 Référence de la classe calculator : :RadianParameterCommand

Graphe d'héritage de calculator : :RadianParameterCommand :



Fonctions membres publiques

- virtual [RadianParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètre dans leur état précédent.

Fonctions membres protégées

- [RadianParameterCommand](#) (const std : :string name)
- [RadianParameterCommand](#) (const [RadianParameterCommand](#) &orig)

Amis

- class [CommandMap](#)

3.72.1 Description détaillée

Définition à la ligne 15 du fichier RadianParameterCommand.h.

3.72.2 Documentation des fonctions membres

3.72.2.1 void calculator : :RadianParameterCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator](#) : :[ParametersCommand](#).

Définition à la ligne 27 du fichier RadianParameterCommand.cpp.

3.72.2.2 [RadianParameterCommand](#) * calculator : :RadianParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator](#) : :[ParametersCommand](#).

Définition à la ligne 23 du fichier RadianParameterCommand.cpp.

3.72.2.3 const [Memento](#) * calculator : :RadianParameterCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 31 du fichier RadianParameterCommand.cpp.

3.72.2.4 void calculator : :RadianParameterCommand : :restoreFromMemento (const **Memento** * *memento*) const throw (**MementoException**) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

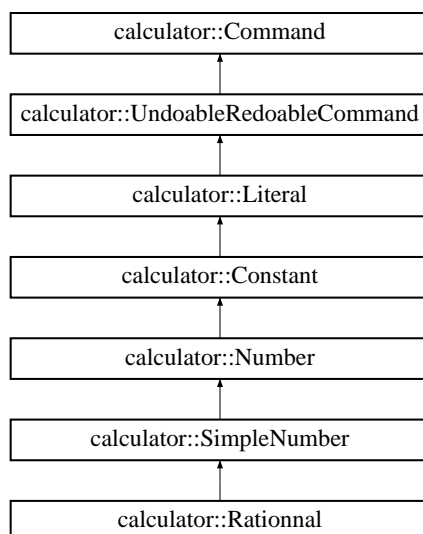
Définition à la ligne 35 du fichier RadianParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RadianParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RadianParameterCommand.cpp

3.73 Référence de la classe calculator : :Rationnal

Graphe d'héritage de calculator : :Rationnal :

**Fonctions membres publiques**

- **Rationnal** (int n, int d=1)
- **Rationnal** (const [Integer](#) &e)
- **Rationnal** (const [Rationnal](#) &orig)
- [Rationnal](#) * *clone* () const
Clone la [Constant](#).
- std : :string *toString* () const
Renvoie la [Command](#) sous la forme d'une std : :string.
- double *value* () const
- [Rationnal](#) & *operator=* (const [Rationnal](#) &r)
- [Rationnal](#) & *operator+=* (const [Rationnal](#) &r)
- [Rationnal](#) & *operator-=* (const [Rationnal](#) &r)
- [Rationnal](#) & *operator*=* (const [Rationnal](#) &r)

- [Rational](#) & **operator/=** (const [Rational](#) &r)
- [Rational](#) & **pow** (const [Integer](#) &e)

Amis

- class **Real**

Additional Inherited Members

3.73.1 Description détaillée

Définition à la ligne 16 du fichier Rational.h.

3.73.2 Documentation des fonctions membres

3.73.2.1 [Rational](#) * calculator : :Rational : :clone () const [virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente [calculator : :SimpleNumber](#).

Définition à la ligne 34 du fichier Rational.cpp.

3.73.2.2 std : :string calculator : :Rational : :toString () const [virtual]

Renvoie la [Command](#) sous la forme d'une std : :string.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme. d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une std : :string.

Implémente [calculator : :SimpleNumber](#).

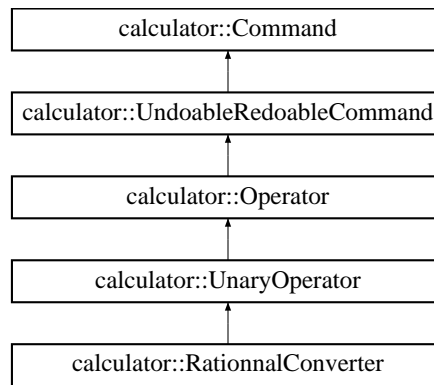
Définition à la ligne 38 du fichier Rational.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Rational.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Rational.cpp

3.74 Référence de la classe calculator : :RationalConverter

Graphe d'héritage de calculator : :RationalConverter :



Fonctions membres publiques

- `std::string isExecutable () const`
Vérifie si la commande est exécutable.
- `RationalConverter * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.74.1 Description détaillée

Définition à la ligne 15 du fichier `RationalConverter.h`.

3.74.2 Documentation des fonctions membres

3.74.2.1 `RationalConverter * calculator : :RationalConverter : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :UnaryOperator`.

Définition à la ligne 29 du fichier `RationalConverter.cpp`.

3.74.2.2 `std::string calculator : :RationalConverter : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant `execute()` afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

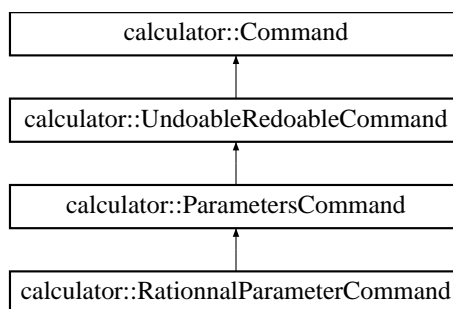
Définition à la ligne 24 du fichier RationalConverter.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/RationalConverter.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/RationalConverter.cpp

3.75 Référence de la classe calculator : :RationalParameterCommand

Graphe d'héritage de calculator : :RationalParameterCommand :

**Fonctions membres publiques**

- virtual [RationalParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- [RationalParameterCommand](#) (const std : :string name)
- [RationalParameterCommand](#) (const [RationalParameterCommand](#) &orig)

Amis

- class [CommandMap](#)

3.75.1 Description détaillée

Définition à la ligne 15 du fichier RationalParameterCommand.h.

3.75.2 Documentation des fonctions membres

3.75.2.1 void calculator : :RationalParameterCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator](#) : [:ParametersCommand](#).

Définition à la ligne 27 du fichier `RationnalParameterCommand.cpp`.

3.75.2.2 `RationnalParameterCommand * calculator : :RationnalParameterCommand : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator](#) : [:ParametersCommand](#).

Définition à la ligne 23 du fichier `RationnalParameterCommand.cpp`.

3.75.2.3 `const Memento * calculator : :RationnalParameterCommand : :createMemento () const throw (CommandException)` [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator](#) : [:ParametersCommand](#).

Définition à la ligne 31 du fichier `RationnalParameterCommand.cpp`.

3.75.2.4 `void calculator : :RationnalParameterCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException)` [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator](#) : [:ParametersCommand](#).

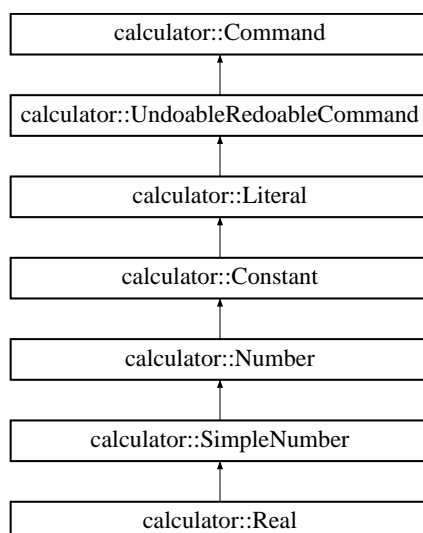
Définition à la ligne 35 du fichier `RationnalParameterCommand.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RationnalParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RationnalParameterCommand.cpp

3.76 Référence de la classe `calculator : :Real`

Graphe d'héritage de `calculator : :Real` :



Fonctions membres publiques

- **Real** (double x)
- **Real** (const **Real** &orig)
- **Real** (const **Integer** &integer)
- **Real** (const **Rationnal** &rational)
- **Real** * **clone** () const
Clone la **Constant**.
- std : :string **toString** () const
Renvoie la **Command** sous la forme d'une std : :string.
- double **value** () const
- **Real** & **operator=** (const **Real** &real)
- **Real** & **operator+=** (const **Real** &real)
- **Real** & **operator-=** (const **Real** &real)
- **Real** & **operator*=** (const **Real** &real)
- **Real** & **operator/=** (const **Real** &real)
- **Real** & **pow** (const **SimpleNumber** &sn)

Additional Inherited Members

3.76.1 Description détaillée

Définition à la ligne 17 du fichier Real.h.

3.76.2 Documentation des fonctions membres

3.76.2.1 **Real** * calculator : :Real : :clone () const [virtual]

Clone la **Constant**.

Renvoie

une copie de la **Constant**.

Implémente **calculator** : :**SimpleNumber**.

Définition à la ligne 32 du fichier Real.cpp.

3.76.2.2 std : :string calculator : :Real : :toString () const [virtual]

Renvoie la **Command** sous la forme d'une std : :string.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme. d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une std : :string.

Implémente [calculator : :SimpleNumber](#).

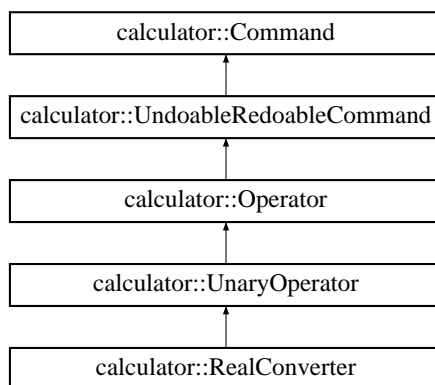
Définition à la ligne 36 du fichier Real.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Real.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Real.cpp

3.77 Référence de la classe calculator : :RealConverter

Graphe d'héritage de calculator : :RealConverter :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- [RealConverter](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * [apply](#) (const [Number](#) *n) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.77.1 Description détaillée

Définition à la ligne 15 du fichier RealConverter.h.

3.77.2 Documentation des fonctions membres

3.77.2.1 [RealConverter](#) * calculator : :RealConverter : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 31 du fichier RealConverter.cpp.

3.77.2.2 std : :string calculator : :RealConverter : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

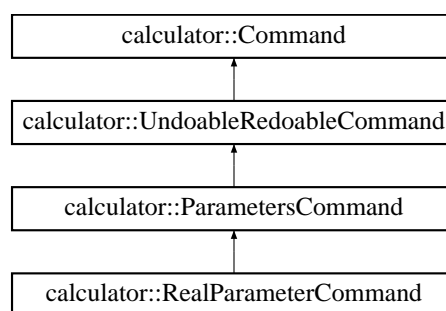
Définition à la ligne 26 du fichier RealConverter.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/RealConverter.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/RealConverter.cpp

3.78 Référence de la classe calculator : :RealParameterCommand

Graphe d'héritage de calculator : :RealParameterCommand :



Fonctions membres publiques

- virtual [RealParameterCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètres dans leur état précédent.

Fonctions membres protégées

- **RealParameterCommand** (const std::string name)
- **RealParameterCommand** (const [RealParameterCommand](#) &orig)

Amis

- class **CommandMap**

3.78.1 Description détaillée

Définition à la ligne 15 du fichier RealParameterCommand.h.

3.78.2 Documentation des fonctions membres

3.78.2.1 void calculator : :RealParameterCommand : :apply (Parameters * parameters) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 27 du fichier RealParameterCommand.cpp.

3.78.2.2 **RealParameterCommand** * calculator : :RealParameterCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 23 du fichier RealParameterCommand.cpp.

3.78.2.3 const **Memento** * calculator : :RealParameterCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 31 du fichier RealParameterCommand.cpp.

3.78.2.4 void calculator : :RealParameterCommand : :restoreFromMemento (const **Memento** * memento) const throw (MementoException) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

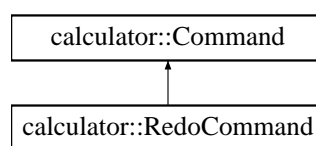
Définition à la ligne 35 du fichier RealParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RealParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/RealParameterCommand.cpp

3.79 Référence de la classe calculator : :RedoCommand

Graphe d'héritage de calculator : :RedoCommand :



Fonctions membres publiques

- virtual [RedoCommand](#) * [clone](#) () const
Clone la commande.
- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- void [execute](#) () const throw (CommandException)
Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Amis

- class **CommandMap**

Additional Inherited Members

3.79.1 Description détaillée

Définition à la ligne 15 du fichier RedoCommand.h.

3.79.2 Documentation des fonctions membres

3.79.2.1 [RedoCommand](#) * calculator : :RedoCommand : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :Command](#).

Définition à la ligne 23 du fichier RedoCommand.cpp.

3.79.2.2 `void calculator : :RedoCommand : :execute () const throw (CommandException) [virtual]`

Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, la méthode appelée est précisée par les filles.

Implémente [calculator : :Command](#).

Définition à la ligne 33 du fichier RedoCommand.cpp.

3.79.2.3 `std : :string calculator : :RedoCommand : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas, ou dans le cas d'une [Expression](#) renvoie le restant non exécuté de l'expression.

Implémente [calculator : :Command](#).

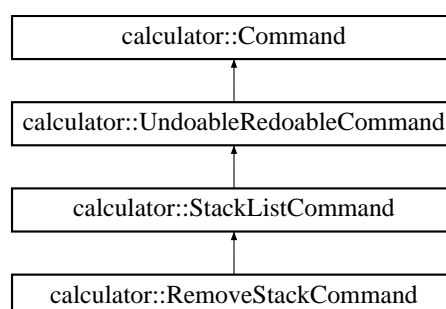
Définition à la ligne 27 du fichier RedoCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/RedoCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/RedoCommand.cpp

3.80 Référence de la classe calculator : :RemoveStackCommand

Graphe d'héritage de calculator : :RemoveStackCommand :



Fonctions membres publiques

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `virtual RemoveStackCommand * clone () const`
Clone la commande.
- `const Memento * createMemento () const throw (CommandException)`
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- `void restoreFromMemento (const Memento *memento) const throw (MementoException)`
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

– class **CommandMap**

Additional Inherited Members

3.80.1 Description détaillée

Définition à la ligne 15 du fichier RemoveStackCommand.h.

3.80.2 Documentation des fonctions membres

3.80.2.1 RemoveStackCommand * calculator : :RemoveStackCommand : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 25 du fichier RemoveStackCommand.cpp.

3.80.2.2 const Memento * calculator : :RemoveStackCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 34 du fichier RemoveStackCommand.cpp.

3.80.2.3 std : :string calculator : :RemoveStackCommand : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 29 du fichier RemoveStackCommand.cpp.

3.80.2.4 void calculator : :RemoveStackCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Implémente [calculator : :StackListCommand](#).

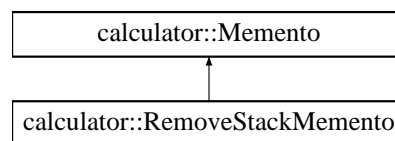
Définition à la ligne 40 du fichier RemoveStackCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/RemoveStackCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/RemoveStackCommand.cpp

3.81 Référence de la classe calculator : :RemoveStackMemento

Graphe d'héritage de calculator : :RemoveStackMemento :



Fonctions membres publiques

- **RemoveStackMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const [Stack](#) *stack, const unsigned int stackIndex)

Amis

- class **RemoveStackCommand**

Additional Inherited Members

3.81.1 Description détaillée

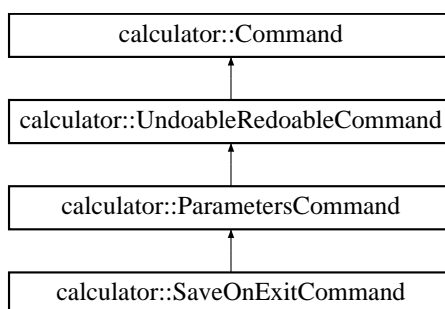
Définition à la ligne 16 du fichier RemoveStackMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/RemoveStackMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/RemoveStackMemento.cpp

3.82 Référence de la classe calculator : :SaveOnExitCommand

Graphe d'héritage de calculator : :SaveOnExitCommand :



Fonctions membres publiques

- virtual [SaveOnExitCommand](#) * [clone](#) () const
Clone l'opérateur.
- void [apply](#) ([Parameters](#) *parameters) const
Méthode appelée par [execute\(\)](#) via un Template Method.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètre dans leur état précédent.

Fonctions membres protégées

- [SaveOnExitCommand](#) (const std : :string name)
- [SaveOnExitCommand](#) (const [SaveOnExitCommand](#) &orig)

Amis

- class [CommandMap](#)

3.82.1 Description détaillée

Définition à la ligne 15 du fichier SaveOnExitCommand.h.

3.82.2 Documentation des fonctions membres

3.82.2.1 void calculator : :SaveOnExitCommand : :apply ([Parameters](#) * *parameters*) const [virtual]

Méthode appelée par [execute\(\)](#) via un Template Method.

Paramètres

<i>parameters</i>	les paramètres à appliquer au Context de l'application
-------------------	--

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 27 du fichier SaveOnExitCommand.cpp.

3.82.2.2 [SaveOnExitCommand](#) * calculator : :SaveOnExitCommand : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente `calculator : :ParametersCommand`.

Définition à la ligne 23 du fichier `SaveOnExitCommand.cpp`.

3.82.2.3 `const Memento * calculator : :SaveOnExitCommand : :createMemento () const throw (CommandException)`
[virtual]

Crée un `Memento` propre à chaque `Command` contenant une copie des données qui vont être modifiées.

Renvoie

un `Memento` spécialisé.

Implémente `calculator : :ParametersCommand`.

Définition à la ligne 31 du fichier `SaveOnExitCommand.cpp`.

3.82.2.4 `void calculator : :SaveOnExitCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException)` [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<code>memento</code>	le <code>Memento</code> contenant les paramètres à restituer.
----------------------	---

Implémente `calculator : :ParametersCommand`.

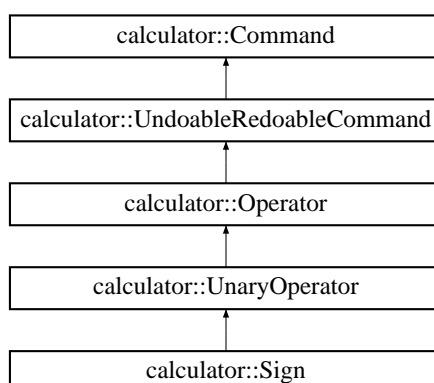
Définition à la ligne 35 du fichier `SaveOnExitCommand.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/SaveOnExitCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/SaveOnExitCommand.cpp

3.83 Référence de la classe calculator : :Sign

Graphe d'héritage de `calculator : :Sign` :



Fonctions membres publiques

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.

- `Sign * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.83.1 Description détaillée

Définition à la ligne 15 du fichier Sign.h.

3.83.2 Documentation des fonctions membres

3.83.2.1 `Sign * calculator : :Sign : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 31 du fichier Sign.cpp.

3.83.2.2 `std : :string calculator : :Sign : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

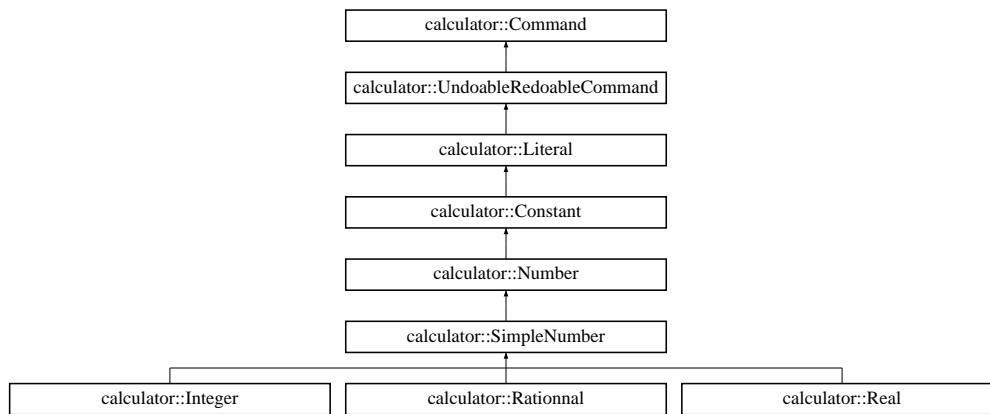
Définition à la ligne 26 du fichier Sign.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sign.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sign.cpp

3.84 Référence de la classe calculator : :SimpleNumber

Graphe d'héritage de calculator : :SimpleNumber :



Fonctions membres publiques

- virtual std : :string [toString](#) () const =0
Renvoie la [Command](#) sous la forme d'une std : :string.
- virtual [SimpleNumber](#) * [clone](#) () const =0
Clone la [Constant](#).
- virtual double [value](#) () const =0
- [operator double](#) () const
Retourne la valeur du [SimpleNumber](#).

Additional Inherited Members

3.84.1 Description détaillée

Définition à la ligne 16 du fichier SimpleNumber.h.

3.84.2 Documentation des fonctions membres

3.84.2.1 virtual [SimpleNumber](#)* calculator : :[SimpleNumber](#) : :[clone](#) () const [pure virtual]

Clone la [Constant](#).

Renvoie

une copie de la [Constant](#).

Implémente calculator : :[Number](#).

Implémenté dans calculator : :[Integer](#), calculator : :[Real](#), et calculator : :[Rationnal](#).

3.84.2.2 calculator : :[SimpleNumber](#) : :[operator double](#) () const [inline]

Retourne la valeur du [SimpleNumber](#).

Renvoie

un double valant la valeur du nombre "simple".

Définition à la ligne 27 du fichier SimpleNumber.h.

3.84.2.3 virtual std : :string calculator : :[SimpleNumber](#) : :[toString](#) () const [pure virtual]

Renvoie la [Command](#) sous la forme d'une std : :string.

D'une manière générale on retourne le nom de la commande, dans le cas des [Constant](#), on retourne la valeur sous la forme. d'une chaîne.

Renvoie

la [Command](#) sous la forme d'une std : :string.

Implémente [calculator : :Literal](#).

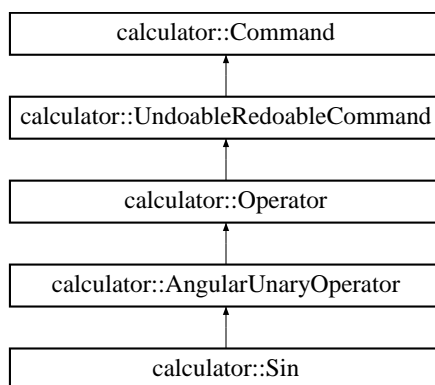
Implémenté dans [calculator : :Integer](#), [calculator : :Real](#), et [calculator : :Rationnal](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/SimpleNumber.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/SimpleNumber.cpp

3.85 Référence de la classe calculator : :Sin

Graphe d'héritage de calculator : :Sin :



Fonctions membres publiques

- [Sin](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * [apply](#) (const [SimpleNumber](#) *n) const throw (ArithmeticException)

Amis

- class [CommandMap](#)

Additional Inherited Members

3.85.1 Description détaillée

Définition à la ligne 15 du fichier Sin.h.

3.85.2 Documentation des fonctions membres

3.85.2.1 [Sin](#) * calculator : :Sin : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :AngularUnaryOperator`.

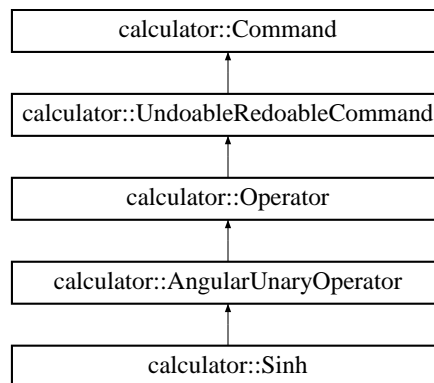
Définition à la ligne 25 du fichier Sin.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sin.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sin.cpp

3.86 Référence de la classe calculator : :Sinh

Graphe d'héritage de calculator : :Sinh :



Fonctions membres publiques

- `Sinh * clone () const`
Clone l'opérateur.
- `const Number * apply (const SimpleNumber *n) const throw (ArithmeticException)`

Amis

- class `CommandMap`

Additional Inherited Members

3.86.1 Description détaillée

Définition à la ligne 15 du fichier Sinh.h.

3.86.2 Documentation des fonctions membres

3.86.2.1 `Sinh * calculator : :Sinh : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :AngularUnaryOperator`.

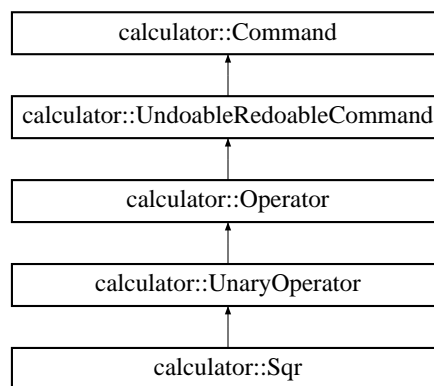
Définition à la ligne 25 du fichier `Sinh.cpp`.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sinh.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sinh.cpp

3.87 Référence de la classe calculator : :Sqr

Graphe d'héritage de calculator : :Sqr :

**Fonctions membres publiques**

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `Sqr * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members**3.87.1 Description détaillée**

Définition à la ligne 15 du fichier `Sqr.h`.

3.87.2 Documentation des fonctions membres**3.87.2.1 Sqr * calculator : :Sqr : :clone () const [virtual]**

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du `CommandManager`, on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 28 du fichier Sqr.cpp.

3.87.2.2 `std : :string calculator : :Sqr : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

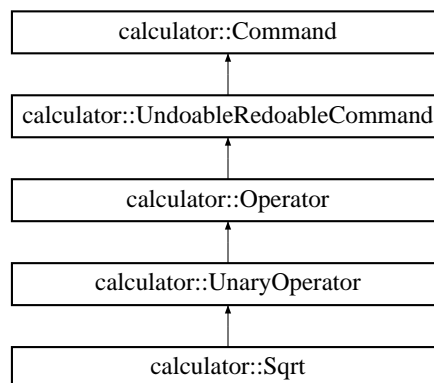
Définition à la ligne 23 du fichier Sqr.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sqr.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sqr.cpp

3.88 Référence de la classe calculator : :Sqrt

Graphe d'héritage de calculator : :Sqrt :

**Fonctions membres publiques**

- `std : :string isExecutable () const`
Vérifie si la commande est exécutable.
- `Sqrt * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.88.1 Description détaillée

Définition à la ligne 15 du fichier Sqrt.h.

3.88.2 Documentation des fonctions membres

3.88.2.1 `Sqrt * calculator : :Sqrt : :clone () const [virtual]`

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :UnaryOperator](#).

Définition à la ligne 34 du fichier Sqrt.cpp.

3.88.2.2 `std : :string calculator : :Sqrt : :isExecutable () const [virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Réimplémentée à partir de [calculator : :UnaryOperator](#).

Définition à la ligne 26 du fichier Sqrt.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

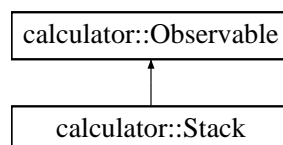
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sqrt.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Sqrt.cpp

3.89 Référence de la classe calculator : :Stack

Classe conteneur de [Constant](#).

```
#include <Stack.h>
```

Graphe d'héritage de calculator : :Stack :



Fonctions membres publiques

- [Stack](#) ()
Constructeur de [Stack](#).
- virtual [~Stack](#) ()
Destructeur de [Stack](#).
- unsigned int [size](#) () const
Nombre de [Constant](#) contenues dans la [Stack](#).
- [Stack](#) * [clone](#) () const
Clone la [Stack](#) et son contenu.
- const [Constant](#) * [getConstant](#) (unsigned int n) const throw (CommandException)
Getter vers la [Constant](#) désignée.
- void [push](#) (const [Constant](#) *constant)
Empile une [Constant](#) en tête de la [Stack](#).
- const [Constant](#) * [top](#) () const
Getter vers la première [Constant](#) de la [Stack](#).
- void [pop](#) ()
Dépile la première [Constant](#) de la [Stack](#) si elle n'est pas vide.
- void [clear](#) ()
Vide la [Stack](#) de son contenu et libère la mémoire occupée par ce contenu.
- bool [empty](#) () const
Indique si la [Stack](#) est vide ou non.
- void [swapTwoFirstConstant](#) ()
Echange les deux premières [Constant](#) de la [Stack](#).
- void [swapAnyConstant](#) (const unsigned int pos1, const unsigned int pos2) throw (ContextException)
Echange les [Constant](#) désignées par leur index respectif.

Amis

- class [XMLParser](#)
Même remarque que pour [StackList](#) à propos de l'amitié avec [XMLParser](#).

Additional Inherited Members

3.89.1 Description détaillée

Classe conteneur de [Constant](#).

Utilise une liste pour pouvoir être parcourue, Les modifications possible sont cependant limitées à celle d'une pile. Une [Stack](#) peut être associée à un [TabStackWidget](#), si une [Stack](#) subit une modification, une mise à jour de la vue est automatiquement demandée

Définition à la ligne 27 du fichier [Stack.h](#).

3.89.2 Documentation des constructeurs et destructeur

3.89.2.1 calculator : :Stack : :Stack ()

Constructeur de [Stack](#).

Crée une [Stack](#) vide.

Définition à la ligne 14 du fichier [Stack.cpp](#).

3.89.2.2 calculator : :Stack : :~Stack () [virtual]

Destructeur de [Stack](#).

Libère la mémoire occupée par les [Constant](#) que la [Stack](#) contient.

Définition à la ligne 17 du fichier [Stack.cpp](#).

3.89.3 Documentation des fonctions membres

3.89.3.1 Stack * calculator : :Stack : :clone () const

Clone la [Stack](#) et son contenu.

Renvoie

une copie de la [Stack](#) et des [Constant](#) qu'elle contient.

Définition à la ligne 53 du fichier Stack.cpp.

3.89.3.2 bool calculator : :Stack : :empty () const

Indique si la [Stack](#) est vide ou non.

Renvoie

true si la [Stack](#) est vide, false sinon.

Définition à la ligne 63 du fichier Stack.cpp.

3.89.3.3 const Constant * calculator : :Stack : :getConstant (unsigned int *n*) const throw (CommandException)

Getter vers la [Constant](#) désignée.

lance une [ContextException](#) si l'index sort de la pile. index : [0 ; [size\(\)](#) - 1].

Paramètres

<i>n</i>	l'index de la Constant désirée parmi la Stack .
----------	---

Renvoie

la [Stack](#) désignée.

Définition à la ligne 71 du fichier Stack.cpp.

3.89.3.4 void calculator : :Stack : :push (const Constant * *constant*)

Empile une [Constant](#) en tête de la [Stack](#).

Paramètres

<i>constant</i>	la Constant à ajouter.
-----------------	--

Définition à la ligne 26 du fichier Stack.cpp.

3.89.3.5 unsigned int calculator : :Stack : :size () const

Nombre de [Constant](#) contenues dans la [Stack](#).

Renvoie

le nombre de [Constant](#) contenues dans la [Stack](#).

Définition à la ligne 67 du fichier Stack.cpp.

3.89.3.6 void calculator : :Stack : :swapAnyConstant (const unsigned int *pos1*, const unsigned int *pos2*) throw (ContextException)

Echange les [Constant](#) désignées par leur index respectif.

Les index commencent à 0

Définition à la ligne 99 du fichier Stack.cpp.

3.89.3.7 const Constant * calculator : :Stack : :top () const

Getter vers la première [Constant](#) de la [Stack](#).

Renvoie

la première [Constant](#) de la [Stack](#), 0 si elle est vide.

Définition à la ligne 31 du fichier Stack.cpp.

3.89.4 Documentation des fonctions amies et associées

3.89.4.1 friend class XMLParser [friend]

Même remarque que pour [StackList](#) à propos de l'amitié avec [XMLParser](#).

Voir également

[StackList](#).

Définition à la ligne 32 du fichier Stack.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Stack.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/Stack.cpp

3.90 Référence de la classe calculator : :StackList

Classe conteneur de [Stack](#).

```
#include <StackList.h>
```

Fonctions membres publiques

- virtual ~StackList ()
Destructeur de StackList.
- StackList * clone () const
Clone l'ensemble des Stack.
- Stack * getStack (const unsigned int n) const throw (ContextException)
Getter vers la Stack désignée.
- Stack * getCurrentStack () const
Getter vers la Stack courante.
- void setCurrentStack (const unsigned int n) throw (ContextException)
Setter de la Stack courante.
- unsigned int getCurrentStackIndex () const
Getter de l'index de la Stack courante parmi la StackList.
- void newStack (const Stack *stack=0)
Ajoute une Stack à la fin de la StackList.
- void removeCurrentStack ()
Supprime de la mémoire la Stack courante.
- void duplicateCurrentStack ()
Duplique la Stack courante et la place après cette dernière.

- void [insertStackAtIndex](#) ([Stack](#) *stack, const unsigned int index)
Insère la [Stack](#) à l'index spécifié dans la [StackList](#).
- bool [empty](#) () const
Indique si la [StackList](#) est vide ou non.
- unsigned int [size](#) () const
Nombre de [Stack](#) contenues dans la [StackList](#).

Fonctions membres publiques statiques

- static [StackList](#) * [getInstance](#) ()
Singleton [StackList](#).
- static void [deleteInstance](#) ()
Singleton [StackList](#).

Amis

- class [XMLParser](#)
Amitié avec [XMLParser](#) pour faciliter la manipulation de la [StackList](#).

3.90.1 Description détaillée

Classe conteneur de [Stack](#).

Maintient à jour un itérateur vers la [Stack](#) courante du [Context](#).

Définition à la ligne 22 du fichier StackList.h.

3.90.2 Documentation des fonctions membres

3.90.2.1 [StackList](#) * calculator : :StackList : :clone () const

Clone l'ensemble des [Stack](#).

Renvoie

une copie des [Stack](#).

Définition à la ligne 44 du fichier StackList.cpp.

3.90.2.2 void calculator : :StackList : :deleteInstance () [static]

Singleton [StackList](#).

Détruit l'unique instance de [StackList](#). Libère la mémoire utilisée par toutes les [Stack](#).

Définition à la ligne 105 du fichier StackList.cpp.

3.90.2.3 void calculator : :StackList : :duplicateCurrentStack ()

Duplique la [Stack](#) courante et la place après cette dernière.

Déplace l'itérateur de la [Stack](#) courante vers la nouvelle [Stack](#).

Définition à la ligne 146 du fichier StackList.cpp.

3.90.2.4 bool calculator : :StackList : :empty () const

Indique si la [StackList](#) est vide ou non.

Renvoie

true si la [StackList](#) est vide, false sinon.

Définition à la ligne 176 du fichier StackList.cpp.

3.90.2.5 Stack * calculator : :StackList : :getCurrentStack () const

Getter vers la [Stack](#) courante.

Renvoie

la [Stack](#) courante du [Context](#).

Définition à la ligne 67 du fichier StackList.cpp.

3.90.2.6 unsigned int calculator : :StackList : :getCurrentStackIndex () const

Getter de l'index de la [Stack](#) courante parmi la [StackList](#).

Renvoie

index de la [Stack](#) courante.

Définition à la ligne 71 du fichier StackList.cpp.

3.90.2.7 StackList * calculator : :StackList : :getInstance () [static]

Singleton [StackList](#).

Renvoie

crée et/ou renvoie l'unique instance de [StackList](#).

Définition à la ligne 98 du fichier StackList.cpp.

3.90.2.8 Stack * calculator : :StackList : :getStack (const unsigned int *n*) const throw (ContextException)

Getter vers la [Stack](#) désignée.

lance une [ContextException](#) si l'index sort de la liste. index : [0 ; [size\(\)](#) - 1].

Paramètres

<i>n</i>	l'index de la Stack désirée parmi la StackList .
----------	--

Renvoie

la [Stack](#) désignée.

Définition à la ligne 56 du fichier StackList.cpp.

3.90.2.9 void calculator : :StackList : :insertStackAtIndex (Stack * *stack*, const unsigned int *index*)

Insère la [Stack](#) à l'index spécifié dans la [StackList](#).

Place l'itérateur sur cette nouvelle [Stack](#). Ne crée pas de copie de la [Stack](#). Si l'index est hors de la liste, l'insertion sera faite à la fin de la [StackList](#)

Paramètres

<i>stack</i>	la stack à insérer dans la StackList
<i>index</i>	la position d'insertion

Définition à la ligne 165 du fichier StackList.cpp.

3.90.2.10 void calculator : :StackList : :newStack (const Stack * *stack* = 0)

Ajoute une [Stack](#) à la fin de la [StackList](#).

Place l'itérateur de la pile courante sur cette nouvelle [Stack](#) Si aucune [Stack](#) en argument, on en crée une nouvelle, sinon on la copie.

Paramètres

<i>stack</i>	une Stack à copier
--------------	------------------------------------

Définition à la ligne 111 du fichier StackList.cpp.

3.90.2.11 void calculator : :StackList : :removeCurrentStack ()

Supprime de la mémoire la [Stack](#) courante.

Déplace l'itérateur sur la [Stack](#) suivante si elle existe, la précédente sinon.

Définition à la ligne 125 du fichier StackList.cpp.

3.90.2.12 void calculator : :StackList : :setCurrentStack (const unsigned int *n*) throw (ContextException)

Setter de la [Stack](#) courante.

lance une [ContextException](#) si l'index sort de la liste. index : [0 ; [size\(\)](#) - 1].

Paramètres

<i>n</i>	index de la Stack contenue dans la StackList .
----------	--

Définition à la ligne 83 du fichier StackList.cpp.

3.90.2.13 unsigned int calculator : :StackList : :size () const [inline]

Nombre de [Stack](#) contenues dans la [StackList](#).

Renvoie

le nombre de [Stack](#) contenues dans la [StackList](#)

Définition à la ligne 127 du fichier StackList.h.

3.90.3 Documentation des fonctions amies et associées

3.90.3.1 friend class XMLParser [friend]

Amitié avec [XMLParser](#) pour faciliter la manipulation de la [StackList](#).

Entorse à l'encapsulation volontaire (manque de temps et côté pratique, plus rapide que de réimplémenter un itérateur, on aurait aussi pu hériter de std : :list).

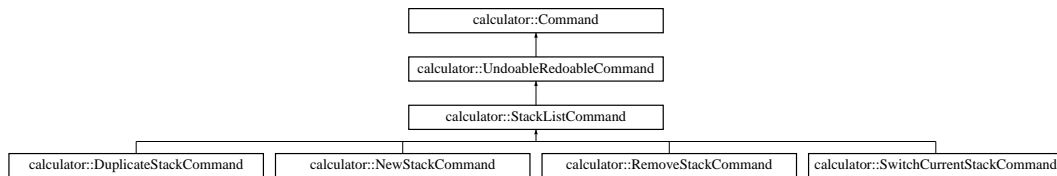
Définition à la ligne 30 du fichier StackList.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/StackList.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/literal/StackList.cpp

3.91 Référence de la classe calculator : :StackListCommand

Graphe d'héritage de calculator : :StackListCommand :



Fonctions membres publiques

- virtual std : :string [isExecutable](#) () const =0
Vérifie si la commande est exécutable.
- void [execute](#) () const throw (CommandException)
Méthode d'entrée du Design Pattern [Command](#).
- virtual [StackListCommand](#) * [clone](#) () const =0
Clone la commande.
- virtual [~StackListCommand](#) ()
Destructeur de [StackListCommand](#).

Fonctions membres protégées

- virtual const [Memento](#) * [createMemento](#) () const =0 throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- virtual void [restoreFromMemento](#) (const [Memento](#) *memento) const =0 throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.
- virtual void [apply](#) ([StackList](#) *stacklist) const =0 throw (CommandException)
Méthode appelée par [execute\(\)](#) via un Template Method.
- [StackListCommand](#) (const std : :string name)
Constructeur de [StackListCommand](#).

3.91.1 Description détaillée

Définition à la ligne 16 du fichier StackListCommand.h.

3.91.2 Documentation des constructeurs et destructeur

3.91.2.1 calculator : :StackListCommand : :StackListCommand (const std : :string *name*) [protected]

Constructeur de [StackListCommand](#).

Chaque [StackListCommand](#) est défini par son nom.

Voir également

[UndoableRedoableCommand](#)

Paramètres

<i>name</i>	le nom de la commande.
-------------	------------------------

Définition à la ligne 15 du fichier StackListCommand.cpp.

3.91.3 Documentation des fonctions membres

3.91.3.1 `virtual void calculator : :StackListCommand : :apply (StackList * stacklist) const throw (CommandException)`
`[protected], [pure virtual]`

Méthode appelée par `execute()` via un Template Method.

Paramètres

<code>stacklist</code>	la liste de Stack sur laquelle on applique la commande.
------------------------	---

3.91.3.2 `virtual StackListCommand* calculator : :StackListCommand : :clone () const` `[pure virtual]`

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente `calculator : :UndoableRedoableCommand`.

Implémenté dans `calculator : :DuplicateStackCommand`, `calculator : :NewStackCommand`, `calculator : :RemoveStackCommand`, et `calculator : :SwitchCurrentStackCommand`.

3.91.3.3 `virtual const Memento* calculator : :StackListCommand : :createMemento () const throw (CommandException)`
`[protected], [pure virtual]`

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente `calculator : :UndoableRedoableCommand`.

Implémenté dans `calculator : :DuplicateStackCommand`, `calculator : :NewStackCommand`, `calculator : :RemoveStackCommand`, et `calculator : :SwitchCurrentStackCommand`.

3.91.3.4 `void calculator : :StackListCommand : :execute () const throw (CommandException)` `[virtual]`

Méthode d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, appelle la méthode `apply(StackList* stacklist)`, une [StackList-Command](#) ayant pour but de gérer l'ensemble des [Stack](#)

Implémente `calculator : :UndoableRedoableCommand`.

Définition à la ligne 21 du fichier `StackListCommand.cpp`.

3.91.3.5 `virtual std : :string calculator : :StackListCommand : :isExecutable () const` `[pure virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant `execute()` afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :UndoableRedoableCommand](#).

Implémenté dans [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), et [calculator : :SwitchCurrentStackCommand](#).

3.91.3.6 virtual void calculator : :StackListCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [protected],[pure virtual]

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Implémente [calculator : :UndoableRedoableCommand](#).

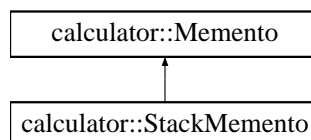
Implémenté dans [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), et [calculator : :SwitchCurrentStackCommand](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/StackListCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/StackListCommand.cpp

3.92 Référence de la classe calculator : :StackMemento

Graphe d'héritage de calculator : :StackMemento :

**Amis**

- class **Operator**
- class **UnaryOperator**
- class **AngularUnaryOperator**
- class **BinaryOperator**
- class **NaryOperator**
- class **Clear**
- class **Drop**
- class **Eval**
- class **RemoveStackCommand**
- class **Swap**

Additional Inherited Members**3.92.1 Description détaillée**

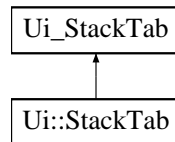
Définition à la ligne 17 du fichier StackMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/StackMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/StackMemento.cpp

3.93 Référence de la classe Ui : :StackTab

Graphe d'héritage de Ui : :StackTab :



Additional Inherited Members

3.93.1 Description détaillée

Définition à la ligne 45 du fichier ui_StackTab.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/ui_StackTab.h

3.94 Référence de la classe calculator : :StackTabList

Classe conteneur de [StackTabWidget](#).

```
#include <StackTabList.h>
```

Fonctions membres publiques

- [StackTabList](#) (const [Engine](#) *const engine)
Constructeur de [StackTabList](#).
- virtual [~StackTabList](#) ()
Destructeur de [StackTabList](#).
- [StackTabWidget](#) * [getTab](#) (const unsigned int n) const throw (ViewException)
Getter vers le [StackTabWidget](#) désigné.
- [StackTabWidget](#) * [getCurrentTab](#) () const
Getter vers le [StackTabWidget](#) courant.
- void [setCurrentTab](#) (const unsigned int n) throw (ViewException)
Setter du [StackTabWidget](#) courant.
- unsigned int [getCurrentTabIndex](#) () const
Getter de l'index du [StackTabWidget](#) courant parmi la [StackTabList](#).
- void [newTab](#) ()
Ajoute un [StackTabWidget](#) à la fin de la [StackTabList](#).
- void [removeCurrentTab](#) () throw (ViewException)
Supprime de la mémoire le [StackTabWidget](#) courant.
- void [duplicateCurrentTab](#) ()
Duplique le [StackTabWidget](#) courant et le place après ce dernier.
- unsigned int [size](#) () const
Nombre de [StackTabWidget](#) contenues dans la [StackTabList](#).

3.94.1 Description détaillée

Classe conteneur de [StackTabWidget](#).

Maintient à jour un itérateur vers le [StackTabWidget](#) actuellement affiché.

Définition à la ligne 23 du fichier StackTabList.h.

3.94.2 Documentation des constructeurs et destructeur

3.94.2.1 calculator : :StackTabList : :StackTabList (const Engine *const engine)

Constructeur de [StackTabList](#).

Paramètres

<i>engine</i>	le moteur gérant l'affichage des Stack .
---------------	--

Définition à la ligne 17 du fichier StackTabList.cpp.

3.94.2.2 calculator : :StackTabList : :~StackTabList () [virtual]

Destructeur de [StackTabList](#).

Libère la mémoire utilisée par les StackTab.

Définition à la ligne 34 du fichier StackTabList.cpp.

3.94.3 Documentation des fonctions membres

3.94.3.1 void calculator : :StackTabList : :duplicateCurrentTab ()

Duplique le [StackTabWidget](#) courant et le place après ce dernier.

Déplace l'itérateur du [StackTabWidget](#) courant vers le nouveau [StackTabWidget](#).

Définition à la ligne 103 du fichier StackTabList.cpp.

3.94.3.2 StackTabWidget * calculator : :StackTabList : :getCurrentTab () const

Getter vers le [StackTabWidget](#) courant.

Renvoie

le [StackTabWidget](#) courant de la vue (et donc du contexte)

Définition à la ligne 43 du fichier StackTabList.cpp.

3.94.3.3 unsigned int calculator : :StackTabList : :getCurrentTabIndex () const

Getter de l'index du [StackTabWidget](#) courant parmi la [StackTabList](#).

Renvoie

index du [StackTabWidget](#) courant.

Définition à la ligne 58 du fichier StackTabList.cpp.

3.94.3.4 StackTabWidget * calculator : :StackTabList : :getTab (const unsigned int n) const throw (ViewException)

Getter vers le [StackTabWidget](#) désigné.

lance une [ViewException](#) si l'index sort de la liste. index : [0 ; size() - 1].

Paramètres

<i>n</i>	l'index du StackTabWidget désiré parmi la StackTabList .
----------	--

Renvoie

le [StackTabWidget](#) désignée.

Définition à la ligne 47 du fichier StackTabList.cpp.

3.94.3.5 void calculator : :StackTabList : :newTab ()

Ajoute un [StackTabWidget](#) à la fin de la [StackTabList](#).

Place l'itérateur de la pile courante affichée sur ce nouveau [StackTabWidget](#)

Définition à la ligne 80 du fichier StackTabList.cpp.

3.94.3.6 void calculator : :StackTabList : :removeCurrentTab () throw (ViewException)

Supprime de la mémoire le [StackTabWidget](#) courant.

Déplace l'itérateur sur le [StackTabWidget](#) suivante si il existe, le précédent sinon.

Définition à la ligne 90 du fichier StackTabList.cpp.

3.94.3.7 void calculator : :StackTabList : :setCurrentTab (const unsigned int *n*) throw (ViewException)

Setter du [StackTabWidget](#) courant.

lance une [ViewException](#) si l'index sort de la liste. index : [0 ; [size\(\)](#) - 1].

Paramètres

<i>n</i>	index du StackTabWidget contenu dans la StackTabList .
----------	--

Définition à la ligne 68 du fichier StackTabList.cpp.

3.94.3.8 unsigned int calculator : :StackTabList : :size () const [inline]

Nombre de [StackTabWidget](#) contenues dans la [StackTabList](#).

Renvoie

le nombre de [Stack](#) contenues dans la [StackTabList](#)

Définition à la ligne 89 du fichier StackTabList.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

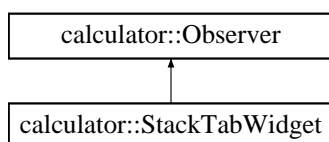
- F :/NetBeansProjects/LO21/Calculatrice/view/StackTabList.h
- F :/NetBeansProjects/LO21/Calculatrice/view/StackTabList.cpp

3.95 Référence de la classe calculator : :StackTabWidget

Élément du TabWidget spécialisé pour afficher les [Stack](#).

```
#include <StackTabWidget.h>
```

Graphe d'héritage de calculator : :StackTabWidget :



Fonctions membres publiques

- [StackTabWidget](#) ([Stack](#) *const stack, [QWidget](#) *parent=0)
Constructeur de [StackTabWidget](#).
- virtual [~StackTabWidget](#) ()
Destructeur de [StackTabWidget](#).
- void [reloadTabContent](#) ()
Recharge le contenu affiché à partir de la [Stack](#).
- void [updateObserver](#) ()
Entraine le rechargement de l'affichage de la pile.

Additional Inherited Members

3.95.1 Description détaillée

Elément du [TabWidget](#) spécialisé pour afficher les [Stack](#).
 Utilise le Design Pattern [Observer](#) pour se maintenir à jour.
 Définition à la ligne 22 du fichier [StackTabWidget.h](#).

3.95.2 Documentation des constructeurs et destructeur

3.95.2.1 calculator : :StackTabWidget : :StackTabWidget ([Stack](#) *const stack, [QWidget](#) * parent = 0)

Constructeur de [StackTabWidget](#).
 Ajoute un observateur à la [Stack](#) associée à ce widget.

Paramètres

<i>stack</i>	la stack associée à ce widget.
<i>parent</i>	le widget parent.

Définition à la ligne 14 du fichier [StackTabWidget.cpp](#).

3.95.2.2 calculator : :StackTabWidget : :~StackTabWidget () [virtual]

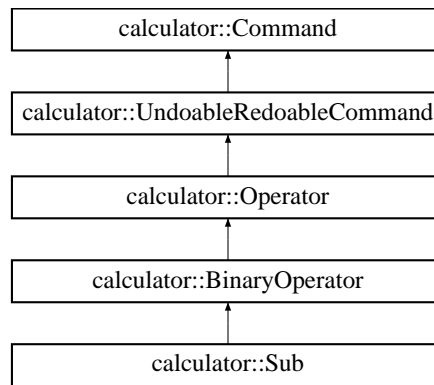
Destructeur de [StackTabWidget](#).
 retire l'observateur de la [Stack](#) associée.
 Définition à la ligne 26 du fichier [StackTabWidget.cpp](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/view/StackTabWidget.h
- F :/NetBeansProjects/LO21/Calculatrice/view/StackTabWidget.cpp

3.96 Référence de la classe calculator : :Sub

Graphe d'héritage de calculator : :Sub :



Fonctions membres publiques

- `Sub * clone () const`
Clone l'opérateur.
- `const Number * apply (const Number *n1, const Number *n2) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.96.1 Description détaillée

Définition à la ligne 15 du fichier Sub.h.

3.96.2 Documentation des fonctions membres

3.96.2.1 `Sub * calculator : :Sub : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :BinaryOperator](#).

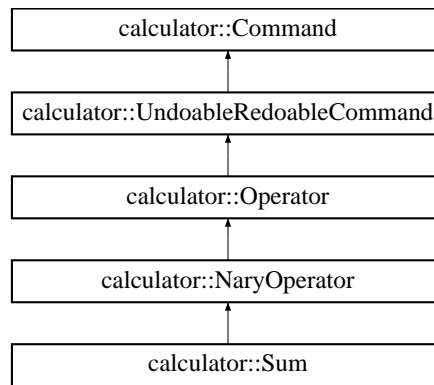
Définition à la ligne 24 du fichier Sub.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Sub.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/binary/Sub.cpp

3.97 Référence de la classe calculator : :Sum

Graphe d'héritage de calculator : :Sum :



Fonctions membres publiques

- virtual [Sum](#) * [clone](#) () const
Clone l'opérateur.
- const [Number](#) * **apply** (const [Stack](#) *stack) const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.97.1 Description détaillée

Définition à la ligne 15 du fichier Sum.h.

3.97.2 Documentation des fonctions membres

3.97.2.1 `Sum * calculator::Sum::clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator::NaryOperator](#).

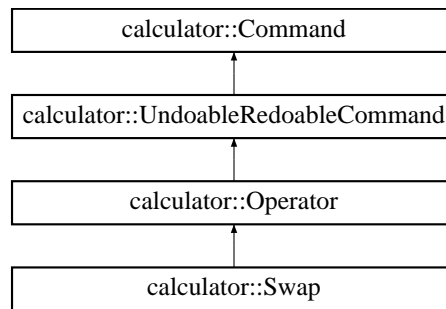
Définition à la ligne 23 du fichier Sum.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/Sum.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/nary/Sum.cpp

3.98 Référence de la classe calculator : :Swap

Graphe d'héritage de calculator : :Swap :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [Swap](#) * [clone](#) () const
Clone l'opérateur.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Amis

- class **CommandMap**

Additional Inherited Members

3.98.1 Description détaillée

Définition à la ligne 15 du fichier Swap.h.

3.98.2 Documentation des fonctions membres

3.98.2.1 [Swap](#) * calculator : :Swap : :clone () const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Définition à la ligne 25 du fichier Swap.cpp.

3.98.2.2 const [Memento](#) * calculator : :Swap : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 29 du fichier Swap.cpp.

3.98.2.3 std : :string calculator : :Swap : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

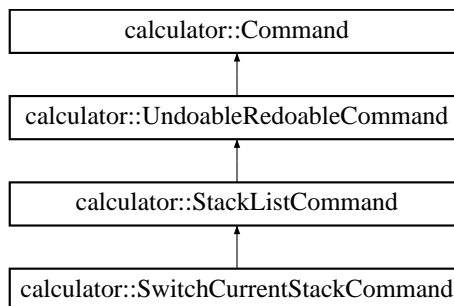
Définition à la ligne 33 du fichier Swap.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/Swap.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/Swap.cpp

3.99 Référence de la classe calculator : :SwitchCurrentStackCommand

Graphe d'héritage de calculator : :SwitchCurrentStackCommand :



Fonctions membres publiques

- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- virtual [SwitchCurrentStackCommand](#) * [clone](#) () const
Clone la commande.
- const [Memento](#) * [createMemento](#) () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void [restoreFromMemento](#) (const [Memento](#) *memento) const throw (MementoException)
Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Amis

- class [CommandMap](#)

Additional Inherited Members

3.99.1 Description détaillée

Définition à la ligne 15 du fichier SwitchCurrentStackCommand.h.

3.99.2 Documentation des fonctions membres

3.99.2.1 SwitchCurrentStackCommand * calculator : :SwitchCurrentStackCommand : :clone () const [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 26 du fichier SwitchCurrentStackCommand.cpp.

3.99.2.2 const Memento * calculator : :SwitchCurrentStackCommand : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 40 du fichier SwitchCurrentStackCommand.cpp.

3.99.2.3 std : :string calculator : :SwitchCurrentStackCommand : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :StackListCommand](#).

Définition à la ligne 30 du fichier SwitchCurrentStackCommand.cpp.

3.99.2.4 void calculator : :SwitchCurrentStackCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [virtual]

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Le comportement de base de cette méthode est de vider la pile courante pour y remettre celle sauvegardée. On suppose que le [Memento](#) utilisé est un [StackMemento](#), sinon le comportement de cette fonction doit aussi être redéfini. Si ce n'est pas fait, on lève une [MementoException](#).

Paramètres

<i>memento</i>	le Memento contenant les données à restituer.
----------------	---

Implémente [calculator : :StackListCommand](#).

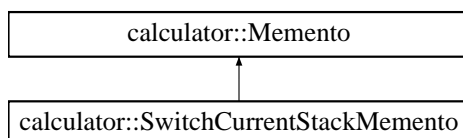
Définition à la ligne 47 du fichier SwitchCurrentStackCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/SwitchCurrentStackCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/stacklist/SwitchCurrentStackCommand.cpp

3.100 Référence de la classe calculator : :SwitchCurrentStackMemento

Graphe d'héritage de calculator : :SwitchCurrentStackMemento :



Fonctions membres publiques

- **SwitchCurrentStackMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const [Integer](#) *integer, int currentMementoIndex)

Amis

- class **SwitchCurrentStackCommand**

Additional Inherited Members

3.100.1 Description détaillée

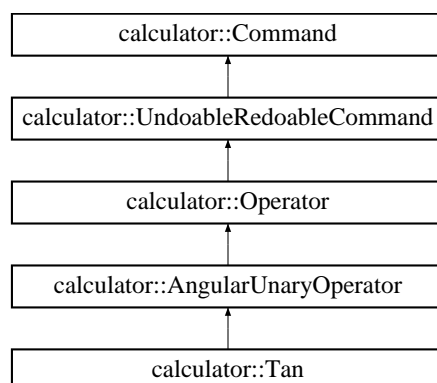
Définition à la ligne 17 du fichier SwitchCurrentStackMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/SwitchCurrentStackMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/SwitchCurrentStackMemento.cpp

3.101 Référence de la classe calculator : :Tan

Graphe d'héritage de calculator : :Tan :



Fonctions membres publiques

- `Tan * clone () const`
Clone l'opérateur.
- `const Number * apply (const SimpleNumber *n) const throw (ArithmeticException)`

Amis

- class **CommandMap**

Additional Inherited Members

3.101.1 Description détaillée

Définition à la ligne 15 du fichier Tan.h.

3.101.2 Documentation des fonctions membres

3.101.2.1 `Tan * calculator : :Tan : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :AngularUnaryOperator](#).

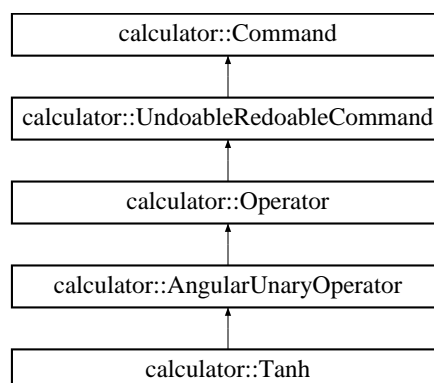
Définition à la ligne 25 du fichier Tan.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Tan.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Tan.cpp

3.102 Référence de la classe calculator : :Tanh

Grappe d'héritage de calculator : :Tanh :



Fonctions membres publiques

- `Tanh * clone ()` const
Clone l'opérateur.
- `const Number * apply (const SimpleNumber *n)` const throw (ArithmeticException)

Amis

- class **CommandMap**

Additional Inherited Members

3.102.1 Description détaillée

Définition à la ligne 15 du fichier Tanh.h.

3.102.2 Documentation des fonctions membres

3.102.2.1 `Tanh * calculator : :Tanh : :clone ()` const [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente `calculator : :AngularUnaryOperator`.

Définition à la ligne 25 du fichier Tanh.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Tanh.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/operator/unary/Tanh.cpp

3.103 Référence de la structure calculator : :toupper

Foncteur permettant de transformer une chaîne de caractère en majuscules.

```
#include <toupper.h>
```

Fonctions membres publiques

- `char operator() (char c)` const
Converti un caractère en majuscule.

3.103.1 Description détaillée

Foncteur permettant de transformer une chaîne de caractère en majuscules.

Définition à la ligne 20 du fichier toupper.h.

3.103.2 Documentation des fonctions membres

3.103.2.1 char calculator : toupper : operator() (char c) const [inline]

Converti un caractère en majuscule.

Paramètres

/e	caractère à transformer en majuscule.
----	---------------------------------------

Renvoie

ce caractère en majuscule

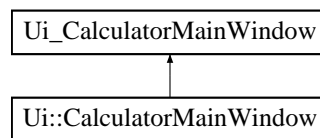
Définition à la ligne 27 du fichier toupper.h.

La documentation de cette structure a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/tool/toupper.h

3.104 Référence de la classe Ui_CalculatorMainWindow

Graphe d'héritage de Ui_CalculatorMainWindow :



Fonctions membres publiques

- void **setupUi** (QMainWindow *CalculatorMainWindow)
- void **retranslateUi** (QMainWindow *CalculatorMainWindow)

Attributs publics

- QAction * **menuItemExit**
- QAction * **menuItemUndo**
- QAction * **menuItemRedo**
- QAction * **menuItemParameters**
- QAction * **menuItemHelp**
- QAction * **menuItemAbout**
- QAction * **menuItemShowKeyboard**
- QAction * **menuItemHideKeyboard**
- QAction * **menuItemNewStack**
- QAction * **menuItemDeleteStack**
- QAction * **menuItemDupStack**
- QAction * **menuItemSave**
- QWidget * **centralWidget**
- QWidget * **verticalLayoutWidget**
- QVBoxLayout * **mainLayout**
- QWidget * **upperWidget**
- QVBoxLayout * **upperDisplay**
- QTabWidget * **stackTabContainer**
- QHBoxLayout * **inputOutputBox**
- QLineEdit * **inputLine**
- QPushButton * **run**
- QWidget * **keyboardDisplay**
- QVBoxLayout * **verticalLayout**
- QGridLayout * **keyboardGridLayout**
- QPushButton * **numeral3**
- QPushButton * **drop**
- QPushButton * **numeral2**
- QPushButton * **sub**

- QPushButton * **mul**
- QPushButton * **div**
- QPushButton * **numeral7**
- QPushButton * **numeral4**
- QPushButton * **numeral8**
- QPushButton * **numeral5**
- QPushButton * **numeral9**
- QPushButton * **numeral6**
- QPushButton * **numeral1**
- QPushButton * **add**
- QPushButton * **clear**
- QPushButton * **swap**
- QPushButton * **dot**
- QPushButton * **dup**
- QPushButton * **backSpace**
- QPushButton * **space**
- QPushButton * **expression**
- QPushButton * **eval**
- QPushButton * **raz**
- QPushButton * **fact**
- QPushButton * **inv**
- QHBoxLayout * **modeBoxLayout**
- QGroupBox * **constantTypeRadioButtonLayout**
- QHBoxLayout * **constantTypeRadioButtonsLayout**
- QRadioButton * **integerModeRadioButton**
- QRadioButton * **rationnalModeRadioButton**
- QRadioButton * **realModeRadioButton**
- QFrame * **line_2**
- QGroupBox * **angleUnitRadioButtonLayout**
- QHBoxLayout * **angleUnitRadioButtonsLayout**
- QRadioButton * **radianModeRadioButton**
- QRadioButton * **degreeModeRadioButton**
- QFrame * **line**
- QHBoxLayout * **complexModeCheckboxButtonLayout**
- QCheckBox * **complexModeCheckButton**
- QPushButton * **cos**
- QPushButton * **sin**
- QPushButton * **tanh**
- QPushButton * **tan**
- QPushButton * **cosh**
- QPushButton * **sinh**
- QPushButton * **cube**
- QPushButton * **mean**
- QPushButton * **pow**
- QPushButton * **userCommand**
- QPushButton * **sign**
- QPushButton * **ln**
- QPushButton * **dollar**
- QPushButton * **sum**
- QPushButton * **log**
- QPushButton * **mod**
- QPushButton * **sqr**
- QPushButton * **sqrt**
- QPushButton * **numeral0**
- QFrame * **line_3**
- QMenuBar * **menuBar**
- QMenu * **menuFile**
- QMenu * **menuStack**
- QMenu * **menuEdit**
- QMenu * **menuOption**
- QMenu * **menuAide**
- QToolBar * **mainToolBar**
- QStatusBar * **statusBar**

3.104.1 Description détaillée

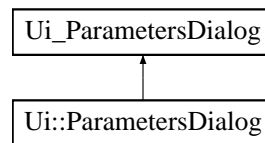
Définition à la ligne 37 du fichier ui_CalculatorMainWindow.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/ui_CalculatorMainWindow.h

3.105 Référence de la classe Ui_ParametersDialog

Graphe d'héritage de Ui_ParametersDialog :



Fonctions membres publiques

- void **setupUi** (QDialog *ParametersDialog)
- void **retranslateUi** (QDialog *ParametersDialog)

Attributs publics

- QWidget * **gridLayoutWidget**
- QGridLayout * **gridLayout**
- QFormLayout * **formLayout**
- QLabel * **visibleStackSizeLabel**
- QLabel * **constantTypeLabel**
- QComboBox * **constantTypeComboBox**
- QLabel * **angleUnitLabel**
- QComboBox * **angleUnitComboBox**
- QLabel * **complexModeLabel**
- QCheckBox * **complexCheckBox**
- QLabel * **displayKeyboardLabel**
- QCheckBox * **displayKeyboardParameter**
- QLabel * **integerDivisionLabel**
- QCheckBox * **integerDivisionCheckBox**
- QLabel * **instantComputeLabel**
- QCheckBox * **instantComputeCheckBox**
- QSpinBox * **visibleStackSizeSpinBox**
- QLabel * **saveOnExitLabel**
- QCheckBox * **saveOnExitCheckBox**
- QHBoxLayout * **cancelSaveButtons**
- QPushButton * **cancelButton**
- QPushButton * **saveButton**

3.105.1 Description détaillée

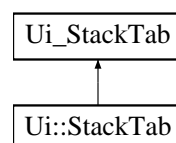
Définition à la ligne 31 du fichier ui_ParametersDialog.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/ui_ParametersDialog.h

3.106 Référence de la classe Ui_StackTab

Graphe d'héritage de Ui_StackTab :



Fonctions membres publiques

- void **setupUi** (QWidget *StackTab)
- void **retranslateUi** (QWidget *StackTab)

3.106.1 Description détaillée

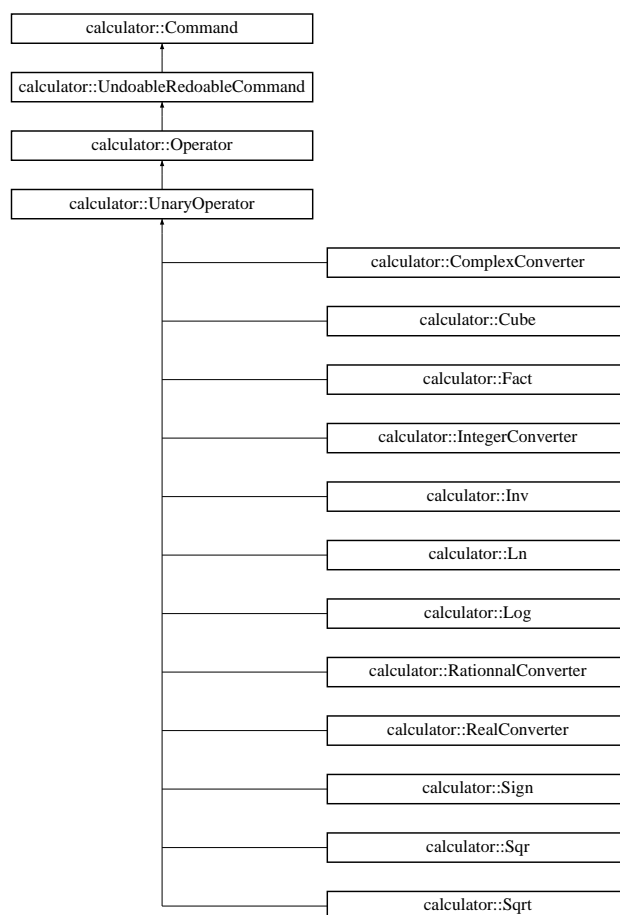
Définition à la ligne 22 du fichier ui_StackTab.h.

La documentation de cette classe a été générée à partir du fichier suivant :

- F :/NetBeansProjects/LO21/Calculatrice/ui_StackTab.h

3.107 Référence de la classe calculator : :UnaryOperator

Graphe d'héritage de calculator : :UnaryOperator :



Fonctions membres publiques

- virtual std : :string **isExecutable** () const
Vérifie si la commande est exécutable.
- virtual **UnaryOperator** * **clone** () const =0
Clone l'opérateur.
- const **Memento** * **createMemento** () const throw (CommandException)
*Crée un **Memento** propre à chaque **Command** contenant une copie des données qui vont être modifiées.*

Fonctions membres protégées

- virtual const [Number](#) * **apply** (const [Number](#) *number) const =0 throw (ArithmeticException)
- **UnaryOperator** (const std : :string name)

3.107.1 Description détaillée

Définition à la ligne 16 du fichier UnaryOperator.h.

3.107.2 Documentation des fonctions membres

3.107.2.1 virtual **UnaryOperator*** calculator : :UnaryOperator : :clone () const [pure virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de l'opérateur

Implémente [calculator : :Operator](#).

Implémenté dans [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationnalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), et [calculator : :Inv](#).

3.107.2.2 const **Memento** * calculator : :UnaryOperator : :createMemento () const throw (CommandException) [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :Operator](#).

Définition à la ligne 33 du fichier UnaryOperator.cpp.

3.107.2.3 std : :string calculator : :UnaryOperator : :isExecutable () const [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une std : :string vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas.

Implémente [calculator : :Operator](#).

Réimplémentée dans [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationnalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), et [calculator : :Sqrt](#).

3.108.2 Documentation des constructeurs et destructeur

3.108.2.1 calculator : :UndoableRedoableCommand : :UndoableRedoableCommand (const std : :string *name*)
[protected]

Constructeur de [UndoableRedoableCommand](#).

Chaque commande est défini par son nom.

Voir également

[Command](#)

Paramètres

<i>name</i>	le nom de la commande.
-------------	------------------------

Définition à la ligne 13 du fichier UndoableRedoableCommand.cpp.

3.108.3 Documentation des fonctions membres

3.108.3.1 virtual UndoableRedoableCommand* calculator : :UndoableRedoableCommand : :clone () const [pure virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande.

Implémente [calculator : :Command](#).

Implémenté dans [calculator : :Expression](#), [calculator : :Complex](#), [calculator : :Operator](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :Constant](#), [calculator : :Literal](#), [calculator : :Integer](#), [calculator : :Real](#), [calculator : :Rationnal](#), [calculator : :Eval](#), [calculator : :Div](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :Swap](#), [calculator : :AngularUnaryOperator](#), [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationnalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), [calculator : :AllParametersCommand](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :Add](#), [calculator : :Mul](#), [calculator : :Sub](#), [calculator : :Cos](#), [calculator : :Cosh](#), [calculator : :Inv](#), [calculator : :Sin](#), [calculator : :Sinh](#), [calculator : :Tan](#), [calculator : :Tanh](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :SimpleNumber](#), [calculator : :NaryOperator](#), [calculator : :Number](#), [calculator : :BinaryOperator](#), [calculator : :Mean](#), [calculator : :Sum](#), [calculator : :Dup](#), [calculator : :UnaryOperator](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), et [calculator : :NullaryOperator](#).

3.108.3.2 virtual const Memento* calculator : :UndoableRedoableCommand : :createMemento () const throw (CommandException) [pure virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémenté dans [calculator : :StackListCommand](#), [calculator : :ParametersCommand](#), [calculator : :Operator](#), [calculator : :Eval](#), [calculator : :AllParametersCommand](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :Swap](#), [calculator : :AngularUnaryOperator](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :NaryOperator](#), [calculator : :BinaryOperator](#), [calculator : :NullaryOperator](#), [calculator : :Dup](#), et [calculator : :UnaryOperator](#).

3.108.3.3 `virtual void calculator : :UndoableRedoableCommand : :execute () const throw (CommandException) [pure virtual]`

Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, la méthode appelée est précisée par les filles.

Implémente [calculator : :Command](#).

Implémenté dans [calculator : :Operator](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :Literal](#), [calculator : :Constant](#), [calculator : :Eval](#), et [calculator : :VisibleStackSizeParameterCommand](#).

3.108.3.4 `virtual std : :string calculator : :UndoableRedoableCommand : :isExecutable () const [pure virtual]`

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas, ou dans le cas d'une [Expression](#) renvoie le restant non exécuté de l'expression.

Implémente [calculator : :Command](#).

Implémenté dans [calculator : :Complex](#), [calculator : :Operator](#), [calculator : :Eval](#), [calculator : :Literal](#), [calculator : :ParametersCommand](#), [calculator : :StackListCommand](#), [calculator : :AngularUnaryOperator](#), [calculator : :ComplexConverter](#), [calculator : :Cube](#), [calculator : :Fact](#), [calculator : :IntegerConverter](#), [calculator : :Ln](#), [calculator : :Log](#), [calculator : :RationalConverter](#), [calculator : :RealConverter](#), [calculator : :Sign](#), [calculator : :Sqr](#), [calculator : :Sqrt](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :Mod](#), [calculator : :Pow](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :NaryOperator](#), [calculator : :Swap](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :BinaryOperator](#), [calculator : :Dup](#), [calculator : :UnaryOperator](#), et [calculator : :NullaryOperator](#).

3.108.3.5 `void calculator : :UndoableRedoableCommand : :redo () const throw (CommandException)`

Rejoue la commande mémorisée et déplace l'itérateur.

de [MementoCaretaker](#) vers le [Memento](#) suivant.

Définition à la ligne 32 du fichier `UndoableRedoableCommand.cpp`.

3.108.3.6 virtual void calculator : :UndoableRedoableCommand : :restoreFromMemento (const Memento * memento) const throw (MementoException) [pure virtual]

Restaure le [Context](#) dans l'état précédent l'exécution de la commande.

Paramètres

<i>memento</i>	le Memento contenant les données à restituer
----------------	--

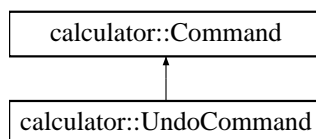
Implémenté dans [calculator : :StackListCommand](#), [calculator : :Operator](#), [calculator : :ParametersCommand](#), [calculator : :Eval](#), [calculator : :AllParametersCommand](#), [calculator : :Clear](#), [calculator : :Drop](#), [calculator : :DuplicateStackCommand](#), [calculator : :NewStackCommand](#), [calculator : :RemoveStackCommand](#), [calculator : :SwitchCurrentStackCommand](#), [calculator : :ComplexParameterCommand](#), [calculator : :DegreeParameterCommand](#), [calculator : :InstantComputeParametersCommand](#), [calculator : :IntegerDivisionParameterCommand](#), [calculator : :IntegerParameterCommand](#), [calculator : :KeyboardParameterCommand](#), [calculator : :RadianParameterCommand](#), [calculator : :RationnalParameterCommand](#), [calculator : :RealParameterCommand](#), [calculator : :SaveOnExitCommand](#), [calculator : :VisibleStackSizeParameterCommand](#), [calculator : :NullaryOperator](#), et [calculator : :Dup](#).

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/UndoableRedoableCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/UndoableRedoableCommand.cpp

3.109 Référence de la classe calculator : :UndoCommand

Graphe d'héritage de calculator : :UndoCommand :



Fonctions membres publiques

- virtual [UndoCommand](#) * [clone](#) () const
Clone la commande.
- std : :string [isExecutable](#) () const
Vérifie si la commande est exécutable.
- void [execute](#) () const throw (CommandException)
Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Amis

- class **CommandMap**

Additional Inherited Members

3.109.1 Description détaillée

Définition à la ligne 15 du fichier UndoCommand.h.

3.109.2 Documentation des fonctions membres

3.109.2.1 `UndoCommand * calculator : :UndoCommand : :clone () const` [virtual]

Clone la commande.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :Command](#).

Définition à la ligne 23 du fichier UndoCommand.cpp.

3.109.2.2 `void calculator : :UndoCommand : :execute () const throw (CommandException)` [virtual]

Méthode virtuelle pure d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, la méthode appelée est précisée par les filles.

Implémente [calculator : :Command](#).

Définition à la ligne 33 du fichier UndoCommand.cpp.

3.109.2.3 `std : :string calculator : :UndoCommand : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Méthode virtuelle pure appelée avant [execute\(\)](#) afin de vérifier les paramètres et l'état des piles.

Voir également

[Parameters](#), [StackList](#), [Stack](#)

Renvoie

une `std : :string` vide si la commande est exécutable, sinon renvoie soit la raison de l'erreur dans la plupart des cas, ou dans le cas d'une [Expression](#) renvoie le restant non exécuté de l'expression.

Implémente [calculator : :Command](#).

Définition à la ligne 27 du fichier UndoCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/UndoCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/UndoCommand.cpp

3.110 Référence de la classe `calculator : :ViewException`

Classe d'exception pour la couche View.

```
#include <ViewException.h>
```

Fonctions membres publiques

- **ViewException** (const std : :string &i)
- const char * **what** () const throw ()

3.110.1 Description détaillée

Classe d'exception pour la couche View.

Exception lancée si la vue se retrouve dans un état inconsistant.

Paramètres

e	la cause de l'exception.
---	--------------------------

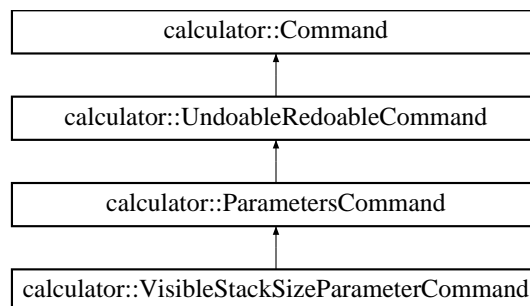
Définition à la ligne 21 du fichier ViewException.h.

La documentation de cette classe a été générée à partir du fichier suivant :

– F :/NetBeansProjects/LO21/Calculatrice/exception/ViewException.h

3.111 Référence de la classe calculator : :VisibleStackSizeParameterCommand

Graphe d'héritage de calculator : :VisibleStackSizeParameterCommand :



Fonctions membres publiques

- std : :string **isExecutable** () const
Vérifie si la commande est exécutable.
- void **execute** () const throw (CommandException)
Méthode d'entrée du Design Pattern [Command](#).
- virtual
[VisibleStackSizeParameterCommand](#) * **clone** () const
Clone l'opérateur.
- const [Memento](#) * **createMemento** () const throw (CommandException)
Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.
- void **restoreFromMemento** (const [Memento](#) *memento) const throw (MementoException)
Restaure les paramètre dans leur état précédent.

Fonctions membres protégées

- **VisibleStackSizeParameterCommand** (const std : :string name)

Amis

- class **CommandMap**

3.111.1 Description détaillée

Définition à la ligne 15 du fichier VisibleStackSizeParameterCommand.h.

3.111.2 Documentation des fonctions membres

3.111.2.1 `VisibleStackSizeParameterCommand * calculator : :VisibleStackSizeParameterCommand : :clone () const` [virtual]

Clone l'opérateur.

On ne peut pas créer de commandes (sauf des constantes) directement, c'est le rôle du [CommandManager](#), on a cependant besoin de les copier pour les sauvegardes et la fonction UNDO/REDO.

Renvoie

une copie de la commande

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 24 du fichier `VisibleStackSizeParameterCommand.cpp`.

3.111.2.2 `const Memento * calculator : :VisibleStackSizeParameterCommand : :createMemento () const throw (CommandException)` [virtual]

Crée un [Memento](#) propre à chaque [Command](#) contenant une copie des données qui vont être modifiées.

Renvoie

un [Memento](#) spécialisé.

Implémente [calculator : :ParametersCommand](#).

Définition à la ligne 57 du fichier `VisibleStackSizeParameterCommand.cpp`.

3.111.2.3 `void calculator : :VisibleStackSizeParameterCommand : :execute () const throw (CommandException)` [virtual]

Méthode d'entrée du Design Pattern [Command](#).

Utilisation avec le Design Pattern Template Method, appelle la méthode `apply(Parameters* parameters)`, une [ParametersCommand](#) ayant pour but de modifier un ou plusieurs paramètres

Réimplémentée à partir de [calculator : :ParametersCommand](#).

Définition à la ligne 35 du fichier `VisibleStackSizeParameterCommand.cpp`.

3.111.2.4 `std : :string calculator : :VisibleStackSizeParameterCommand : :isExecutable () const` [virtual]

Vérifie si la commande est exécutable.

Une commande modifiant les paramètres est toujours exécutable Si un paramètre d'entrée est incorrect, il sera réglé sur les limites acceptables

Voir également

[Parameters](#)

Renvoie

`std : :string("")` (toujours exécutable)

Voir également

[Command](#)

Réimplémentée à partir de [calculator : :ParametersCommand](#).

Définition à la ligne 28 du fichier `VisibleStackSizeParameterCommand.cpp`.

3.111.2.5 void calculator : :VisibleStackSizeParameterCommand : :restoreFromMemento (const Memento * memento) const
throw (MementoException) [virtual]

Restaure les paramètre dans leur état précédent.

Paramètres

<i>memento</i>	le Memento contenant les paramètres à restituer.
----------------	--

Implémente [calculator : :ParametersCommand](#).

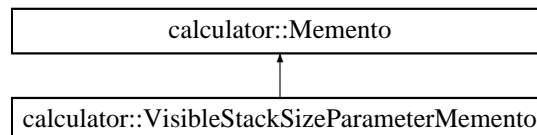
Définition à la ligne 64 du fichier VisibleStackSizeParameterCommand.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/VisibleStackSizeParameterCommand.h
- F :/NetBeansProjects/LO21/Calculatrice/model/command/parameters/VisibleStackSizeParameterCommand.cpp

3.112 Référence de la classe calculator : :VisibleStackSizeParameterMemento

Graphe d'héritage de calculator : :VisibleStackSizeParameterMemento :



Fonctions membres publiques

- **VisibleStackSizeParameterMemento** ([UndoableRedoableCommand](#) *undoableRedoableCommand, const [Integer](#) *integer, int visibleStackSize)

Amis

- class **VisibleStackSizeParameterCommand**

Additional Inherited Members

3.112.1 Description détaillée

Définition à la ligne 17 du fichier VisibleStackSizeParameterMemento.h.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/model/memento/VisibleStackSizeParameterMemento.h
- F :/NetBeansProjects/LO21/Calculatrice/model/memento/VisibleStackSizeParameterMemento.cpp

3.113 Référence de la classe calculator : :XMLParser

Parseur XML/CalculatriceNPI.

```
#include <XMLParser.h>
```

Fonctions membres publiques

- void [loadContextFromFile](#) ([Context](#) *context, const std : :string &saveFilename)

- Méthode de chargement du [Context](#) à partir d'un fichier XML.
- void [saveContextToFile](#) (const [Context](#) *const context, const std : :string &saveFilename)
Méthode d'enregistrement du [Context](#) vers un fichier XML.

Fonctions membres publiques statiques

- static [XMLParser](#) * [getInstance](#) ()
Singleton [XMLParser](#).
- static void [deleteInstance](#) ()
Singleton [XMLParser](#).

3.113.1 Description détaillée

Parseur XML/CalculatriceNPI.

Design Pattern Singleton.

Définition à la ligne 29 du fichier XMLParser.h.

3.113.2 Documentation des fonctions membres

3.113.2.1 void calculator : :XMLParser : :deleteInstance () [static]

Singleton [XMLParser](#).

Détruit l'unique instance de [XMLParser](#).

Définition à la ligne 23 du fichier XMLParser.cpp.

3.113.2.2 XMLParser * calculator : :XMLParser : :getInstance () [static]

Singleton [XMLParser](#).

Renvoie

crée et/ou renvoie l'unique instance de [XMLParser](#).

Définition à la ligne 18 du fichier XMLParser.cpp.

3.113.2.3 void calculator : :XMLParser : :loadContextFromFile (Context * context, const std : :string & saveFilename)

Méthode de chargement du [Context](#) à partir d'un fichier XML.

Paramètres

<i>context</i>	le context à modifier.
<i>saveFilename</i>	le nom du fichier à charger.

Définition à la ligne 33 du fichier XMLParser.cpp.

3.113.2.4 void calculator : :XMLParser : :saveContextToFile (const Context *const context, const std : :string & saveFilename)

Méthode d'enregistrement du [Context](#) vers un fichier XML.

Paramètres

<i>context</i>	le context à sauvegarder.
<i>saveFilename</i>	le nom du fichier vers lequel sauvegarder.

Définition à la ligne 109 du fichier XMLParser.cpp.

La documentation de cette classe a été générée à partir des fichiers suivants :

- F :/NetBeansProjects/LO21/Calculatrice/tool/XMLParser.h
- F :/NetBeansProjects/LO21/Calculatrice/tool/XMLParser.cpp

Index

- ~CalculatorParser
 - calculator : :CalculatorParser, 20
- ~CommandManager
 - calculator : :CommandManager, 27
- ~Engine
 - calculator : :Engine, 52
- ~Expression
 - calculator : :Expression, 58
- ~LogMessage
 - calculator : :LogMessage, 78
- ~Stack
 - calculator : :Stack, 130
- ~StackTabList
 - calculator : :StackTabList, 140
- ~StackTabWidget
 - calculator : :StackTabWidget, 142
- addObserver
 - calculator : :Observable, 93
- AngleUnit
 - calculator : :Parameters, 98
- append
 - calculator : :Expression, 59
- apply
 - calculator : :AllParametersCommand, 11
 - calculator : :ComplexParameterCommand, 33
 - calculator : :DegreeParameterCommand, 43
 - calculator : :InstantComputeParametersCommand, 62
 - calculator : :IntegerDivisionParameterCommand, 66
 - calculator : :IntegerParameterCommand, 68
 - calculator : :KeyboardParameterCommand, 70
 - calculator : :ParametersCommand, 101
 - calculator : :RadianParameterCommand, 107
 - calculator : :RationalParameterCommand, 111
 - calculator : :RealParameterCommand, 116
 - calculator : :SaveOnExitCommand, 121
 - calculator : :StackListCommand, 137
- calculator : :Parameters
 - DEGREE, 98
 - INTEGER, 98
 - RADIAN, 98
 - RATIONAL, 98
 - REAL, 98
- calculator : :Add, 9
 - clone, 9
- calculator : :AllParametersCommand, 10
 - apply, 11
 - clone, 11
 - createMemento, 11
 - restoreFromMemento, 11
- calculator : :AngleUnitParameterMemento, 12
- calculator : :AngularUnaryOperator, 12
 - clone, 13
 - createMemento, 13
 - isExecutable, 13
- calculator : :ArithmeticException, 14
- calculator : :BinaryOperator, 14
 - clone, 15
 - createMemento, 15
 - isExecutable, 15
- calculator : :BooleanParameterMemento, 16
- calculator : :CalculatorMainWindow, 17
 - CalculatorMainWindow, 18
 - getEngine, 18
 - init, 18
 - prepareForShutdown, 18
 - setEngine, 19
- calculator : :CalculatorParser, 19
 - ~CalculatorParser, 20
 - CalculatorParser, 20
 - isCommand, 20
 - isComplex, 20
 - isExpression, 20
 - isInteger, 21
 - isRational, 21
 - isReal, 21
 - isSimpleNumber, 21
 - parse, 22
- calculator : :Clear, 22
 - clone, 23
 - createMemento, 23
 - isExecutable, 23
 - restoreFromMemento, 23
- calculator : :Command, 24
 - clone, 25
 - Command, 25
 - execute, 25
 - isExecutable, 25
 - toString, 26
- calculator : :CommandException, 26
- calculator : :CommandManager, 27
 - ~CommandManager, 27
 - CommandManager, 27
 - executeCommand, 28
- calculator : :CommandMap, 28
 - deleteInstance, 28

- getCommand, 29
- getInstance, 29
- isCommandName, 29
- calculator : :Complex, 29
 - clone, 30
 - isExecutable, 30
 - toString, 31
- calculator : :ComplexConverter, 31
 - clone, 32
 - isExecutable, 32
- calculator : :ComplexParameterCommand, 32
 - apply, 33
 - clone, 33
 - createMemento, 33
 - restoreFromMemento, 34
- calculator : :Constant, 34
 - clone, 35
- calculator : :ConstantTypeParameterMemento, 35
- calculator : :Context, 36
 - deleteInstance, 36
 - getInstance, 37
 - saveContextToXML, 37
- calculator : :ContextException, 37
- calculator : :ContextMemento, 38
- calculator : :Cos, 38
 - clone, 39
- calculator : :Cosh, 39
 - clone, 40
- calculator : :Cube, 40
 - clone, 41
 - isExecutable, 41
- calculator : :CurrentStackMemento, 42
- calculator : :DegreeParameterCommand, 42
 - apply, 43
 - clone, 43
 - createMemento, 43
 - restoreFromMemento, 43
- calculator : :Div, 44
 - clone, 44
- calculator : :Drop, 45
 - clone, 45
 - createMemento, 46
 - isExecutable, 46
 - restoreFromMemento, 46
- calculator : :Dup, 47
 - clone, 47
 - createMemento, 47
 - isExecutable, 48
 - restoreFromMemento, 48
- calculator : :DuplicateStackCommand, 48
 - clone, 49
 - createMemento, 49
 - isExecutable, 50
 - restoreFromMemento, 50
- calculator : :EmptyMemento, 50
- calculator : :Engine, 51
 - ~Engine, 52
 - drop, 52
 - dupStack, 52
 - Engine, 52
 - getContext, 52
 - newStack, 52
 - redo, 53
 - removeCurrentStack, 53
 - run, 53
 - saveAllParameters, 53
 - setAngleUnitTo, 53
 - setConstantTypeTo, 54
 - setCurrentStack, 54
 - setInputText, 54
 - undo, 54
- calculator : :Eval, 55
 - clone, 55
 - createMemento, 56
 - execute, 56
 - isExecutable, 56
 - restoreFromMemento, 56
- calculator : :Expression, 57
 - ~Expression, 58
 - append, 59
 - clone, 59
 - empty, 59
 - Expression, 58
 - getFirstCommand, 59
 - toString, 59
- calculator : :Fact, 60
 - clone, 60
 - isExecutable, 61
- calculator : :InstantComputeParametersCommand, 61
 - apply, 62
 - clone, 62
 - createMemento, 62
 - restoreFromMemento, 62
- calculator : :Integer, 63
 - clone, 64
 - toString, 64
- calculator : :IntegerConverter, 64
 - clone, 65
 - isExecutable, 65
- calculator : :IntegerDivisionParameterCommand, 65
 - apply, 66
 - clone, 66
 - createMemento, 67
 - restoreFromMemento, 67
- calculator : :IntegerParameterCommand, 67
 - apply, 68
 - clone, 68
 - createMemento, 68
 - restoreFromMemento, 68
- calculator : :Inv, 69
 - clone, 69
- calculator : :KeyboardParameterCommand, 70
 - apply, 70
 - clone, 71
 - createMemento, 71
 - restoreFromMemento, 71

- calculator : :Literal, 71
 - clone, 72
 - execute, 73
 - isExecutable, 73
 - Literal, 72
 - toString, 73
- calculator : :Ln, 73
 - clone, 74
 - isExecutable, 74
- calculator : :Log, 75
 - clone, 75
 - isExecutable, 76
- calculator : :LogMessage, 78
 - ~LogMessage, 78
 - getClassname, 79
 - getLevel, 79
 - getMessage, 79
 - LogMessage, 78
- calculator : :LogSystem, 79
 - log, 80
- calculator : :Logger, 76
- calculator : :LoggerManager, 77
 - deleteInstance, 77
 - getInstance, 77
 - getLogger, 77
- calculator : :Mean, 80
 - clone, 81
- calculator : :Memento, 81
 - getInputString, 83
 - getUndoableRedoableCommand, 83
 - Memento, 83
- calculator : :MementoCaretaker, 84
- calculator : :MementoException, 84
- calculator : :Mod, 84
 - clone, 85
 - isExecutable, 85
- calculator : :Mul, 86
 - clone, 86
- calculator : :NaryOperator, 87
 - clone, 87
 - createMemento, 88
 - isExecutable, 88
- calculator : :NewStackCommand, 88
 - clone, 89
 - createMemento, 89
 - isExecutable, 89
 - restoreFromMemento, 90
- calculator : :NullaryOperator, 90
 - clone, 91
 - createMemento, 91
 - isExecutable, 91
 - restoreFromMemento, 91
- calculator : :Number, 92
 - clone, 92
- calculator : :Observable, 93
 - addObserver, 93
 - removeObserver, 94
- calculator : :Observer, 94
- calculator : :Operator, 94
 - clone, 95
 - createMemento, 96
 - execute, 96
 - isExecutable, 96
 - Operator, 95
 - restoreFromMemento, 97
- calculator : :Parameters, 97
 - AngleUnit, 98
 - clone, 99
 - ConstantType, 98
 - deleteInstance, 99
 - getInstance, 99
 - Parameters, 98
- calculator : :ParametersCommand, 99
 - apply, 101
 - clone, 101
 - createMemento, 101
 - execute, 102
 - isExecutable, 102
 - ParametersCommand, 100
 - restoreFromMemento, 102
- calculator : :ParametersDialog, 103
 - ParametersDialog, 103
- calculator : :ParametersMemento, 104
- calculator : :ParseException, 105
- calculator : :Pow, 105
 - clone, 106
 - isExecutable, 106
- calculator : :RadianParameterCommand, 106
 - apply, 107
 - clone, 107
 - createMemento, 107
 - restoreFromMemento, 108
- calculator : :Rationnal, 108
 - clone, 109
 - toString, 109
- calculator : :RationnalConverter, 109
 - clone, 110
 - isExecutable, 110
- calculator : :RationnalParameterCommand, 111
 - apply, 111
 - clone, 112
 - createMemento, 112
 - restoreFromMemento, 112
- calculator : :Real, 112
 - clone, 113
 - toString, 113
- calculator : :RealConverter, 114
 - clone, 114
 - isExecutable, 115
- calculator : :RealParameterCommand, 115
 - apply, 116
 - clone, 116
 - createMemento, 116
 - restoreFromMemento, 116
- calculator : :RedoCommand, 117
 - clone, 117

- execute, 117
- isExecutable, 118
- calculator : :RemoveStackCommand, 118
 - clone, 119
 - createMemento, 119
 - isExecutable, 119
 - restoreFromMemento, 119
- calculator : :RemoveStackMemento, 120
- calculator : :SaveOnExitCommand, 120
 - apply, 121
 - clone, 121
 - createMemento, 122
 - restoreFromMemento, 122
- calculator : :Sign, 122
 - clone, 123
 - isExecutable, 123
- calculator : :SimpleNumber, 123
 - clone, 124
 - operator double, 124
 - toString, 124
- calculator : :Sin, 125
 - clone, 125
- calculator : :Sinh, 126
 - clone, 126
- calculator : :Sqr, 127
 - clone, 127
 - isExecutable, 128
- calculator : :Sqrt, 128
 - clone, 129
 - isExecutable, 129
- calculator : :Stack, 129
 - ~Stack, 130
 - clone, 131
 - empty, 131
 - getConstant, 131
 - push, 131
 - size, 131
 - Stack, 130
 - swapAnyConstant, 131
 - top, 132
 - XMLParser, 132
- calculator : :StackList, 132
 - clone, 133
 - deleteInstance, 133
 - duplicateCurrentStack, 133
 - empty, 133
 - getCurrentStack, 134
 - getCurrentStackIndex, 134
 - getInstance, 134
 - getStack, 134
 - insertStackAtIndex, 134
 - newStack, 135
 - removeCurrentStack, 135
 - setCurrentStack, 135
 - size, 135
 - XMLParser, 135
- calculator : :StackListCommand, 136
 - apply, 137
 - clone, 137
 - createMemento, 137
 - execute, 137
 - isExecutable, 137
 - restoreFromMemento, 138
 - StackListCommand, 136
- calculator : :StackMemento, 138
- calculator : :StackTabList, 139
 - ~StackTabList, 140
 - duplicateCurrentTab, 140
 - getCurrentTab, 140
 - getCurrentTabIndex, 140
 - getTab, 140
 - newTab, 141
 - removeCurrentTab, 141
 - setCurrentTab, 141
 - size, 141
 - StackTabList, 140
- calculator : :StackTabWidget, 141
 - ~StackTabWidget, 142
 - StackTabWidget, 142
- calculator : :Sub, 142
 - clone, 143
- calculator : :Sum, 143
 - clone, 144
- calculator : :Swap, 144
 - clone, 145
 - createMemento, 145
 - isExecutable, 145
- calculator : :SwitchCurrentStackCommand, 146
 - clone, 147
 - createMemento, 147
 - isExecutable, 147
 - restoreFromMemento, 147
- calculator : :SwitchCurrentStackMemento, 148
- calculator : :Tan, 148
 - clone, 149
- calculator : :Tanh, 149
 - clone, 150
- calculator : :UnaryOperator, 154
 - clone, 155
 - createMemento, 155
 - isExecutable, 155
- calculator : :UndoCommand, 159
 - clone, 159
 - execute, 160
 - isExecutable, 160
- calculator : :UndoableRedoableCommand, 156
 - clone, 157
 - createMemento, 157
 - execute, 158
 - isExecutable, 158
 - redo, 158
 - restoreFromMemento, 158
 - UndoableRedoableCommand, 157
- calculator : :ViewException, 160
- calculator : :VisibleStackSizeParameterCommand, 161
 - clone, 162

- createMemento, 162
- execute, 162
- isExecutable, 162
- restoreFromMemento, 162
- calculator : :VisibleStackSizeParameterMemento, 163
- calculator : :XMLParser, 163
 - deleteInstance, 164
 - getInstance, 164
 - loadContextFromFile, 164
 - saveContextToFile, 164
- calculator : :toupper, 150
- operator(), 151
- CalculatorMainWindow
 - calculator : :CalculatorMainWindow, 18
- CalculatorParser
 - calculator : :CalculatorParser, 20
- clone
 - calculator : :Add, 9
 - calculator : :AllParametersCommand, 11
 - calculator : :AngularUnaryOperator, 13
 - calculator : :BinaryOperator, 15
 - calculator : :Clear, 23
 - calculator : :Command, 25
 - calculator : :Complex, 30
 - calculator : :ComplexConverter, 32
 - calculator : :ComplexParameterCommand, 33
 - calculator : :Constant, 35
 - calculator : :Cos, 39
 - calculator : :Cosh, 40
 - calculator : :Cube, 41
 - calculator : :DegreeParameterCommand, 43
 - calculator : :Div, 44
 - calculator : :Drop, 45
 - calculator : :Dup, 47
 - calculator : :DuplicateStackCommand, 49
 - calculator : :Eval, 55
 - calculator : :Expression, 59
 - calculator : :Fact, 60
 - calculator : :InstantComputeParametersCommand, 62
 - calculator : :Integer, 64
 - calculator : :IntegerConverter, 65
 - calculator : :IntegerDivisionParameterCommand, 66
 - calculator : :IntegerParameterCommand, 68
 - calculator : :Inv, 69
 - calculator : :KeyboardParameterCommand, 71
 - calculator : :Literal, 72
 - calculator : :Ln, 74
 - calculator : :Log, 75
 - calculator : :Mean, 81
 - calculator : :Mod, 85
 - calculator : :Mul, 86
 - calculator : :NaryOperator, 87
 - calculator : :NewStackCommand, 89
 - calculator : :NullaryOperator, 91
 - calculator : :Number, 92
 - calculator : :Operator, 95
 - calculator : :Parameters, 99
 - calculator : :ParametersCommand, 101
 - calculator : :Pow, 106
 - calculator : :RadianParameterCommand, 107
 - calculator : :Rational, 109
 - calculator : :RationalConverter, 110
 - calculator : :RationalParameterCommand, 112
 - calculator : :Real, 113
 - calculator : :RealConverter, 114
 - calculator : :RealParameterCommand, 116
 - calculator : :RedoCommand, 117
 - calculator : :RemoveStackCommand, 119
 - calculator : :SaveOnExitCommand, 121
 - calculator : :Sign, 123
 - calculator : :SimpleNumber, 124
 - calculator : :Sin, 125
 - calculator : :Sinh, 126
 - calculator : :Sqr, 127
 - calculator : :Sqrt, 129
 - calculator : :Stack, 131
 - calculator : :StackList, 133
 - calculator : :StackListCommand, 137
 - calculator : :Sub, 143
 - calculator : :Sum, 144
 - calculator : :Swap, 145
 - calculator : :SwitchCurrentStackCommand, 147
 - calculator : :Tan, 149
 - calculator : :Tanh, 150
 - calculator : :UnaryOperator, 155
 - calculator : :UndoableRedoableCommand, 157
 - calculator : :UndoCommand, 159
 - calculator : :VisibleStackSizeParameterCommand, 162
- Command
 - calculator : :Command, 25
- CommandManager
 - calculator : :CommandManager, 27
- ConstantType
 - calculator : :Parameters, 98
- createMemento
 - calculator : :AllParametersCommand, 11
 - calculator : :AngularUnaryOperator, 13
 - calculator : :BinaryOperator, 15
 - calculator : :Clear, 23
 - calculator : :ComplexParameterCommand, 33
 - calculator : :DegreeParameterCommand, 43
 - calculator : :Drop, 46
 - calculator : :Dup, 47
 - calculator : :DuplicateStackCommand, 49
 - calculator : :Eval, 56
 - calculator : :InstantComputeParametersCommand, 62
 - calculator : :IntegerDivisionParameterCommand, 67
 - calculator : :IntegerParameterCommand, 68
 - calculator : :KeyboardParameterCommand, 71
 - calculator : :NaryOperator, 88
 - calculator : :NewStackCommand, 89

- calculator : :NullaryOperator, 91
- calculator : :Operator, 96
- calculator : :ParametersCommand, 101
- calculator : :RadianParameterCommand, 107
- calculator : :RationalParameterCommand, 112
- calculator : :RealParameterCommand, 116
- calculator : :RemoveStackCommand, 119
- calculator : :SaveOnExitCommand, 122
- calculator : :StackListCommand, 137
- calculator : :Swap, 145
- calculator : :SwitchCurrentStackCommand, 147
- calculator : :UnaryOperator, 155
- calculator : :UndoableRedoableCommand, 157
- calculator : :VisibleStackSizeParameterCommand, 162
- DEGREE
 - calculator : :Parameters, 98
- deleteInstance
 - calculator : :CommandMap, 28
 - calculator : :Context, 36
 - calculator : :LoggerManager, 77
 - calculator : :Parameters, 99
 - calculator : :StackList, 133
 - calculator : :XMLParser, 164
- drop
 - calculator : :Engine, 52
- dupStack
 - calculator : :Engine, 52
- duplicateCurrentStack
 - calculator : :StackList, 133
- duplicateCurrentTab
 - calculator : :StackTabList, 140
- empty
 - calculator : :Expression, 59
 - calculator : :Stack, 131
 - calculator : :StackList, 133
- Engine
 - calculator : :Engine, 52
- execute
 - calculator : :Command, 25
 - calculator : :Eval, 56
 - calculator : :Literal, 73
 - calculator : :Operator, 96
 - calculator : :ParametersCommand, 102
 - calculator : :RedoCommand, 117
 - calculator : :StackListCommand, 137
 - calculator : :UndoableRedoableCommand, 158
 - calculator : :UndoCommand, 160
 - calculator : :VisibleStackSizeParameterCommand, 162
- executeCommand
 - calculator : :CommandManager, 28
- Expression
 - calculator : :Expression, 58
- getClassname
 - calculator : :LogMessage, 79
- getCommand
 - calculator : :CommandMap, 29
- getConstant
 - calculator : :Stack, 131
- getContext
 - calculator : :Engine, 52
- getCurrentStack
 - calculator : :StackList, 134
- getCurrentStackIndex
 - calculator : :StackList, 134
- getCurrentTab
 - calculator : :StackTabList, 140
- getCurrentTabIndex
 - calculator : :StackTabList, 140
- getEngine
 - calculator : :CalculatorMainWindow, 18
- getFirstCommand
 - calculator : :Expression, 59
- getInputString
 - calculator : :Memento, 83
- getInstance
 - calculator : :CommandMap, 29
 - calculator : :Context, 37
 - calculator : :LoggerManager, 77
 - calculator : :Parameters, 99
 - calculator : :StackList, 134
 - calculator : :XMLParser, 164
- getLevel
 - calculator : :LogMessage, 79
- getLogger
 - calculator : :LoggerManager, 77
- getMessage
 - calculator : :LogMessage, 79
- getStack
 - calculator : :StackList, 134
- getTab
 - calculator : :StackTabList, 140
- getUndoableRedoableCommand
 - calculator : :Memento, 83
- INTEGER
 - calculator : :Parameters, 98
- init
 - calculator : :CalculatorMainWindow, 18
- insertStackAtIndex
 - calculator : :StackList, 134
- isCommand
 - calculator : :CalculatorParser, 20
- isCommandName
 - calculator : :CommandMap, 29
- isComplex
 - calculator : :CalculatorParser, 20
- isExecutable
 - calculator : :AngularUnaryOperator, 13
 - calculator : :BinaryOperator, 15
 - calculator : :Clear, 23
 - calculator : :Command, 25
 - calculator : :Complex, 30
 - calculator : :ComplexConverter, 32

- calculator : :Cube, 41
- calculator : :Drop, 46
- calculator : :Dup, 48
- calculator : :DuplicateStackCommand, 50
- calculator : :Eval, 56
- calculator : :Fact, 61
- calculator : :IntegerConverter, 65
- calculator : :Literal, 73
- calculator : :Ln, 74
- calculator : :Log, 76
- calculator : :Mod, 85
- calculator : :NaryOperator, 88
- calculator : :NewStackCommand, 89
- calculator : :NullaryOperator, 91
- calculator : :Operator, 96
- calculator : :ParametersCommand, 102
- calculator : :Pow, 106
- calculator : :RationalConverter, 110
- calculator : :RealConverter, 115
- calculator : :RedoCommand, 118
- calculator : :RemoveStackCommand, 119
- calculator : :Sign, 123
- calculator : :Sqr, 128
- calculator : :Sqrt, 129
- calculator : :StackListCommand, 137
- calculator : :Swap, 145
- calculator : :SwitchCurrentStackCommand, 147
- calculator : :UnaryOperator, 155
- calculator : :UndoableRedoableCommand, 158
- calculator : :UndoCommand, 160
- calculator : :VisibleStackSizeParameterCommand, 162
- isExpression
 - calculator : :CalculatorParser, 20
- isInteger
 - calculator : :CalculatorParser, 21
- isRational
 - calculator : :CalculatorParser, 21
- isReal
 - calculator : :CalculatorParser, 21
- isSimpleNumber
 - calculator : :CalculatorParser, 21
- Literal
 - calculator : :Literal, 72
- loadContextFromFile
 - calculator : :XMLParser, 164
- log
 - calculator : :LogSystem, 80
- LogMessage
 - calculator : :LogMessage, 78
- Memento
 - calculator : :Memento, 83
- newStack
 - calculator : :Engine, 52
 - calculator : :StackList, 135
- newTab
 - calculator : :StackTabList, 141
- Operator
 - calculator : :Operator, 95
- operator double
 - calculator : :SimpleNumber, 124
- operator()
 - calculator : :toupper, 151
- Parameters
 - calculator : :Parameters, 98
- ParametersCommand
 - calculator : :ParametersCommand, 100
- ParametersDialog
 - calculator : :ParametersDialog, 103
- parse
 - calculator : :CalculatorParser, 22
- prepareForShutdown
 - calculator : :CalculatorMainWindow, 18
- push
 - calculator : :Stack, 131
- RADIAN
 - calculator : :Parameters, 98
- RATIONAL
 - calculator : :Parameters, 98
- REAL
 - calculator : :Parameters, 98
- redo
 - calculator : :Engine, 53
 - calculator : :UndoableRedoableCommand, 158
- removeCurrentStack
 - calculator : :Engine, 53
 - calculator : :StackList, 135
- removeCurrentTab
 - calculator : :StackTabList, 141
- removeObserver
 - calculator : :Observable, 94
- restoreFromMemento
 - calculator : :AllParametersCommand, 11
 - calculator : :Clear, 23
 - calculator : :ComplexParameterCommand, 34
 - calculator : :DegreeParameterCommand, 43
 - calculator : :Drop, 46
 - calculator : :Dup, 48
 - calculator : :DuplicateStackCommand, 50
 - calculator : :Eval, 56
 - calculator : :InstantComputeParametersCommand, 62
 - calculator : :IntegerDivisionParameterCommand, 67
 - calculator : :IntegerParameterCommand, 68
 - calculator : :KeyboardParameterCommand, 71
 - calculator : :NewStackCommand, 90
 - calculator : :NullaryOperator, 91
 - calculator : :Operator, 97
 - calculator : :ParametersCommand, 102
 - calculator : :RadianParameterCommand, 108
 - calculator : :RationalParameterCommand, 112

- calculator : :RealParameterCommand, 116
- calculator : :RemoveStackCommand, 119
- calculator : :SaveOnExitCommand, 122
- calculator : :StackListCommand, 138
- calculator : :SwitchCurrentStackCommand, 147
- calculator : :UndoableRedoableCommand, 158
- calculator : :VisibleStackSizeParameterCommand, 162
- run
 - calculator : :Engine, 53
- saveAllParameters
 - calculator : :Engine, 53
- saveContextToFile
 - calculator : :XMLParser, 164
- saveContextToXML
 - calculator : :Context, 37
- setAngleUnitTo
 - calculator : :Engine, 53
- setConstantTypeTo
 - calculator : :Engine, 54
- setCurrentStack
 - calculator : :Engine, 54
 - calculator : :StackList, 135
- setCurrentTab
 - calculator : :StackTabList, 141
- setEngine
 - calculator : :CalculatorMainWindow, 19
- setInputText
 - calculator : :Engine, 54
- size
 - calculator : :Stack, 131
 - calculator : :StackList, 135
 - calculator : :StackTabList, 141
- Stack
 - calculator : :Stack, 130
- StackListCommand
 - calculator : :StackListCommand, 136
- StackTabList
 - calculator : :StackTabList, 140
- StackTabWidget
 - calculator : :StackTabWidget, 142
- swapAnyConstant
 - calculator : :Stack, 131
- toString
 - calculator : :Command, 26
 - calculator : :Complex, 31
 - calculator : :Expression, 59
 - calculator : :Integer, 64
 - calculator : :Literal, 73
 - calculator : :Rational, 109
 - calculator : :Real, 113
 - calculator : :SimpleNumber, 124
- top
 - calculator : :Stack, 132
- Ui : :CalculatorMainWindow, 16
- Ui : :ParametersDialog, 104
- Ui : :StackTab, 139
- Ui_CalculatorMainWindow, 151
- Ui_ParametersDialog, 153
- Ui_StackTab, 153
- undo
 - calculator : :Engine, 54
- UndoableRedoableCommand
 - calculator : :UndoableRedoableCommand, 157
- XMLParser
 - calculator : :Stack, 132
 - calculator : :StackList, 135