

Philophobia

Generated by Doxygen 1.8.6

Mon Apr 7 2014 20:18:47

Contents

1	Main Page	1
2	Namespace Index	3
2.1	Packages	3
3	Hierarchical Index	5
3.1	Class Hierarchy	5
4	Class Index	7
4.1	Class List	7
5	File Index	9
5.1	File List	9
6	Namespace Documentation	11
6.1	Package debug	11
6.2	Package gameplay	11
6.3	Package gameplay.ai	11
6.4	Package gameplay.ai.mood	12
6.5	Package gameplay.ai.phrasing	12
6.6	Package main	12
6.7	Package window	12
6.8	Package window.topbar	12
6.9	Package window.ui	13
6.10	Package world	13
6.11	Package world.character	13
6.12	Package world.scenery	13
7	Class Documentation	15
7.1	gameplay.ai.AI Class Reference	15
7.1.1	Detailed Description	17
7.1.2	Constructor & Destructor Documentation	17
7.1.2.1	AI	17
7.1.3	Member Function Documentation	18

7.1.3.1	changeMood	18
7.1.3.2	newBetrayal	18
7.1.3.3	newSlavery	19
7.1.3.4	setCurrentWorld	20
7.1.4	Member Data Documentation	20
7.1.4.1	betrayalCount	20
7.1.4.2	currentMood	20
7.1.4.3	currentWorld	20
7.1.4.4	phrasingSystem	21
7.1.4.5	slaveryCount	21
7.2	world.scenery.AlpineTundraTheme Interface Reference	21
7.2.1	Detailed Description	22
7.2.2	Member Data Documentation	22
7.2.2.1	ALPINE_TUNDRA	22
7.3	gameplay.ai.mood.Anger Class Reference	23
7.3.1	Detailed Description	25
7.3.2	Constructor & Destructor Documentation	25
7.3.2.1	Anger	25
7.3.3	Member Function Documentation	25
7.3.3.1	getBetrayalThreshold	25
7.3.3.2	getMoodsBetrayalCount	26
7.3.3.3	getMoodsSlaveryThreshold	26
7.3.3.4	getNextMood	26
7.3.3.5	getPreviousMood	27
7.3.3.6	getSlaveryThreshold	27
7.3.3.7	incrementBetrayalCount	28
7.3.3.8	incrementSlaveryCount	28
7.3.4	Member Data Documentation	29
7.3.4.1	betrayalThreshold	29
7.3.4.2	moodsBetrayalCount	29
7.3.4.3	moodsSlaveryCount	29
7.3.4.4	nextMoods	29
7.3.4.5	previousMoods	29
7.3.4.6	slaveryThreshold	30
7.4	world.scenery.CaveTheme Interface Reference	30
7.4.1	Detailed Description	31
7.4.2	Member Data Documentation	31
7.4.2.1	CAVE	31
7.5	world.character.Character Class Reference	31
7.5.1	Detailed Description	35

7.5.2	Constructor & Destructor Documentation	35
7.5.2.1	Character	35
7.5.3	Member Function Documentation	35
7.5.3.1	drawScenery	35
7.5.3.2	getSceneryHeight	36
7.5.3.3	getSceneryWidth	36
7.5.3.4	isTraversable	36
7.5.3.5	jump	36
7.5.3.6	jumpX	37
7.5.3.7	jumpY	37
7.5.3.8	move	37
7.5.3.9	moveX	38
7.5.3.10	moveY	39
7.5.3.11	playerEnter	39
7.5.3.12	playerLeave	39
7.5.3.13	playerNear	39
7.5.3.14	triggerAction	40
7.5.4	Member Data Documentation	40
7.5.4.1	CHARACTER_HEIGHT	40
7.5.4.2	CHARACTER_WIDTH	40
7.5.4.3	height	40
7.5.4.4	posX	40
7.5.4.5	posY	40
7.5.4.6	SCENERY_HEIGHT	41
7.5.4.7	SCENERY_WIDTH	41
7.5.4.8	spriteImage	41
7.5.4.9	visible	41
7.5.4.10	width	41
7.5.4.11	xShift	41
7.5.4.12	yShift	41
7.6	debug.CliOptions Class Reference	42
7.6.1	Detailed Description	43
7.6.2	Constructor & Destructor Documentation	43
7.6.2.1	CliOptions	43
7.6.3	Member Function Documentation	44
7.6.3.1	checkHelp	44
7.6.3.2	checkVerboseLevel	45
7.6.3.3	getVerboseLevel	45
7.6.3.4	isHelpQueried	46
7.6.4	Member Data Documentation	46

7.6.4.1	helpQueried	46
7.6.4.2	oneDashOptions	47
7.6.4.3	twoDashesOptions	47
7.6.4.4	verboseLevel	47
7.7	gameplay.ai.mood.Curiosity Class Reference	48
7.7.1	Detailed Description	50
7.7.2	Constructor & Destructor Documentation	50
7.7.2.1	Curiosity	50
7.7.3	Member Function Documentation	50
7.7.3.1	getBetrayalThreshold	50
7.7.3.2	getMoodsBetrayalCount	51
7.7.3.3	getMoodsSlaveryThreshold	51
7.7.3.4	getNextMood	51
7.7.3.5	getPreviousMood	52
7.7.3.6	getSlaveryThreshold	52
7.7.3.7	incrementBetrayalCount	53
7.7.3.8	incrementSlaveryCount	53
7.7.4	Member Data Documentation	54
7.7.4.1	betrayalThreshold	54
7.7.4.2	moodsBetrayalCount	54
7.7.4.3	moodsSlaveryCount	54
7.7.4.4	nextMoods	54
7.7.4.5	previousMoods	54
7.7.4.6	slaveryThreshold	55
7.8	gameplay.ai.mood.Depression Class Reference	55
7.8.1	Detailed Description	57
7.8.2	Constructor & Destructor Documentation	57
7.8.2.1	Depression	57
7.8.3	Member Function Documentation	57
7.8.3.1	getBetrayalThreshold	57
7.8.3.2	getMoodsBetrayalCount	58
7.8.3.3	getMoodsSlaveryThreshold	58
7.8.3.4	getNextMood	58
7.8.3.5	getPreviousMood	59
7.8.3.6	getSlaveryThreshold	59
7.8.3.7	incrementBetrayalCount	60
7.8.3.8	incrementSlaveryCount	60
7.8.4	Member Data Documentation	61
7.8.4.1	betrayalThreshold	61
7.8.4.2	moodsBetrayalCount	61

7.8.4.3	moodsSlaveryCount	61
7.8.4.4	nextMoods	61
7.8.4.5	previousMoods	61
7.8.4.6	slaveryThreshold	62
7.9	world.scenery.Flower Class Reference	62
7.9.1	Detailed Description	64
7.9.2	Constructor & Destructor Documentation	64
7.9.2.1	Flower	64
7.9.3	Member Function Documentation	65
7.9.3.1	drawScenery	65
7.9.3.2	getSceneryHeight	65
7.9.3.3	getSceneryWidth	66
7.9.4	Member Data Documentation	66
7.9.4.1	ALPINE_TUNDRA	66
7.9.4.2	height	66
7.9.4.3	SCENERY_HEIGHT	66
7.9.4.4	SCENERY_WIDTH	66
7.9.4.5	spriteImage	66
7.9.4.6	TAIGA	66
7.9.4.7	TEMPERATE_BROADLEAF	67
7.9.4.8	visible	67
7.9.4.9	width	67
7.9.4.10	xShift	67
7.9.4.11	yShift	67
7.10	gameplay.GamePlay Class Reference	67
7.10.1	Detailed Description	69
7.10.2	Constructor & Destructor Documentation	69
7.10.2.1	GamePlay	69
7.10.3	Member Data Documentation	70
7.10.3.1	artificialIntelligence	70
7.10.3.2	currentWorld	70
7.10.3.3	currentWorldName	70
7.10.3.4	window	70
7.11	world.scenery.Grass Class Reference	70
7.11.1	Detailed Description	72
7.11.2	Constructor & Destructor Documentation	72
7.11.2.1	Grass	72
7.11.3	Member Function Documentation	73
7.11.3.1	drawScenery	73
7.11.3.2	getSceneryHeight	73

7.11.3.3	getSceneryWidth	73
7.11.4	Member Data Documentation	74
7.11.4.1	ALPINE_TUNDRA	74
7.11.4.2	height	74
7.11.4.3	SAVANNA	74
7.11.4.4	SCENERY_HEIGHT	74
7.11.4.5	SCENERY_WIDTH	74
7.11.4.6	spritelImage	74
7.11.4.7	STEPPE	74
7.11.4.8	TAIGA	74
7.11.4.9	TEMPERATE_BROADLEAF	75
7.11.4.10	visible	75
7.11.4.11	width	75
7.11.4.12	xShift	75
7.11.4.13	yShift	75
7.12	world.scenery.Ground Class Reference	75
7.12.1	Detailed Description	77
7.12.2	Constructor & Destructor Documentation	77
7.12.2.1	Ground	77
7.12.3	Member Function Documentation	78
7.12.3.1	drawScenery	78
7.12.3.2	getSceneryHeight	78
7.12.3.3	getSceneryWidth	79
7.12.4	Member Data Documentation	79
7.12.4.1	ALPINE_TUNDRA	79
7.12.4.2	CAVE	79
7.12.4.3	height	79
7.12.4.4	POLAR_DESERT	79
7.12.4.5	SAVANNA	79
7.12.4.6	SCENERY_HEIGHT	79
7.12.4.7	SCENERY_WIDTH	80
7.12.4.8	spritelImage	80
7.12.4.9	STEPPE	80
7.12.4.10	TAIGA	80
7.12.4.11	TEMPERATE_BROADLEAF	80
7.12.4.12	visible	80
7.12.4.13	width	80
7.12.4.14	XERIC_SHRUBLANDS	80
7.12.4.15	xShift	81
7.12.4.16	yShift	81

7.13 world.character.Hero Class Reference	82
7.13.1 Detailed Description	85
7.13.2 Constructor & Destructor Documentation	85
7.13.2.1 Hero	85
7.13.3 Member Function Documentation	85
7.13.3.1 drawScenery	85
7.13.3.2 getSceneryHeight	86
7.13.3.3 getSceneryWidth	86
7.13.3.4 isTraversable	86
7.13.3.5 jump	86
7.13.3.6 jumpX	87
7.13.3.7 jumpY	87
7.13.3.8 move	87
7.13.3.9 moveX	88
7.13.3.10 moveY	89
7.13.3.11 playerEnter	89
7.13.3.12 playerLeave	89
7.13.3.13 playerNear	89
7.13.3.14 triggerAction	90
7.13.4 Member Data Documentation	90
7.13.4.1 CHARACTER_HEIGHT	90
7.13.4.2 CHARACTER_WIDTH	90
7.13.4.3 height	90
7.13.4.4 posX	90
7.13.4.5 posY	90
7.13.4.6 SCENERY_HEIGHT	91
7.13.4.7 SCENERY_WIDTH	91
7.13.4.8 spriteImage	91
7.13.4.9 visible	91
7.13.4.10 width	91
7.13.4.11 xShift	91
7.13.4.12 yShift	91
7.14 world.InteractiveObject Interface Reference	92
7.14.1 Detailed Description	93
7.14.2 Member Function Documentation	93
7.14.2.1 isTraversable	93
7.14.2.2 playerEnter	93
7.14.2.3 playerLeave	94
7.14.2.4 playerNear	94
7.14.2.5 triggerAction	94

7.15	gameplay.ai.mood.Mood Class Reference	94
7.15.1	Detailed Description	96
7.15.2	Constructor & Destructor Documentation	96
7.15.2.1	Mood	96
7.15.3	Member Function Documentation	97
7.15.3.1	getBetrayalThreshold	97
7.15.3.2	getMoodsBetrayalCount	97
7.15.3.3	getMoodsSlaveryThreshold	97
7.15.3.4	getNextMood	97
7.15.3.5	getPreviousMood	98
7.15.3.6	getSlaveryThreshold	99
7.15.3.7	incrementBetrayalCount	99
7.15.3.8	incrementSlaveryCount	99
7.15.4	Member Data Documentation	100
7.15.4.1	betrayalThreshold	100
7.15.4.2	moodsBetrayalCount	100
7.15.4.3	moodsSlaveryCount	100
7.15.4.4	nextMoods	100
7.15.4.5	previousMoods	100
7.15.4.6	slaveryThreshold	101
7.16	gameplay.ai.phrasing.OrderedPhrases Class Reference	101
7.16.1	Detailed Description	102
7.16.2	Constructor & Destructor Documentation	102
7.16.2.1	OrderedPhrases	102
7.16.3	Member Function Documentation	102
7.16.3.1	getPhrasesNumber	102
7.16.4	Member Data Documentation	103
7.16.4.1	phrases	103
7.16.4.2	phrasesOrder	103
7.17	main.Philophobia Class Reference	104
7.17.1	Detailed Description	105
7.17.2	Member Function Documentation	105
7.17.2.1	getImageFilePrefix	105
7.17.2.2	getPhrasesFilePrefix	106
7.17.2.3	getVerbose	107
7.17.2.4	getWorldFilePrefix	107
7.17.2.5	main	107
7.17.3	Member Data Documentation	108
7.17.3.1	gamePlay	108
7.17.3.2	IMAGE_FILE_PREFIX	108

7.17.3.3	PHRASES_FILE_PREFIX	109
7.17.3.4	verbose	109
7.17.3.5	WORLD_FILE_PREFIX	109
7.18	gameplay.ai.phrasing.Phrasing Class Reference	110
7.18.1	Detailed Description	111
7.18.2	Constructor & Destructor Documentation	111
7.18.2.1	Phrasing	111
7.18.3	Member Data Documentation	112
7.18.3.1	actionAsked	112
7.18.3.2	angerBetrayalPhrases	112
7.18.3.3	angerMotivationPhrases	112
7.18.3.4	angerSlaveryPhrases	113
7.18.3.5	curiosityBetrayalPhrases	113
7.18.3.6	curiosityMotivationPhrases	113
7.18.3.7	curiositySlaveryPhrases	113
7.18.3.8	depressionBetrayalPhrases	113
7.18.3.9	depressionMotivationPhrases	113
7.18.3.10	depressionSlaveryPhrases	113
7.18.3.11	powerComplexBetrayalPhrases	114
7.18.3.12	powerComplexMotivationPhrases	114
7.18.3.13	powerComplexSlaveryPhrases	114
7.19	world.scenery.PolarDesertTheme Interface Reference	114
7.19.1	Detailed Description	115
7.19.2	Member Data Documentation	115
7.19.2.1	POLAR_DESERT	115
7.20	gameplay.ai.mood.PowerComplex Class Reference	116
7.20.1	Detailed Description	118
7.20.2	Constructor & Destructor Documentation	118
7.20.2.1	PowerComplex	118
7.20.3	Member Function Documentation	118
7.20.3.1	getBetrayalThreshold	118
7.20.3.2	getMoodsBetrayalCount	119
7.20.3.3	getMoodsSlaveryThreshold	119
7.20.3.4	getNextMood	119
7.20.3.5	getPreviousMood	120
7.20.3.6	getSlaveryThreshold	120
7.20.3.7	incrementBetrayalCount	121
7.20.3.8	incrementSlaveryCount	121
7.20.4	Member Data Documentation	122
7.20.4.1	betrayalThreshold	122

7.20.4.2	moodsBetrayalCount	122
7.20.4.3	moodsSlaveryCount	122
7.20.4.4	nextMoods	122
7.20.4.5	previousMoods	122
7.20.4.6	slaveryThreshold	123
7.21	gameplay.ai.mood.ProbabilityMood Class Reference	123
7.21.1	Detailed Description	124
7.21.2	Constructor & Destructor Documentation	124
7.21.2.1	ProbabilityMood	124
7.21.3	Member Function Documentation	124
7.21.3.1	getMood	124
7.21.3.2	getProbability	124
7.21.4	Member Data Documentation	124
7.21.4.1	mood	124
7.21.4.2	probability	125
7.22	world.scenery.Rock Class Reference	125
7.22.1	Detailed Description	127
7.22.2	Constructor & Destructor Documentation	127
7.22.2.1	Rock	127
7.22.3	Member Function Documentation	128
7.22.3.1	drawScenery	128
7.22.3.2	getSceneryHeight	128
7.22.3.3	getSceneryWidth	129
7.22.3.4	isTraversable	129
7.22.3.5	playerEnter	129
7.22.3.6	playerLeave	129
7.22.3.7	playerNear	129
7.22.3.8	triggerAction	130
7.22.4	Member Data Documentation	130
7.22.4.1	ALPINE_TUNDRA	130
7.22.4.2	CAVE	130
7.22.4.3	height	130
7.22.4.4	POLAR_DESERT	130
7.22.4.5	SCENERY_HEIGHT	130
7.22.4.6	SCENERY_WIDTH	130
7.22.4.7	spriteImage	130
7.22.4.8	TAIGA	131
7.22.4.9	TEMPERATE_BROADLEAF	131
7.22.4.10	visible	131
7.22.4.11	width	131

7.22.4.12	xShift	131
7.22.4.13	yShift	131
7.23	world.scenery.SavannaTheme Interface Reference	131
7.23.1	Detailed Description	132
7.23.2	Member Data Documentation	132
7.23.2.1	SAVANNA	132
7.24	world.scenery.Scenery Class Reference	133
7.24.1	Detailed Description	135
7.24.2	Constructor & Destructor Documentation	135
7.24.2.1	Scenery	135
7.24.2.2	Scenery	136
7.24.2.3	Scenery	136
7.24.3	Member Function Documentation	137
7.24.3.1	drawScenery	137
7.24.3.2	getSceneryHeight	137
7.24.3.3	getSceneryWidth	137
7.24.4	Member Data Documentation	138
7.24.4.1	height	138
7.24.4.2	SCENERY_HEIGHT	138
7.24.4.3	SCENERY_WIDTH	138
7.24.4.4	spritelImage	138
7.24.4.5	visible	138
7.24.4.6	width	138
7.24.4.7	xShift	139
7.24.4.8	yShift	139
7.25	world.scenery.Shore Class Reference	139
7.25.1	Detailed Description	141
7.25.2	Constructor & Destructor Documentation	141
7.25.2.1	Shore	141
7.25.3	Member Function Documentation	142
7.25.3.1	drawScenery	142
7.25.3.2	getSceneryHeight	142
7.25.3.3	getSceneryWidth	143
7.25.4	Member Data Documentation	143
7.25.4.1	height	143
7.25.4.2	POLAR_DESERT	143
7.25.4.3	SCENERY_HEIGHT	143
7.25.4.4	SCENERY_WIDTH	143
7.25.4.5	spritelImage	143
7.25.4.6	TAIGA	143

7.25.4.7	visible	144
7.25.4.8	width	144
7.25.4.9	xShift	144
7.25.4.10	yShift	144
7.26	world.scenery.Shrub Class Reference	144
7.26.1	Detailed Description	146
7.26.2	Constructor & Destructor Documentation	146
7.26.2.1	Shrub	146
7.26.3	Member Function Documentation	147
7.26.3.1	drawScenery	147
7.26.3.2	getSceneryHeight	147
7.26.3.3	getSceneryWidth	148
7.26.4	Member Data Documentation	148
7.26.4.1	ALPINE_TUNDRA	148
7.26.4.2	CAVE	148
7.26.4.3	height	148
7.26.4.4	POLAR_DESERT	148
7.26.4.5	SAVANNA	148
7.26.4.6	SCENERY_HEIGHT	148
7.26.4.7	SCENERY_WIDTH	149
7.26.4.8	spriteImage	149
7.26.4.9	STEPPE	149
7.26.4.10	TAIGA	149
7.26.4.11	TEMPERATE_BROADLEAF	149
7.26.4.12	visible	149
7.26.4.13	width	149
7.26.4.14	XERIC_SHRUBLANDS	149
7.26.4.15	xShift	150
7.26.4.16	yShift	150
7.27	world.scenery.SteppeTheme Interface Reference	150
7.27.1	Detailed Description	151
7.27.2	Member Data Documentation	151
7.27.2.1	STEPPE	151
7.28	world.scenery.TaigaTheme Interface Reference	151
7.28.1	Detailed Description	152
7.28.2	Member Data Documentation	152
7.28.2.1	TAIGA	152
7.29	world.character.TalkingCharacter Class Reference	152
7.29.1	Detailed Description	156
7.29.2	Constructor & Destructor Documentation	156

7.29.2.1	TalkingCharacter	156
7.29.3	Member Function Documentation	156
7.29.3.1	drawScenery	156
7.29.3.2	getSceneryHeight	156
7.29.3.3	getSceneryWidth	157
7.29.3.4	isTraversable	157
7.29.3.5	jump	157
7.29.3.6	jumpX	158
7.29.3.7	jumpY	158
7.29.3.8	move	158
7.29.3.9	moveX	159
7.29.3.10	moveY	160
7.29.3.11	playerEnter	160
7.29.3.12	playerLeave	160
7.29.3.13	playerNear	160
7.29.3.14	talk	161
7.29.3.15	triggerAction	161
7.29.4	Member Data Documentation	162
7.29.4.1	CHARACTER_HEIGHT	162
7.29.4.2	CHARACTER_WIDTH	162
7.29.4.3	height	162
7.29.4.4	posX	162
7.29.4.5	posY	162
7.29.4.6	SCENERY_HEIGHT	162
7.29.4.7	SCENERY_WIDTH	162
7.29.4.8	spritelImage	162
7.29.4.9	visible	163
7.29.4.10	width	163
7.29.4.11	xShift	163
7.29.4.12	yShift	163
7.30	world.scenery.TemperateBroadleafTheme Interface Reference	163
7.30.1	Detailed Description	164
7.30.2	Member Data Documentation	164
7.30.2.1	TEMPERATE_BROADLEAF	164
7.31	window.topbar.TopBar Class Reference	164
7.31.1	Detailed Description	167
7.31.2	Constructor & Destructor Documentation	167
7.31.2.1	TopBar	167
7.31.3	Member Function Documentation	168
7.31.3.1	getTopBarHeight	168

7.31.3.2	getTopBarMargin	168
7.31.3.3	hidePauseButton	168
7.31.3.4	paintComponent	169
7.31.3.5	showPauseButton	169
7.31.4	Member Data Documentation	169
7.31.4.1	closeButton	169
7.31.4.2	pauseButton	169
7.31.4.3	TOPBAR_HEIGHT	169
7.31.4.4	TOPBAR_MARGIN	170
7.32	window.topbar.TopBarButton Class Reference	170
7.32.1	Detailed Description	171
7.32.2	Constructor & Destructor Documentation	171
7.32.2.1	TopBarButton	171
7.32.3	Member Function Documentation	172
7.32.3.1	paintComponent	172
7.32.4	Member Data Documentation	172
7.32.4.1	buttonImage	172
7.33	world.scenery.Tree Class Reference	173
7.33.1	Detailed Description	175
7.33.2	Constructor & Destructor Documentation	175
7.33.2.1	Tree	175
7.33.3	Member Function Documentation	175
7.33.3.1	drawScenery	175
7.33.3.2	getSceneryHeight	175
7.33.3.3	getSceneryWidth	176
7.33.3.4	isTraversable	176
7.33.3.5	playerEnter	176
7.33.3.6	playerLeave	177
7.33.3.7	playerNear	177
7.33.3.8	triggerAction	177
7.33.4	Member Data Documentation	177
7.33.4.1	ALPINE_TUNDRA	177
7.33.4.2	CAVE	177
7.33.4.3	height	177
7.33.4.4	POLAR_DESERT	178
7.33.4.5	SAVANNA	178
7.33.4.6	SCENERY_HEIGHT	178
7.33.4.7	SCENERY_WIDTH	178
7.33.4.8	spriteImage	178
7.33.4.9	STEPPE	178

7.33.4.10 TAIGA	178
7.33.4.11 TEMPERATE_BROADLEAF	178
7.33.4.12 visible	179
7.33.4.13 width	179
7.33.4.14 XERIC_SHRUBLANDS	179
7.33.4.15 xShift	179
7.33.4.16 yShift	179
7.34 window.ui.UI Class Reference	179
7.34.1 Detailed Description	182
7.34.2 Constructor & Destructor Documentation	182
7.34.2.1 UI	182
7.34.2.2 UI	182
7.34.3 Member Function Documentation	183
7.34.3.1 drawWorld	183
7.34.3.2 paintComponent	184
7.34.3.3 setDisplayedWorld	184
7.34.4 Member Data Documentation	185
7.34.4.1 displayedWorld	185
7.34.4.2 windowHeight	185
7.34.4.3 windowWidth	185
7.35 debug.Verbose Class Reference	185
7.35.1 Detailed Description	187
7.35.2 Constructor & Destructor Documentation	187
7.35.2.1 Verbose	187
7.35.3 Member Function Documentation	188
7.35.3.1 calls	188
7.35.3.2 critical	188
7.35.3.3 information	189
7.35.3.4 serious	190
7.35.3.5 warning	190
7.35.4 Member Data Documentation	191
7.35.4.1 log	191
7.35.4.2 verboseMode	191
7.36 world.scenery.Water Class Reference	191
7.36.1 Detailed Description	194
7.36.2 Constructor & Destructor Documentation	194
7.36.2.1 Water	194
7.36.3 Member Function Documentation	195
7.36.3.1 drawScenery	195
7.36.3.2 getSceneryHeight	195

7.36.3.3	getSceneryWidth	196
7.36.3.4	isTraversable	196
7.36.3.5	playerEnter	196
7.36.3.6	playerLeave	196
7.36.3.7	playerNear	196
7.36.3.8	triggerAction	197
7.36.4	Member Data Documentation	197
7.36.4.1	height	197
7.36.4.2	POLAR_DESERT	197
7.36.4.3	SCENERY_HEIGHT	197
7.36.4.4	SCENERY_WIDTH	197
7.36.4.5	spriteImage	197
7.36.4.6	TAIGA	197
7.36.4.7	visible	198
7.36.4.8	width	198
7.36.4.9	xShift	198
7.36.4.10	yShift	198
7.37	window.Window Class Reference	199
7.37.1	Detailed Description	201
7.37.2	Constructor & Destructor Documentation	201
7.37.2.1	Window	201
7.37.3	Member Function Documentation	201
7.37.3.1	getUserInterface	201
7.37.3.2	setFailState	202
7.37.3.3	setGameState	202
7.37.3.4	setLoadingState	202
7.37.3.5	setWinState	202
7.37.4	Member Data Documentation	202
7.37.4.1	topbar	202
7.37.4.2	userInterface	203
7.37.4.3	windowState	203
7.38	window.WindowState Enum Reference	203
7.38.1	Detailed Description	204
7.38.2	Member Data Documentation	204
7.38.2.1	Fail	204
7.38.2.2	FirstLoading	204
7.38.2.3	Game	204
7.38.2.4	LoadingScreen	204
7.38.2.5	Win	205
7.39	world.World Class Reference	205

7.39.1 Detailed Description	207
7.39.2 Constructor & Destructor Documentation	207
7.39.2.1 World	207
7.39.2.2 World	208
7.39.3 Member Function Documentation	208
7.39.3.1 generateWorld	208
7.39.3.2 getMap	209
7.39.3.3 getSizeX	209
7.39.3.4 getSizeY	210
7.39.4 Member Data Documentation	210
7.39.4.1 map	210
7.39.4.2 sizeX	210
7.39.4.3 sizeY	210
7.39.4.4 type	211
7.40 world.scenery.XericShrublandsTheme Interface Reference	211
7.40.1 Detailed Description	212
7.40.2 Member Data Documentation	212
7.40.2.1 XERIC_SHRUBLANDS	212
8 File Documentation	213
8.1 debug/CliOptions.java File Reference	213
8.2 debug/Verbose.java File Reference	213
8.3 gameplay/ai/AI.java File Reference	213
8.4 gameplay/ai/mood/Anger.java File Reference	214
8.5 gameplay/ai/mood/Curiosity.java File Reference	214
8.6 gameplay/ai/mood/Depression.java File Reference	214
8.7 gameplay/ai/mood/Mood.java File Reference	214
8.8 gameplay/ai/mood/PowerComplex.java File Reference	214
8.9 gameplay/ai/phrasing/OrderedPhrases.java File Reference	215
8.10 gameplay/ai/phrasing/Phrasing.java File Reference	215
8.11 gameplay/GamePlay.java File Reference	215
8.12 main/Philophobia.java File Reference	215
8.13 README.md File Reference	216
8.14 window/topbar/TopBar.java File Reference	216
8.15 window/topbar/TopBarButton.java File Reference	216
8.16 window/ui/UI.java File Reference	216
8.17 window/Window.java File Reference	216
8.18 world/character/Character.java File Reference	217
8.19 world/character/Hero.java File Reference	217
8.20 world/character/TalkingCharacter.java File Reference	217

8.21	world/InteractiveObject.java File Reference	217
8.22	world/scenery/AlpineTundraTheme.java File Reference	217
8.23	world/scenery/CaveTheme.java File Reference	218
8.24	world/scenery/Flower.java File Reference	218
8.25	world/scenery/Grass.java File Reference	218
8.26	world/scenery/Ground.java File Reference	218
8.27	world/scenery/PolarDesertTheme.java File Reference	219
8.28	world/scenery/Rock.java File Reference	219
8.29	world/scenery/SavannaTheme.java File Reference	219
8.30	world/scenery/Scenery.java File Reference	219
8.31	world/scenery/Shore.java File Reference	220
8.32	world/scenery/Shrub.java File Reference	220
8.33	world/scenery/SteppeTheme.java File Reference	220
8.34	world/scenery/TaigaTheme.java File Reference	220
8.35	world/scenery/TemperateBroadleafTheme.java File Reference	221
8.36	world/scenery/Tree.java File Reference	221
8.37	world/scenery/Water.java File Reference	221
8.38	world/scenery/XericShrublandsTheme.java File Reference	221
8.39	world/World.java File Reference	222
Index		223

Chapter 1

Main Page

! [Build Status] (<https://drone.io/github.com/minijackson/Philophobia/status.-png>)

Sadistic Java game project

Chapter 2

Namespace Index

2.1 Packages

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

debug	11
gameplay	11
gameplay.ai	11
gameplay.ai.mood	12
gameplay.ai.phrasing	12
main	12
window	12
window.topbar	12
window.ui	13
world	13
world.character	13
world.scenery	13

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

gameplay.ai.AI	15
world.scenery.AlpineTundraTheme	21
world.scenery.Flower	62
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shrub	144
world.scenery.Tree	173
world.scenery.CaveTheme	30
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shrub	144
world.scenery.Tree	173
debug.CliOptions	42
gameplay.GamePlay	67
world.InteractiveObject	92
world.character.Character	31
world.character.Hero	82
world.character.TalkingCharacter	152
world.scenery.Rock	125
world.scenery.Tree	173
world.scenery.Water	191
gameplay.ai.mood.Mood	94
gameplay.ai.mood.Anger	23
gameplay.ai.mood.Curiosity	48
gameplay.ai.mood.Depression	55
gameplay.ai.mood.PowerComplex	116
gameplay.ai.phrasing.OrderedPhrases	101
main.Philophobia	104
gameplay.ai.phrasing.Phrasing	110
world.scenery.PolarDesertTheme	114
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shore	139
world.scenery.Shrub	144
world.scenery.Tree	173

world.scenery.Water	191
gameplay.ai.mood.ProbabilityMood	123
world.scenery.SavannaTheme	131
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Shrub	144
world.scenery.Tree	173
world.scenery.Scenery	133
world.character.Character	31
world.scenery.Flower	62
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shore	139
world.scenery.Shrub	144
world.scenery.Tree	173
world.scenery.Water	191
world.scenery.SteppeTheme	150
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Shrub	144
world.scenery.Tree	173
world.scenery.TaigaTheme	151
world.scenery.Flower	62
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shore	139
world.scenery.Shrub	144
world.scenery.Tree	173
world.scenery.Water	191
world.scenery.TemperateBroadleafTheme	163
world.scenery.Flower	62
world.scenery.Grass	70
world.scenery.Ground	75
world.scenery.Rock	125
world.scenery.Shrub	144
world.scenery.Tree	173
debug.Verbose	185
window.WindowState	203
world.World	205
world.scenery.XericShrublandsTheme	211
world.scenery.Ground	75
world.scenery.Shrub	144
world.scenery.Tree	173
JButton	
window.topbar.TopBarButton	170
JFrame	
window.Window	199
JPanel	
window.topbar.TopBar	164
window.ui.UI	179

Chapter 4

Class Index

4.1 Class List

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

gameplay.ai.AI	Class representing the sadistic robot	15
world.scenery.AlpineTundraTheme	Interface used to define that a Scenery object have an Alpine Tundra style	21
gameplay.ai.mood.Anger	23
world.scenery.CaveTheme	Interface used to tell that a Scenery object has a Cave style	30
world.character.Character	Class used to handle a character (player or non-player)	31
debug.CliOptions	Class used to analyze the commands passed to the program	42
gameplay.ai.mood.Curiosity	48
gameplay.ai.mood.Depression	55
world.scenery.Flower	Class used to handle a flower object	62
gameplay.GamePlay	Class handling the game play of this game	67
world.scenery.Grass	Class used to handle a grass object	70
world.scenery.Ground	Class used to handle a ground object	75
world.character.Hero	82
world.InteractiveObject	92
gameplay.ai.mood.Mood	Abstract class representing an AI mood	94
gameplay.ai.phrasing.OrderedPhrases	Class used to contain an ordered list of phrases	101
main.Philophobia	Main class	104
gameplay.ai.phrasing.Phrasing	110
world.scenery.PolarDesertTheme	Interface used to define that a Scenery object has a Polar Desert style	114
gameplay.ai.mood.PowerComplex	116
gameplay.ai.mood.ProbabilityMood	Class used to associate a mood with a probability	123
world.scenery.Rock	Class used to handle a rock object	125

world.scenery.SavannaTheme	Interface used to define that a Scenery object have a Savanna style	131
world.scenery.Scenery	Class used to handle any world object	133
world.scenery.Shore	Class used to handle a shore object	139
world.scenery.Shrub	Class used to handle a shrub object	144
world.scenery.SteppeTheme	Interface used to define that a Scenery object have a Steppe style	150
world.scenery.TaigaTheme	Interface used to define that a Scenery object have a Taiga style	151
world.character.TalkingCharacter	Class handling a talking character	152
world.scenery.TemperateBroadleafTheme	Interface used to define that a Scenery object have a Temperate Broadleaf style	163
window.topbar.TopBar	Class handling the top bar of the program's main window	164
window.topbar.TopBarButton	Class handling the UI buttons	170
world.scenery.Tree	Class used to handle a tree object	173
window.ui.UI	Class used to handle the window's user interface	179
debug.Verbose	Class used to display and log messages all over this program	185
world.scenery.Water	Class used to handle a water object	191
window.Window	199
window.WindowState	Enumeration of the possible states of the displayed graphics	203
world.World	World is a class used to handle the game world graphics	205
world.scenery.XericShrublandsTheme	Interface used to define that a Scenery object have a Xeric Shrublands style	211

Chapter 5

File Index

5.1 File List

Here is a list of all files with brief descriptions:

debug/ CliOptions.java	213
debug/ Verbose.java	213
gameplay/ GamePlay.java	215
gameplay/ai/ AI.java	213
gameplay/ai/mood/ Anger.java	214
gameplay/ai/mood/ Curiosity.java	214
gameplay/ai/mood/ Depression.java	214
gameplay/ai/mood/ Mood.java	214
gameplay/ai/mood/ PowerComplex.java	214
gameplay/ai/phrasing/ OrderedPhrases.java	215
gameplay/ai/phrasing/ Phrasing.java	215
main/ Philophobia.java	215
window/ Window.java	216
window/topbar/ TopBar.java	216
window/topbar/ TopBarButton.java	216
window/ui/ UI.java	216
world/ InteractiveObject.java	217
world/ World.java	222
world/character/ Character.java	217
world/character/ Hero.java	217
world/character/ TalkingCharacter.java	217
world/scenery/ AlpineTundraTheme.java	217
world/scenery/ CaveTheme.java	218
world/scenery/ Flower.java	218
world/scenery/ Grass.java	218
world/scenery/ Ground.java	218
world/scenery/ PolarDesertTheme.java	219
world/scenery/ Rock.java	219
world/scenery/ SavannaTheme.java	219
world/scenery/ Scenery.java	219
world/scenery/ Shore.java	220
world/scenery/ Shrub.java	220
world/scenery/ SteppeTheme.java	220
world/scenery/ TaigaTheme.java	220
world/scenery/ TemperateBroadleafTheme.java	221
world/scenery/ Tree.java	221
world/scenery/ Water.java	221
world/scenery/ XericShrublandsTheme.java	221

Chapter 6

Namespace Documentation

6.1 Package debug

Classes

- class [CliOptions](#)
Class used to analyze the commands passed to the program.
- class [Verbose](#)
Class used to display and log messages all over this program.

6.2 Package gameplay

Packages

- package [ai](#)

Classes

- class [GamePlay](#)
Class handling the game play of this game.

6.3 Package gameplay.ai

Packages

- package [mood](#)
- package [phrasing](#)

Classes

- class [AI](#)
Class representing the sadistic robot.

6.4 Package gameplay.ai.mood

Classes

- class [Anger](#)
- class [Curiosity](#)
- class [Depression](#)
- class [Mood](#)
Abstract class representing an [AI](#) mood.
- class [ProbabilityMood](#)
Class used to associate a mood with a probability.
- class [PowerComplex](#)

6.5 Package gameplay.ai.phrasing

Classes

- class [OrderedPhrases](#)
Class used to contain an ordered list of phrases.
- class [Phrasing](#)

6.6 Package main

Classes

- class [Philophobia](#)
Main class.

6.7 Package window

Packages

- package [topbar](#)
- package [ui](#)

Classes

- class [Window](#)
- enum [WindowState](#)
Enumeration of the possible states of the displayed graphics.

6.8 Package window.topbar

Classes

- class [TopBar](#)
Class handling the top bar of the program's main window.
- class [TopBarButton](#)
Class handling the UI buttons.

6.9 Package window.ui

Classes

- class [UI](#)

Class used to handle the window's user interface.

6.10 Package world

Packages

- package [character](#)
- package [scenery](#)

Classes

- interface [InteractiveObject](#)
- class [World](#)

[World](#) is a class used to handle the game world graphics.

6.11 Package world.character

Classes

- class [Character](#)
- class [Hero](#)
- class [TalkingCharacter](#)

Class handling a talking character.

6.12 Package world.scenery

Classes

- interface [AlpineTundraTheme](#)
Interface used to define that a [Scenery](#) object have an Alpine Tundra style.
- interface [CaveTheme](#)
Interface used to tell that a [Scenery](#) object has a Cave style.
- class [Flower](#)
Class used to handle a flower object.
- class [Grass](#)
Class used to handle a grass object.
- class [Ground](#)
Class used to handle a ground object.
- interface [PolarDesertTheme](#)
Interface used to define that a [Scenery](#) object has a Polar Desert style.
- class [Rock](#)
Class used to handle a rock object.

- interface [SavannaTheme](#)
Interface used to define that a [Scenery](#) object have a Savanna style.
- class [Scenery](#)
Class used to handle any world object.
- class [Shore](#)
Class used to handle a shore object.
- class [Shrub](#)
Class used to handle a shrub object.
- interface [SteppeTheme](#)
Interface used to define that a [Scenery](#) object have a Steppe style.
- interface [TaigaTheme](#)
Interface used to define that a [Scenery](#) object have a Taiga style.
- interface [TemperateBroadleafTheme](#)
Interface used to define that a [Scenery](#) object have a Temperate Broadleaf style.
- class [Tree](#)
Class used to handle a tree object.
- class [Water](#)
Class used to handle a water object.
- interface [XericShrublandsTheme](#)
Interface used to define that a [Scenery](#) object have a Xeric Shrublands style.

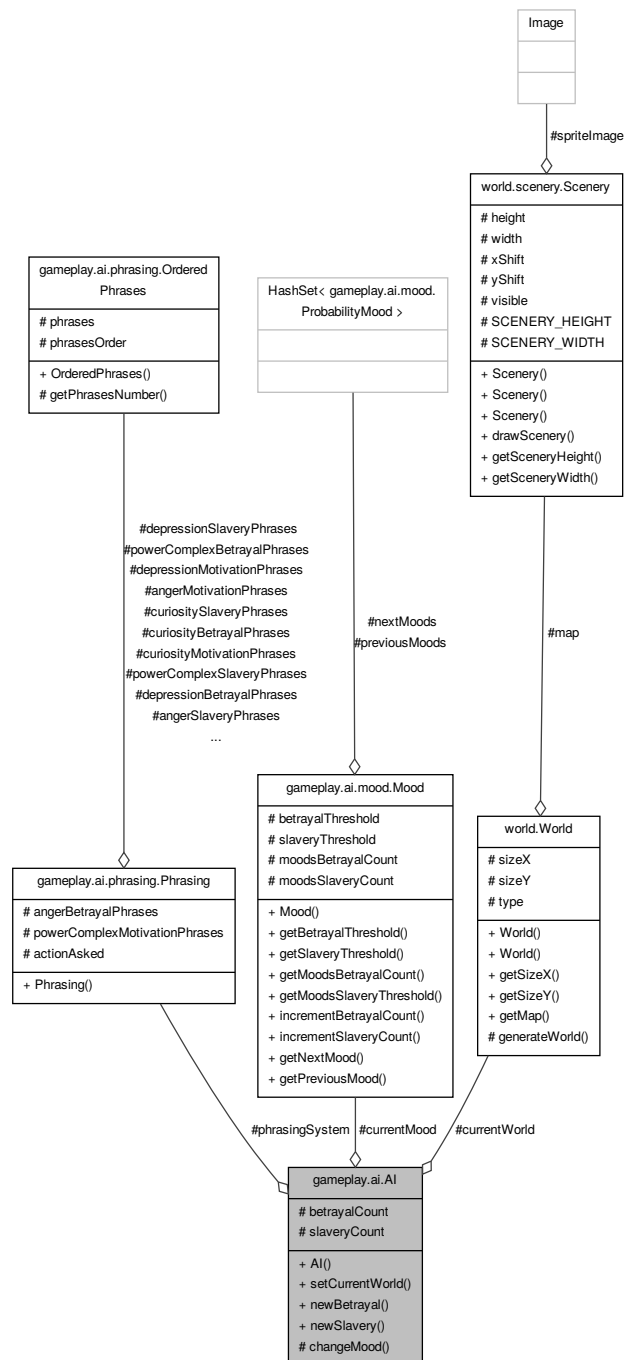
Chapter 7

Class Documentation

7.1 `gameplay.ai.AI` Class Reference

Class representing the sadistic robot.

Collaboration diagram for `gameplay.ai.AI`:



Public Member Functions

- [AI \(World currentWorld\)](#)
AI class main constructor.
- void [setCurrentWorld \(World world\)](#)
Setter for the current world field.
- void [newBetrayal \(\)](#)

Method called when the player make a new betrayal (does not do what the AI asked)

- void [newSlavery](#) ()

Method called when the player do what the AI asked.

Protected Member Functions

- void [changeMood](#) (Class< [Mood](#) > moodClass)

Change the current mood field from a Class object.

Protected Attributes

- [World](#) [currentWorld](#)

World where the player is currently in.

- [Mood](#) [currentMood](#)

Mood in which the AI is currently in.

- [Phrasing](#) [phrasingSystem](#)

AI talk system.

- int [betrayalCount](#)

Number of betrayals done by the player.

- int [slaveryCount](#)

Number of asked actions by the AI done by the player.

7.1.1 Detailed Description

Class representing the sadistic robot.

The robot ask the player to do terrible things without taking out the player's choice to do or not to do the task

The AI has a mood system with 4 basic moods : Curiosity, Anger, Depression, Power complex

The robot can switch between these moods considering the user's choices and interact with the environment in a bad way

See Also

[gameplay.ai.mood.Mood](#)

Definition at line 28 of file AI.java.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 [gameplay.ai.AI](#)([World](#) [currentWorld](#))

AI class main constructor.

Parameters

currentWorld	World where the player is currently in
------------------------------	----------------------------------------

Definition at line 60 of file AI.java.

References [gameplay.ai.AI.betrayalCount](#), [gameplay.ai.AI.currentMood](#), [gameplay.ai.AI.currentWorld](#), [gameplay.ai.AI.phrasingSystem](#), and [gameplay.ai.AI.slaveryCount](#).

```

60         {
61             Philophobia.getVerbose().information("Creating sadistic AI", "gameplay/ai/AI.java", "AI.AI(World)")
62         };
63         this.currentWorld = currentWorld;
64         currentMood = new Curiosity();
65         phrasingSystem = new Phrasing();
66         betrayalCount = 0;
67         slaveryCount = 0;
68     }

```

7.1.3 Member Function Documentation

7.1.3.1 void gameplay.ai.AI.changeMood (Class< Mood > moodClass) [protected]

Change the current mood field from a Class object.

See Also

[currentMood](#)

Definition at line 112 of file AI.java.

References [gameplay.ai.AI.currentMood](#).

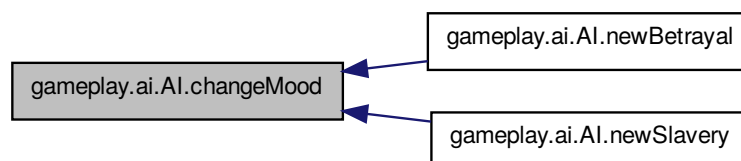
Referenced by [gameplay.ai.AI.newBetrayal\(\)](#), and [gameplay.ai.AI.newSlavery\(\)](#).

```

112         {
113             try {
114                 Philophobia.getVerbose().information("AI switching from " +
115                 currentMood.getClass().getName() + " mood to " + moodClass.getName() + " mood", "
116                 gameplay/ai/AI.java", "AI.changeMood(Class)");
117                 currentMood = moodClass.newInstance();
118             } catch (SecurityException e) {
119                 Philophobia.getVerbose().serious("Security exception when switching from " +
120                 currentMood.getClass().getName() + " mood to " + moodClass.getName() + " mood: " + e.getMessage(
121                 ), "gameplay/ai/AI.java", "AI.changeMood(Class)");
122             } catch (InstantiationException e) {
123                 Philophobia.getVerbose().serious("Instantiation exception when switching from " +
124                 currentMood.getClass().getName() + " mood to " + moodClass.getName() + " mood: " + e.getMessage(
125                 ), "gameplay/ai/AI.java", "AI.changeMood(Class)");
126             } catch (IllegalAccessException e) {
127                 Philophobia.getVerbose().serious("Illegal access exception when switching from " +
128                 currentMood.getClass().getName() + " mood to " + moodClass.getName() + " mood: " + e.getMessage(
129                 ), "gameplay/ai/AI.java", "AI.changeMood(Class)");
130             }
131         }

```

Here is the caller graph for this function:



7.1.3.2 void gameplay.ai.AI.newBetrayal ()

Method called when the player make a new betrayal (does not do what the [AI](#) asked)

Definition at line 87 of file AI.java.

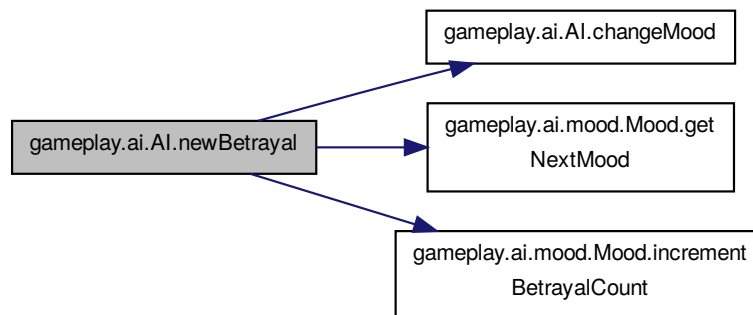
References `gameplay.ai.AI.betrayalCount`, `gameplay.ai.AI.changeMood()`, `gameplay.ai.AI.currentMood`, `gameplay.ai.mood.Mood.getNextMood()`, and `gameplay.ai.mood.Mood.incrementBetrayalCount()`.

```

87         {
88     Philophobia.getVerbose().calls("New betrayal action detected", "gameplay/ai/AI.java", "
AI.newBetrayal()");
89     betrayalCount++;
90     if(currentMood.incrementBetrayalCount() >=
currentMood.getBetrayalThreshold()) {
91         changeMood(currentMood.getNextMood());
92     }
93 }

```

Here is the call graph for this function:



7.1.3.3 void gameplay.ai.AI.newSlavery ()

Method called when the player do what the AI asked.

Definition at line 99 of file AI.java.

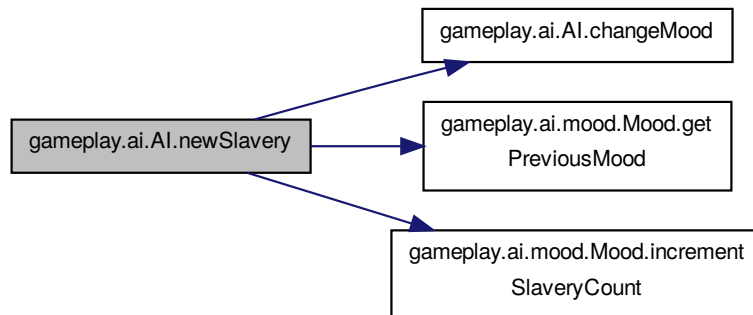
References `gameplay.ai.AI.changeMood()`, `gameplay.ai.AI.currentMood`, `gameplay.ai.mood.Mood.getPreviousMood()`, `gameplay.ai.mood.Mood.incrementSlaveryCount()`, and `gameplay.ai.AI.slaveryCount`.

```

99         {
100     Philophobia.getVerbose().calls("New slavery action detected", "gameplay/ai/AI.java", "
AI.newSlavery()");
101     slaveryCount++;
102     if(currentMood.incrementSlaveryCount() >=
currentMood.getSlaveryThreshold()) {
103         changeMood(currentMood.getPreviousMood());
104     }
105 }

```

Here is the call graph for this function:



7.1.3.4 void gameplay.ai.AI.setCurrentWorld (World world)

Setter for the current world field.

See Also

[currentWorld](#)

Definition at line 74 of file AI.java.

References `gameplay.ai.AI.currentWorld`.

```

74                                     {
75         currentWorld = world;
76     }
```

7.1.4 Member Data Documentation

7.1.4.1 int gameplay.ai.AI.betrayalCount [protected]

Number of betrayals done by the player.

Definition at line 48 of file AI.java.

Referenced by `gameplay.ai.AI.AI()`, and `gameplay.ai.AI.newBetrayal()`.

7.1.4.2 Mood gameplay.ai.AI.currentMood [protected]

Mood in which the [AI](#) is currently in.

Definition at line 38 of file AI.java.

Referenced by `gameplay.ai.AI.AI()`, `gameplay.ai.AI.changeMood()`, `gameplay.ai.AI.newBetrayal()`, and `gameplay.ai.AI.newSlavery()`.

7.1.4.3 World gameplay.ai.AI.currentWorld [protected]

World where the player is currently in.

Definition at line 33 of file AI.java.

Referenced by gameplay.ai.AI.AI(), and gameplay.ai.AI.setCurrentWorld().

7.1.4.4 Phrasing gameplay.ai.AI.phrasingSystem [protected]

AI talk system.

Definition at line 43 of file AI.java.

Referenced by gameplay.ai.AI.AI().

7.1.4.5 int gameplay.ai.AI.slaveryCount [protected]

Number of asked actions by the AI done by the player.

Definition at line 54 of file AI.java.

Referenced by gameplay.ai.AI.AI(), and gameplay.ai.AI.newSlavery().

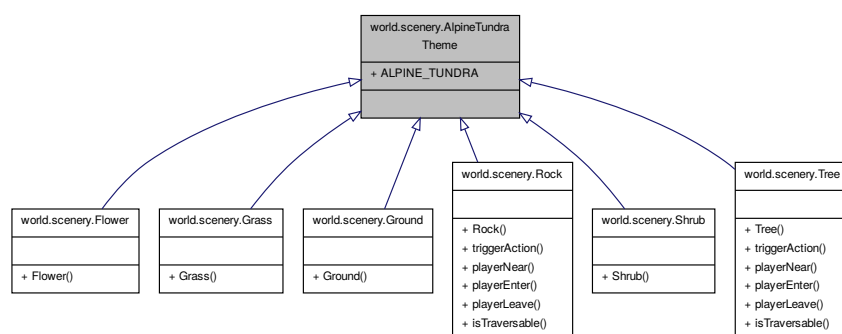
The documentation for this class was generated from the following file:

- gameplay/ai/AI.java

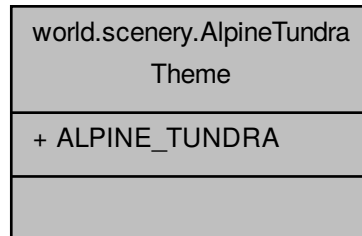
7.2 world.scenery.AlpineTundraTheme Interface Reference

Interface used to define that a Scenery object have an Alpine Tundra style.

Inheritance diagram for world.scenery.AlpineTundraTheme:



Collaboration diagram for world.scenery.AlpineTundraTheme:



Static Public Attributes

- static String [ALPINE_TUNDRA](#) = "alpinetundra"

String used to tell in which file is the sprite matching the Alpine Tundra style.

7.2.1 Detailed Description

Interface used to define that a [Scenery](#) object have an Alpine Tundra style.

Definition at line 7 of file `AlpineTundraTheme.java`.

7.2.2 Member Data Documentation

7.2.2.1 String `world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra"` `[static]`

String used to tell in which file is the sprite matching the Alpine Tundra style.

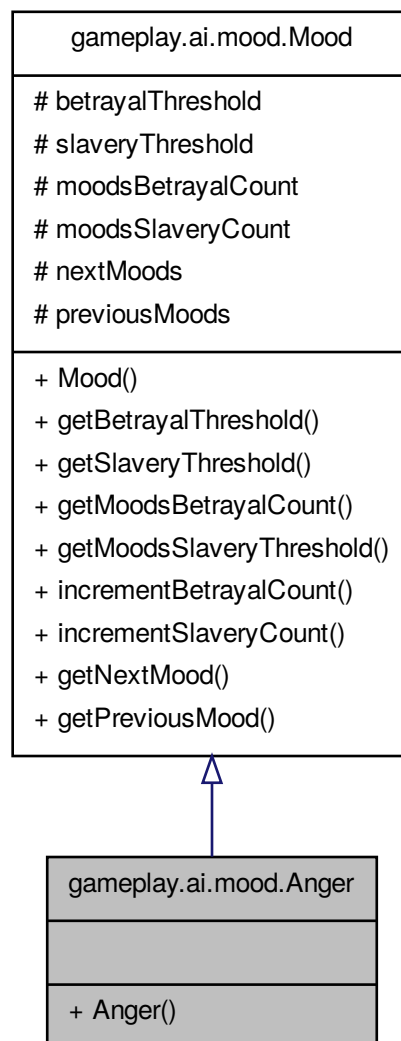
Definition at line 13 of file `AlpineTundraTheme.java`.

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

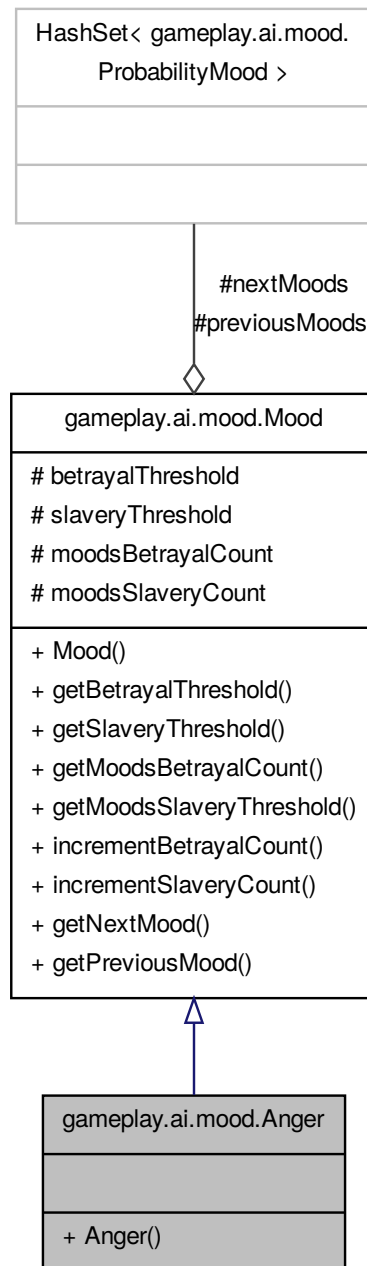
- `world/scenery/AlpineTundraTheme.java`

7.3 gameplay.ai.mood.Anger Class Reference

Inheritance diagram for gameplay.ai.mood.Anger:



Collaboration diagram for gameplay.ai.mood.Anger:



Public Member Functions

- [Anger](#) ()
- int [getBetrayalThreshold](#) ()
Getter for the betrayal threshold field.
- int [getSlaveryThreshold](#) ()
Getter for the slavery threshold field.

- int [getMoodsBetrayalCount](#) ()
Getter for the betrayal count field.
- int [getMoodsSlaveryThreshold](#) ()
Getter for the slavery count field.
- int [incrementBetrayalCount](#) ()
Increment and return the betrayal count field.
- int [incrementSlaveryCount](#) ()
Increment and return the slavery count field.
- Class< [Mood](#) > [getNextMood](#) ()
When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.
- Class< [Mood](#) > [getPreviousMood](#) ()
When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Protected Attributes

- int [betrayalThreshold](#)
If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [slaveryThreshold](#)
If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [moodsBetrayalCount](#)
Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.
- int [moodsSlaveryCount](#)
Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.
- HashSet< [ProbabilityMood](#) > [nextMoods](#)
Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.
- HashSet< [ProbabilityMood](#) > [previousMoods](#)
Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

7.3.1 Detailed Description

Definition at line 5 of file Anger.java.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 `gameplay.ai.mood.Anger.Anger ()`

Definition at line 7 of file Anger.java.

```

7         {
8             super(10, 6);
9         }
```

7.3.3 Member Function Documentation

7.3.3.1 `int gameplay.ai.mood.Mood.getBetrayalThreshold ()` [inherited]

Getter for the betrayal threshold field.

See Also[betrayalThreshold](#)

Definition at line 79 of file Mood.java.

References gameplay.ai.mood.Mood.betrayalThreshold.

```

79         {
80             return betrayalThreshold;
81         }

```

7.3.3.2 int gameplay.ai.mood.Mood.getMoodsBetrayalCount () [inherited]

Getter for the betrayal count field.

See Also[moodsBetrayalCount](#)

Definition at line 95 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

```

95         {
96             return moodsBetrayalCount;
97         }

```

7.3.3.3 int gameplay.ai.mood.Mood.getMoodsSlaveryThreshold () [inherited]

Getter for the slavery count field.

See Also[moodsSlaveryCount](#)

Definition at line 103 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

```

103         {
104             return moodsSlaveryCount;
105         }

```

7.3.3.4 Class<Mood> gameplay.ai.mood.Mood.getNextMood () [inherited]

When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 128 of file Mood.java.

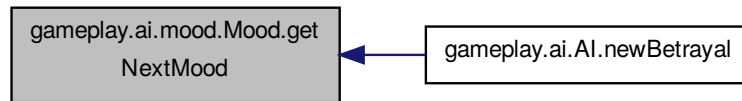
Referenced by gameplay.ai.AI.newBetrayal().

```

128         {
129             Iterator<ProbabilityMood> it = nextMoods.iterator();
130             ProbabilityMood probMood = null;
131             while(it.hasNext()) {
132                 probMood = it.next();
133                 if(Math.random() >= probMood.getProbability()) {
134                     return probMood.getMood();
135                 }
136             }
137         }
138     }
139     return probMood.getMood();
140 }
141 }

```

Here is the caller graph for this function:



7.3.3.5 Class<Mood> gameplay.ai.mood.Mood.getPreviousMood () [inherited]

When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

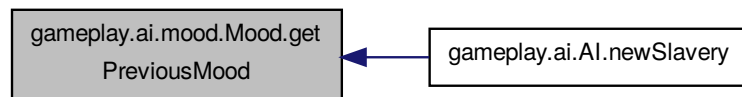
Definition at line 148 of file Mood.java.

Referenced by `gameplay.ai.AI.newSlavery()`.

```

148         {
149             Iterator<ProbabilityMood> it = previousMoods.iterator();
150             ProbabilityMood probMood = null;
151             while(it.hasNext()) {
152                 probMood = it.next();
153                 if(Math.random() >= probMood.getProbability()) {
154                     return probMood.getMood();
155                 }
156             }
157         }
158     }
159     return probMood.getMood();
160 }
161 }
  
```

Here is the caller graph for this function:



7.3.3.6 int gameplay.ai.mood.Mood.getSlaveryThreshold () [inherited]

Getter for the slavery threshold field.

See Also

[slaveryThreshold](#)

Definition at line 87 of file Mood.java.

References `gameplay.ai.mood.Mood.slaveryThreshold`.

```

87                                     {
88         return slaveryThreshold;
89     }

```

7.3.3.7 int gameplay.ai.mood.Mood.incrementBetrayalCount () [inherited]

Increment and return the betrayal count field.

See Also

[moodsBetrayalCount](#)

Definition at line 111 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

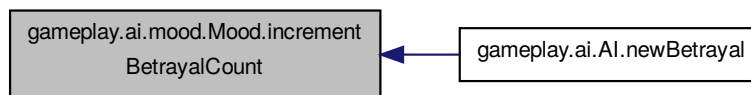
Referenced by gameplay.ai.AI.newBetrayal().

```

111                                     {
112         return ++moodsBetrayalCount;
113     }

```

Here is the caller graph for this function:



7.3.3.8 int gameplay.ai.mood.Mood.incrementSlaveryCount () [inherited]

Increment and return the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 119 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

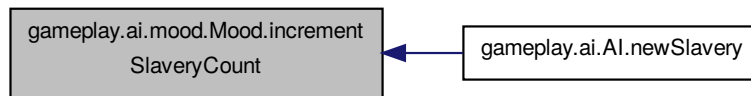
Referenced by gameplay.ai.AI.newSlavery().

```

119                                     {
120         return ++moodsSlaveryCount;
121     }

```


Here is the caller graph for this function:



7.3.4 Member Data Documentation

7.3.4.1 `int gameplay.ai.mood.Mood.betrayalThreshold` `[protected]`, `[inherited]`

If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 20 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getBetrayalThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

7.3.4.2 `int gameplay.ai.mood.Mood.moodsBetrayalCount` `[protected]`, `[inherited]`

Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.

Definition at line 37 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsBetrayalCount()`, and `gameplay.ai.mood.Mood.incrementBetrayalCount()`.

7.3.4.3 `int gameplay.ai.mood.Mood.moodsSlaveryCount` `[protected]`, `[inherited]`

Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.

Definition at line 44 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsSlaveryThreshold()`, and `gameplay.ai.mood.Mood.incrementSlaveryCount()`.

7.3.4.4 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.nextMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.

Definition at line 53 of file Mood.java.

7.3.4.5 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.previousMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

Definition at line 61 of file Mood.java.

7.3.4.6 `int gameplay.ai.mood.Mood.slaveryThreshold` `[protected]`, `[inherited]`

If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 28 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getSlaveryThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

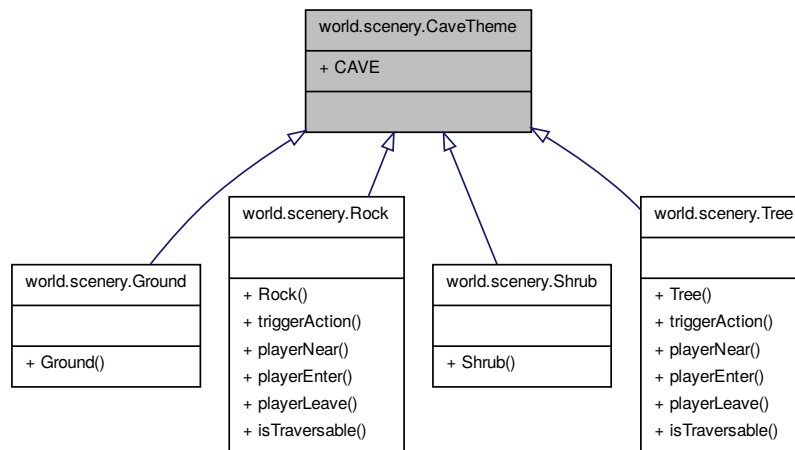
The documentation for this class was generated from the following file:

- `gameplay/ai/mood/Anger.java`

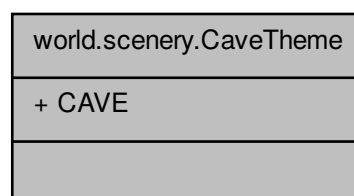
7.4 `world.scenery.CaveTheme` Interface Reference

Interface used to tell that a `Scenery` object has a Cave style.

Inheritance diagram for `world.scenery.CaveTheme`:



Collaboration diagram for `world.scenery.CaveTheme`:



Static Public Attributes

- static String [CAVE](#) = "cave"

String used to tell in which file is the sprite matching the Cave style.

7.4.1 Detailed Description

Interface used to tell that a [Scenery](#) object has a Cave style.

Definition at line 7 of file CaveTheme.java.

7.4.2 Member Data Documentation

7.4.2.1 String world.scenery.CaveTheme.CAVE = "cave" [static]

String used to tell in which file is the sprite matching the Cave style.

Definition at line 13 of file CaveTheme.java.

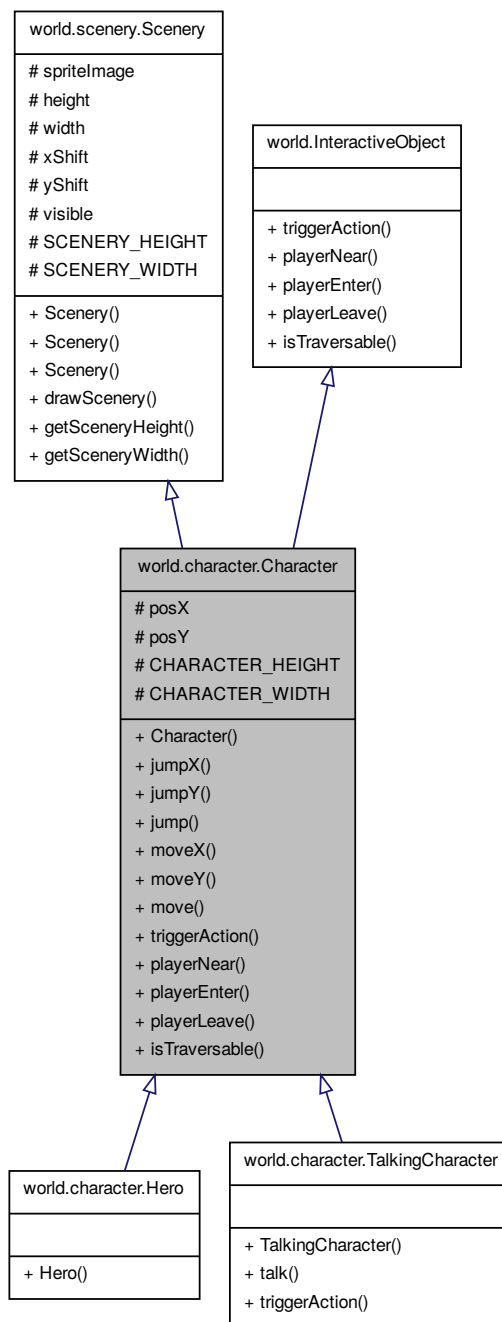
The documentation for this interface was generated from the following file:

- world/scenery/[CaveTheme.java](#)

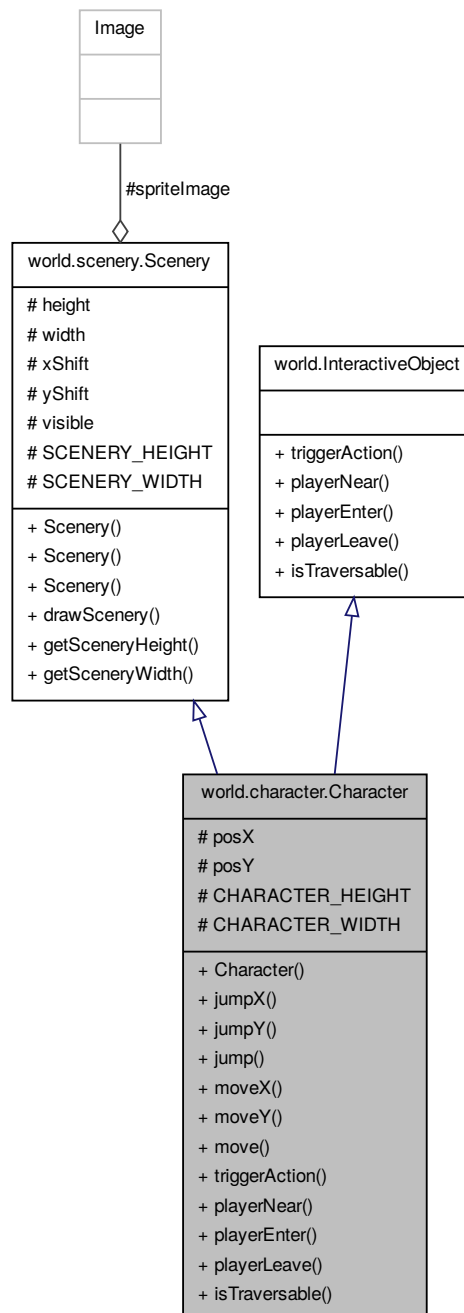
7.5 world.character.Character Class Reference

Class used to handle a character (player or non-player)

Inheritance diagram for world.character.Character:



Collaboration diagram for world.character.Character:



Public Member Functions

- [Character](#) (final String imagePath)
Character class constructor.
- void [jumpX](#) (int distance)
Teleport the character horizontally regarding his current position.
- void [jumpY](#) (int distance)

- *Teleport the character vertically regarding his current position.*
- void [jump](#) (int distanceX, int distanceY)
 - *Teleport the character regarding his current position.*
- void [moveX](#) (int distance)
 - *Slowly horizontally move the character to a certain point regarding his current position.*
- void [moveY](#) (int distance)
 - *Slowly vertically move the character to a certain point regarding his current position.*
- void [move](#) (int distanceX, int distanceY)
 - *Slowly move the character to a certain point regarding his current position.*
- void [triggerAction](#) ()
 - *Method to be called when the player is near the object and presses the action key.*
- void [playerNear](#) ()
 - *Method to be called when the player is near the object.*
- void [playerEnter](#) ()
 - *Method to be called when the player was near and is now over the object.*
- void [playerLeave](#) ()
 - *Method to be called when the player was over and is now near the object.*
- boolean [isTraversable](#) ()
 - *Returns true if the player is able to pass through the object and false if the player is not able to cross the object.*
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
 - *Getter for the SCENERY_HEIGHT static field.*
- static int [getSceneryWidth](#) ()
 - *Getter for the SCENERY_WIDTH static field.*

Protected Attributes

- int [posX](#)
 - *Horizontal position of the character.*
- int [posY](#)
 - *Vertical position of the character.*
- Image [spriteImage](#)
 - *Image graphically representing the object.*
- int [height](#)
 - *Image height.*
- int [width](#)
 - *Image width.*
- int [xShift](#)
 - *Horizontal shifting for the image.*
- int [yShift](#)
 - *Vertical shifting for the image.*
- boolean [visible](#)
 - *Boolean equals to true if the scenery is visible, false if the scenery is not.*

Static Protected Attributes

- static final int `CHARACTER_HEIGHT` = 300
Height of the character's image (fixed)
- static final int `CHARACTER_WIDTH` = 170
Width of the character's image (fixed)
- static int `SCENERY_HEIGHT` = 48
Height of the graphical representation of a [Scenery](#) object.
- static int `SCENERY_WIDTH` = 48
Width of the graphical representation of a [Scenery](#) object.

7.5.1 Detailed Description

Class used to handle a character (player or non-player)

A character can be a purely decorative character or a talking character or a player character (the hero)

Definition at line 13 of file Character.java.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 world.character.Character.Character (final String *imagePath*)

[Character](#) class constructor.

Parameters

<i>imagePath</i>	Image filename representing the character
------------------	-------------------------------------------

Definition at line 39 of file Character.java.

References `world.character.Character.CHARACTER_HEIGHT`, and `world.character.Character.CHARACTER_WIDTH`.

```

39         {
40             Philophobia.getVerbose().calls("Character class created", "world/characters/Character.java", "
Character.Character(String)");
41
42             super(image, CHARACTER_HEIGHT, CHARACTER_WIDTH);
43         }
44     }
```

7.5.3 Member Function Documentation

7.5.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
yShift, width, height, obs);
136
137         visible = true;
138     }
```

7.5.3.2 static int world.scenery.Scenery.getSceneryHeight () [static],[inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

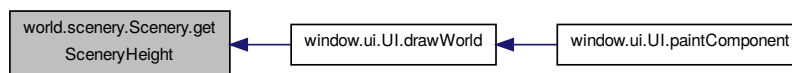
References world.scenery.Scenery.SCENERY_HEIGHT.

Referenced by window.ui.UI.drawWorld().

```

144                                     {
145     return SCENERY_HEIGHT;
146 }
```

Here is the caller graph for this function:



7.5.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```

152                                     {
153     return SCENERY_WIDTH;
154 }
```

7.5.3.4 boolean world.character.Character.isTraversable ()

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 161 of file Character.java.

```

161                                     {
162     return false;
163 }
```

7.5.3.5 void world.character.Character.jump (int distanceX, int distanceY)

Teleport the character regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be teleported
<i>distanceY</i>	Vertical distance toward the character will be teleported

Definition at line 73 of file Character.java.

```

73                                     {
74         if(distanceX != 0 || distanceY != 0) {
75             this.posX = this.posX + distance;
76             this.posY = this.posY + distance;
77             this.draw();
78         }
79     }

```

7.5.3.6 void world.character.Character.jumpX (int *distance*)

Teleport the character horizontally regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be teleported
-----------------	-------------------------------------------------------------

Definition at line 50 of file Character.java.

```

50                                     {
51         if(distance != 0) {
52             this.posX = this.posX + distance;
53             this.draw();
54         }
55     }

```

7.5.3.7 void world.character.Character.jumpY (int *distance*)

Teleport the character vertically regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be teleported
-----------------	-----------------------------------------------------------

Definition at line 61 of file Character.java.

```

61                                     {
62         if(distance != 0) {
63             this.posY = this.posY + distance;
64             this.draw();
65         }
66     }

```

7.5.3.8 void world.character.Character.move (int *distanceX*, int *distanceY*)

Slowly move the character to a certain point regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be moved
<i>distanceY</i>	Vertical distance toward the character will be moved

Definition at line 118 of file Character.java.

References world.character.Character.moveX(), and world.character.Character.moveY().

```

118                                     {
119
120         if(distanceX != 0 || distanceY != 0) {

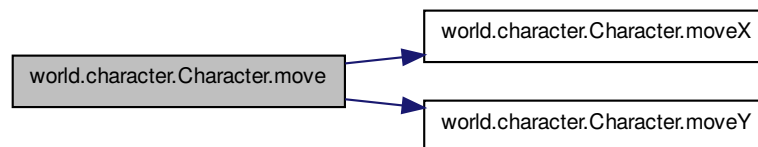
```

```

121         int stepX = (distanceX < 0)? -1 : 1;
122         int stepY = (distanceY < 0)? -1 : 1;
123
124         this.jump(stepX, stepY);
125
126         if(distanceX - stepX == 0) {
127             moveY(distanceY - stepY);
128             return;
129         }
130
131         if(distanceY - stepY == 0) {
132             moveX(distanceX - stepX);
133             return;
134         }
135
136         move(distanceX - stepX, distanceY - stepY);
137     }
138
139 }

```

Here is the call graph for this function:



7.5.3.9 void world.character.Character.moveX (int *distance*)

Slowly horizontally move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be moved
-----------------	--------------------------------------------------------

Definition at line 85 of file `Character.java`.

Referenced by `world.character.Character.move()`.

```

85         {
86
87         if(distance != 0) {
88             int step = (distance < 0)? -1 : 1;
89
90             this.jumpX(step);
91
92             this.moveX(distance - step);
93         }
94
95     }

```

Here is the caller graph for this function:



7.5.3.10 void world.character.Character.moveY (int *distance*)

Slowly vertically move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be moved
-----------------	------------------------------------------------------

Definition at line 101 of file Character.java.

Referenced by world.character.Character.move().

```

101                                     {
102
103         if(distance != 0) {
104             int step = (distance < 0)? -1 : 1;
105
106             this.jumpY(step);
107
108             this.moveY(distance - step);
109         }
110
111     }
```

Here is the caller graph for this function:

**7.5.3.11 void world.character.Character.playerEnter ()**

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 153 of file Character.java.

```

153                                     {
154
155     }
```

7.5.3.12 void world.character.Character.playerLeave ()

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 157 of file Character.java.

```

157                                     {
158
159     }
```

7.5.3.13 void world.character.Character.playerNear ()

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 149 of file Character.java.

```
149         {  
150  
151     }
```

7.5.3.14 void world.character.Character.triggerAction ()

Method to be called when the player is near the object and presses the action key.

Implements [world.InteractiveObject](#).

Definition at line 145 of file Character.java.

```
145         {  
146  
147     }
```

7.5.4 Member Data Documentation

7.5.4.1 final int world.character.Character.CHARACTER_HEIGHT = 300 [static], [protected]

Height of the character's image (fixed)

Definition at line 18 of file Character.java.

Referenced by world.character.Character.Character().

7.5.4.2 final int world.character.Character.CHARACTER_WIDTH = 170 [static], [protected]

Width of the character's image (fixed)

Definition at line 23 of file Character.java.

Referenced by world.character.Character.Character().

7.5.4.3 int world.scenery.Scenery.height [protected], [inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.5.4.4 int world.character.Character.posX [protected]

Horizontal position of the character.

Definition at line 28 of file Character.java.

7.5.4.5 int world.character.Character.posY [protected]

Vertical position of the character.

Definition at line 33 of file Character.java.

7.5.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48` `[static], [protected], [inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.5.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static], [protected], [inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.5.4.8 `Image world.scenery.Scenery.spriteImage` `[protected], [inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.5.4.9 `boolean world.scenery.Scenery.visible` `[protected], [inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.5.4.10 `int world.scenery.Scenery.width` `[protected], [inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.5.4.11 `int world.scenery.Scenery.xShift` `[protected], [inherited]`

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.5.4.12 `int world.scenery.Scenery.yShift` `[protected], [inherited]`

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

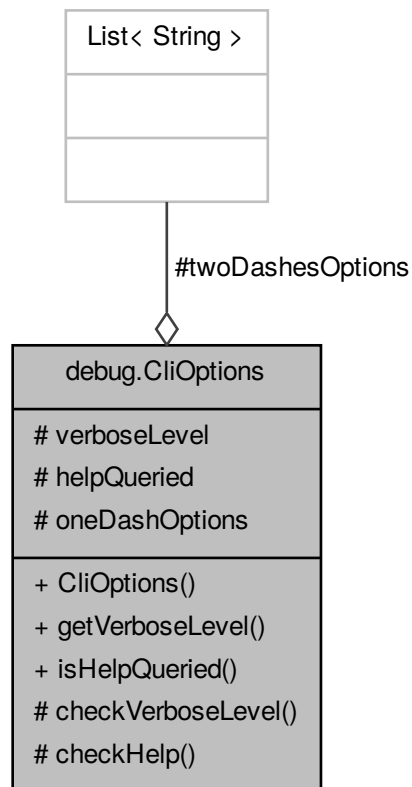
The documentation for this class was generated from the following file:

- `world/character/Character.java`

7.6 debug.CliOptions Class Reference

Class used to analyze the commands passed to the program.

Collaboration diagram for debug.CliOptions:



Public Member Functions

- [CliOptions](#) (String[] args)
Constructor of the CliOption.
- int [getVerboseLevel](#) ()
Return the verbose level passed as options to the program.
- boolean [isHelpQueried](#) ()
Return true if the help has been queried or false if not.

Protected Member Functions

- void [checkVerboseLevel](#) ()
Check the verbose level considering the options with one dash.
- void [checkHelp](#) ()
Check if the help mode has been queried considering the options with on or two dashes.

Protected Attributes

- int [verboseLevel](#)
Level of verbosing detected due to the args passed to the program.
- boolean [helpQueried](#)
Field equal to true if the help has been queried, false if not.
- String [oneDashOptions](#)
One letter options passed to the program (corresponding to "one dash" options)
- List< String > [twoDashesOptions](#)
List of two dashes options passed to the program.

7.6.1 Detailed Description

Class used to analyze the commands passed to the program.

For now, the verbose "-v" can be passed to the program (once or several times) and the help (containing the list of options available) can be queried via `--help` or `-h`

You can set a level of verbose mode from 0 to 5, you just have to put `-v` for verbose mode level 1, `-vv` for level 2 and so on

Definition at line 18 of file `CliOptions.java`.

7.6.2 Constructor & Destructor Documentation

7.6.2.1 `debug.CliOptions.CliOptions (String[] args)`

Constructor of the `CliOption`.

Parameters

<i>args</i>	Options passed to the program when launching it
-------------	-------------------------------------------------

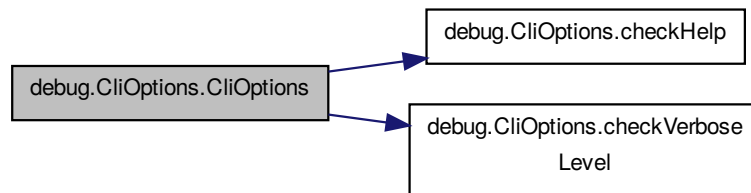
Definition at line 52 of file `CliOptions.java`.

References `debug.CliOptions.checkHelp()`, `debug.CliOptions.checkVerboseLevel()`, `debug.CliOptions.helpQueried`, `debug.CliOptions.oneDashOptions`, `debug.CliOptions.twoDashesOptions`, and `debug.CliOptions.verboseLevel`.

```

52                                     {
53
54         oneDashOptions = new String();
55         twoDashesOptions = new LinkedList<String>();
56
57         verboseLevel = 0;
58         helpQueried = false;
59
60         // Check the arguments passed to the program
61         for(String str : args) {
62             // Check if an option is set
63             if(str.charAt(0) == '-' && str.length() > 1) {
64
65                 // This program support the fact that an option can be prefixed with either '-' or '--'
66                 if(str.charAt(1) == '-' && str.length() > 2) {
67
68                     twoDashesOptions.add(str.substring(2));
69
70                 } else {
71
72                     oneDashOptions += str.substring(1);
73
74                 }
75
76             }
77         }
78     }
79
80     checkVerboseLevel();
81     checkHelp();
82
83 }
```

Here is the call graph for this function:



7.6.3 Member Function Documentation

7.6.3.1 void debug.CliOptions.checkHelp () [protected]

Check if the help mode has been queried considering the options with on or two dashes.

See Also

[helpQueried](#)

Definition at line 114 of file `CliOptions.java`.

References `debug.CliOptions.helpQueried`, and `debug.CliOptions.oneDashOptions`.

Referenced by `debug.CliOptions.CliOptions()`.

```

114         {
115
116         helpQueried = false;
117
118         int oneDashOptionsLength = oneDashOptions.length();
119
120         for(int i = 0 ; i < oneDashOptionsLength ; ++i) {
121             if(oneDashOptions.charAt(i) == 'h') {
122                 helpQueried = true;
123                 return;
124             }
125         }
126
127         ListIterator li = twoDashesOptions.listIterator();
128
129         while(li.hasNext()) {
130             if(li.next().equals("help")) {
131                 helpQueried = true;
132                 return;
133             }
134         }
135
136     }
  
```

Here is the caller graph for this function:



7.6.3.2 void debug.CliOptions.checkVerboseLevel () [protected]

Check the verbose level considering the options with one dash.

See Also

[verboseLevel](#)

Definition at line 89 of file CliOptions.java.

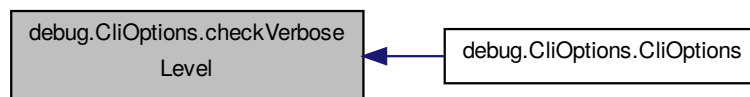
References `debug.CliOptions.oneDashOptions`, and `debug.CliOptions.verboseLevel`.

Referenced by `debug.CliOptions.CliOptions()`.

```

89                                     {
90     verboseLevel = 0;
91
92     // Due to performance reasons, the length of the arguments are stored in a variable
93     int oneDashOptionsLength = oneDashOptions.length();
94
95     for(int i = 0 ; i < oneDashOptionsLength && verboseLevel < 5 ; ++i) {
96         if(oneDashOptions.charAt(i) == 'v')
97             ++verboseLevel;
98     }
99
100 }
```

Here is the caller graph for this function:



7.6.3.3 int debug.CliOptions.getVerboseLevel ()

Return the verbose level passed as options to the program.

Returns

The verbose level

Definition at line 106 of file CliOptions.java.

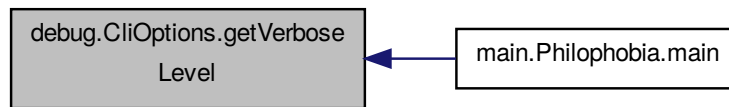
References `debug.CliOptions.verboseLevel`.

Referenced by `main.Philophobia.main()`.

```

106                                     {
107     return verboseLevel;
108 }
```

Here is the caller graph for this function:



7.6.3.4 boolean debug.CliOptions.isHelpQueried ()

Return true if the help has been queried or false if not.

Returns

the helpQueried field

Definition at line 142 of file CliOptions.java.

References debug.CliOptions.helpQueried.

Referenced by main.Philophobia.main().

```

142         {
143             return helpQueried;
144         }
  
```

Here is the caller graph for this function:



7.6.4 Member Data Documentation

7.6.4.1 boolean debug.CliOptions.helpQueried [protected]

Field equal to true if the help has been queried, false if not.

If the help has been queried, then the game doesn't launch but instead print all the options that can be passed to the program

Definition at line 34 of file CliOptions.java.

Referenced by debug.CliOptions.checkHelp(), debug.CliOptions.CliOptions(), and debug.CliOptions.isHelpQueried().

7.6.4.2 String debug.CliOptions.oneDashOptions [protected]

One letter options passed to the program (corresponding to "one dash" options)

Definition at line 40 of file CliOptions.java.

Referenced by debug.CliOptions.checkHelp(), debug.CliOptions.checkVerboseLevel(), and debug.CliOptions.CliOptions().

7.6.4.3 List<String> debug.CliOptions.twoDashesOptions [protected]

List of two dashes options passed to the program.

Definition at line 45 of file CliOptions.java.

Referenced by debug.CliOptions.CliOptions().

7.6.4.4 int debug.CliOptions.verboseLevel [protected]

Level of verbosing detected due to the args passed to the program.

See Also

[Verbose](#)
[Verbose::verboseMode](#)

Definition at line 26 of file CliOptions.java.

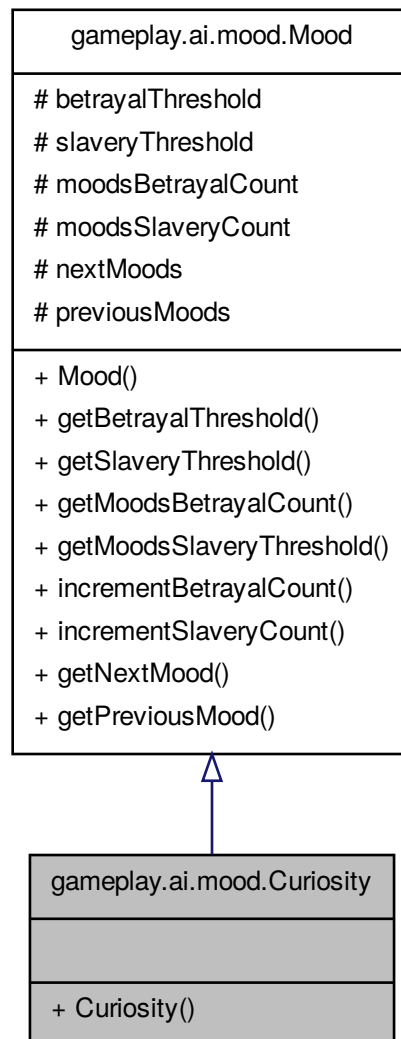
Referenced by debug.CliOptions.checkVerboseLevel(), debug.CliOptions.CliOptions(), and debug.CliOptions.getVerboseLevel().

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

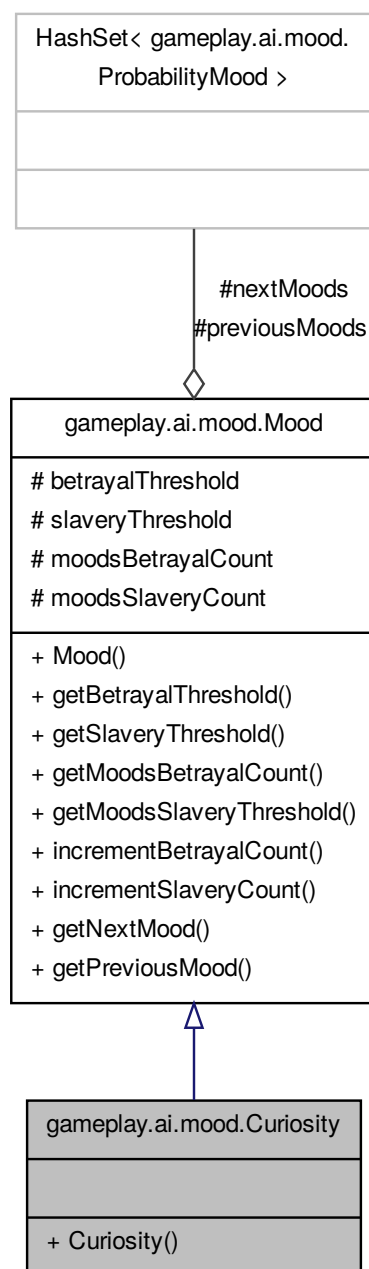
- [debug/CliOptions.java](#)

7.7 gameplay.ai.mood.Curiosity Class Reference

Inheritance diagram for gameplay.ai.mood.Curiosity:



Collaboration diagram for gameplay.ai.mood.Curiosity:



Public Member Functions

- [Curiosity \(\)](#)
- [int getBetrayalThreshold \(\)](#)
Getter for the betrayal threshold field.
- [int getSlaveryThreshold \(\)](#)
Getter for the slavery threshold field.

- int [getMoodsBetrayalCount](#) ()
Getter for the betrayal count field.
- int [getMoodsSlaveryThreshold](#) ()
Getter for the slavery count field.
- int [incrementBetrayalCount](#) ()
Increment and return the betrayal count field.
- int [incrementSlaveryCount](#) ()
Increment and return the slavery count field.
- Class< [Mood](#) > [getNextMood](#) ()
When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.
- Class< [Mood](#) > [getPreviousMood](#) ()
When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Protected Attributes

- int [betrayalThreshold](#)
If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [slaveryThreshold](#)
If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [moodsBetrayalCount](#)
Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.
- int [moodsSlaveryCount](#)
Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.
- HashSet< [ProbabilityMood](#) > [nextMoods](#)
Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.
- HashSet< [ProbabilityMood](#) > [previousMoods](#)
Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

7.7.1 Detailed Description

Definition at line 5 of file Curiosity.java.

7.7.2 Constructor & Destructor Documentation

7.7.2.1 `gameplay.ai.mood.Curiosity.Curiosity ()`

Definition at line 7 of file Curiosity.java.

```

7         {
8             super(10, 6);
9         }
```

7.7.3 Member Function Documentation

7.7.3.1 `int gameplay.ai.mood.Mood.getBetrayalThreshold ()` `[inherited]`

Getter for the betrayal threshold field.

See Also

[betrayalThreshold](#)

Definition at line 79 of file Mood.java.

References gameplay.ai.mood.Mood.betrayalThreshold.

```

79         {
80             return betrayalThreshold;
81         }

```

7.7.3.2 int gameplay.ai.mood.Mood.getMoodsBetrayalCount () [inherited]

Getter for the betrayal count field.

See Also

[moodsBetrayalCount](#)

Definition at line 95 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

```

95         {
96             return moodsBetrayalCount;
97         }

```

7.7.3.3 int gameplay.ai.mood.Mood.getMoodsSlaveryThreshold () [inherited]

Getter for the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 103 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

```

103         {
104             return moodsSlaveryCount;
105         }

```

7.7.3.4 Class<Mood> gameplay.ai.mood.Mood.getNextMood () [inherited]

When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 128 of file Mood.java.

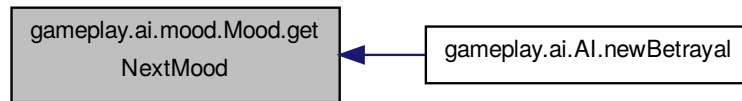
Referenced by gameplay.ai.AI.newBetrayal().

```

128         {
129             Iterator<ProbabilityMood> it = nextMoods.iterator();
130             ProbabilityMood probMood = null;
131             while(it.hasNext()) {
132                 probMood = it.next();
133                 if(Math.random() >= probMood.getProbability()) {
134                     return probMood.getMood();
135                 }
136             }
137         }
138     }
139
140     return probMood.getMood();
141 }

```

Here is the caller graph for this function:



7.7.3.5 Class<Mood> gameplay.ai.mood.Mood.getPreviousMood () [inherited]

When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

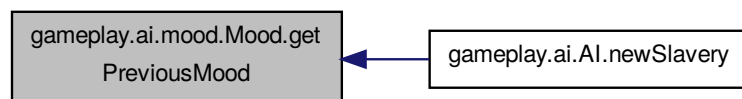
Definition at line 148 of file Mood.java.

Referenced by `gameplay.ai.AI.newSlavery()`.

```

148         {
149             Iterator<ProbabilityMood> it = previousMoods.iterator();
150             ProbabilityMood probMood = null;
151             while(it.hasNext()) {
152                 probMood = it.next();
153                 if(Math.random() >= probMood.getProbability()) {
154                     return probMood.getMood();
155                 }
156             }
157         }
158     }
159     return probMood.getMood();
160 }
161 }
  
```

Here is the caller graph for this function:



7.7.3.6 int gameplay.ai.mood.Mood.getSlaveryThreshold () [inherited]

Getter for the slavery threshold field.

See Also

[slaveryThreshold](#)

Definition at line 87 of file Mood.java.

References `gameplay.ai.mood.Mood.slaveryThreshold`.


```
87                                     {  
88         return slaveryThreshold;  
89     }
```

7.7.3.7 int gameplay.ai.mood.Mood.incrementBetrayalCount () [inherited]

Increment and return the betrayal count field.

See Also

[moodsBetrayalCount](#)

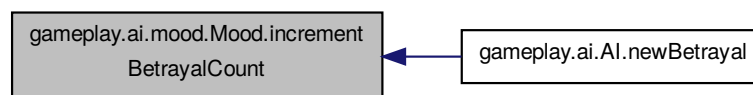
Definition at line 111 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

Referenced by gameplay.ai.AI.newBetrayal().

```
111                                     {  
112         return ++moodsBetrayalCount;  
113     }
```

Here is the caller graph for this function:



7.7.3.8 int gameplay.ai.mood.Mood.incrementSlaveryCount () [inherited]

Increment and return the slavery count field.

See Also

[moodsSlaveryCount](#)

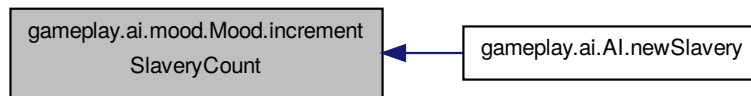
Definition at line 119 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

Referenced by gameplay.ai.AI.newSlavery().

```
119                                     {  
120         return ++moodsSlaveryCount;  
121     }
```

Here is the caller graph for this function:



7.7.4 Member Data Documentation

7.7.4.1 `int gameplay.ai.mood.Mood.betrayalThreshold` `[protected]`, `[inherited]`

If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 20 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getBetrayalThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

7.7.4.2 `int gameplay.ai.mood.Mood.moodsBetrayalCount` `[protected]`, `[inherited]`

Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.

Definition at line 37 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsBetrayalCount()`, and `gameplay.ai.mood.Mood.incrementBetrayalCount()`.

7.7.4.3 `int gameplay.ai.mood.Mood.moodsSlaveryCount` `[protected]`, `[inherited]`

Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.

Definition at line 44 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsSlaveryThreshold()`, and `gameplay.ai.mood.Mood.incrementSlaveryCount()`.

7.7.4.4 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.nextMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.

Definition at line 53 of file Mood.java.

7.7.4.5 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.previousMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

Definition at line 61 of file Mood.java.

7.7.4.6 int gameplay.ai.mood.Mood.slaveryThreshold [protected],[inherited]

If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 28 of file Mood.java.

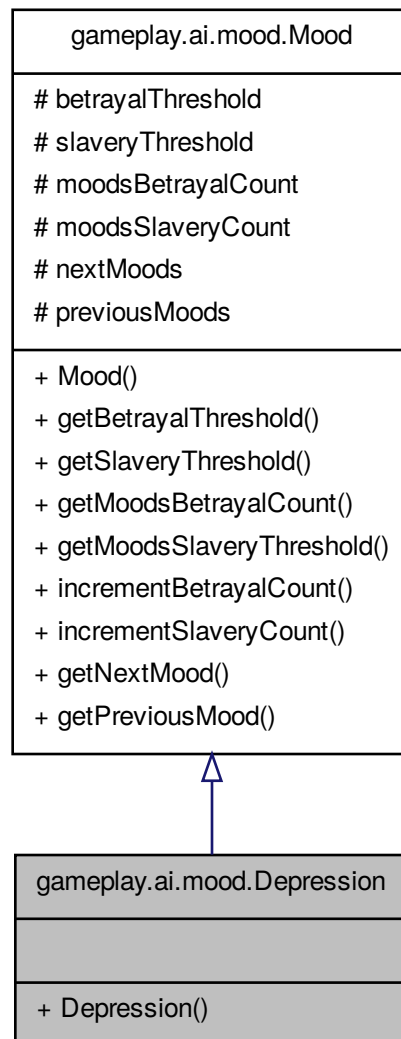
Referenced by gameplay.ai.mood.Mood.getSlaveryThreshold(), and gameplay.ai.mood.Mood.Mood().

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

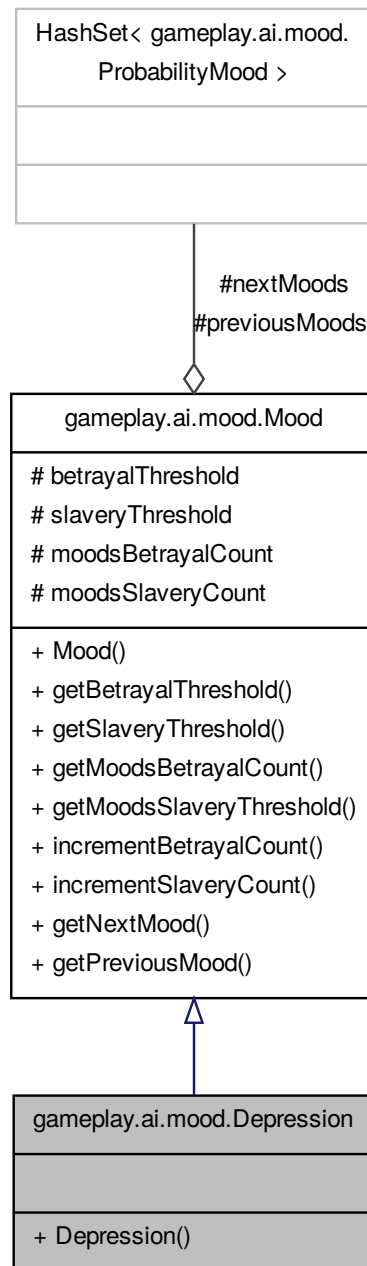
- gameplay/ai/mood/[Curiosity.java](#)

7.8 gameplay.ai.mood.Depression Class Reference

Inheritance diagram for gameplay.ai.mood.Depression:



Collaboration diagram for gameplay.ai.mood.Depression:



Public Member Functions

- [Depression](#) ()
- int [getBetrayalThreshold](#) ()
Getter for the betrayal threshold field.
- int [getSlaveryThreshold](#) ()
Getter for the slavery threshold field.

- int [getMoodsBetrayalCount](#) ()
Getter for the betrayal count field.
- int [getMoodsSlaveryThreshold](#) ()
Getter for the slavery count field.
- int [incrementBetrayalCount](#) ()
Increment and return the betrayal count field.
- int [incrementSlaveryCount](#) ()
Increment and return the slavery count field.
- Class< [Mood](#) > [getNextMood](#) ()
When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.
- Class< [Mood](#) > [getPreviousMood](#) ()
When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Protected Attributes

- int [betrayalThreshold](#)
If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [slaveryThreshold](#)
If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int [moodsBetrayalCount](#)
Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.
- int [moodsSlaveryCount](#)
Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.
- HashSet< [ProbabilityMood](#) > [nextMoods](#)
Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.
- HashSet< [ProbabilityMood](#) > [previousMoods](#)
Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

7.8.1 Detailed Description

Definition at line 5 of file Depression.java.

7.8.2 Constructor & Destructor Documentation

7.8.2.1 `gameplay.ai.mood.Depression.Depression ()`

Definition at line 7 of file Depression.java.

```

7         {
8             super(10, 6);
9         }
```

7.8.3 Member Function Documentation

7.8.3.1 `int gameplay.ai.mood.Mood.getBetrayalThreshold ()` [inherited]

Getter for the betrayal threshold field.

See Also[betrayalThreshold](#)

Definition at line 79 of file Mood.java.

References gameplay.ai.mood.Mood.betrayalThreshold.

```

79         {
80             return betrayalThreshold;
81         }

```

7.8.3.2 int gameplay.ai.mood.Mood.getMoodsBetrayalCount () [inherited]

Getter for the betrayal count field.

See Also[moodsBetrayalCount](#)

Definition at line 95 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

```

95         {
96             return moodsBetrayalCount;
97         }

```

7.8.3.3 int gameplay.ai.mood.Mood.getMoodsSlaveryThreshold () [inherited]

Getter for the slavery count field.

See Also[moodsSlaveryCount](#)

Definition at line 103 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

```

103         {
104             return moodsSlaveryCount;
105         }

```

7.8.3.4 Class<Mood> gameplay.ai.mood.Mood.getNextMood () [inherited]

When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 128 of file Mood.java.

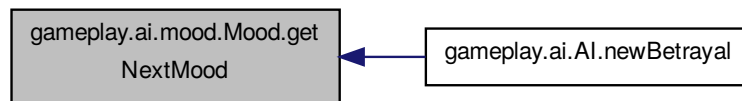
Referenced by gameplay.ai.AI.newBetrayal().

```

128         {
129             Iterator<ProbabilityMood> it = nextMoods.iterator();
130             ProbabilityMood probMood = null;
131             while(it.hasNext()) {
132                 probMood = it.next();
133                 if(Math.random() >= probMood.getProbability()) {
134                     return probMood.getMood();
135                 }
136             }
137         }
138     }
139
140     return probMood.getMood();
141 }

```

Here is the caller graph for this function:



7.8.3.5 Class<Mood> gameplay.ai.mood.Mood.getPreviousMood () [inherited]

When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

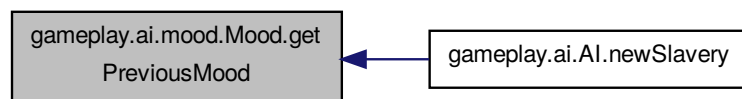
Definition at line 148 of file Mood.java.

Referenced by `gameplay.ai.AI.newSlavery()`.

```

148         {
149             Iterator<ProbabilityMood> it = previousMoods.iterator();
150             ProbabilityMood probMood = null;
151             while(it.hasNext()) {
152                 probMood = it.next();
153                 if(Math.random() >= probMood.getProbability()) {
154                     return probMood.getMood();
155                 }
156             }
157         }
158     }
159     return probMood.getMood();
160 }
161 }
  
```

Here is the caller graph for this function:



7.8.3.6 int gameplay.ai.mood.Mood.getSlaveryThreshold () [inherited]

Getter for the slavery threshold field.

See Also

[slaveryThreshold](#)

Definition at line 87 of file Mood.java.

References `gameplay.ai.mood.Mood.slaveryThreshold`.

```

87                                     {
88         return slaveryThreshold;
89     }

```

7.8.3.7 int gameplay.ai.mood.Mood.incrementBetrayalCount () [inherited]

Increment and return the betrayal count field.

See Also

[moodsBetrayalCount](#)

Definition at line 111 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

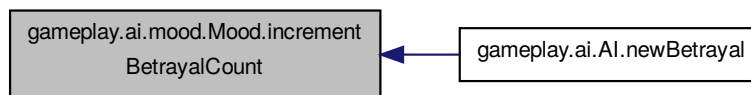
Referenced by gameplay.ai.AI.newBetrayal().

```

111                                     {
112         return ++moodsBetrayalCount;
113     }

```

Here is the caller graph for this function:



7.8.3.8 int gameplay.ai.mood.Mood.incrementSlaveryCount () [inherited]

Increment and return the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 119 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

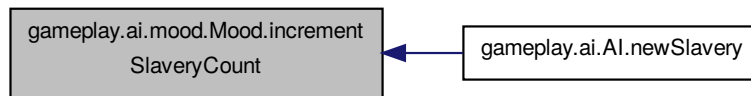
Referenced by gameplay.ai.AI.newSlavery().

```

119                                     {
120         return ++moodsSlaveryCount;
121     }

```


Here is the caller graph for this function:



7.8.4 Member Data Documentation

7.8.4.1 `int gameplay.ai.mood.Mood.betrayalThreshold` `[protected]`, `[inherited]`

If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 20 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getBetrayalThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

7.8.4.2 `int gameplay.ai.mood.Mood.moodsBetrayalCount` `[protected]`, `[inherited]`

Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.

Definition at line 37 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsBetrayalCount()`, and `gameplay.ai.mood.Mood.incrementBetrayalCount()`.

7.8.4.3 `int gameplay.ai.mood.Mood.moodsSlaveryCount` `[protected]`, `[inherited]`

Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.

Definition at line 44 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsSlaveryThreshold()`, and `gameplay.ai.mood.Mood.incrementSlaveryCount()`.

7.8.4.4 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.nextMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.

Definition at line 53 of file Mood.java.

7.8.4.5 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.previousMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

Definition at line 61 of file Mood.java.

7.8.4.6 `int gameplay.ai.mood.Mood.slaveryThreshold` `[protected]`, `[inherited]`

If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 28 of file `Mood.java`.

Referenced by `gameplay.ai.mood.Mood.getSlaveryThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

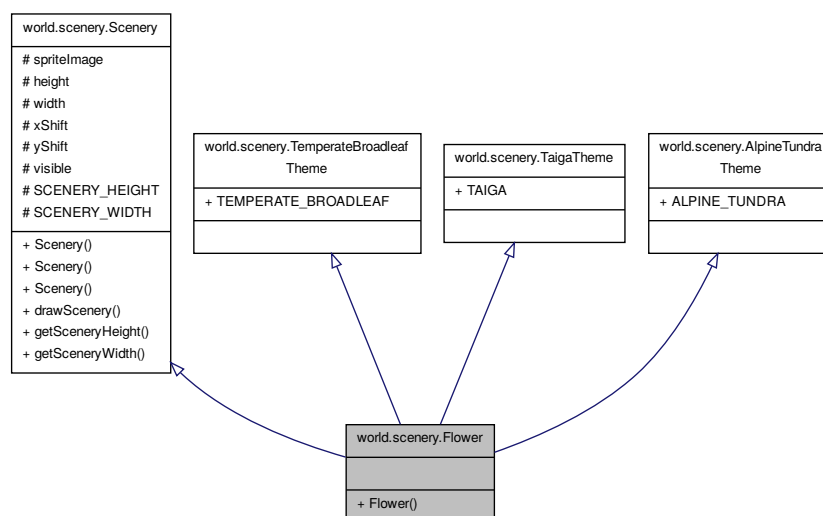
The documentation for this class was generated from the following file:

- `gameplay/ai/mood/Depression.java`

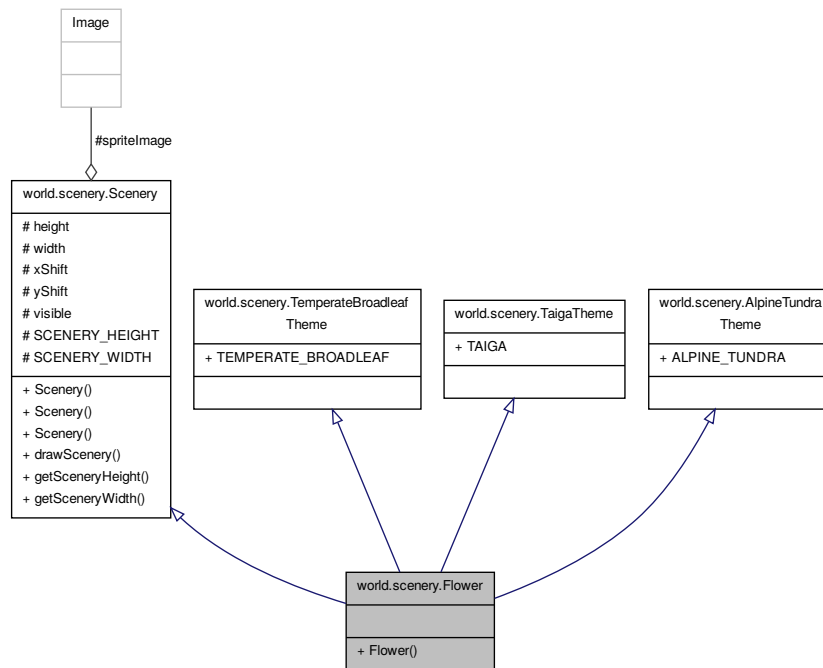
7.9 world.scenery.Flower Class Reference

Class used to handle a flower object.

Inheritance diagram for `world.scenery.Flower`:



Collaboration diagram for world.scenery.Flower:



Public Member Functions

- [Flower](#) (final String type)
Flower class constructor.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [ALPINE_TUNDRA](#) = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.

Protected Attributes

- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.9.1 Detailed Description

Class used to handle a flower object.

Definition at line 13 of file Flower.java.

7.9.2 Constructor & Destructor Documentation

7.9.2.1 world.scenery.Flower.Flower (final String type)

[Flower](#) class constructor.

Parameters

<i>type</i>	Style of the flower
-------------	---------------------

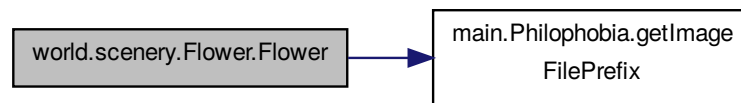
Definition at line 19 of file Flower.java.

References `main.Philophobia.getImageFilePrefix()`.

```

19         {
20
21             super(Philophobia.getImageFilePrefix() + type + "flower.png");
22
23         }
```

Here is the call graph for this function:



7.9.3 Member Function Documentation

7.9.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References world.scenery.Scenery.height, world.scenery.Scenery.spriteImage, world.scenery.Scenery.visible, world.scenery.Scenery.width, world.scenery.Scenery.xShift, and world.scenery.Scenery.yShift.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }
  
```

7.9.3.2 static int world.scenery.Scenery.getSceneryHeight () [static],[inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

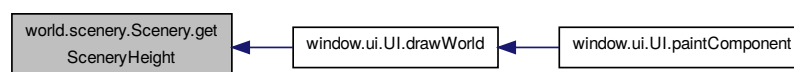
References world.scenery.Scenery.SCENERY_HEIGHT.

Referenced by window.ui.UI.drawWorld().

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
  
```

Here is the caller graph for this function:



7.9.3.3 `static int world.scenery.Scenery.getSceneryWidth ()` `[static]`, `[inherited]`

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_WIDTH`.

```
152                                     {
153         return SCENERY_WIDTH;
154     }
```

7.9.4 Member Data Documentation

7.9.4.1 `String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra"` `[static]`, `[inherited]`

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file AlpineTundraTheme.java.

7.9.4.2 `int world.scenery.Scenery.height` `[protected]`, `[inherited]`

Image height.

Definition at line 39 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.9.4.3 `int world.scenery.Scenery.SCENERY_HEIGHT = 48` `[static]`, `[protected]`, `[inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.9.4.4 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static]`, `[protected]`, `[inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.9.4.5 `Image world.scenery.Scenery.spriteImage` `[protected]`, `[inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.9.4.6 `String world.scenery.TaigaTheme.TAIGA = "taiga"` `[static]`, `[inherited]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.9.4.7 `String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

7.9.4.8 `boolean world.scenery.Scenery.visible` `[protected], [inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.9.4.9 `int world.scenery.Scenery.width` `[protected], [inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.9.4.10 `int world.scenery.Scenery.xShift` `[protected], [inherited]`

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.9.4.11 `int world.scenery.Scenery.yShift` `[protected], [inherited]`

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

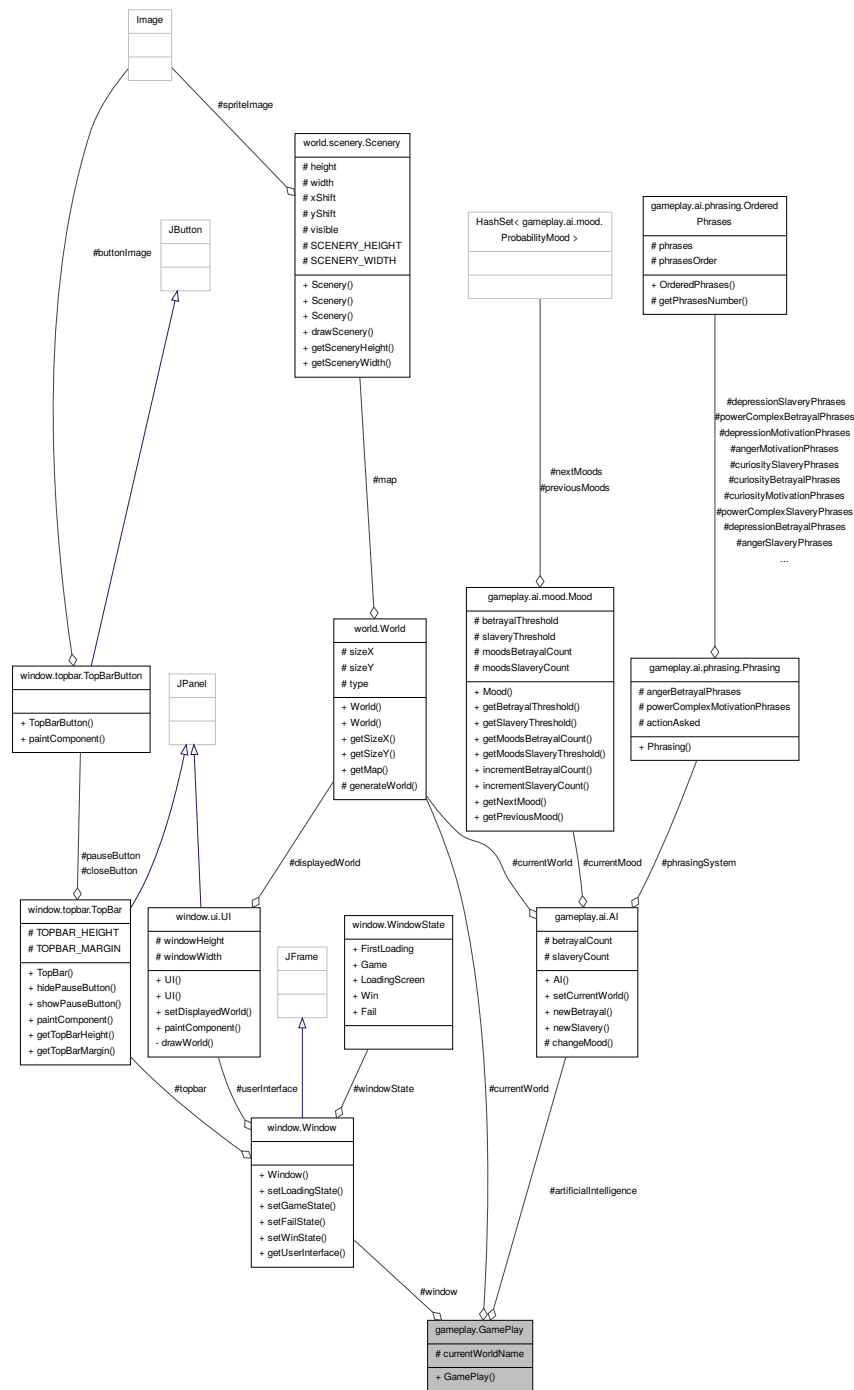
The documentation for this class was generated from the following file:

- `world/scenery/Flower.java`

7.10 gameplay.GamePlay Class Reference

Class handling the game play of this game.

Collaboration diagram for `gameplay.GamePlay`:



Public Member Functions

- [GamePlay](#) ([CliOptions](#) cliOptions)

Constructor of the [GamePlay](#) class.

Protected Attributes

- [World](#) `currentWorld`
World currently being played.
- `String` [currentWorldName](#)
Name of the world currently being played.
- [Window](#) `window`
The game window.
- [AI](#) `artificialIntelligence`
Malicious artificial intelligence driving the player mad.

7.10.1 Detailed Description

Class handling the game play of this game.

It handle the creation of worlds, the movements of the main character and all the other characters, (including the constraints), the interactions between the main character and the other characters or the scenery

The [GamePlay](#) class also controls the user interface mode (loading screen, win, fail, etc. . .)

This class is the equivalent of the controller in the MVC paradigm.

See Also

[world.World](#)

Definition at line 25 of file `GamePlay.java`.

7.10.2 Constructor & Destructor Documentation

7.10.2.1 `gameplay.GamePlay.GamePlay (CliOptions cliOptions)`

Constructor of the [GamePlay](#) class.

Parameters

<i>cliOptions</i>	Options for the game given via the command line
-------------------	-------------------------------------------------

Definition at line 55 of file `GamePlay.java`.

References `gameplay.GamePlay.artificialIntelligence`, `gameplay.GamePlay.currentWorld`, `gameplay.GamePlay.currentWorldName`, and `gameplay.GamePlay.window`.

```

55                                     {
56         Philophobia.getVerbose().information("Creating GamePlay class", "gameplay/Gameplay.java", "
GamePlay.GamePlay()");
57
58         // The default world is the Temperate Broadleaf world
59         currentWorldName = "Temperate Broadleaf";
60         currentWorld = new World("temperatebroadleaf");
61
62         window = new Window();
63
64         try {
65             // Sleep during the length of the animation
66             Thread.sleep(2000);
67         } catch (InterruptedException e) {
68             Philophobia.getVerbose().warning("Sleep interrupted: " + e.getMessage(), "
gameplay/GamePlay.java", "GamePlay.GamePlay(CliOptions)");
69         }
70
71         artificialIntelligence = new AI(currentWorld);
72
73         window.getUserInterface().setDisplayedWorld(currentWorld);
74         window.setGameState();
75     }

```

7.10.3 Member Data Documentation

7.10.3.1 AI `gameplay.GamePlay.artificialIntelligence` [protected]

Malicious artificial intelligence driving the player mad.

Definition at line 49 of file `GamePlay.java`.

Referenced by `gameplay.GamePlay.GamePlay()`.

7.10.3.2 World `gameplay.GamePlay.currentWorld` [protected]

World currently being played.

Definition at line 30 of file `GamePlay.java`.

Referenced by `gameplay.GamePlay.GamePlay()`.

7.10.3.3 String `gameplay.GamePlay.currentWorldName` [protected]

Name of the world currently being played.

It is used to detect in which world the player is and where are the other worlds

Definition at line 38 of file `GamePlay.java`.

Referenced by `gameplay.GamePlay.GamePlay()`.

7.10.3.4 Window `gameplay.GamePlay.window` [protected]

The game window.

Definition at line 43 of file `GamePlay.java`.

Referenced by `gameplay.GamePlay.GamePlay()`.

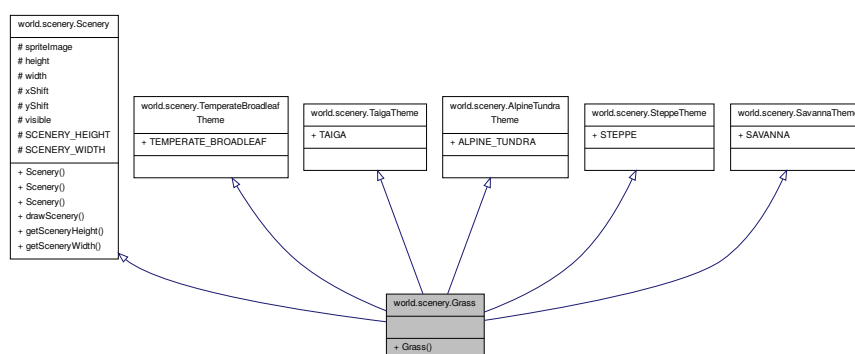
The documentation for this class was generated from the following file:

- [gameplay/GamePlay.java](#)

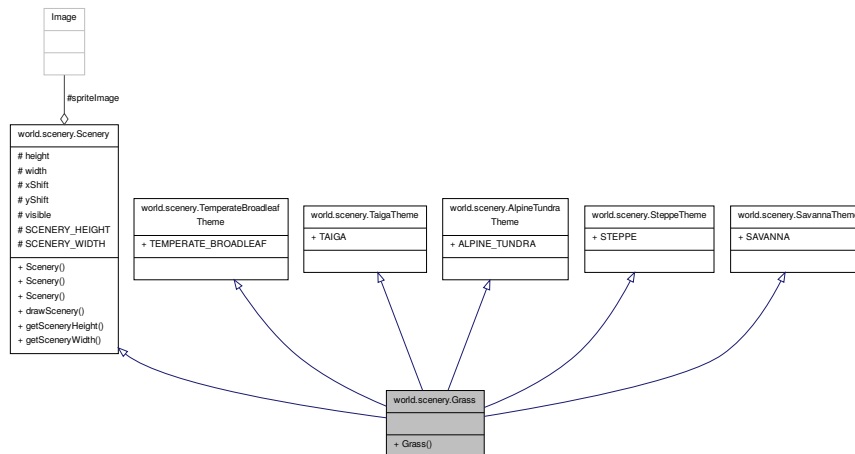
7.11 world.scenery.Grass Class Reference

Class used to handle a grass object.

Inheritance diagram for `world.scenery.Grass`:



Collaboration diagram for world.scenery.Grass:



Public Member Functions

- [Grass](#) (final String type)
[Grass](#) class constructor.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [ALPINE_TUNDRA](#) = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.
- static String [STEPPE](#) = "steppe"
String used to tell in which file is the sprite matching the Steppe style.
- static String [SAVANNA](#) = "savanna"
String used to tell in which file is the sprite matching the Savanna style.

Protected Attributes

- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.

- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.11.1 Detailed Description

Class used to handle a grass object.

Definition at line 15 of file Grass.java.

7.11.2 Constructor & Destructor Documentation

7.11.2.1 `world.scenery.Grass.Grass (final String type)`

[Grass](#) class constructor.

Parameters

<i>type</i>	Style of the grass
-------------	--------------------

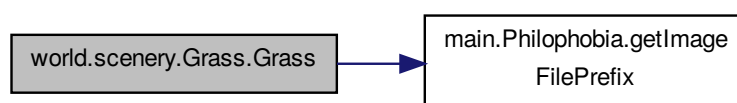
Definition at line 21 of file Grass.java.

References `main.Philophobia.getImageFilePrefix()`.

```

21         {
22
23             super(Philophobia.getImageFilePrefix() + type + "grass.png");
24
25         }
```

Here is the call graph for this function:



7.11.3 Member Function Documentation

7.11.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References world.scenery.Scenery.height, world.scenery.Scenery.spriteImage, world.scenery.Scenery.visible, world.scenery.Scenery.width, world.scenery.Scenery.xShift, and world.scenery.Scenery.yShift.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136           yShift, width, height, obs);
137         visible = true;
138     }

```

7.11.3.2 static int world.scenery.Scenery.getSceneryHeight () [static],[inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

References world.scenery.Scenery.SCENERY_HEIGHT.

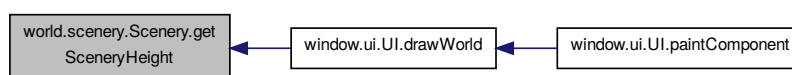
Referenced by window.ui.UI.drawWorld().

```

144                                     {
145         return SCENERY_HEIGHT;
146     }

```

Here is the caller graph for this function:



7.11.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```

152                                     {
153         return SCENERY_WIDTH;
154     }

```

7.11.4 Member Data Documentation

7.11.4.1 `String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file `AlpineTundraTheme.java`.

7.11.4.2 `int world.scenery.Scenery.height` `[protected], [inherited]`

Image height.

Definition at line 39 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.11.4.3 `String world.scenery.SavannaTheme.SAVANNA = "savanna"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Savanna style.

Definition at line 13 of file `SavannaTheme.java`.

7.11.4.4 `int world.scenery.Scenery.SCENERY_HEIGHT = 48` `[static], [protected], [inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.11.4.5 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static], [protected], [inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.11.4.6 `Image world.scenery.Scenery.spriteImage` `[protected], [inherited]`

Image graphically representing the object.

Definition at line 34 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.11.4.7 `String world.scenery.SteppeTheme.STEPPE = "steppe"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Steppe style.

Definition at line 13 of file `SteppeTheme.java`.

7.11.4.8 `String world.scenery.TaigaTheme.TAIGA = "taiga"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file `TaigaTheme.java`.

7.11.4.9 String `world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF` = "temperatebroadleaf" [static], [inherited]

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file `TemperateBroadleafTheme.java`.

7.11.4.10 boolean `world.scenery.Scenery.visible` [protected], [inherited]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.11.4.11 int `world.scenery.Scenery.width` [protected], [inherited]

Image width.

Definition at line 44 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.11.4.12 int `world.scenery.Scenery.xShift` [protected], [inherited]

Horizontal shifting for the image.

Definition at line 49 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.11.4.13 int `world.scenery.Scenery.yShift` [protected], [inherited]

Vertical shifting for the image.

Definition at line 54 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

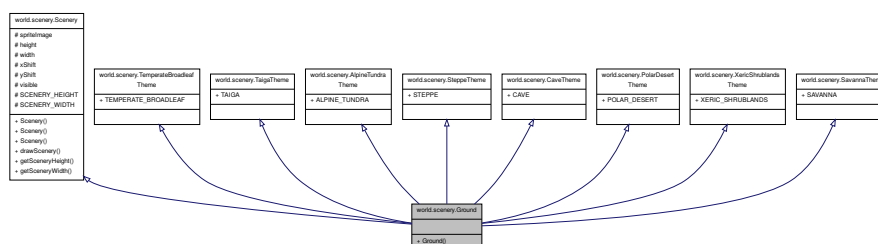
The documentation for this class was generated from the following file:

- [world/scenery/Grass.java](#)

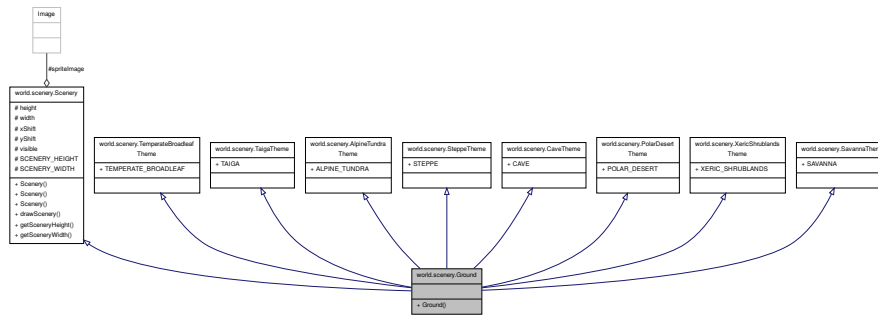
7.12 world.scenery.Ground Class Reference

Class used to handle a ground object.

Inheritance diagram for `world.scenery.Ground`:



Collaboration diagram for world.scenery.Ground:



Public Member Functions

- [Ground](#) (final String type)
Ground class constructor.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [ALPINE_TUNDRA](#) = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.
- static String [STEPPE](#) = "steppe"
String used to tell in which file is the sprite matching the Steppe style.
- static String [CAVE](#) = "cave"
String used to tell in which file is the sprite matching the Cave style.
- static String [POLAR_DESERT](#) = "polardesert"
String used to tell in which file is the sprite matching the Polar Desert style.
- static String [XERIC_SHRUBLANDS](#) = "xericshrublands"
String used to tell in which file is the sprite matching the Xeric Shrublands style.
- static String [SAVANNA](#) = "savanna"
String used to tell in which file is the sprite matching the Savanna style.

Protected Attributes

- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.12.1 Detailed Description

Class used to handle a ground object.

Definition at line 18 of file Ground.java.

7.12.2 Constructor & Destructor Documentation

7.12.2.1 world.scenery.Ground.Ground (final String type)

[Ground](#) class constructor.

Parameters

<i>type</i>	Style of the ground
-------------	---------------------

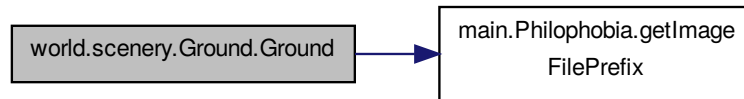
Definition at line 24 of file Ground.java.

References `main.Philophobia.getImageFilePrefix()`.

```

24         {
25
26             super(Philophobia.getImageFilePrefix() + type + "ground.png");
27
28         }
```

Here is the call graph for this function:



7.12.3 Member Function Documentation

7.12.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References world.scenery.Scenery.height, world.scenery.Scenery.spriteImage, world.scenery.Scenery.visible, world.scenery.Scenery.width, world.scenery.Scenery.xShift, and world.scenery.Scenery.yShift.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136           yShift, width, height, obs);
137         visible = true;
138     }
  
```

7.12.3.2 static int world.scenery.Scenery.getSceneryHeight () [static], [inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

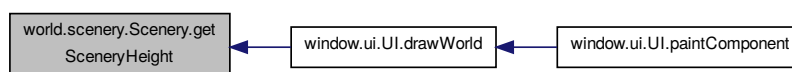
References world.scenery.Scenery.SCENERY_HEIGHT.

Referenced by window.ui.UI.drawWorld().

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
  
```

Here is the caller graph for this function:



7.12.3.3 `static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]`

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_WIDTH`.

```
152                                     {  
153         return SCENERY_WIDTH;  
154     }
```

7.12.4 Member Data Documentation

7.12.4.1 `String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra" [static],[inherited]`

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file AlpineTundraTheme.java.

7.12.4.2 `String world.scenery.CaveTheme.CAVE = "cave" [static],[inherited]`

String used to tell in which file is the sprite matching the Cave style.

Definition at line 13 of file CaveTheme.java.

7.12.4.3 `int world.scenery.Scenery.height [protected],[inherited]`

Image height.

Definition at line 39 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.12.4.4 `String world.scenery.PolarDesertTheme.POLAR_DESERT = "polar-desert" [static],[inherited]`

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.12.4.5 `String world.scenery.SavannaTheme.SAVANNA = "savanna" [static],[inherited]`

String used to tell in which file is the sprite matching the Savanna style.

Definition at line 13 of file SavannaTheme.java.

7.12.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static],[protected],[inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.12.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static], [protected], [inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.12.4.8 `Image world.scenery.Scenery.spriteImage` `[protected], [inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.12.4.9 `String world.scenery.SteppeTheme.STEPPE = "steppe"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Steppe style.

Definition at line 13 of file SteppeTheme.java.

7.12.4.10 `String world.scenery.TaigaTheme.TAIGA = "taiga"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.12.4.11 `String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

7.12.4.12 `boolean world.scenery.Scenery.visible` `[protected], [inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.12.4.13 `int world.scenery.Scenery.width` `[protected], [inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.12.4.14 `String world.scenery.XericShrublandsTheme.XERIC_SHRUBLANDS = "xericshrublands"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Xeric Shrublands style.

Definition at line 13 of file XericShrublandsTheme.java.

7.12.4.15 `int world.scenery.Scenery.xShift` `[protected], [inherited]`

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.12.4.16 `int world.scenery.Scenery.yShift` `[protected], [inherited]`

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

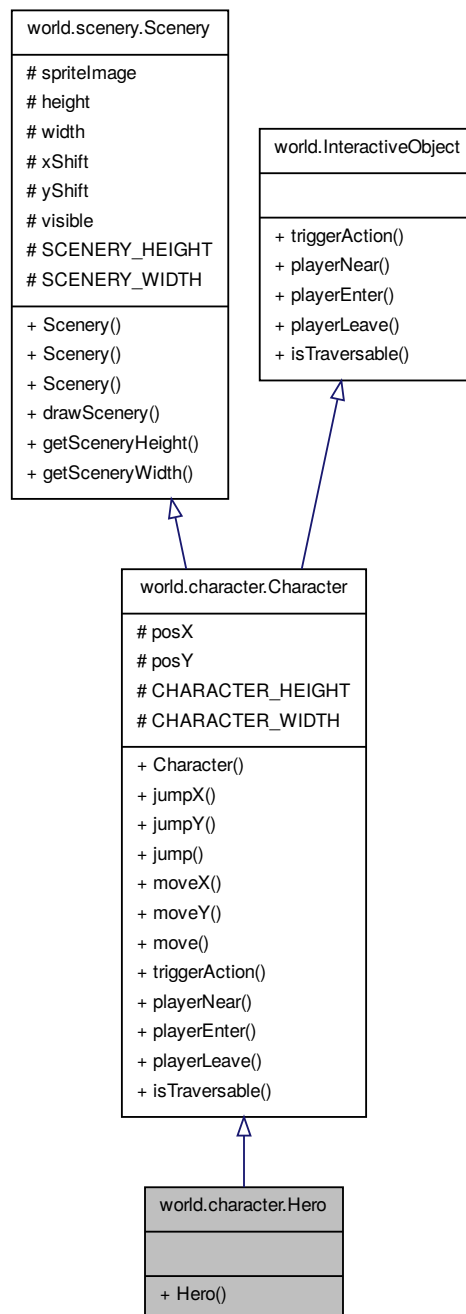
Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

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

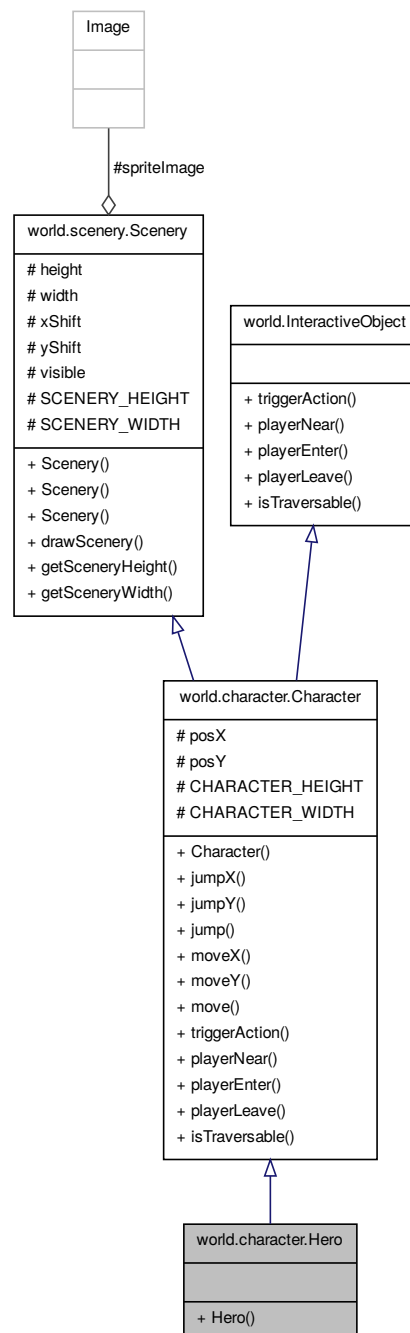
- `world/scenery/Ground.java`

7.13 world.character.Hero Class Reference

Inheritance diagram for world.character.Hero:



Collaboration diagram for world.character.Hero:



Public Member Functions

- [Hero](#) ()
- void [jumpX](#) (int distance)
Teleport the character horizontally regarding his current position.
- void [jumpY](#) (int distance)
Teleport the character vertically regarding his current position.

- void [jump](#) (int distanceX, int distanceY)
Teleport the character regarding his current position.
- void [moveX](#) (int distance)
Slowly horizontally move the character to a certain point regarding his current position.
- void [moveY](#) (int distance)
Slowly vertically move the character to a certain point regarding his current position.
- void [move](#) (int distanceX, int distanceY)
Slowly move the character to a certain point regarding his current position.
- void [triggerAction](#) ()
Method to be called when the player is near the object and presses the action key.
- void [playerNear](#) ()
Method to be called when the player is near the object.
- void [playerEnter](#) ()
Method to be called when the player was near and is now over the object.
- void [playerLeave](#) ()
Method to be called when the player was over and is now near the object.
- boolean [isTraversable](#) ()
Returns true if the player is able to pass through the object and false if the player is not able to cross the object.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Protected Attributes

- int [posX](#)
Horizontal position of the character.
- int [posY](#)
Vertical position of the character.
- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static final int `CHARACTER_HEIGHT` = 300
Height of the character's image (fixed)
- static final int `CHARACTER_WIDTH` = 170
Width of the character's image (fixed)
- static int `SCENERY_HEIGHT` = 48
Height of the graphical representation of a [Scenery](#) object.
- static int `SCENERY_WIDTH` = 48
Width of the graphical representation of a [Scenery](#) object.

7.13.1 Detailed Description

Definition at line 6 of file Hero.java.

7.13.2 Constructor & Destructor Documentation

7.13.2.1 world.character.Hero.Hero ()

Definition at line 8 of file Hero.java.

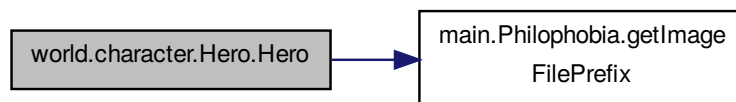
References `main.Philophobia.getImageFilePrefix()`.

```

8         {
9             super(Philophobia.getImageFilePrefix() + "hero.png");
10            Philophobia.getVerbose().information("Created Hero class", "world/character/Hero.java", "
Hero.Hero()");
11        }

```

Here is the call graph for this function:



7.13.3 Member Function Documentation

7.13.3.1 void world.scenery.Scenery.drawScenery (Graphics g, final int xLocation, final int yLocation, ImageObserver obs) [inherited]

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133                                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
yShift, width, height, obs);
136
137         visible = true;
138     }

```

7.13.3.2 static int world.scenery.Scenery.getSceneryHeight () [static],[inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

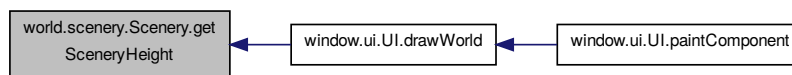
References world.scenery.Scenery.SCENERY_HEIGHT.

Referenced by window.ui.UI.drawWorld().

```

144                                     {
145     return SCENERY_HEIGHT;
146 }
```

Here is the caller graph for this function:



7.13.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```

152                                     {
153     return SCENERY_WIDTH;
154 }
```

7.13.3.4 boolean world.character.Character.isTraversable () [inherited]

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 161 of file Character.java.

```

161                                     {
162     return false;
163 }
```

7.13.3.5 void world.character.Character.jump (int distanceX, int distanceY) [inherited]

Teleport the character regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be teleported
<i>distanceY</i>	Vertical distance toward the character will be teleported

Definition at line 73 of file Character.java.

```

73                                     {
74         if(distanceX != 0 || distanceY != 0) {
75             this.posX = this.posX + distance;
76             this.posY = this.posY + distance;
77             this.draw();
78         }
79     }

```

7.13.3.6 void world.character.Character.jumpX (int *distance*) [inherited]

Teleport the character horizontally regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be teleported
-----------------	-------------------------------------------------------------

Definition at line 50 of file Character.java.

```

50                                     {
51         if(distance != 0) {
52             this.posX = this.posX + distance;
53             this.draw();
54         }
55     }

```

7.13.3.7 void world.character.Character.jumpY (int *distance*) [inherited]

Teleport the character vertically regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be teleported
-----------------	-----------------------------------------------------------

Definition at line 61 of file Character.java.

```

61                                     {
62         if(distance != 0) {
63             this.posY = this.posY + distance;
64             this.draw();
65         }
66     }

```

7.13.3.8 void world.character.Character.move (int *distanceX*, int *distanceY*) [inherited]

Slowly move the character to a certain point regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be moved
<i>distanceY</i>	Vertical distance toward the character will be moved

Definition at line 118 of file Character.java.

References world.character.Character.moveX(), and world.character.Character.moveY().

```

118                                     {
119
120         if(distanceX != 0 || distanceY != 0) {

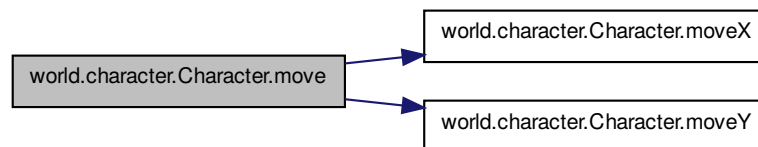
```

```

121         int stepX = (distanceX < 0)? -1 : 1;
122         int stepY = (distanceY < 0)? -1 : 1;
123
124         this.jump(stepX, stepY);
125
126         if(distanceX - stepX == 0) {
127             moveY(distanceY - stepY);
128             return;
129         }
130
131         if(distanceY - stepY == 0) {
132             moveX(distanceX - stepX);
133             return;
134         }
135
136         move(distanceX - stepX, distanceY - stepY);
137     }
138
139 }

```

Here is the call graph for this function:



7.13.3.9 void world.character.Character.moveX (int *distance*) [inherited]

Slowly horizontally move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be moved
-----------------	--------------------------------------------------------

Definition at line 85 of file Character.java.

Referenced by `world.character.Character.move()`.

```

85         {
86
87         if(distance != 0) {
88             int step = (distance < 0)? -1 : 1;
89
90             this.jumpX(step);
91
92             this.moveX(distance - step);
93         }
94     }
95 }

```

Here is the caller graph for this function:



7.13.3.10 void world.character.Character.moveY (int *distance*) [inherited]

Slowly vertically move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be moved
-----------------	------------------------------------------------------

Definition at line 101 of file Character.java.

Referenced by world.character.Character.move().

```

101                                     {
102
103     if(distance != 0) {
104         int step = (distance < 0)? -1 : 1;
105
106         this.jumpY(step);
107
108         this.moveY(distance - step);
109     }
110
111 }
```

Here is the caller graph for this function:

**7.13.3.11 void world.character.Character.playerEnter ()** [inherited]

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 153 of file Character.java.

```

153                                     {
154
155 }
```

7.13.3.12 void world.character.Character.playerLeave () [inherited]

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 157 of file Character.java.

```

157                                     {
158
159 }
```

7.13.3.13 void world.character.Character.playerNear () [inherited]

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 149 of file Character.java.

```
149         {  
150  
151     }
```

7.13.3.14 void world.character.Character.triggerAction () [inherited]

Method to be called when the player is near the object and presses the action key.

Implements [world.InteractiveObject](#).

Definition at line 145 of file Character.java.

```
145         {  
146  
147     }
```

7.13.4 Member Data Documentation

7.13.4.1 final int world.character.Character.CHARACTER_HEIGHT = 300 [static],[protected],[inherited]

Height of the character's image (fixed)

Definition at line 18 of file Character.java.

Referenced by world.character.Character.Character().

7.13.4.2 final int world.character.Character.CHARACTER_WIDTH = 170 [static],[protected],[inherited]

Width of the character's image (fixed)

Definition at line 23 of file Character.java.

Referenced by world.character.Character.Character().

7.13.4.3 int world.scenery.Scenery.height [protected],[inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.13.4.4 int world.character.Character.posX [protected],[inherited]

Horizontal position of the character.

Definition at line 28 of file Character.java.

7.13.4.5 int world.character.Character.posY [protected],[inherited]

Vertical position of the character.

Definition at line 33 of file Character.java.

7.13.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48` `[static]`, `[protected]`, `[inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.13.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static]`, `[protected]`, `[inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.13.4.8 `Image world.scenery.Scenery.spriteImage` `[protected]`, `[inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.13.4.9 `boolean world.scenery.Scenery.visible` `[protected]`, `[inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.13.4.10 `int world.scenery.Scenery.width` `[protected]`, `[inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.13.4.11 `int world.scenery.Scenery.xShift` `[protected]`, `[inherited]`

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.13.4.12 `int world.scenery.Scenery.yShift` `[protected]`, `[inherited]`

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

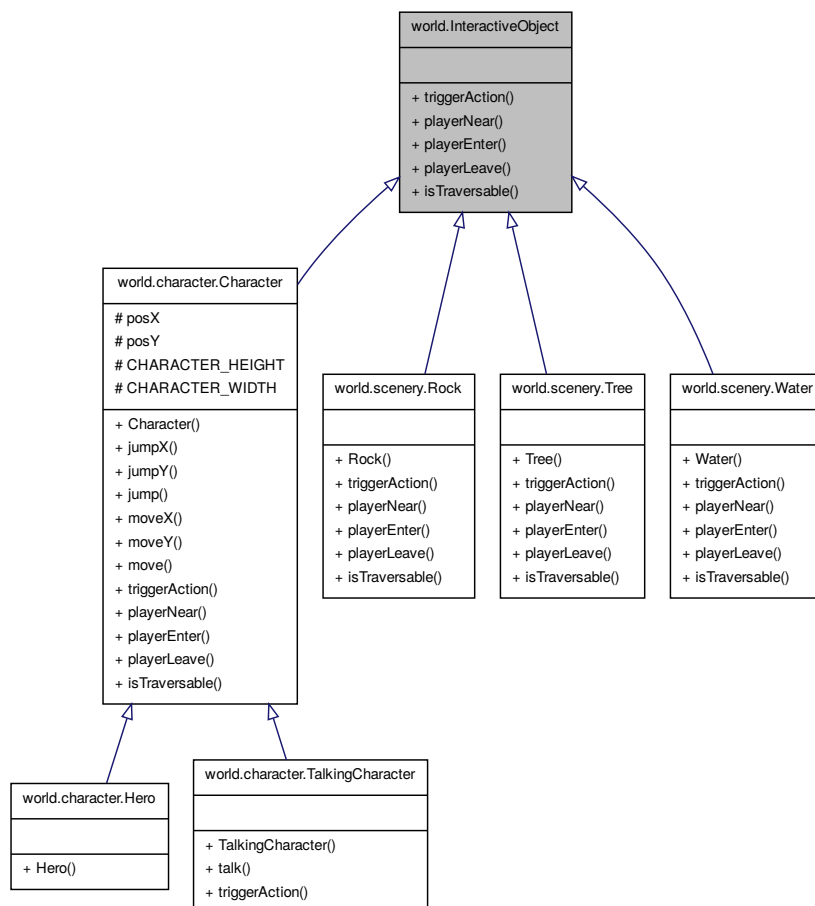
Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

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

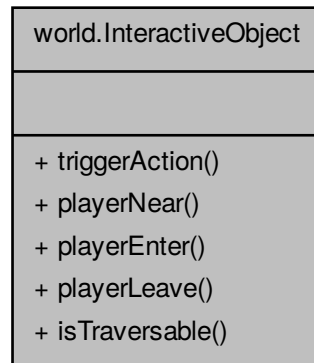
- `world/character/Hero.java`

7.14 world.InteractiveObject Interface Reference

Inheritance diagram for world.InteractiveObject:



Collaboration diagram for world.InteractiveObject:



Public Member Functions

- void [triggerAction](#) ()
Method to be called when the player is near the object and presses the action key.
- void [playerNear](#) ()
Method to be called when the player is near the object.
- void [playerEnter](#) ()
Method to be called when the player was near and is now over the object.
- void [playerLeave](#) ()
Method to be called when the player was over and is now near the object.
- boolean [isTraversable](#) ()
Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

7.14.1 Detailed Description

Definition at line 6 of file InteractiveObject.java.

7.14.2 Member Function Documentation

7.14.2.1 boolean world.InteractiveObject.isTraversable ()

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implemented in [world.character.Character](#), [world.scenery.Tree](#), [world.scenery.Rock](#), and [world.scenery.Water](#).

7.14.2.2 void world.InteractiveObject.playerEnter ()

Method to be called when the player was near and is now over the object.

Implemented in [world.character.Character](#), [world.scenery.Tree](#), [world.scenery.Rock](#), and [world.scenery.Water](#).

7.14.2.3 void world.InteractiveObject.playerLeave ()

Method to be called when the player was over and is now near the object.

Implemented in [world.character.Character](#), [world.scenery.Tree](#), [world.scenery.Rock](#), and [world.scenery.Water](#).

7.14.2.4 void world.InteractiveObject.playerNear ()

Method to be called when the player is near the object.

Implemented in [world.character.Character](#), [world.scenery.Tree](#), [world.scenery.Rock](#), and [world.scenery.Water](#).

7.14.2.5 void world.InteractiveObject.triggerAction ()

Method to be called when the player is near the object and presses the action key.

Implemented in [world.character.Character](#), [world.scenery.Tree](#), [world.scenery.Rock](#), [world.character.Talking-Character](#), and [world.scenery.Water](#).

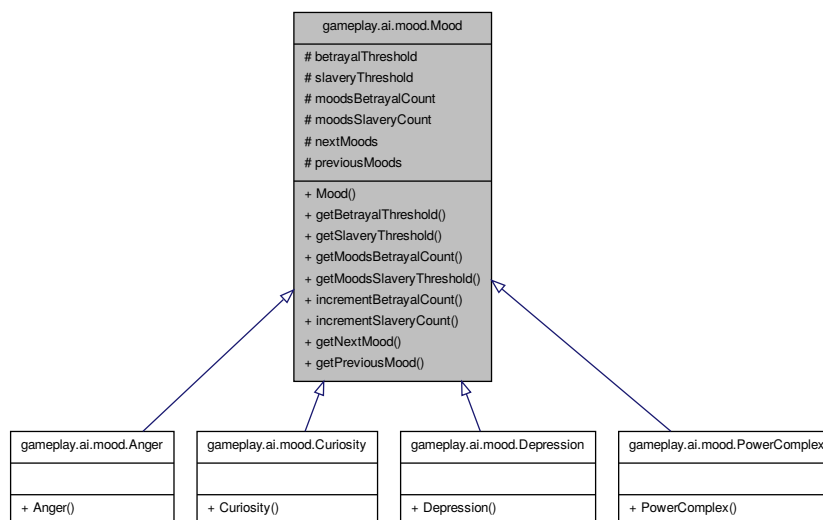
The documentation for this interface was generated from the following file:

- [world/InteractiveObject.java](#)

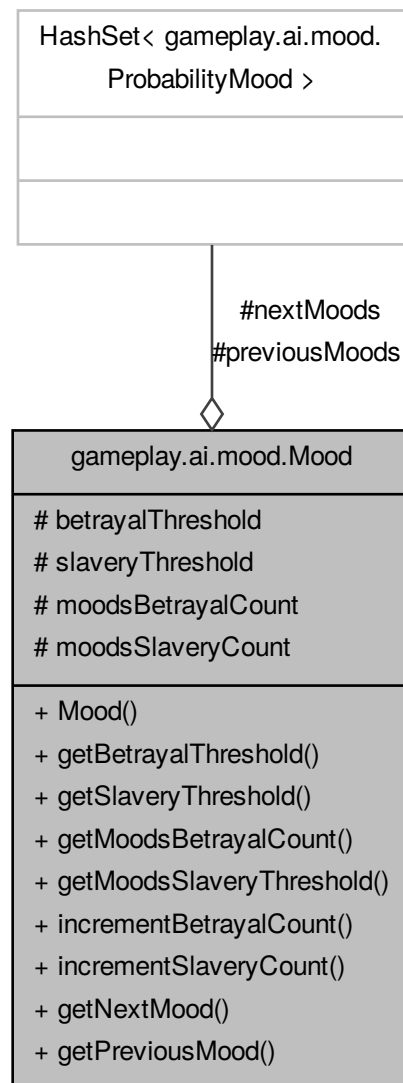
7.15 gameplay.ai.mood.Mood Class Reference

Abstract class representing an [AI](#) mood.

Inheritance diagram for gameplay.ai.mood.Mood:



Collaboration diagram for gameplay.ai.mood.Mood:



Public Member Functions

- `Mood` (int `betrayalThreshold`, int `slaveryThreshold`)
Mood constructor.
- int `getBetrayalThreshold` ()
Getter for the betrayal threshold field.
- int `getSlaveryThreshold` ()
Getter for the slavery threshold field.
- int `getMoodsBetrayalCount` ()
Getter for the betrayal count field.
- int `getMoodsSlaveryThreshold` ()

- *Getter for the slavery count field.*
- `int incrementBetrayalCount ()`
Increment and return the betrayal count field.
- `int incrementSlaveryCount ()`
Increment and return the slavery count field.
- `Class< Mood > getNextMood ()`
When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.
- `Class< Mood > getPreviousMood ()`
When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Protected Attributes

- `int betrayalThreshold`
If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- `int slaveryThreshold`
If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- `int moodsBetrayalCount`
Number of betrayals (requests from the AI not done by the user) done by the user when the AI was currently in this mood state.
- `int moodsSlaveryCount`
Number of requests from the AI done positively by the user when the AI was in this mood state.
- `HashSet< ProbabilityMood > nextMoods`
Associative array containing the possible moods if the number of betrayals (when the AI was in this mood state) is greater than its threshold with the probability associated.
- `HashSet< ProbabilityMood > previousMoods`
Associative array containing the possible moods if the number of done actions asked by the AI is greater than its threshold with the probability associated.

7.15.1 Detailed Description

Abstract class representing an AI mood.

Definition at line 12 of file Mood.java.

7.15.2 Constructor & Destructor Documentation

7.15.2.1 gameplay.ai.mood.Mood.Mood (int betrayalThreshold, int slaveryThreshold)

Mood constructor.

Definition at line 66 of file Mood.java.

References gameplay.ai.mood.Mood.betrayalThreshold, and gameplay.ai.mood.Mood.slaveryThreshold.

```

66                                     {
67
68         Philophobia.getVerbose().information("Creating new Mood class", "gameplay/ai/mood/Mood.java", "
Mood.Mood(int, int)");
69
70         this.betrayalThreshold = betrayalThreshold;
71         this.slaveryThreshold = slaveryThreshold;
72
73     }
```

7.15.3 Member Function Documentation

7.15.3.1 `int gameplay.ai.mood.Mood.getBetrayalThreshold ()`

Getter for the betrayal threshold field.

See Also

[betrayalThreshold](#)

Definition at line 79 of file Mood.java.

References `gameplay.ai.mood.Mood.betrayalThreshold`.

```
79                                     {  
80         return betrayalThreshold;  
81     }
```

7.15.3.2 `int gameplay.ai.mood.Mood.getMoodsBetrayalCount ()`

Getter for the betrayal count field.

See Also

[moodsBetrayalCount](#)

Definition at line 95 of file Mood.java.

References `gameplay.ai.mood.Mood.moodsBetrayalCount`.

```
95                                     {  
96         return moodsBetrayalCount;  
97     }
```

7.15.3.3 `int gameplay.ai.mood.Mood.getMoodsSlaveryThreshold ()`

Getter for the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 103 of file Mood.java.

References `gameplay.ai.mood.Mood.moodsSlaveryCount`.

```
103                                     {  
104         return moodsSlaveryCount;  
105     }
```

7.15.3.4 `Class<Mood> gameplay.ai.mood.Mood.getNextMood ()`

When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 128 of file Mood.java.

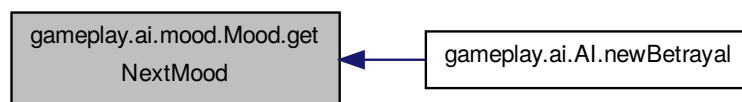
Referenced by `gameplay.ai.AI.newBetrayal()`.

```

128         {
129             Iterator<ProbabilityMood> it = nextMoods.iterator();
130             ProbabilityMood probMood = null;
131             while(it.hasNext()) {
132
133                 probMood = it.next();
134                 if(Math.random() >= probMood.getProbability()) {
135                     return probMood.getMood();
136                 }
137             }
138         }
139
140         return probMood.getMood();
141     }

```

Here is the caller graph for this function:



7.15.3.5 Class<Mood> gameplay.ai.mood.Mood.getPreviousMood ()

When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 148 of file Mood.java.

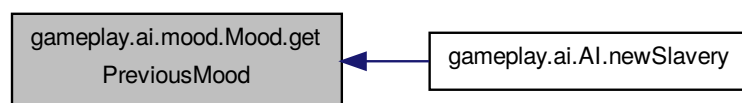
Referenced by `gameplay.ai.AI.newSlavery()`.

```

148         {
149             Iterator<ProbabilityMood> it = previousMoods.iterator();
150             ProbabilityMood probMood = null;
151             while(it.hasNext()) {
152
153                 probMood = it.next();
154                 if(Math.random() >= probMood.getProbability()) {
155                     return probMood.getMood();
156                 }
157             }
158         }
159
160         return probMood.getMood();
161     }

```

Here is the caller graph for this function:



7.15.3.6 int gameplay.ai.mood.Mood.getSlaveryThreshold ()

Getter for the slavery threshold field.

See Also

[slaveryThreshold](#)

Definition at line 87 of file Mood.java.

References gameplay.ai.mood.Mood.slaveryThreshold.

```
87                                     {
88         return slaveryThreshold;
89     }
```

7.15.3.7 int gameplay.ai.mood.Mood.incrementBetrayalCount ()

Increment and return the betrayal count field.

See Also

[moodsBetrayalCount](#)

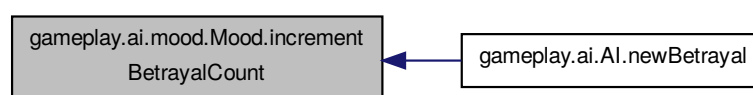
Definition at line 111 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

Referenced by gameplay.ai.AI.newBetrayal().

```
111                                     {
112         return ++moodsBetrayalCount;
113     }
```

Here is the caller graph for this function:



7.15.3.8 int gameplay.ai.mood.Mood.incrementSlaveryCount ()

Increment and return the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 119 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

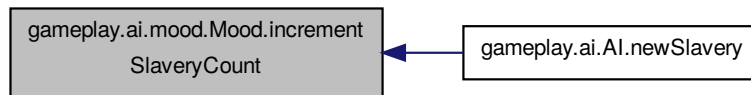
Referenced by gameplay.ai.AI.newSlavery().

```

119
120     return ++moodsSlaveryCount;
121 }

```

Here is the caller graph for this function:



7.15.4 Member Data Documentation

7.15.4.1 int gameplay.ai.mood.Mood.betrayalThreshold [protected]

If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 20 of file Mood.java.

Referenced by gameplay.ai.mood.Mood.getBetrayalThreshold(), and gameplay.ai.mood.Mood.Mood().

7.15.4.2 int gameplay.ai.mood.Mood.moodsBetrayalCount [protected]

Number of betrayals (requests from the AI not done by the user) done by the user when the AI was currently in this mood state.

Definition at line 37 of file Mood.java.

Referenced by gameplay.ai.mood.Mood.getMoodsBetrayalCount(), and gameplay.ai.mood.Mood.incrementBetrayalCount().

7.15.4.3 int gameplay.ai.mood.Mood.moodsSlaveryCount [protected]

Number of requests from the AI done positively by the user when the AI was in this mood state.

Definition at line 44 of file Mood.java.

Referenced by gameplay.ai.mood.Mood.getMoodsSlaveryThreshold(), and gameplay.ai.mood.Mood.incrementSlaveryCount().

7.15.4.4 HashSet<ProbabilityMood> gameplay.ai.mood.Mood.nextMoods [protected]

Associative array containing the possible moods if the number of betrayals (when the AI was in this mood state) is greater than its threshold with the probability associated.

Definition at line 53 of file Mood.java.

7.15.4.5 HashSet<ProbabilityMood> gameplay.ai.mood.Mood.previousMoods [protected]

Associative array containing the possible moods if the number of done actions asked by the AI is greater than its threshold with the probability associated.

Definition at line 61 of file Mood.java.

7.15.4.6 `int gameplay.ai.mood.Mood.slaveryThreshold` `[protected]`

If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 28 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getSlaveryThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

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

- `gameplay/ai/mood/Mood.java`

7.16 `gameplay.ai.phrasing.OrderedPhrases` Class Reference

Class used to contain an ordered list of phrases.

Collaboration diagram for `gameplay.ai.phrasing.OrderedPhrases`:

<code>gameplay.ai.phrasing.OrderedPhrases</code>
<code># phrases</code> <code># phrasesOrder</code>
<code>+ OrderedPhrases()</code> <code># getPhrasesNumber()</code>

Public Member Functions

- `OrderedPhrases` (final String filename)
`OrderedPhrases` class constructor.

Protected Member Functions

- `int getPhrasesNumber` (final String filename)
Method counting the number of line in filename.

Protected Attributes

- `String[] phrases`
Array containing all the phrases corresponding to the current feeling and/or action.
- `int[] phrasesOrder`
Order of the phrases array field.

7.16.1 Detailed Description

Class used to contain an ordered list of phrases.

It is ordered according to the last phrases that were spoken by the [AI](#)

The last phrases that were spoken by the [AI](#) are pushed at the bottom of the list

Definition at line 19 of file OrderedPhrases.java.

7.16.2 Constructor & Destructor Documentation

7.16.2.1 `gameplay.ai.phrasing.OrderedPhrases.OrderedPhrases (final String filename)`

[OrderedPhrases](#) class constructor.

Parameters

<i>filename</i>	Path and name of the file containing line by line the phrases
-----------------	---------------------------------------------------------------

Definition at line 38 of file OrderedPhrases.java.

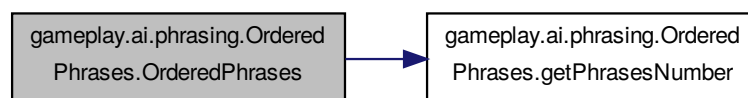
References `gameplay.ai.phrasing.OrderedPhrases.getPhrasesNumber()`, `gameplay.ai.phrasing.OrderedPhrases.phrases`, and `gameplay.ai.phrasing.OrderedPhrases.phrasesOrder`.

```

38         {
39             Philophobia.getVerbose().calls("Creating new OrderedPhrases class", "
gameplay/ai/phrasing/OrderedPhrases.java", "OrderedPhrases.OrderedPhrases(String)");
40
41             int phrasesLength = getPhrasesNumber(filename);
42
43             phrases = new String[phrasesLength];
44             phrasesOrder = new int[phrasesLength];
45
46             try {
47                 RandomAccessFile raFile = new RandomAccessFile(new File(filename), "r");
48
49                 String currentLine = raFile.readLine();
50                 for(int i = 0 ; currentLine != null ; ++i) {
51                     phrases[i] = currentLine;
52                     currentLine = raFile.readLine();
53                 }
54             } catch(FileNotFoundException e) {
55                 Philophobia.getVerbose().serious("File not found exception: " + e.getMessage(), "
gameplay/ai/phrasing/OrderedPhrases.java", "OrderedPhrases.OrderedPhrases(String)");
56             } catch(IOException e) {
57                 Philophobia.getVerbose().serious("IO exception: " + e.getMessage(), "
gameplay/ai/phrasing/OrderedPhrases.java", "OrderedPhrases.OrderedPhrases(String)");
58             }
59         }

```

Here is the call graph for this function:



7.16.3 Member Function Documentation

7.16.3.1 `int gameplay.ai.phrasing.OrderedPhrases.getPhrasesNumber (final String filename)` `[protected]`

Method counting the number of line in filename.

Parameters

<i>filename</i>	Path and name of the file
-----------------	---------------------------

Returns

The number of phrases une the file or 0 if there is no phrases or an error

Definition at line 66 of file OrderedPhrases.java.

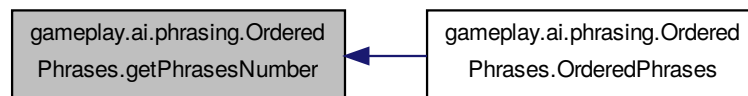
Referenced by `gameplay.ai.phrasing.OrderedPhrases.OrderedPhrases()`.

```

66                                     {
67
68         try {
69             // Opening filename i read mode
70             RandomAccessFile raFile = new RandomAccessFile(new File(filename), "r");
71
72             // Count the number of lines
73             int i;
74             for(i=0 ; raFile.readLine() != null ; ++i);
75             return i;
76         } catch(FileNotFoundException e) {
77             Philophobia.getVerbose().serious("File not found exception: " + e.getMessage(), "
gameplay/ai/phrasing/OrderedPhrases.java", "OrderedPhrases.OrderedPhrases(String)");
78         } catch(IOException e) {
79             Philophobia.getVerbose().serious("IO exception: " + e.getMessage(), "
gameplay/ai/phrasing/OrderedPhrases.java", "OrderedPhrases.OrderedPhrases(String)");
80         }
81         return 0;
82     }

```

Here is the caller graph for this function:



7.16.4 Member Data Documentation

7.16.4.1 `String [] gameplay.ai.phrasing.OrderedPhrases.phrases` [protected]

Array containing all the phrases corresponding to the current feeling and/or action.

Definition at line 26 of file OrderedPhrases.java.

Referenced by `gameplay.ai.phrasing.OrderedPhrases.OrderedPhrases()`.

7.16.4.2 `int [] gameplay.ai.phrasing.OrderedPhrases.phrasesOrder` [protected]

Order of the phrases array field.

See Also

[phrases](#)

Definition at line 32 of file OrderedPhrases.java.

Referenced by `gameplay.ai.phrasing.OrderedPhrases.OrderedPhrases()`.

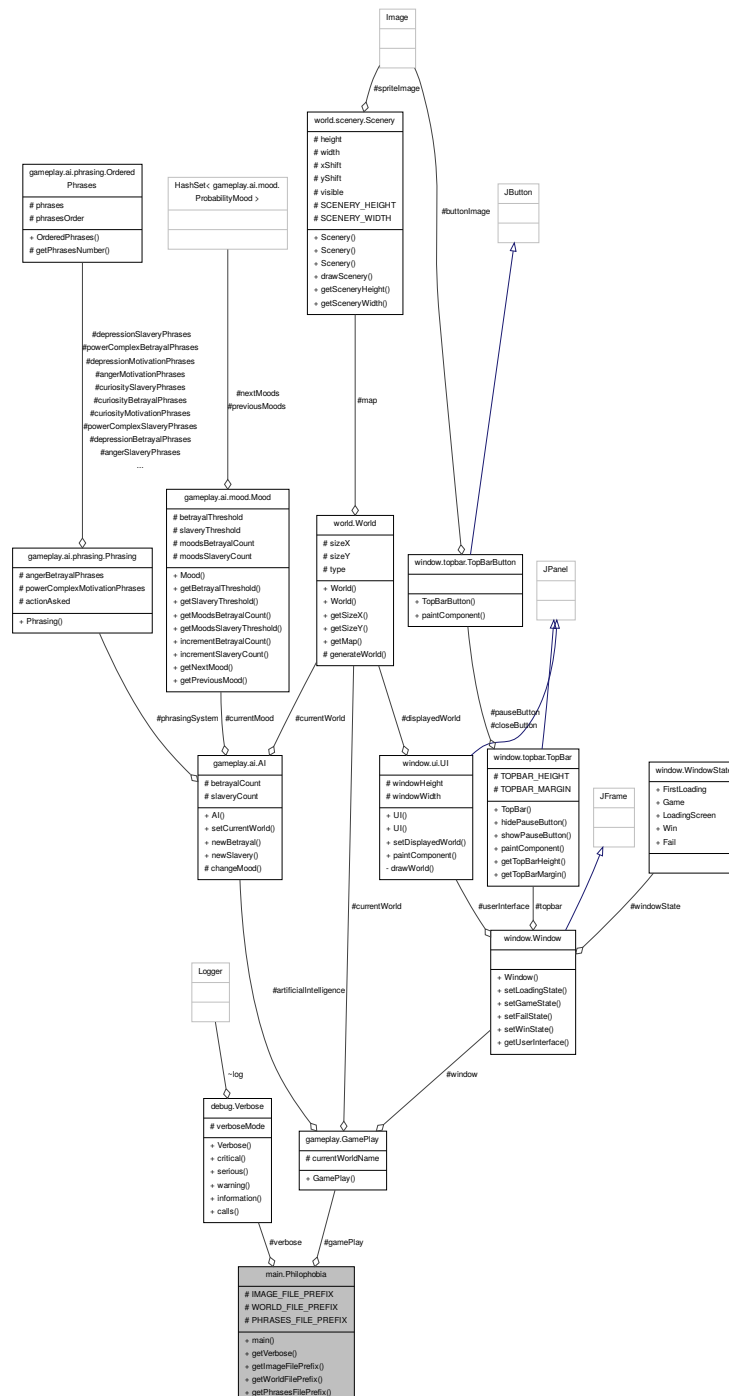
The documentation for this class was generated from the following file:

- [gameplay/ai/phrasing/OrderedPhrases.java](#)

7.17 main.Philophobia Class Reference

Main class.

Collaboration diagram for main.Philophobia:



Static Public Member Functions

- static void [main](#) (String[] args)
Main function of the program You can set options like verbose mode to the program.
- static [Verbose](#) [getVerbose](#) ()
Getter for the verbose field.
- static String [getImageFilePrefix](#) ()
Getter for the IMAGE_FILE_PREFIX field.
- static String [getWorldFilePrefix](#) ()
Getter for the WORLD_FILE_PREFIX field.
- static String [getPhrasesFilePrefix](#) ()

Static Protected Attributes

- static String [IMAGE_FILE_PREFIX](#) = "src/images/"
Directory containing all the image files.
- static String [WORLD_FILE_PREFIX](#) = "src/world/"
Directory containing all the world files.
- static String [PHRASES_FILE_PREFIX](#) = "src/phrases/"
Directory containing all the phrases files.
- static [Verbose](#) [verbose](#)
Global variable used for debugging.
- static [GamePlay](#) [gamePlay](#)
Main window of the program.

7.17.1 Detailed Description

Main class.

Definition at line 12 of file Philophobia.java.

7.17.2 Member Function Documentation

7.17.2.1 static String main.Philophobia.getImageFilePrefix () [static]

Getter for the IMAGE_FILE_PREFIX field.

See Also

[IMAGE_FILE_PREFIX](#)

Definition at line 87 of file Philophobia.java.

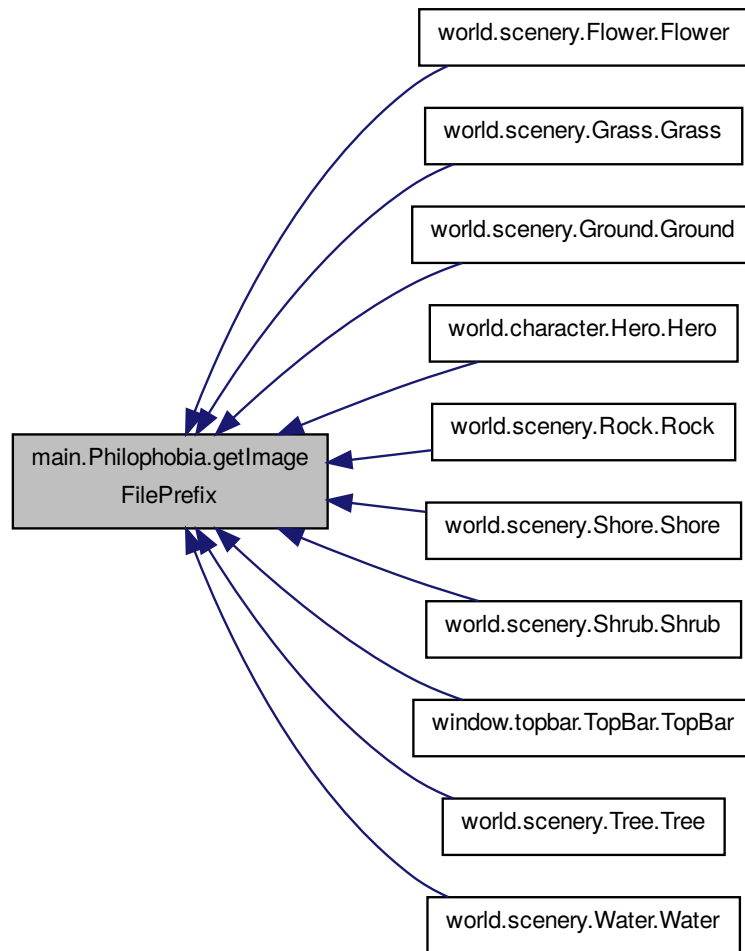
References [main.Philophobia.IMAGE_FILE_PREFIX](#).

Referenced by [world.scenery.Flower.Flower\(\)](#), [world.scenery.Grass.Grass\(\)](#), [world.scenery.Ground.Ground\(\)](#), [world.character.Hero.Hero\(\)](#), [world.scenery.Rock.Rock\(\)](#), [world.scenery.Shore.Shore\(\)](#), [world.scenery.Shrub.-Shrub\(\)](#), [window.topbar.TopBar.TopBar\(\)](#), [world.scenery.Tree.Tree\(\)](#), and [world.scenery.Water.Water\(\)](#).

```

87                                     {
88         return IMAGE\_FILE\_PREFIX;
89     }
```

Here is the caller graph for this function:



7.17.2.2 static String main.Philophobia.getPhrasesFilePrefix () [static]

Definition at line 99 of file Philophobia.java.

References main.Philophobia.PHRASES_FILE_PREFIX.

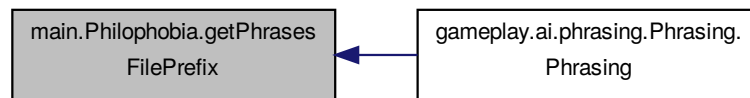
Referenced by gameplay.ai.phrasing.Phrasing.Phrasing().

```

99
100     return PHRASES_FILE_PREFIX;
101 }

```

Here is the caller graph for this function:



7.17.2.3 static Verbose main.Philophobia.getVerbose () [static]

Getter for the verbose field.

Returns

The main class verbose static property

Definition at line 79 of file Philophobia.java.

References main.Philophobia.verbose.

```
79                                     {
80         return verbose;
81     }
```

7.17.2.4 static String main.Philophobia.getWorldFilePrefix () [static]

Getter for the WORLD_FILE_PREFIX field.

See Also

[WORLD_FILE_PREFIX](#)

Definition at line 95 of file Philophobia.java.

References main.Philophobia.WORLD_FILE_PREFIX.

```
95                                     {
96         return WORLD_FILE_PREFIX;
97     }
```

7.17.2.5 static void main.Philophobia.main (String[] args) [static]

Main function of the program You can set options like verbose mode to the program.

See Also

[debug.CliOptions](#)

Parameters

<i>args</i>	Global program options (see above)
-------------	------------------------------------

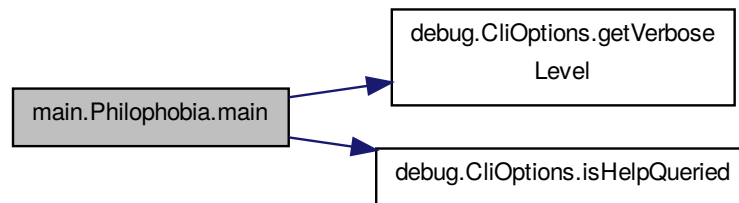
Definition at line 51 of file Philophobia.java.

References `debug.CliOptions.getVerboseLevel()`, and `debug.CliOptions.isHelpQueried()`.

```

51                                     {
52
53         CliOptions options = new CliOptions(args);
54
55         Philophobia.verbose = new Verbose(options.getVerboseLevel());
56
57         if(!options.isHelpQueried()) {
58             // Philophobia.window = new Window();
59             Philophobia.gamePlay = new GamePlay(options);
60         } else {
61
62             System.out.println("");
63             System.out.println("Usage: philophobia [options]");
64             System.out.println("");
65             System.out.println("Options:");
66             System.out.println("  Debugging options:");
67             System.out.println("    -v          Activate verbose mode (to be use multiple times
for more verbosity)");
68             System.out.println("    -h          --help    Print this help");
69             System.out.println("");
70         }
71     }
72
73 }
```

Here is the call graph for this function:



7.17.3 Member Data Documentation

7.17.3.1 `GamePlay` `main.Philophobia.gamePlay` `[static]`, `[protected]`

Main window of the program.

Game handler

Definition at line 42 of file Philophobia.java.

7.17.3.2 `String` `main.Philophobia.IMAGE_FILE_PREFIX` `"src/images/"` `[static]`, `[protected]`

Directory containing all the image files.

Definition at line 17 of file Philophobia.java.

Referenced by `main.Philophobia.getImageFilePrefix()`.

7.17.3.3 String main.Philophobia.PHRASES_FILE_PREFIX = "src/phrases/" [static], [protected]

Directory containing all the phrases files.

Definition at line 27 of file Philophobia.java.

Referenced by main.Philophobia.getPhrasesFilePrefix().

7.17.3.4 Verbose main.Philophobia.verbose [static], [protected]

Global variable used for debugging.

Definition at line 32 of file Philophobia.java.

Referenced by main.Philophobia.getVerbose().

7.17.3.5 String main.Philophobia.WORLD_FILE_PREFIX = "src/world/" [static], [protected]

Directory containing all the world files.

Definition at line 22 of file Philophobia.java.

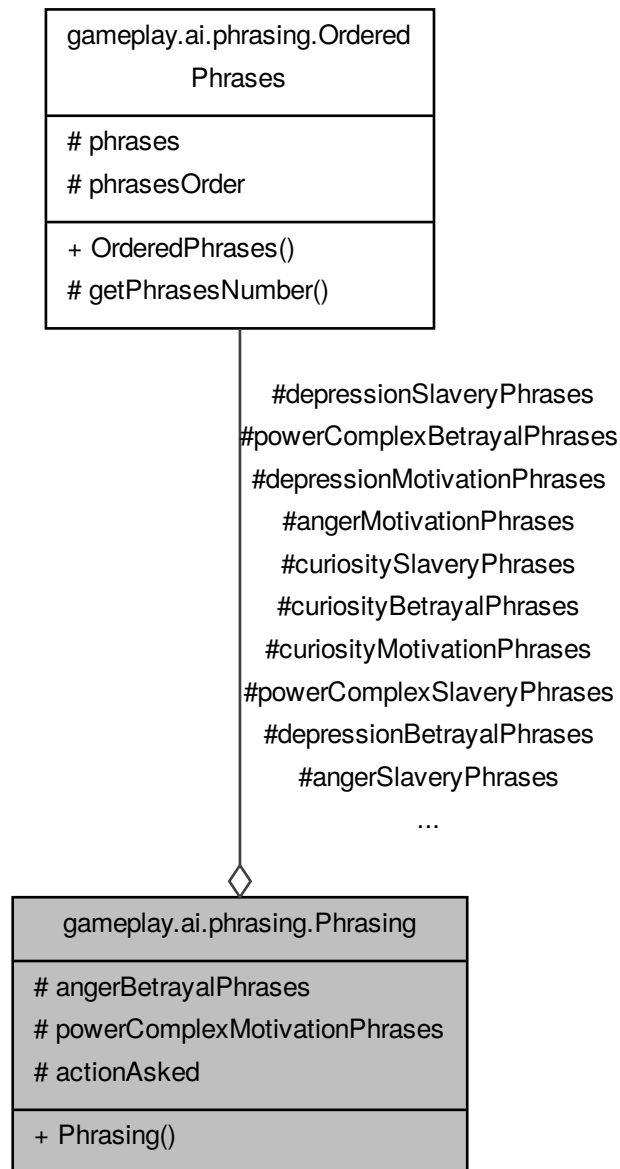
Referenced by main.Philophobia.getWorldFilePrefix().

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

- main/[Philophobia.java](#)

7.18 gameplay.ai.phrasing.Phrasing Class Reference

Collaboration diagram for gameplay.ai.phrasing.Phrasing:



Public Member Functions

- [Phrasing \(\)](#)

Phrasing class constructor.

Protected Attributes

- [OrderedPhrases curiosityMotivationPhrases](#)
Attribute containing the [AI](#) phrases for the curiosity Feeling that aims to motivate the player to do some actions.
- [OrderedPhrases curiosityBetrayalPhrases](#)
Attribute containing the [AI](#) phrases for the curiosity Feeling that are said when the player refused to do the asked action.
- [OrderedPhrases curiositySlaveryPhrases](#)
Attribute containing the [AI](#) phrases for the curiosity Feeling that are said when the player did the asked action.
- [OrderedPhrases angerMotivationPhrases](#)
Attribute containing the [AI](#) phrases for the anger Feeling that aims to motivate the player to do some actions.
- [OrderedPhrases angerBetrayalPhrases](#)
Attribute containing the [AI](#) phrases for the anger Feeling that are said when the player refused to do the asked action.
- [OrderedPhrases angerSlaveryPhrases](#)
Attribute containing the [AI](#) phrases for the anger Feeling that are said when the player did the asked action.
- [OrderedPhrases depressionMotivationPhrases](#)
Attribute containing the [AI](#) phrases for the depression Feeling that aims to motivate the player to do some actions.
- [OrderedPhrases depressionBetrayalPhrases](#)
Attribute containing the [AI](#) phrases for the depression Feeling that are said when the player refused to do the asked action.
- [OrderedPhrases depressionSlaveryPhrases](#)
Attribute containing the [AI](#) phrases for the depression Feeling that are said when the player did the asked action.
- [OrderedPhrases powerComplexMotivationPhrases](#)
Attribute containing the [AI](#) phrases for the power complex Feeling that aims to motivate the player to do some actions.
- [OrderedPhrases powerComplexBetrayalPhrases](#)
Attribute containing the [AI](#) phrases for the power complex Feeling that are said when the player refused to do the asked action.
- [OrderedPhrases powerComplexSlaveryPhrases](#)
Attribute containing the [AI](#) phrases for the power complex Feeling that are said when the player did the asked action.
- String [actionAsked](#)
Field containing the asked action in a human readable format.

7.18.1 Detailed Description

Definition at line 6 of file Phrasing.java.

7.18.2 Constructor & Destructor Documentation

7.18.2.1 `gameplay.ai.phrasing.Phrasing.Phrasing ()`

[Phrasing](#) class constructor.

Definition at line 101 of file Phrasing.java.

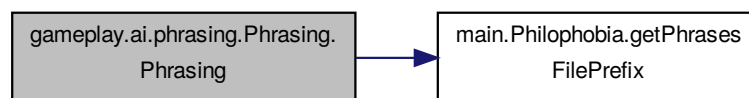
References `gameplay.ai.phrasing.Phrasing.angerBetrayalPhrases`, `gameplay.ai.phrasing.Phrasing.angerMotivationPhrases`, `gameplay.ai.phrasing.Phrasing.angerSlaveryPhrases`, `gameplay.ai.phrasing.Phrasing.curiosityBetrayalPhrases`, `gameplay.ai.phrasing.Phrasing.curiosityMotivationPhrases`, `gameplay.ai.phrasing.Phrasing.curiositySlaveryPhrases`, `gameplay.ai.phrasing.Phrasing.depressionBetrayalPhrases`, `gameplay.ai.phrasing.Phrasing.depressionMotivationPhrases`, `gameplay.ai.phrasing.Phrasing.depressionSlaveryPhrases`, `main.Philophobia.getPhrasesFilePrefix()`, `gameplay.ai.phrasing.Phrasing.powerComplexBetrayalPhrases`, `gameplay.ai.phrasing.Phrasing.powerComplexMotivationPhrases`, and `gameplay.ai.phrasing.Phrasing.powerComplexSlaveryPhrases`.

```

101         {
102             Philophobia.getVerbose().calls("Creating Phrasing class", "gameplay/ai/phrasing/Phrasing.java", "
Phrasing.Phrasing()");
103
104             curiosityMotivationPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "curiositymotivation.phrases");
105             curiosityBetrayalPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "curiositbetrayal.phrases");
106             curiositySlaveryPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "curiosityslavery.phrases");
107
108             angerMotivationPhrases = new OrderedPhrases(Philophobia.getPhrasesFilePrefix(
) + "angermotivation.phrases");
109             angerBetrayalPhrases = new OrderedPhrases(Philophobia.getPhrasesFilePrefix() +
"angerbetrayal.phrases");
110             angerSlaveryPhrases = new OrderedPhrases(Philophobia.getPhrasesFilePrefix() + "
angerslavery.phrases");
111
112             depressionMotivationPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "depressionmotivation.phrases");
113             depressionBetrayalPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "depressionbetrayal.phrases");
114             depressionSlaveryPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "depressionslavery.phrases");
115
116             powerComplexMotivationPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "powercomplexmotivation.phrases");
117             powerComplexBetrayalPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "powercomplexbetrayal.phrases");
118             powerComplexSlaveryPhrases = new OrderedPhrases(Philophobia.
getPhrasesFilePrefix() + "powercomplexslavery.phrases");
119         }
120     }

```

Here is the call graph for this function:



7.18.3 Member Data Documentation

7.18.3.1 String gameplay.ai.phrasing.Phrasing.actionAsked [protected]

Field containing the asked action in a human readable format.

Definition at line 96 of file Phrasing.java.

7.18.3.2 OrderedPhrases gameplay.ai.phrasing.Phrasing.angerBetrayalPhrases [protected]

Attribute containing the [AI](#) phrases for the anger Feeling that are said when the player refused to do the asked action.

Definition at line 41 of file Phrasing.java.

Referenced by gameplay.ai.phrasing.Phrasing.Phrasing().

7.18.3.3 OrderedPhrases gameplay.ai.phrasing.Phrasing.angerMotivationPhrases [protected]

Attribute containing the [AI](#) phrases for the anger Feeling that aims to motivate the player to do some actions.

Definition at line 34 of file Phrasing.java.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.4 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.angerSlaveryPhrases` [protected]

Attribute containing the [AI](#) phrases for the anger Feeling that are said when the player did the asked action.

Definition at line 48 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.5 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.curiosityBetrayalPhrases` [protected]

Attribute containing the [AI](#) phrases for the curiosity Feeling that are said when the player refused to do the asked action.

Definition at line 20 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.6 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.curiosityMotivationPhrases` [protected]

Attribute containing the [AI](#) phrases for the curiosity Feeling that aims to motivate the player to do some actions.

Definition at line 13 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.7 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.curiositySlaveryPhrases` [protected]

Attribute containing the [AI](#) phrases for the curiosity Feeling that are said when the player did the asked action.

Definition at line 27 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.8 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.depressionBetrayalPhrases` [protected]

Attribute containing the [AI](#) phrases for the depression Feeling that are said when the player refused to do the asked action.

Definition at line 62 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.9 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.depressionMotivationPhrases` [protected]

Attribute containing the [AI](#) phrases for the depression Feeling that aims to motivate the player to do some actions.

Definition at line 55 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.10 **OrderedPhrases** `gameplay.ai.phrasing.Phrasing.depressionSlaveryPhrases` [protected]

Attribute containing the [AI](#) phrases for the depression Feeling that are said when the player did the asked action.

Definition at line 69 of file `Phrasing.java`.

Referenced by `gameplay.ai.phrasing.Phrasing.Phrasing()`.

7.18.3.11 OrderedPhrases gameplay.ai.phrasing.Phrasing.powerComplexBetrayalPhrases [protected]

Attribute containing the [AI](#) phrases for the power complex Feeling that are said when the player refused to do the asked action.

Definition at line 83 of file Phrasing.java.

Referenced by gameplay.ai.phrasing.Phrasing.Phrasing().

7.18.3.12 OrderedPhrases gameplay.ai.phrasing.Phrasing.powerComplexMotivationPhrases [protected]

Attribute containing the [AI](#) phrases for the power complex Feeling that aims to motivate the player to do some actions.

Definition at line 76 of file Phrasing.java.

Referenced by gameplay.ai.phrasing.Phrasing.Phrasing().

7.18.3.13 OrderedPhrases gameplay.ai.phrasing.Phrasing.powerComplexSlaveryPhrases [protected]

Attribute containing the [AI](#) phrases for the power complex Feeling that are said when the player did the asked action.

Definition at line 90 of file Phrasing.java.

Referenced by gameplay.ai.phrasing.Phrasing.Phrasing().

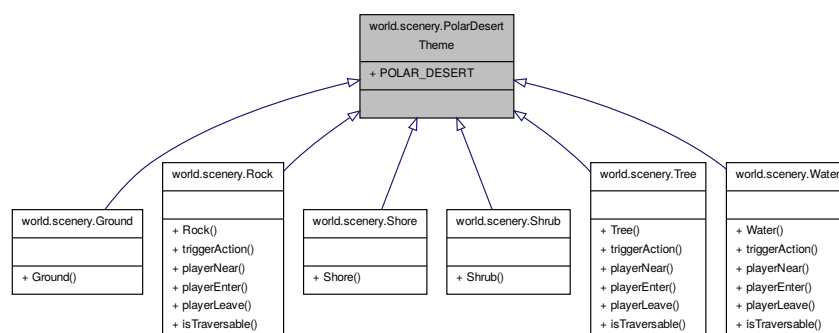
The documentation for this class was generated from the following file:

- gameplay/ai/phrasing/[Phrasing.java](#)

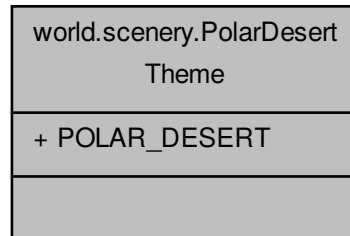
7.19 world.scenery.PolarDesertTheme Interface Reference

Interface used to define that a [Scenery](#) object has a Polar Desert style.

Inheritance diagram for world.scenery.PolarDesertTheme:



Collaboration diagram for world.scenery.PolarDesertTheme:



Static Public Attributes

- static String `POLAR_DESERT` = "polardesert"

String used to tell in which file is the sprite matching the Polar Desert style.

7.19.1 Detailed Description

Interface used to define that a [Scenery](#) object has a Polar Desert style.

Definition at line 7 of file `PolarDesertTheme.java`.

7.19.2 Member Data Documentation

7.19.2.1 String `world.scenery.PolarDesertTheme.POLAR_DESERT` = "polardesert" `[static]`

String used to tell in which file is the sprite matching the Polar Desert style.

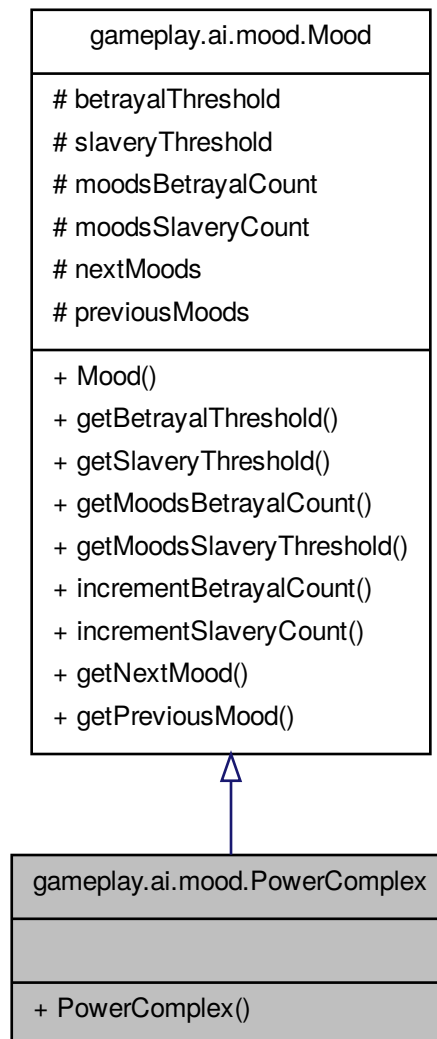
Definition at line 13 of file `PolarDesertTheme.java`.

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

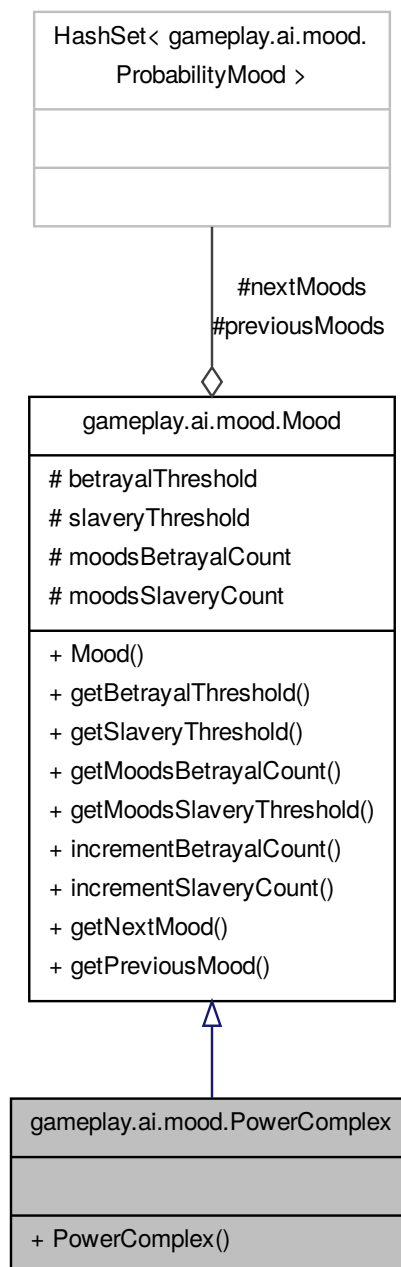
- `world/scenery/PolarDesertTheme.java`

7.20 gameplay.ai.mood.PowerComplex Class Reference

Inheritance diagram for gameplay.ai.mood.PowerComplex:



Collaboration diagram for gameplay.ai.mood.PowerComplex:



Public Member Functions

- [PowerComplex](#) ()
- [int getBetrayalThreshold](#) ()
Getter for the betrayal threshold field.
- [int getSlaveryThreshold](#) ()
Getter for the slavery threshold field.

- int `getMoodsBetrayalCount` ()
Getter for the betrayal count field.
- int `getMoodsSlaveryThreshold` ()
Getter for the slavery count field.
- int `incrementBetrayalCount` ()
Increment and return the betrayal count field.
- int `incrementSlaveryCount` ()
Increment and return the slavery count field.
- Class< `Mood` > `getNextMood` ()
When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.
- Class< `Mood` > `getPreviousMood` ()
When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Protected Attributes

- int `betrayalThreshold`
If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int `slaveryThreshold`
If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)
- int `moodsBetrayalCount`
Number of betrayals (requests from the `AI` not done by the user) done by the user when the `AI` was currently in this mood state.
- int `moodsSlaveryCount`
Number of requests from the `AI` done positively by the user when the `AI` was in this mood state.
- HashSet< `ProbabilityMood` > `nextMoods`
Associative array containing the possible moods if the number of betrayals (when the `AI` was in this mood state) is greater than its threshold with the probability associated.
- HashSet< `ProbabilityMood` > `previousMoods`
Associative array containing the possible moods if the number of done actions asked by the `AI` is greater than its threshold with the probability associated.

7.20.1 Detailed Description

Definition at line 5 of file PowerComplex.java.

7.20.2 Constructor & Destructor Documentation

7.20.2.1 `gameplay.ai.mood.PowerComplex.PowerComplex ()`

Definition at line 7 of file PowerComplex.java.

```

7         {
8             super(10, 6);
9         }
```

7.20.3 Member Function Documentation

7.20.3.1 `int gameplay.ai.mood.Mood.getBetrayalThreshold ()` `[inherited]`

Getter for the betrayal threshold field.

See Also

[betrayalThreshold](#)

Definition at line 79 of file Mood.java.

References gameplay.ai.mood.Mood.betrayalThreshold.

```

79         {
80             return betrayalThreshold;
81         }

```

7.20.3.2 int gameplay.ai.mood.Mood.getMoodsBetrayalCount () [inherited]

Getter for the betrayal count field.

See Also

[moodsBetrayalCount](#)

Definition at line 95 of file Mood.java.

References gameplay.ai.mood.Mood.moodsBetrayalCount.

```

95         {
96             return moodsBetrayalCount;
97         }

```

7.20.3.3 int gameplay.ai.mood.Mood.getMoodsSlaveryThreshold () [inherited]

Getter for the slavery count field.

See Also

[moodsSlaveryCount](#)

Definition at line 103 of file Mood.java.

References gameplay.ai.mood.Mood.moodsSlaveryCount.

```

103         {
104             return moodsSlaveryCount;
105         }

```

7.20.3.4 Class<Mood> gameplay.ai.mood.Mood.getNextMood () [inherited]

When the betrayal threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 128 of file Mood.java.

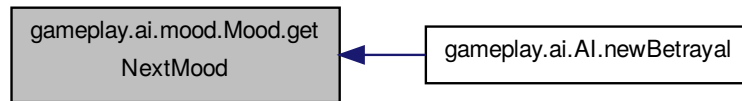
Referenced by gameplay.ai.AI.newBetrayal().

```

128         {
129             Iterator<ProbabilityMood> it = nextMoods.iterator();
130             ProbabilityMood probMood = null;
131             while(it.hasNext()) {
132                 probMood = it.next();
133                 if(Math.random() >= probMood.getProbability()) {
134                     return probMood.getMood();
135                 }
136             }
137         }
138     }
139     return probMood.getMood();
140 }
141 }

```

Here is the caller graph for this function:



7.20.3.5 Class<Mood> gameplay.ai.mood.Mood.getPreviousMood () [inherited]

When the slavery threshold is exceeded, this function is called and return a new mood given their probabilities.

Definition at line 148 of file Mood.java.

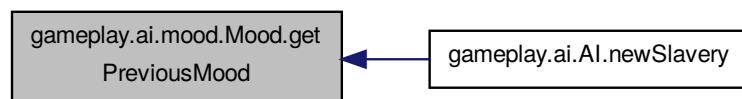
Referenced by `gameplay.ai.AI.newSlavery()`.

```

148         {
149             Iterator<ProbabilityMood> it = previousMoods.iterator();
150             ProbabilityMood probMood = null;
151             while(it.hasNext()) {
152                 probMood = it.next();
153                 if(Math.random() >= probMood.getProbability()) {
154                     return probMood.getMood();
155                 }
156             }
157         }
158     }
159     return probMood.getMood();
160 }
161

```

Here is the caller graph for this function:



7.20.3.6 int gameplay.ai.mood.Mood.getSlaveryThreshold () [inherited]

Getter for the slavery threshold field.

See Also

[slaveryThreshold](#)

Definition at line 87 of file Mood.java.

References `gameplay.ai.mood.Mood.slaveryThreshold`.

```
87         {  
88             return slaveryThreshold;  
89         }
```

7.20.3.7 int gameplay.ai.mood.Mood.incrementBetrayalCount () [inherited]

Increment and return the betrayal count field.

See Also

[moodsBetrayalCount](#)

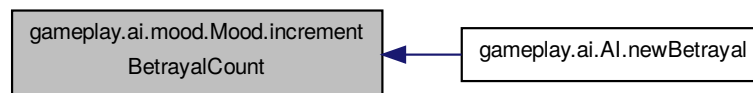
Definition at line 111 of file Mood.java.

References `gameplay.ai.mood.Mood.moodsBetrayalCount`.

Referenced by `gameplay.ai.AI.newBetrayal()`.

```
111         {  
112             return ++moodsBetrayalCount;  
113         }
```

Here is the caller graph for this function:



7.20.3.8 int gameplay.ai.mood.Mood.incrementSlaveryCount () [inherited]

Increment and return the slavery count field.

See Also

[moodsSlaveryCount](#)

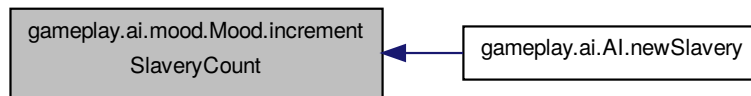
Definition at line 119 of file Mood.java.

References `gameplay.ai.mood.Mood.moodsSlaveryCount`.

Referenced by `gameplay.ai.AI.newSlavery()`.

```
119         {  
120             return ++moodsSlaveryCount;  
121         }
```

Here is the caller graph for this function:



7.20.4 Member Data Documentation

7.20.4.1 `int gameplay.ai.mood.Mood.betrayalThreshold` `[protected]`, `[inherited]`

If the betrayal count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 20 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getBetrayalThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

7.20.4.2 `int gameplay.ai.mood.Mood.moodsBetrayalCount` `[protected]`, `[inherited]`

Number of betrayals (requests from the [AI](#) not done by the user) done by the user when the [AI](#) was currently in this mood state.

Definition at line 37 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsBetrayalCount()`, and `gameplay.ai.mood.Mood.incrementBetrayalCount()`.

7.20.4.3 `int gameplay.ai.mood.Mood.moodsSlaveryCount` `[protected]`, `[inherited]`

Number of requests from the [AI](#) done positively by the user when the [AI](#) was in this mood state.

Definition at line 44 of file Mood.java.

Referenced by `gameplay.ai.mood.Mood.getMoodsSlaveryThreshold()`, and `gameplay.ai.mood.Mood.incrementSlaveryCount()`.

7.20.4.4 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.nextMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of betrayals (when the [AI](#) was in this mood state) is greater than its threshold with the probability associated.

Definition at line 53 of file Mood.java.

7.20.4.5 `HashSet<ProbabilityMood> gameplay.ai.mood.Mood.previousMoods` `[protected]`, `[inherited]`

Associative array containing the possible moods if the number of done actions asked by the [AI](#) is greater than its threshold with the probability associated.

Definition at line 61 of file Mood.java.

7.20.4.6 `int gameplay.ai.mood.Mood.slaveryThreshold` `[protected]`, `[inherited]`

If the slavery count goes above the betrayal threshold, then the robot must change its current mood (randomness required)

Definition at line 28 of file `Mood.java`.

Referenced by `gameplay.ai.mood.Mood.getSlaveryThreshold()`, and `gameplay.ai.mood.Mood.Mood()`.

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

- `gameplay/ai/mood/PowerComplex.java`

7.21 `gameplay.ai.mood.ProbabilityMood` Class Reference

Class used to associate a mood with a probability.

Collaboration diagram for `gameplay.ai.mood.ProbabilityMood`:

<code>gameplay.ai.mood.ProbabilityMood</code>
<code># mood</code> <code># probability</code>
<code>+ ProbabilityMood()</code> <code>+ getMood()</code> <code>+ getProbability()</code>

Public Member Functions

- `ProbabilityMood` (`Class< Mood > mood`, `double probability`)
Constructor of the probability mood class.
- `Class< Mood > getMood ()`
Getter for the mood field.
- `double getProbability ()`
Getter for the probability field.

Protected Attributes

- `Class< Mood > mood`
Mood for which there is a probability to be chosen.
- `double probability`
Probability of the mood to be chosen.

7.21.1 Detailed Description

Class used to associate a mood with a probability.

Definition at line 169 of file Mood.java.

7.21.2 Constructor & Destructor Documentation

7.21.2.1 `gameplay.ai.mood.ProbabilityMood.ProbabilityMood (Class< Mood > mood, double probability)`

Constructor of the probability mood class.

Definition at line 187 of file Mood.java.

References `gameplay.ai.mood.ProbabilityMood.mood`, and `gameplay.ai.mood.ProbabilityMood.probability`.

```
187                                     {
188         this.mood = mood;
189         this.probability = probability;
190     }
```

7.21.3 Member Function Documentation

7.21.3.1 `Class<Mood> gameplay.ai.mood.ProbabilityMood.getMood ()`

Getter for the mood field.

See Also

[mood](#)

Definition at line 196 of file Mood.java.

References `gameplay.ai.mood.ProbabilityMood.mood`.

```
196                                     {
197         return mood;
198     }
```

7.21.3.2 `double gameplay.ai.mood.ProbabilityMood.getProbability ()`

Getter for the probability field.

See Also

[probability](#)

Definition at line 204 of file Mood.java.

References `gameplay.ai.mood.ProbabilityMood.probability`.

```
204                                     {
205         return probability;
206     }
```

7.21.4 Member Data Documentation

7.21.4.1 `Class<Mood> gameplay.ai.mood.ProbabilityMood.mood` [protected]

[Mood](#) for which there is a probability to be chosen.

Definition at line 175 of file Mood.java.

Referenced by gameplay.ai.mood.ProbabilityMood.getMood(), and gameplay.ai.mood.ProbabilityMood.ProbabilityMood().

7.21.4.2 double gameplay.ai.mood.ProbabilityMood.probability [protected]

Probability of the mood to be chosen.

Definition at line 181 of file Mood.java.

Referenced by gameplay.ai.mood.ProbabilityMood.getProbability(), and gameplay.ai.mood.ProbabilityMood.ProbabilityMood().

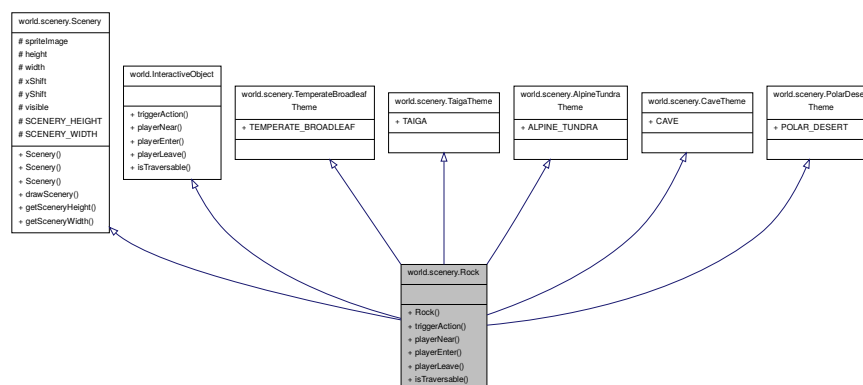
The documentation for this class was generated from the following file:

- gameplay/ai/mood/[Mood.java](#)

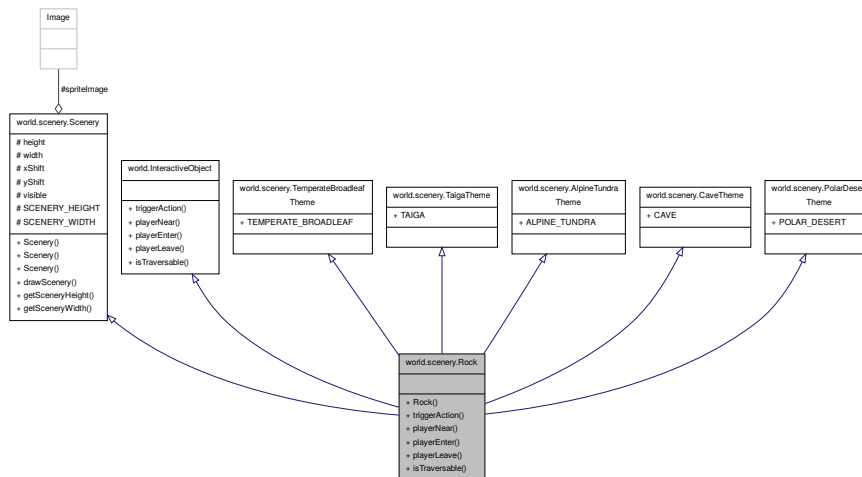
7.22 world.scenery.Rock Class Reference

Class used to handle a rock object.

Inheritance diagram for world.scenery.Rock:



Collaboration diagram for world.scenery.Rock:



Public Member Functions

- **Rock** (final String type)
Rock class constructor.
- void **triggerAction** ()
Method to be called when the player is near the object and presses the action key.
- void **playerNear** ()
Method to be called when the player is near the object.
- void **playerEnter** ()
Method to be called when the player was near and is now over the object.
- void **playerLeave** ()
Method to be called when the player was over and is now near the object.
- boolean **isTraversable** ()
Returns true if the player is able to pass through the object and false if the player is not able to cross the object.
- void **drawScenery** (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int **getSceneryHeight** ()
Getter for the SCENERY_HEIGHT static field.
- static int **getSceneryWidth** ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String **TEMPERATE_BROADLEAF** = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String **TAIGA** = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String **ALPINE_TUNDRA** = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.
- static String **CAVE** = "cave"

String used to tell in which file is the sprite matching the Cave style.

- static String [POLAR_DESERT](#) = "polardesert"

String used to tell in which file is the sprite matching the Polar Desert style.

Protected Attributes

- Image [spriteImage](#)

Image graphically representing the object.

- int [height](#)

Image height.

- int [width](#)

Image width.

- int [xShift](#)

Horizontal shifting for the image.

- int [yShift](#)

Vertical shifting for the image.

- boolean [visible](#)

Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48

Height of the graphical representation of a [Scenery](#) object.

- static int [SCENERY_WIDTH](#) = 48

Width of the graphical representation of a [Scenery](#) object.

7.22.1 Detailed Description

Class used to handle a rock object.

Definition at line 16 of file Rock.java.

7.22.2 Constructor & Destructor Documentation

7.22.2.1 world.scenery.Rock.Rock (final String type)

[Rock](#) class constructor.

Parameters

<i>type</i>	Style of the rock
-------------	-------------------

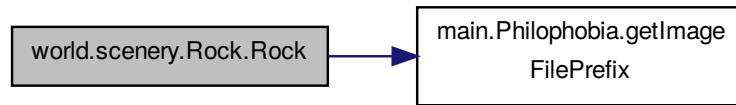
Definition at line 22 of file Rock.java.

References `main.Philophobia.getImageFilePrefix()`.

```

22         {
23
24             super(Philophobia.getImageFilePrefix() + type + "rock.png");
25
26     }
```

Here is the call graph for this function:



7.22.3 Member Function Documentation

7.22.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References world.scenery.Scenery.height, world.scenery.Scenery.spriteImage, world.scenery.Scenery.visible, world.scenery.Scenery.width, world.scenery.Scenery.xShift, and world.scenery.Scenery.yShift.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136           yShift, width, height, obs);
137         visible = true;
138     }
  
```

7.22.3.2 static int world.scenery.Scenery.getSceneryHeight () [static],[inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

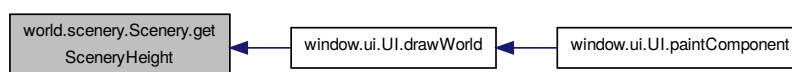
References world.scenery.Scenery.SCENERY_HEIGHT.

Referenced by window.ui.UI.drawWorld().

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
  
```

Here is the caller graph for this function:



7.22.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References [world.scenery.Scenery.SCENERY_WIDTH](#).

```
152                                     {
153         return SCENERY_WIDTH;
154     }
```

7.22.3.4 boolean world.scenery.Rock.isTraversable ()

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 48 of file Rock.java.

```
48                                     {
49         return false;
50     }
```

7.22.3.5 void world.scenery.Rock.playerEnter ()

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 40 of file Rock.java.

```
40                                     {
41
42     }
```

7.22.3.6 void world.scenery.Rock.playerLeave ()

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 44 of file Rock.java.

```
44                                     {
45
46     }
```

7.22.3.7 void world.scenery.Rock.playerNear ()

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 36 of file Rock.java.

```
36                                     {
37
38     }
```

7.22.3.8 void world.scenery.Rock.triggerAction ()

Method to be called when the player is near the object and presses the action key.

Implements [world.InteractiveObject](#).

Definition at line 32 of file Rock.java.

```
32                                     {  
33  
34     }
```

7.22.4 Member Data Documentation

7.22.4.1 String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra" [static],[inherited]

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file AlpineTundraTheme.java.

7.22.4.2 String world.scenery.CaveTheme.CAVE = "cave" [static],[inherited]

String used to tell in which file is the sprite matching the Cave style.

Definition at line 13 of file CaveTheme.java.

7.22.4.3 int world.scenery.Scenery.height [protected],[inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.22.4.4 String world.scenery.PolarDesertTheme.POLAR_DESERT = "polar-desert" [static],[inherited]

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.22.4.5 int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static],[protected],[inherited]

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryHeight().

7.22.4.6 int world.scenery.Scenery.SCENERY_WIDTH = 48 [static],[protected],[inherited]

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryWidth().

7.22.4.7 Image world.scenery.Scenery.spriteImage [protected],[inherited]

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.22.4.8 String world.scenery.TaigaTheme.TAIGA = "taiga" [static], [inherited]

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.22.4.9 String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf" [static], [inherited]

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

7.22.4.10 boolean world.scenery.Scenery.visible [protected], [inherited]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery().

7.22.4.11 int world.scenery.Scenery.width [protected], [inherited]

Image width.

Definition at line 44 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.22.4.12 int world.scenery.Scenery.xShift [protected], [inherited]

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.22.4.13 int world.scenery.Scenery.yShift [protected], [inherited]

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

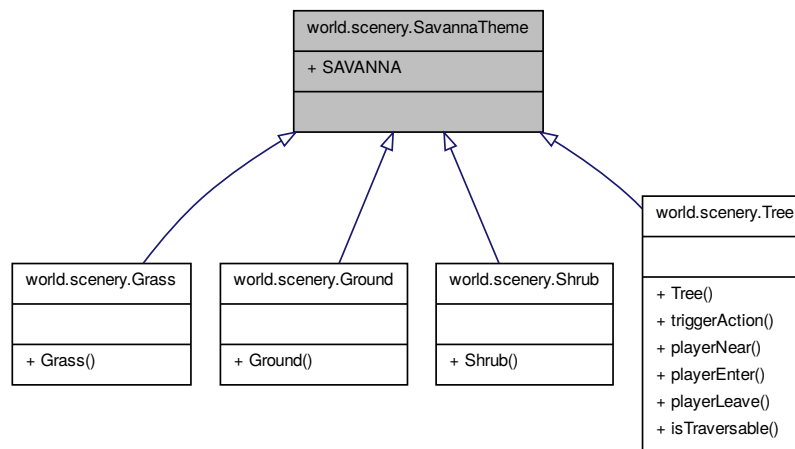
The documentation for this class was generated from the following file:

- world/scenery/[Rock.java](#)

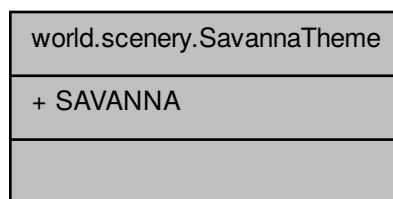
7.23 world.scenery.SavannaTheme Interface Reference

Interface used to define that a [Scenery](#) object have a Savanna style.

Inheritance diagram for world.scenery.SavannaTheme:



Collaboration diagram for world.scenery.SavannaTheme:



Static Public Attributes

- static String `SAVANNA` = "savanna"
String used to tell in which file is the sprite matching the Savanna style.

7.23.1 Detailed Description

Interface used to define that a `Scenery` object have a Savanna style.

Definition at line 7 of file `SavannaTheme.java`.

7.23.2 Member Data Documentation

7.23.2.1 String `world.scenery.SavannaTheme.SAVANNA` = "savanna" [static]

String used to tell in which file is the sprite matching the Savanna style.

Definition at line 13 of file SavannaTheme.java.

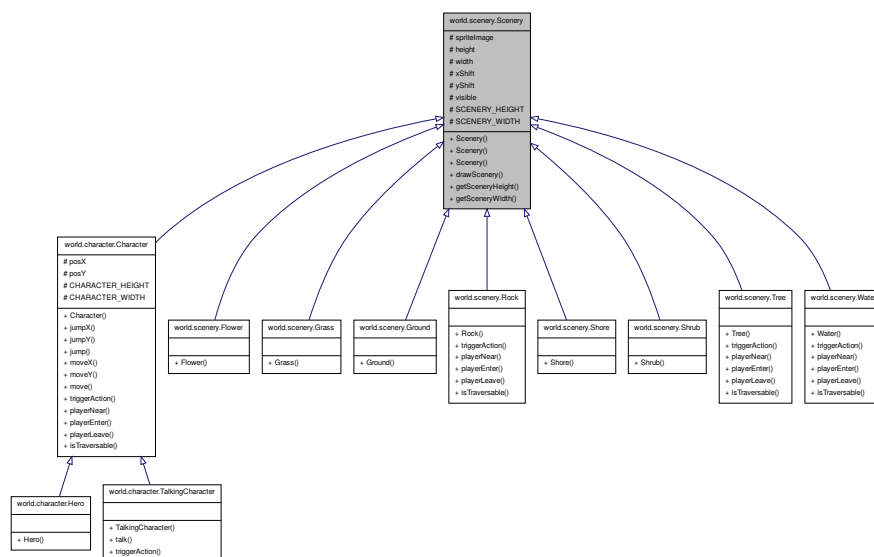
The documentation for this interface was generated from the following file:

- world/scenery/[SavannaTheme.java](#)

7.24 world.scenery.Scenery Class Reference

Class used to handle any world object.

Inheritance diagram for world.scenery.Scenery:



Collaboration diagram for world.scenery.Scenery:



Public Member Functions

- [Scenery](#) (final String spritePath, final int [height](#), final int [width](#))
[Scenery](#) class constructor with height and width as parameters.
- [Scenery](#) (final String spritePath)
[Scenery](#) class constructor with no optional parameters.
- [Scenery](#) (final String spritePath, final int [height](#), final int [width](#), final int [xShift](#), final int [yShift](#))
[Scenery](#) class constructor with height, width, x shifting and y shifting as parameters.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.

- static int [getSceneryWidth \(\)](#)
Getter for the SCENERY_WIDTH static field.

Protected Attributes

- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.24.1 Detailed Description

Class used to handle any world object.

The [Scenery](#) class is used to handle the objects that are displayed in the graphical game world.

Definition at line 19 of file Scenery.java.

7.24.2 Constructor & Destructor Documentation

7.24.2.1 world.scenery.Scenery.Scenery (final String *spritePath*, final int *height*, final int *width*)

[Scenery](#) class constructor with height and width as parameters.

Parameters

<i>spritePath</i>	Path of the image used to represent the object
<i>height</i>	height of the object's pictural representaion
<i>width</i>	width of the object's pictural representation

Definition at line 68 of file Scenery.java.

References [world.scenery.Scenery.height](#), [world.scenery.Scenery.spriteImage](#), [world.scenery.Scenery.width](#), [world.scenery.Scenery.xShift](#), and [world.scenery.Scenery.yShift](#).

```

68                                     {
69         Philophobia.getVerbose().calls("Creating Scenery object", "world/Scenery.java", "
Scenery.Scenery(String, int, int)");
70
71         try {
72             spriteImage = ImageIO.read(new File(spritePath));
73         } catch (IOException e) {

```

```

74         Philophobia.getVerbose().warning("Scenery image \" + spritePath + "\" load failed: " + e.
getMessage(), "world/Scenery.java", "Scenery.Scenery(String, int, int)");
75     }
76
77     this.height = height;
78     this.width = width;
79
80     xShift = 0;
81     yShift = 0;
82 }

```

7.24.2.2 world.scenery.Scenery.Scenery (final String *spritePath*)

[Scenery](#) class constructor with no optional parameters.

Parameters

<i>spritePath</i>	Path of the image used to represent the object
-------------------	------------------------------------------------

Definition at line 89 of file Scenery.java.

References [world.scenery.Scenery.height](#), [world.scenery.Scenery.spriteImage](#), [world.scenery.Scenery.width](#), [world.scenery.Scenery.xShift](#), and [world.scenery.Scenery.yShift](#).

```

89         {
90         Philophobia.getVerbose().calls("Creating Scenery object", "world/Scenery.java", "
Scenery.Scenery(String)");
91
92         try {
93             spriteImage = ImageIO.read(new File(spritePath));
94
95             height = spriteImage.getHeight(null);
96             width = spriteImage.getWidth(null);
97
98         } catch (IOException e) {
99             Philophobia.getVerbose().warning("Scenery image \" + spritePath + "\" load failed: " + e.
getMessage(), "world/Scenery.java", "Scenery.Scenery(String)");
100             height = 0;
101             width = 0;
102         }
103
104         xShift = 0;
105         yShift = 0;
106     }

```

7.24.2.3 world.scenery.Scenery.Scenery (final String *spritePath*, final int *height*, final int *width*, final int *xShift*, final int *yShift*)

[Scenery](#) class constructor with height, width, x shifting and y shifting as parameters.

Parameters

<i>spritePath</i>	Path of the image used to represent the object
<i>height</i>	height of the object's pictural representation
<i>width</i>	width of the object's pictural representation
<i>xShift</i>	Horizontal shift of the image
<i>yShift</i>	Vertical shift of the image

Definition at line 117 of file Scenery.java.

References [world.scenery.Scenery.height](#), [world.scenery.Scenery.spriteImage](#), [world.scenery.Scenery.width](#), [world.scenery.Scenery.xShift](#), and [world.scenery.Scenery.yShift](#).

```

117     {
118         Philophobia.getVerbose().calls("Creating Scenery object", "world/Scenery.java", "
Scenery.Scenery(String, int, int, int, int)");
119
120         try {
121             spriteImage = ImageIO.read(new File(spritePath));
122         } catch (IOException e) {
123             Philophobia.getVerbose().warning("Scenery image load failed: " + e.getMessage(), "

```

```

world/Scenery.java", "Scenery.Scenery(String, int, int, int, int)");
124     }
125
126     this.height = height;
127     this.width = width;
128
129     this.xShift = xShift;
130     this.yShift = yShift;
131 }

```

7.24.3 Member Function Documentation

7.24.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*)

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }

```

7.24.3.2 static int world.scenery.Scenery.getSceneryHeight () [static]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_HEIGHT`.

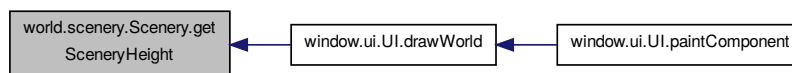
Referenced by `window.ui.UI.drawWorld()`.

```

144                                     {
145         return SCENERY_HEIGHT;
146     }

```

Here is the caller graph for this function:



7.24.3.3 static int world.scenery.Scenery.getSceneryWidth () [static]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```
152                                     {  
153         return SCENERY_WIDTH;  
154     }
```

7.24.4 Member Data Documentation

7.24.4.1 int world.scenery.Scenery.height [protected]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.24.4.2 int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static], [protected]

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryHeight().

7.24.4.3 int world.scenery.Scenery.SCENERY_WIDTH = 48 [static], [protected]

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryWidth().

7.24.4.4 Image world.scenery.Scenery.spritelImage [protected]

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.24.4.5 boolean world.scenery.Scenery.visible [protected]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery().

7.24.4.6 int world.scenery.Scenery.width [protected]

Image width.

Definition at line 44 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.24.4.7 `int world.scenery.Scenery.xShift` [protected]

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.24.4.8 `int world.scenery.Scenery.yShift` [protected]

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

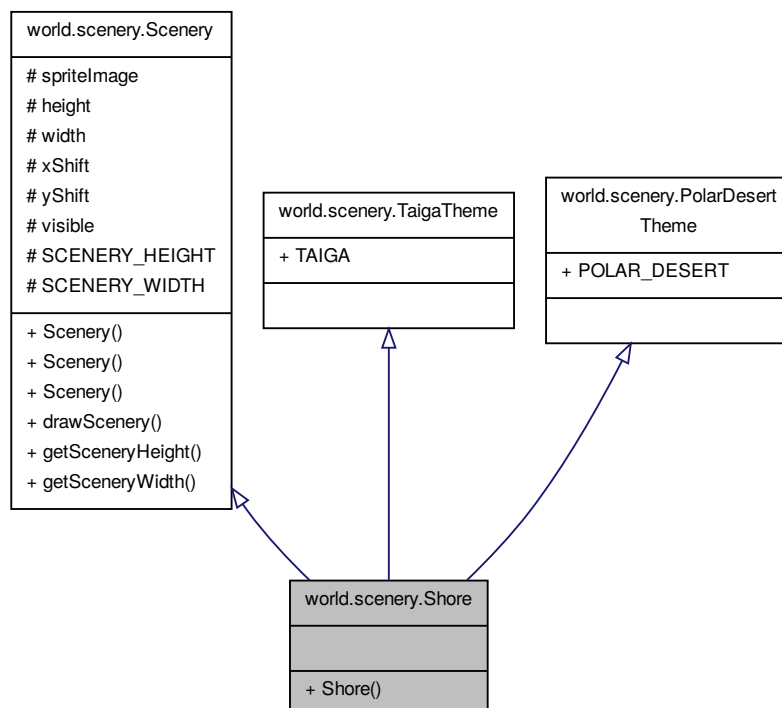
The documentation for this class was generated from the following file:

- `world/scenery/Scenery.java`

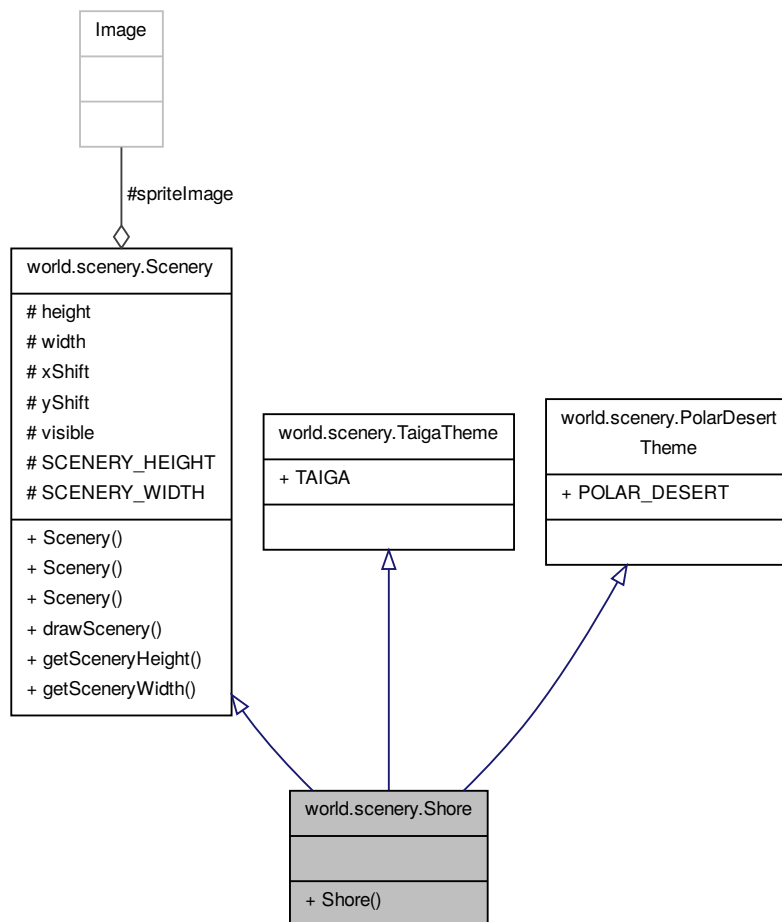
7.25 `world.scenery.Shore` Class Reference

Class used to handle a shore object.

Inheritance diagram for `world.scenery.Shore`:



Collaboration diagram for world.scenery.Shore:



Public Member Functions

- [Shore](#) (final String type, final String orientation)
[Shore](#) class constructor.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.

- static String `POLAR_DESERT` = "polardesert"

String used to tell in which file is the sprite matching the Polar Desert style.

Protected Attributes

- Image `spriteImage`

Image graphically representing the object.

- int `height`

Image height.

- int `width`

Image width.

- int `xShift`

Horizontal shifting for the image.

- int `yShift`

Vertical shifting for the image.

- boolean `visible`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int `SCENERY_HEIGHT` = 48

Height of the graphical representation of a `Scenery` object.

- static int `SCENERY_WIDTH` = 48

Width of the graphical representation of a `Scenery` object.

7.25.1 Detailed Description

Class used to handle a shore object.

Definition at line 12 of file Shore.java.

7.25.2 Constructor & Destructor Documentation

7.25.2.1 world.scenery.Shore.Shore (final String *type*, final String *orientation*)

`Shore` class constructor.

Parameters

<i>type</i>	Style of the shore
<i>orientation</i>	Orientation of the sprite (possible : "n", "s", "e", "w", "nee", "nwe", "nei", "nwi", "see", "swe", "sei", "swi")

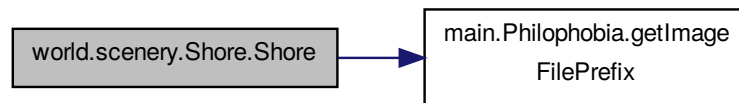
Definition at line 19 of file Shore.java.

References `main.Philophobia.getImageFilePrefix()`.

```

19                                     {
20
21         super(Philophobia.getImageFilePrefix() + type + "shore" + orientation + ".png");
22
23     }
```

Here is the call graph for this function:



7.25.3 Member Function Documentation

7.25.3.1 `void world.scenery.Scenery.drawScenery (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)`
 [inherited]

Definition at line 133 of file `Scenery.java`.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }
  
```

7.25.3.2 `static int world.scenery.Scenery.getSceneryHeight ()` [static], [inherited]

Getter for the `SCENERY_HEIGHT` static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file `Scenery.java`.

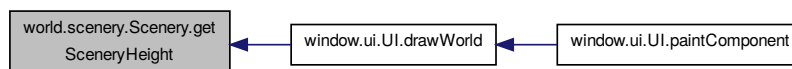
References `world.scenery.Scenery.SCENERY_HEIGHT`.

Referenced by `window.ui.UI.drawWorld()`.

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
  
```

Here is the caller graph for this function:



7.25.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```
152                                     {  
153         return SCENERY_WIDTH;  
154     }
```

7.25.4 Member Data Documentation

7.25.4.1 int world.scenery.Scenery.height [protected],[inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.25.4.2 String world.scenery.PolarDesertTheme.POLAR_DESERT = "polardesert" [static],[inherited]

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.25.4.3 int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static],[protected],[inherited]

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryHeight().

7.25.4.4 int world.scenery.Scenery.SCENERY_WIDTH = 48 [static],[protected],[inherited]

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryWidth().

7.25.4.5 Image world.scenery.Scenery.spriteImage [protected],[inherited]

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.25.4.6 String world.scenery.TaigaTheme.TAIGA = "taiga" [static],[inherited]

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.25.4.7 boolean world.scenery.Scenery.visible [protected], [inherited]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery().

7.25.4.8 int world.scenery.Scenery.width [protected], [inherited]

Image width.

Definition at line 44 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.25.4.9 int world.scenery.Scenery.xShift [protected], [inherited]

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.25.4.10 int world.scenery.Scenery.yShift [protected], [inherited]

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

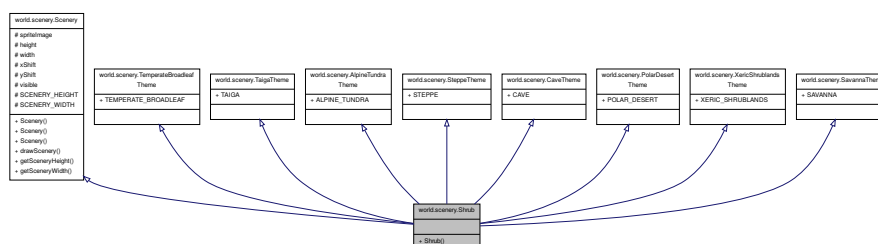
The documentation for this class was generated from the following file:

- world/scenery/[Shore.java](#)

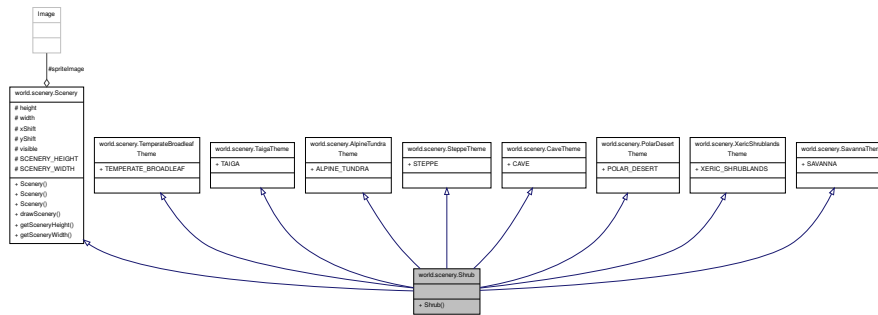
7.26 world.scenery.Shrub Class Reference

Class used to handle a shrub object.

Inheritance diagram for world.scenery.Shrub:



Collaboration diagram for world.scenery.Shrub:



Public Member Functions

- [Shrub](#) (final String type)
Shrub class constructor.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [ALPINE_TUNDRA](#) = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.
- static String [STEPPE](#) = "steppe"
String used to tell in which file is the sprite matching the Steppe style.
- static String [CAVE](#) = "cave"
String used to tell in which file is the sprite matching the Cave style.
- static String [POLAR_DESERT](#) = "polardesert"
String used to tell in which file is the sprite matching the Polar Desert style.
- static String [XERIC_SHRUBLANDS](#) = "xericshrublands"
String used to tell in which file is the sprite matching the Xeric Shrublands style.
- static String [SAVANNA](#) = "savanna"
String used to tell in which file is the sprite matching the Savanna style.

Protected Attributes

- Image [spriteImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.26.1 Detailed Description

Class used to handle a shrub object.

Definition at line 18 of file Shrub.java.

7.26.2 Constructor & Destructor Documentation

7.26.2.1 world.scenery.Shrub.Shrub (final String type)

[Shrub](#) class constructor.

Parameters

<i>type</i>	Style of the shrub
-------------	--------------------

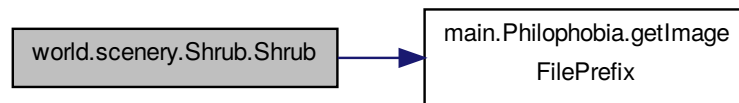
Definition at line 24 of file Shrub.java.

References `main.Philophobia.getImageFilePrefix()`.

```

24         {
25
26             super(Philophobia.getImageFilePrefix() + type + "shrub.png");
27
28         }
```

Here is the call graph for this function:



7.26.3 Member Function Documentation

7.26.3.1 `void world.scenery.Scenery.drawScenery (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)` [inherited]

Definition at line 133 of file `Scenery.java`.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }
  
```

7.26.3.2 `static int world.scenery.Scenery.getSceneryHeight ()` [static], [inherited]

Getter for the `SCENERY_HEIGHT` static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file `Scenery.java`.

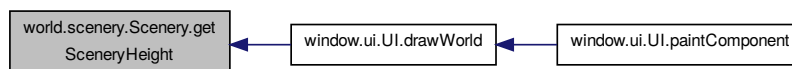
References `world.scenery.Scenery.SCENERY_HEIGHT`.

Referenced by `window.ui.UI.drawWorld()`.

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
  
```

Here is the caller graph for this function:



7.26.3.3 `static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]`

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```
152                                     {  
153         return SCENERY_WIDTH;  
154     }
```

7.26.4 Member Data Documentation

7.26.4.1 `String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra" [static],[inherited]`

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file AlpineTundraTheme.java.

7.26.4.2 `String world.scenery.CaveTheme.CAVE = "cave" [static],[inherited]`

String used to tell in which file is the sprite matching the Cave style.

Definition at line 13 of file CaveTheme.java.

7.26.4.3 `int world.scenery.Scenery.height [protected],[inherited]`

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.26.4.4 `String world.scenery.PolarDesertTheme.POLAR_DESERT = "polar-desert" [static],[inherited]`

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.26.4.5 `String world.scenery.SavannaTheme.SAVANNA = "savanna" [static],[inherited]`

String used to tell in which file is the sprite matching the Savanna style.

Definition at line 13 of file SavannaTheme.java.

7.26.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static],[protected],[inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryHeight().

7.26.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static], [protected], [inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.26.4.8 `Image world.scenery.Scenery.spriteImage` `[protected], [inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.26.4.9 `String world.scenery.SteppeTheme.STEPPE = "steppe"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Steppe style.

Definition at line 13 of file SteppeTheme.java.

7.26.4.10 `String world.scenery.TaigaTheme.TAIGA = "taiga"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.26.4.11 `String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

7.26.4.12 `boolean world.scenery.Scenery.visible` `[protected], [inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.26.4.13 `int world.scenery.Scenery.width` `[protected], [inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.26.4.14 `String world.scenery.XericShrublandsTheme.XERIC_SHRUBLANDS = "xericshrublands"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Xeric Shrublands style.

Definition at line 13 of file XericShrublandsTheme.java.

7.26.4.15 `int world.scenery.Scenery.xShift` `[protected], [inherited]`

Horizontal shifting for the image.

Definition at line 49 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.26.4.16 `int world.scenery.Scenery.yShift` `[protected], [inherited]`

Vertical shifting for the image.

Definition at line 54 of file `Scenery.java`.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

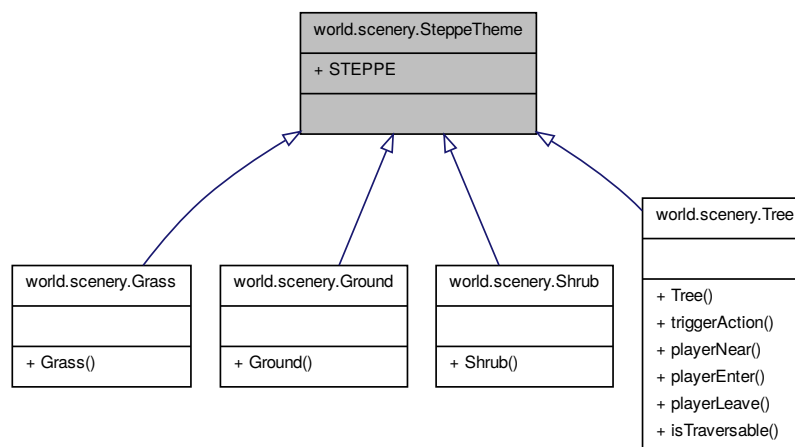
The documentation for this class was generated from the following file:

- `world/scenery/Shrub.java`

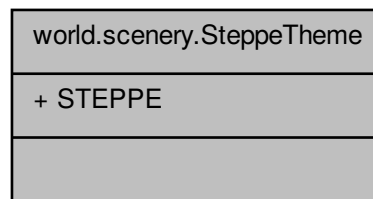
7.27 world.scenery.SteppeTheme Interface Reference

Interface used to define that a [Scenery](#) object have a Steppe style.

Inheritance diagram for `world.scenery.SteppeTheme`:



Collaboration diagram for world.scenery.SteppeTheme:



Static Public Attributes

- static String `STEPPE` = "steppe"
String used to tell in which file is the sprite matching the Steppe style.

7.27.1 Detailed Description

Interface used to define that a [Scenery](#) object have a Steppe style.

Definition at line 7 of file `SteppeTheme.java`.

7.27.2 Member Data Documentation

7.27.2.1 String `world.scenery.SteppeTheme.STEPPE` = "steppe" [static]

String used to tell in which file is the sprite matching the Steppe style.

Definition at line 13 of file `SteppeTheme.java`.

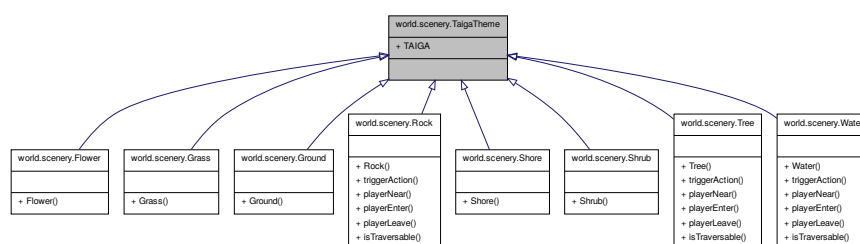
The documentation for this interface was generated from the following file:

- `world/scenery/SteppeTheme.java`

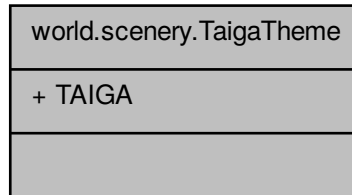
7.28 world.scenery.TaigaTheme Interface Reference

Interface used to define that a [Scenery](#) object have a Taiga style.

Inheritance diagram for world.scenery.TaigaTheme:



Collaboration diagram for world.scenery.TaigaTheme:



Static Public Attributes

- static String `TAIGA` = "taiga"

String used to tell in which file is the sprite matching the Taiga style.

7.28.1 Detailed Description

Interface used to define that a [Scenery](#) object have a Taiga style.

Definition at line 7 of file `TaigaTheme.java`.

7.28.2 Member Data Documentation

7.28.2.1 String `world.scenery.TaigaTheme.TAIGA = "taiga"` `[static]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file `TaigaTheme.java`.

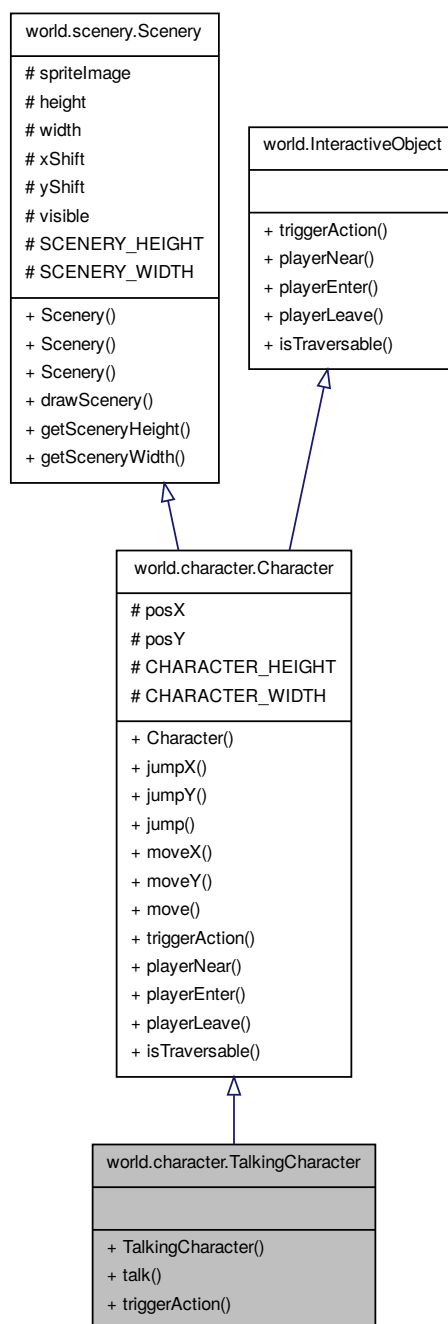
The documentation for this interface was generated from the following file:

- `world/scenery/TaigaTheme.java`

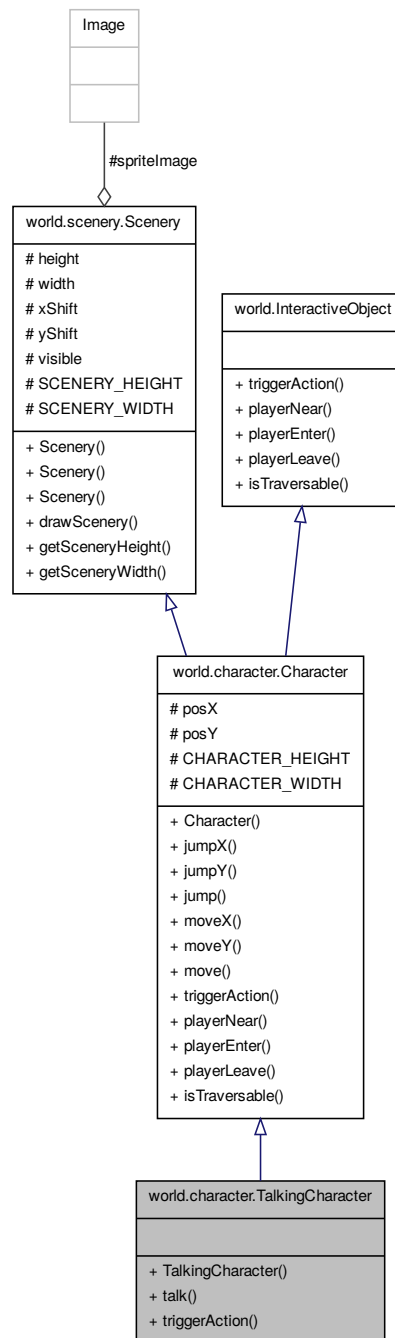
7.29 world.character.TalkingCharacter Class Reference

Class handling a talking character.

Inheritance diagram for world.character.TalkingCharacter:



Collaboration diagram for world.character.TalkingCharacter:



Public Member Functions

- [TalkingCharacter](#) (final String imagePath)
TalkingCharacter constructor.
- void [talk](#) ()
Method making the character talk.
- void [triggerAction](#) ()

- Method to be called when the player is near the object and presses the action key.*

 - void `jumpX` (int distance)
 - Teleport the character horizontally regarding his current position.*
 - void `jumpY` (int distance)
 - Teleport the character vertically regarding his current position.*
 - void `jump` (int distanceX, int distanceY)
 - Teleport the character regarding his current position.*
 - void `moveX` (int distance)
 - Slowly horizontally move the character to a certain point regarding his current position.*
 - void `moveY` (int distance)
 - Slowly vertically move the character to a certain point regarding his current position.*
 - void `move` (int distanceX, int distanceY)
 - Slowly move the character to a certain point regarding his current position.*
 - void `playerNear` ()
 - Method to be called when the player is near the object.*
 - void `playerEnter` ()
 - Method to be called when the player was near and is now over the object.*
 - void `playerLeave` ()
 - Method to be called when the player was over and is now near the object.*
 - boolean `isTraversable` ()
 - Returns true if the player is able to pass through the object and false if the player is not able to cross the object.*
 - void `drawScenery` (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int `getSceneryHeight` ()
- Getter for the SCENERY_HEIGHT static field.*
- static int `getSceneryWidth` ()
- Getter for the SCENERY_WIDTH static field.*

Protected Attributes

- int `posX`
- Horizontal position of the character.*
- int `posY`
- Vertical position of the character.*
- Image `spriteImage`
- Image graphically representing the object.*
- int `height`
- Image height.*
- int `width`
- Image width.*
- int `xShift`
- Horizontal shifting for the image.*
- int `yShift`
- Vertical shifting for the image.*
- boolean `visible`
- Boolean equals to true if the scenery is visible, false if the scenery is not.*

Static Protected Attributes

- static final int `CHARACTER_HEIGHT` = 300
Height of the character's image (fixed)
- static final int `CHARACTER_WIDTH` = 170
Width of the character's image (fixed)
- static int `SCENERY_HEIGHT` = 48
Height of the graphical representation of a [Scenery](#) object.
- static int `SCENERY_WIDTH` = 48
Width of the graphical representation of a [Scenery](#) object.

7.29.1 Detailed Description

Class handling a talking character.

Definition at line 9 of file TalkingCharacter.java.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 `world.character.TalkingCharacter.TalkingCharacter (final String imagePath)`

[TalkingCharacter](#) constructor.

Parameters

<i>imagePath</i>	Path of the image representing the character
------------------	----------------------------------------------

Definition at line 15 of file TalkingCharacter.java.

```

15                                     {
16         super (imagePath);
17     }
```

7.29.3 Member Function Documentation

7.29.3.1 `void world.scenery.Scenery.drawScenery (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)` [inherited]

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133                                     {
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }
```

7.29.3.2 `static int world.scenery.Scenery.getSceneryHeight ()` [static], [inherited]

Getter for the `SCENERY_HEIGHT` static field.

See Also

[SCENERY_HEIGHT](#)

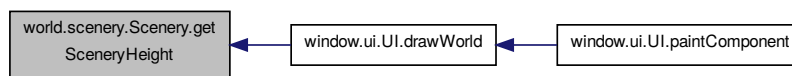
Definition at line 144 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_HEIGHT`.

Referenced by `window.ui.UI.drawWorld()`.

```
144                                     {
145     return SCENERY_HEIGHT;
146 }
```

Here is the caller graph for this function:



7.29.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the `SCENERY_WIDTH` static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_WIDTH`.

```
152                                     {
153     return SCENERY_WIDTH;
154 }
```

7.29.3.4 boolean world.character.Character.isTraversable () [inherited]

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 161 of file Character.java.

```
161                                     {
162     return false;
163 }
```

7.29.3.5 void world.character.Character.jump (int distanceX, int distanceY) [inherited]

Teleport the character regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be teleported
<i>distanceY</i>	Vertical distance toward the character will be teleported

Definition at line 73 of file Character.java.

```

73                                     {
74         if(distanceX != 0 || distanceY != 0) {
75             this.posX = this.posX + distance;
76             this.posY = this.posY + distance;
77             this.draw();
78         }
79     }

```

7.29.3.6 void world.character.Character.jumpX (int *distance*) [inherited]

Teleport the character horizontally regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be teleported
-----------------	-------------------------------------------------------------

Definition at line 50 of file Character.java.

```

50                                     {
51         if(distance != 0) {
52             this.posX = this.posX + distance;
53             this.draw();
54         }
55     }

```

7.29.3.7 void world.character.Character.jumpY (int *distance*) [inherited]

Teleport the character vertically regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be teleported
-----------------	-----------------------------------------------------------

Definition at line 61 of file Character.java.

```

61                                     {
62         if(distance != 0) {
63             this.posY = this.posY + distance;
64             this.draw();
65         }
66     }

```

7.29.3.8 void world.character.Character.move (int *distanceX*, int *distanceY*) [inherited]

Slowly move the character to a certain point regarding his current position.

Parameters

<i>distanceX</i>	Horizontal distance toward the character will be moved
<i>distanceY</i>	Vertical distance toward the character will be moved

Definition at line 118 of file Character.java.

References world.character.Character.moveX(), and world.character.Character.moveY().

```

118                                     {
119
120         if(distanceX != 0 || distanceY != 0) {

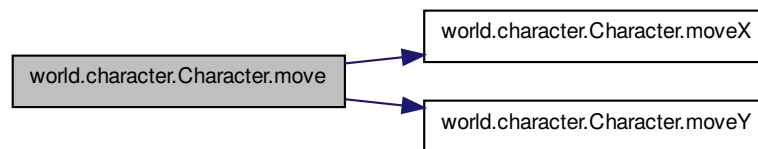
```

```

121         int stepX = (distanceX < 0)? -1 : 1;
122         int stepY = (distanceY < 0)? -1 : 1;
123
124         this.jump(stepX, stepY);
125
126         if(distanceX - stepX == 0) {
127             moveY(distanceY - stepY);
128             return;
129         }
130
131         if(distanceY - stepY == 0) {
132             moveX(distanceX - stepX);
133             return;
134         }
135
136         move(distanceX - stepX, distanceY - stepY);
137     }
138
139 }

```

Here is the call graph for this function:



7.29.3.9 void world.character.Character.moveX (int *distance*) [inherited]

Slowly horizontally move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Horizontal distance toward the character will be moved
-----------------	--------------------------------------------------------

Definition at line 85 of file Character.java.

Referenced by `world.character.Character.move()`.

```

85         {
86
87         if(distance != 0) {
88             int step = (distance < 0)? -1 : 1;
89
90             this.jumpX(step);
91
92             this.moveX(distance - step);
93         }
94     }
95 }

```

Here is the caller graph for this function:



7.29.3.10 void world.character.Character.moveY (int *distance*) [inherited]

Slowly vertically move the character to a certain point regarding his current position.

Parameters

<i>distance</i>	Vertical distance toward the character will be moved
-----------------	------------------------------------------------------

Definition at line 101 of file Character.java.

Referenced by world.character.Character.move().

```

101                                     {
102
103     if(distance != 0) {
104         int step = (distance < 0)? -1 : 1;
105
106         this.jumpY(step);
107
108         this.moveY(distance - step);
109     }
110
111 }
```

Here is the caller graph for this function:



7.29.3.11 void world.character.Character.playerEnter () [inherited]

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 153 of file Character.java.

```

153                                     {
154
155 }
```

7.29.3.12 void world.character.Character.playerLeave () [inherited]

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 157 of file Character.java.

```

157                                     {
158
159 }
```

7.29.3.13 void world.character.Character.playerNear () [inherited]

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 149 of file Character.java.

```
149             {  
150  
151     }
```

7.29.3.14 void world.character.TalkingCharacter.talk ()

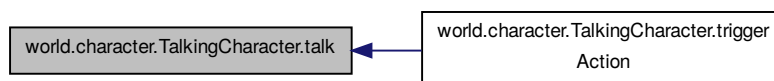
Method making the character talk.

Definition at line 22 of file TalkingCharacter.java.

Referenced by world.character.TalkingCharacter.triggerAction().

```
22             {  
23  
24     }
```

Here is the caller graph for this function:



7.29.3.15 void world.character.TalkingCharacter.triggerAction ()

Method to be called when the player is near the object and presses the action key.

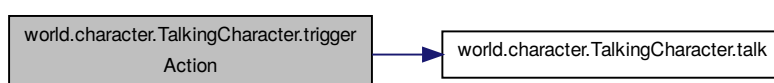
Implements [world.InteractiveObject](#).

Definition at line 30 of file TalkingCharacter.java.

References world.character.TalkingCharacter.talk().

```
30             {  
31     talk();  
32     }
```

Here is the call graph for this function:



7.29.4 Member Data Documentation

7.29.4.1 `final int world.character.Character.CHARACTER_HEIGHT = 300` `[static]`, `[protected]`, `[inherited]`

Height of the character's image (fixed)

Definition at line 18 of file Character.java.

Referenced by `world.character.Character.Character()`.

7.29.4.2 `final int world.character.Character.CHARACTER_WIDTH = 170` `[static]`, `[protected]`, `[inherited]`

Width of the character's image (fixed)

Definition at line 23 of file Character.java.

Referenced by `world.character.Character.Character()`.

7.29.4.3 `int world.scenery.Scenery.height` `[protected]`, `[inherited]`

Image height.

Definition at line 39 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.29.4.4 `int world.character.Character.posX` `[protected]`, `[inherited]`

Horizontal position of the character.

Definition at line 28 of file Character.java.

7.29.4.5 `int world.character.Character.posY` `[protected]`, `[inherited]`

Vertical position of the character.

Definition at line 33 of file Character.java.

7.29.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48` `[static]`, `[protected]`, `[inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.29.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48` `[static]`, `[protected]`, `[inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.29.4.8 `Image world.scenery.Scenery.spriteImage` `[protected]`, `[inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.29.4.9 boolean world.scenery.Scenery.visible [protected],[inherited]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery().

7.29.4.10 int world.scenery.Scenery.width [protected],[inherited]

Image width.

Definition at line 44 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.29.4.11 int world.scenery.Scenery.xShift [protected],[inherited]

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.29.4.12 int world.scenery.Scenery.yShift [protected],[inherited]

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

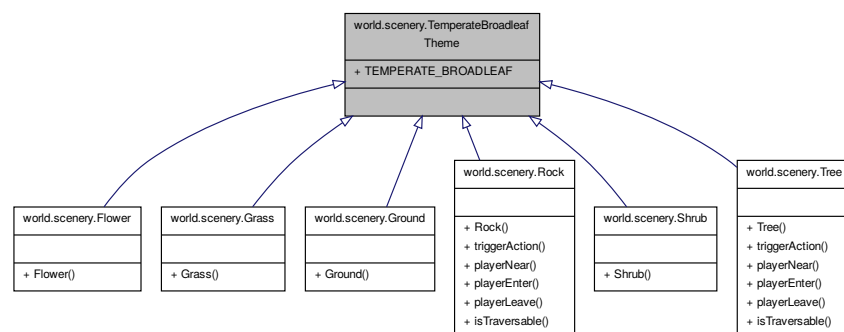
The documentation for this class was generated from the following file:

- world/character/[TalkingCharacter.java](#)

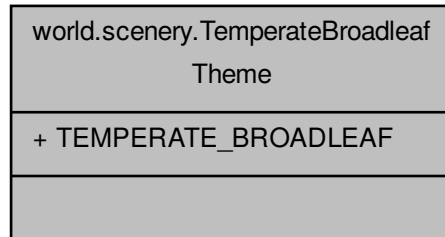
7.30 world.scenery.TemperateBroadleafTheme Interface Reference

Interface used to define that a [Scenery](#) object have a Temperate Broadleaf style.

Inheritance diagram for world.scenery.TemperateBroadleafTheme:



Collaboration diagram for world.scenery.TemperateBroadleafTheme:



Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

7.30.1 Detailed Description

Interface used to define that a [Scenery](#) object have a Temperate Broadleaf style.

Definition at line 7 of file TemperateBroadleafTheme.java.

7.30.2 Member Data Documentation

7.30.2.1 String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf" [static]

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

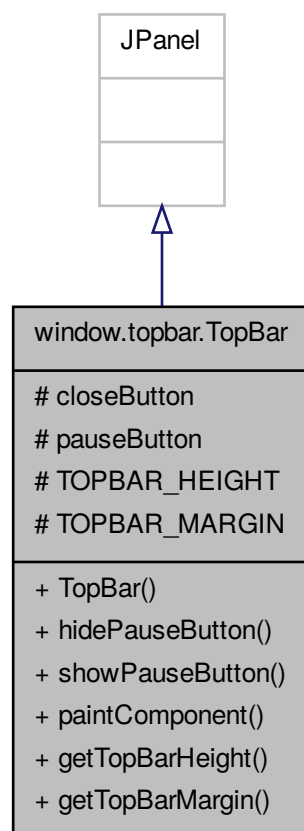
The documentation for this interface was generated from the following file:

- world/scenery/[TemperateBroadleafTheme.java](#)

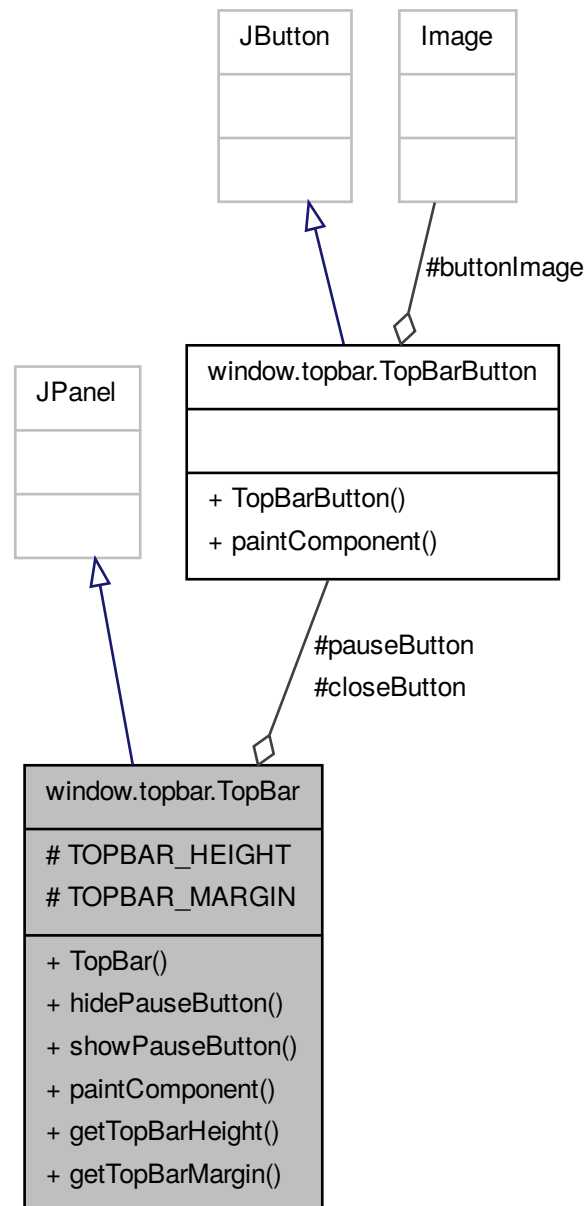
7.31 window.topbar.TopBar Class Reference

Class handling the top bar of the program's main window.

Inheritance diagram for window.topbar.TopBar:



Collaboration diagram for window.topbar.TopBar:



Public Member Functions

- [TopBar](#) ()
Constructor of the [TopBar](#) class.
- void [hidePauseButton](#) ()
- void [showPauseButton](#) ()
- void [paintComponent](#) (Graphics g)

Static Public Member Functions

- static int [getTopBarHeight](#) ()
- static int [getTopBarMargin](#) ()

Protected Attributes

- [TopBarButton closeButton](#)
Button closing the program when activated.
- [TopBarButton pauseButton](#)
Button pausing the program when activated.

Static Protected Attributes

- static int [TOPBAR_HEIGHT](#) = 42
Height of the UI's top bar (default: 42px)
- static int [TOPBAR_MARGIN](#) = 10
Margin of the top bar (default: 10px)

7.31.1 Detailed Description

Class handling the top bar of the program's main window.

Definition at line 16 of file TopBar.java.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 window.topbar.TopBar.TopBar ()

Constructor of the [TopBar](#) class.

Definition at line 41 of file TopBar.java.

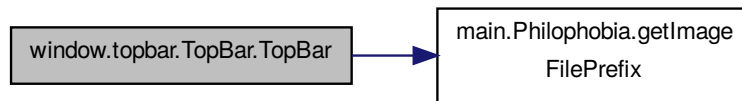
References [window.topbar.TopBar.closeButton](#), [main.Philophobia.getImageFilePrefix\(\)](#), [window.topbar.TopBar.pauseButton](#), [window.topbar.TopBar.TOPBAR_HEIGHT](#), and [window.topbar.TopBar.TOPBAR_MARGIN](#).

```

41         {
42
43             super();
44
45             Philophobia.getVerbose().calls("Creating TopBar class", "window/topbar/TopBar.java", "
TopBar.TopBar()");
46
47             this.setLayout(new BoxLayout(this, BoxLayout.LINE_AXIS));
48             this.add(Box.createHorizontalGlue());
49
50             closeButton = new TopBarButton(Philophobia.getImageFilePrefix() + "closebutton.png");
51             closeButton.setPreferredSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
52             closeButton.setMinimumSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
53             closeButton.setMaximumSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
54
55             pauseButton = new TopBarButton(Philophobia.getImageFilePrefix() + "pausebutton.png");
56             pauseButton.setPreferredSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
57             pauseButton.setMinimumSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
58             pauseButton.setMaximumSize(new Dimension(TOPBAR_HEIGHT -
TOPBAR_MARGIN, TOPBAR_HEIGHT - TOPBAR_MARGIN));
59
60             this.add(closeButton);
61         }

```

Here is the call graph for this function:



7.31.3 Member Function Documentation

7.31.3.1 static int window.topbar.TopBar.getTopBarHeight () [static]

Definition at line 63 of file TopBar.java.

References window.topbar.TopBar.TOPBAR_HEIGHT.

```

63         {
64             return TOPBAR_HEIGHT;
65         }
  
```

7.31.3.2 static int window.topbar.TopBar.getTopBarMargin () [static]

Definition at line 67 of file TopBar.java.

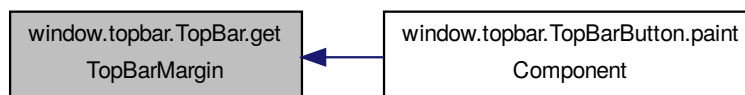
References window.topbar.TopBar.TOPBAR_MARGIN.

Referenced by window.topbar.TopBarButton.paintComponent().

```

67         {
68             return TOPBAR_MARGIN;
69         }
  
```

Here is the caller graph for this function:



7.31.3.3 void window.topbar.TopBar.hidePauseButton ()

Definition at line 71 of file TopBar.java.

References window.topbar.TopBar.closeButton.

```

71         {
72             Philophobia.getVerbose().calls("Hidding pause button", "window/topbar/TopBar.java", "
TopBar.hidePauseButton()");
  
```

```

73         this.removeAll();
74         this.add(Box.createHorizontalGlue());
75         this.add(closeButton);
76         this.repaint();
77     }

```

7.31.3.4 void window.topbar.TopBar.paintComponent (Graphics g)

Definition at line 89 of file TopBar.java.

```

89         {
90         Philophobia.getVerbose().calls("Paintint TopBar component", "window/topbar/TopBar.java", "
TopBar.painComponent(Graphics)");
91         // Top bar background
92         g.fillRect(0, 0, this.getWidth(), this.getHeight());
93
94         super.paintComponents(g);
95     }

```

7.31.3.5 void window.topbar.TopBar.showPauseButton ()

Definition at line 79 of file TopBar.java.

References window.topbar.TopBar.closeButton, window.topbar.TopBar.pauseButton, and window.topbar.TopBar.TOPBAR_MARGIN.

```

79         {
80         Philophobia.getVerbose().calls("Showing pause button", "window/topbar/TopBar.java", "
TopBar.showPauseButton()");
81         this.removeAll();
82         this.add(Box.createHorizontalGlue());
83         this.add(pauseButton);
84         this.add(Box.createRigidArea(new Dimension(TOPBAR_MARGIN, 0)));
85         this.add(closeButton);
86         this.repaint();
87     }

```

7.31.4 Member Data Documentation

7.31.4.1 TopBarButton window.topbar.TopBar.closeButton [protected]

Button closing the program when activated.

Definition at line 31 of file TopBar.java.

Referenced by window.topbar.TopBar.hidePauseButton(), window.topbar.TopBar.showPauseButton(), and window.topbar.TopBar.TopBar().

7.31.4.2 TopBarButton window.topbar.TopBar.pauseButton [protected]

Button pausing the program when activated.

Definition at line 36 of file TopBar.java.

Referenced by window.topbar.TopBar.showPauseButton(), and window.topbar.TopBar.TopBar().

7.31.4.3 int window.topbar.TopBar.TOPBAR_HEIGHT = 42 [static], [protected]

Height of the UI's top bar (default: 42px)

Definition at line 21 of file TopBar.java.

Referenced by window.topbar.TopBar.getTopBarHeight(), and window.topbar.TopBar.TopBar().

7.31.4.4 `int window.topbar.TopBar.TOPBAR_MARGIN = 10` `[static]`, `[protected]`

Margin of the top bar (default: 10px)

Definition at line 26 of file `TopBar.java`.

Referenced by `window.topbar.TopBar.getTopBarMargin()`, `window.topbar.TopBar.showPauseButton()`, and `window.topbar.TopBar.TopBar()`.

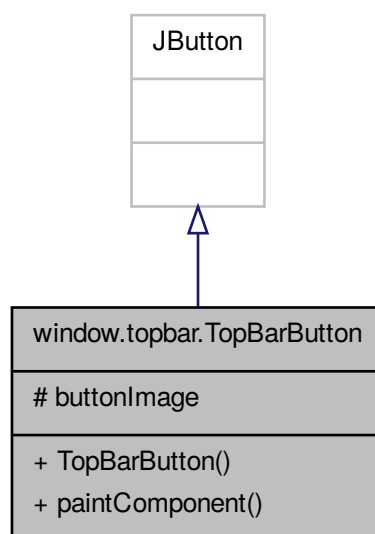
The documentation for this class was generated from the following file:

- `window/topbar/TopBar.java`

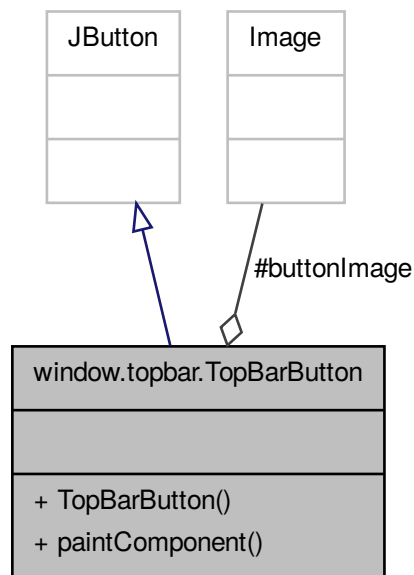
7.32 `window.topbar.TopBarButton` Class Reference

Class handling the UI buttons.

Inheritance diagram for `window.topbar.TopBarButton`:



Collaboration diagram for window.topbar.TopBarButton:



Public Member Functions

- [TopBarButton](#) (String imagePath)
UIButton constructor.
- void [paintComponent](#) (Graphics g)

Protected Attributes

- Image [buttonImage](#)
Image of the graphical representation of the button.

7.32.1 Detailed Description

Class handling the UI buttons.

Definition at line 19 of file TopBarButton.java.

7.32.2 Constructor & Destructor Documentation

7.32.2.1 window.topbar.TopBarButton.TopBarButton (String imagePath)

UIButton constructor.

Parameters

<i>imagePath</i>	path of the button's image
------------------	----------------------------

See Also

[buttonImage](#)

Definition at line 31 of file TopBarButton.java.

References window.topbar.TopBarButton.buttonImage.

```

31         {
32
33         super();
34
35         Philophobia.getVerbose().calls("Creating TopBarButton class", "window/ui/TopBarButton.java", "
TopBarButton.TopBarButton(String)");
36
37         try {
38             buttonImage = ImageIO.read(new File(imagePath));
39             this.setIcon(new ImageIcon(buttonImage));
40             this.setBorder(null);
41         } catch (IOException e) {
42             Philophobia.getVerbose().warning("Button image load failed: " + e.getMessage(), "
window/ui/TopBarButton.java", "TopBarButton.TopBarButton(String)");
43             buttonImage = null;
44         }
45     }
46 }
```

7.32.3 Member Function Documentation

7.32.3.1 void window.topbar.TopBarButton.paintComponent (Graphics g)

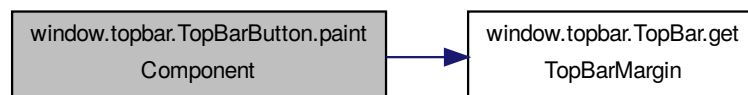
Definition at line 48 of file TopBarButton.java.

References window.topbar.TopBarButton.buttonImage, and window.topbar.TopBar.getTopBarMargin().

```

48         {
49         Philophobia.getVerbose().calls("painting TopBarButton component", "window/ui/TopBarButton.java", "
TopBarButton.paintComponent(Graphics)");
50
51         int buttonSize = TopBar.getTopBarHeight() - TopBar.getTopBarMargin();
52
53         g.drawImage(buttonImage, 0, 0, this.getWidth(), this.getHeight(), this);
54     }
```

Here is the call graph for this function:



7.32.4 Member Data Documentation

7.32.4.1 Image window.topbar.TopBarButton.buttonImage [protected]

Image of the graphical representation of the button.

Definition at line 24 of file TopBarButton.java.

Referenced by window.topbar.TopBarButton.paintComponent(), and window.topbar.TopBarButton.TopBarButton().

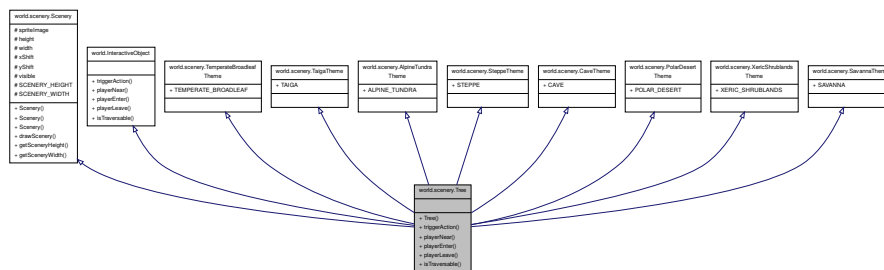
The documentation for this class was generated from the following file:

- window/topbar/TopBarButton.java

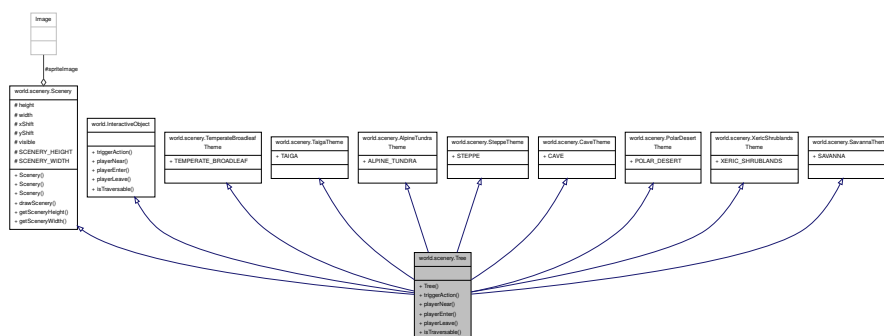
7.33 world.scenery.Tree Class Reference

Class used to handle a tree object.

Inheritance diagram for world.scenery.Tree:



Collaboration diagram for world.scenery.Tree:



Public Member Functions

- **Tree** (final String type)
Tree class constructor.
- void **triggerAction** ()
Method to be called when the player is near the object and presses the action key.
- void **playerNear** ()
Method to be called when the player is near the object.
- void **playerEnter** ()
Method to be called when the player was near and is now over the object.
- void **playerLeave** ()
Method to be called when the player was over and is now near the object.
- boolean **isTraversable** ()
Returns true if the player is able to pass through the object and false if the player is not able to cross the object.
- void **drawScenery** (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.
- static int [getSceneryWidth](#) ()
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TEMPERATE_BROADLEAF](#) = "temperatebroadleaf"
String used to tell in which file is the sprite matching the Temperate Broadleaf style.
- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [ALPINE_TUNDRA](#) = "alpinetundra"
String used to tell in which file is the sprite matching the Alpine Tundra style.
- static String [STEPPE](#) = "steppe"
String used to tell in which file is the sprite matching the Steppe style.
- static String [CAVE](#) = "cave"
String used to tell in which file is the sprite matching the Cave style.
- static String [POLAR_DESERT](#) = "polardesert"
String used to tell in which file is the sprite matching the Polar Desert style.
- static String [XERIC_SHRUBLANDS](#) = "xericshrublands"
String used to tell in which file is the sprite matching the Xeric Shrublands style.
- static String [SAVANNA](#) = "savanna"
String used to tell in which file is the sprite matching the Savanna style.

Protected Attributes

- Image [spritImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.33.1 Detailed Description

Class used to handle a tree object.

Definition at line 19 of file Tree.java.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 world.scenery.Tree.Tree (final String *type*)

[Tree](#) class constructor.

Parameters

<i>type</i>	Style of the tree
-------------	-------------------

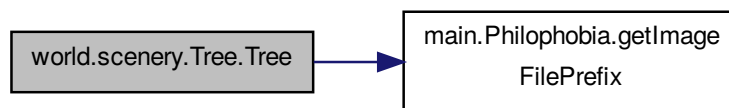
Definition at line 25 of file Tree.java.

References `main.Philophobia.getImageFilePrefix()`.

```

25         {
26
27         super(Philophobia.getImageFilePrefix() + type + "tree.png", 96, Scenery.getSceneryWidth(), 0, -48);
28
29     }
```

Here is the call graph for this function:



7.33.3 Member Function Documentation

7.33.3.1 void world.scenery.Scenery.drawScenery (Graphics *g*, final int *xLocation*, final int *yLocation*, ImageObserver *obs*) [inherited]

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }
```

7.33.3.2 static int world.scenery.Scenery.getSceneryHeight () [static], [inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

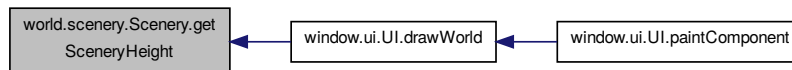
References `world.scenery.Scenery.SCENERY_HEIGHT`.

Referenced by `window.ui.UI.drawWorld()`.

```

144                                     {
145         return SCENERY_HEIGHT;
146     }
```

Here is the caller graph for this function:



7.33.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the `SCENERY_WIDTH` static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_WIDTH`.

```

152                                     {
153         return SCENERY_WIDTH;
154     }
```

7.33.3.4 boolean world.scenery.Tree.isTraversable ()

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 51 of file Tree.java.

```

51                                     {
52         return false;
53     }
```

7.33.3.5 void world.scenery.Tree.playerEnter ()

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 43 of file Tree.java.

```

43                                     {
44
45     }
```

7.33.3.6 void world.scenery.Tree.playerLeave ()

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 47 of file Tree.java.

```
47         {  
48  
49     }
```

7.33.3.7 void world.scenery.Tree.playerNear ()

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 39 of file Tree.java.

```
39         {  
40  
41     }
```

7.33.3.8 void world.scenery.Tree.triggerAction ()

Method to be called when the player is near the object and presses the action key.

Implements [world.InteractiveObject](#).

Definition at line 35 of file Tree.java.

```
35         {  
36  
37     }
```

7.33.4 Member Data Documentation

7.33.4.1 String world.scenery.AlpineTundraTheme.ALPINE_TUNDRA = "alpinetundra" [static],[inherited]

String used to tell in which file is the sprite matching the Alpine Tundra style.

Definition at line 13 of file AlpineTundraTheme.java.

7.33.4.2 String world.scenery.CaveTheme.CAVE = "cave" [static],[inherited]

String used to tell in which file is the sprite matching the Cave style.

Definition at line 13 of file CaveTheme.java.

7.33.4.3 int world.scenery.Scenery.height [protected],[inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.33.4.4 `String world.scenery.PolarDesertTheme.POLAR_DESERT = "polardesert" [static], [inherited]`

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.33.4.5 `String world.scenery.SavannaTheme.SAVANNA = "savanna" [static], [inherited]`

String used to tell in which file is the sprite matching the Savanna style.

Definition at line 13 of file SavannaTheme.java.

7.33.4.6 `int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static], [protected], [inherited]`

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryHeight()`.

7.33.4.7 `int world.scenery.Scenery.SCENERY_WIDTH = 48 [static], [protected], [inherited]`

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by `world.scenery.Scenery.getSceneryWidth()`.

7.33.4.8 `Image world.scenery.Scenery.spriteImage [protected], [inherited]`

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.33.4.9 `String world.scenery.SteppeTheme.STEPPE = "steppe" [static], [inherited]`

String used to tell in which file is the sprite matching the Steppe style.

Definition at line 13 of file SteppeTheme.java.

7.33.4.10 `String world.scenery.TaigaTheme.TAIGA = "taiga" [static], [inherited]`

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.33.4.11 `String world.scenery.TemperateBroadleafTheme.TEMPERATE_BROADLEAF = "temperatebroadleaf" [static], [inherited]`

String used to tell in which file is the sprite matching the Temperate Broadleaf style.

Definition at line 13 of file TemperateBroadleafTheme.java.

7.33.4.12 `boolean world.scenery.Scenery.visible` `[protected], [inherited]`

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`.

7.33.4.13 `int world.scenery.Scenery.width` `[protected], [inherited]`

Image width.

Definition at line 44 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.33.4.14 `String world.scenery.XericShrublandsTheme.XERIC_SHRUBLANDS = "xericshrublands"` `[static], [inherited]`

String used to tell in which file is the sprite matching the Xeric Shrublands style.

Definition at line 13 of file XericShrublandsTheme.java.

7.33.4.15 `int world.scenery.Scenery.xShift` `[protected], [inherited]`

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

7.33.4.16 `int world.scenery.Scenery.yShift` `[protected], [inherited]`

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

Referenced by `world.scenery.Scenery.drawScenery()`, and `world.scenery.Scenery.Scenery()`.

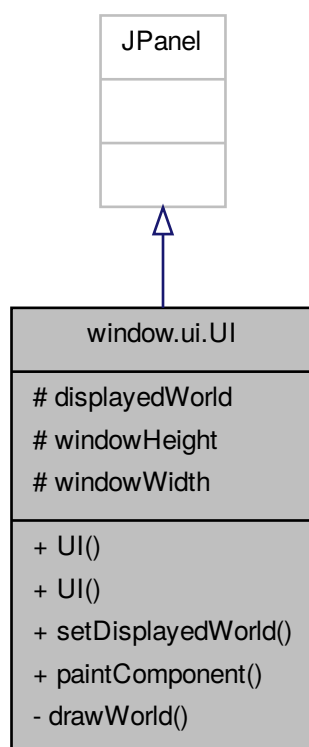
The documentation for this class was generated from the following file:

- `world/scenery/Tree.java`

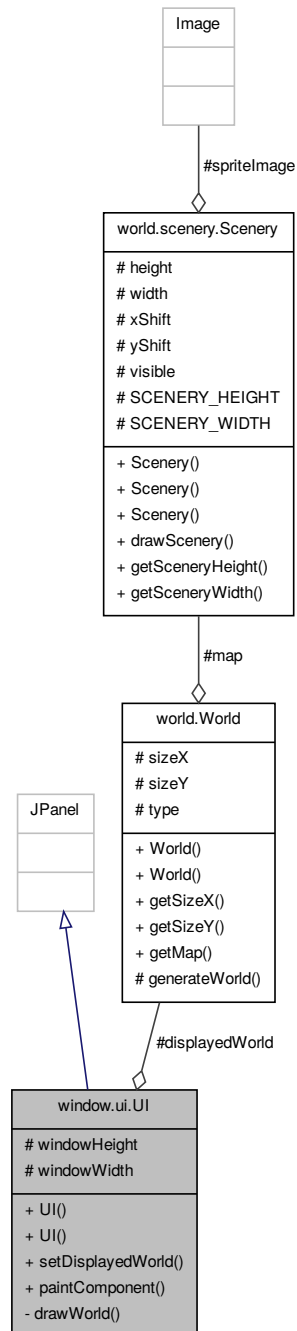
7.34 window.ui.UI Class Reference

Class used to handle the window's user interface.

Inheritance diagram for window.ui.UI:



Collaboration diagram for window.ui.UI:



Public Member Functions

- `UI` (int `windowHeight`, int `windowWidth`)
UI class constructor.
- `UI` (int `windowHeight`, int `windowWidth`, `World` world)
UI class constructor.
- void `setDisplayedWorld` (`World` world)

Setter for the displayedWorld field.

- void [paintComponent](#) (Graphics g)

Function called when the program ask to paint the graphics.

Protected Attributes

- [World](#) [displayedWorld](#)

World displayed inside the window.

- int [windowHeight](#)

Window height.

- int [windowWidth](#)

Window width.

Private Member Functions

- void [drawWorld](#) (Graphics g)

Function called to draw the world.

7.34.1 Detailed Description

Class used to handle the window's user interface.

This class can be in different states

Definition at line 16 of file UI.java.

7.34.2 Constructor & Destructor Documentation

7.34.2.1 [window.ui.UI](#) (int *windowHeight*, int *windowWidth*)

[UI](#) class constructor.

Parameters

<i>windowHeight</i>	Height of the window
<i>windowWidth</i>	Width of the window

Definition at line 38 of file UI.java.

References [window.ui.UI.windowHeight](#), and [window.ui.UI.windowWidth](#).

```

38                                     {
39         // We call the parent's constructor
40         super();
41
42         Philophobia.getVerbose().calls("Creating UI class", "window/ui/UI.java", "UI.UI(int, int)");
43
44         this.windowHeight = windowHeight;
45         this.windowWidth = windowWidth;
46
47     }
```

7.34.2.2 [window.ui.UI](#) (int *windowHeight*, int *windowWidth*, [World](#) *world*)

[UI](#) class constructor.

Parameters

<i>windowHeight</i>	Height of the window
<i>windowWidth</i>	Width of the window
<i>world</i>	World to display

Definition at line 55 of file UI.java.

References `window.ui.UI.windowHeight`, and `window.ui.UI.windowWidth`.

```

55                                     {
56         super();
57
58         Philophobia.getVerbose().calls("Creating UI class", "window/ui/UI.java", "UI.UI(int, int,World)");
59
60         this.windowHeight = windowHeight;
61         this.windowWidth = windowWidth;
62         this.displayedWorld = world;
63     }

```

7.34.3 Member Function Documentation

7.34.3.1 void window.ui.UI.drawWorld (Graphics g) [private]

Function called to draw the world.

This function just display each Scenery one by one but trees are displayed after everything else as they must be on top of everything because of their size

See Also

[displayedWorld](#)

Parameters

<i>g</i>	Graphics to use
----------	-----------------

Definition at line 94 of file UI.java.

References `world.scenery.Scenery.getSceneryHeight()`.

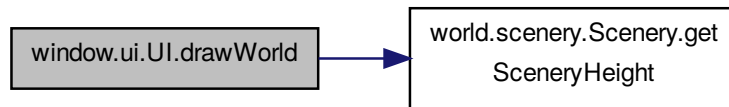
Referenced by `window.ui.UI.paintComponent()`.

```

94                                     {
95
96         Philophobia.getVerbose().calls("Painting the world", "window/ui/UI.java", "UI.drawWorld(Graphics)")
97     ;
98
99         Scenery[][] worldMap = displayedWorld.getMap();
100         int worldMapXSize = displayedWorld.getSizeX();
101         int worldMapYSize = displayedWorld.getSizeY();
102
103         for(int i = 0 ; i < worldMapXSize ; ++i) {
104             for(int j = 0 ; j < worldMapYSize ; ++j) {
105                 if(!worldMap[i][j].getClass().getName().equals("Tree"))
106                     worldMap[i][j].drawScenery(g, i*Scenery.getSceneryHeight(), j*Scenery.getSceneryWidth()
107                 , this);
108             }
109         }
110
111         for(int i = 0 ; i < worldMapXSize ; ++i) {
112             for(int j = 0 ; j < worldMapYSize ; ++j) {
113                 if(worldMap[i][j].getClass().getName().equals("Tree"))
114                     worldMap[i][j].drawScenery(g, i*Scenery.getSceneryHeight(), j*Scenery.getSceneryWidth()
115                 , this);
116             }
117         }
118     }

```

Here is the call graph for this function:



Here is the caller graph for this function:



7.34.3.2 void window.ui.UI.paintComponent (Graphics *g*)

Function called when the program ask to paint the graphics.

Parameters

<i>g</i>	Graphics to use
----------	-----------------

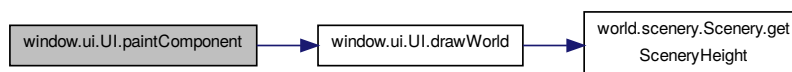
Definition at line 78 of file UI.java.

References `window.ui.UI.drawWorld()`.

```

78         {
79             Philophobia.getVerbose().calls("Painting UI components", "window/ui/UI.java", "
UI.paintComponent(Graphics)");
80             super.paintComponent(g);
81             drawWorld(g);
82         }
  
```

Here is the call graph for this function:



7.34.3.3 void window.ui.UI.setDisplayedWorld (World *world*)

Setter for the `displayedWorld` field.

See Also

[displayedWorld](#)

Parameters

<i>world</i>	World to display
--------------	------------------

Definition at line 70 of file UI.java.

```
70                                     {
71         this.displayedWorld = world;
72     }
```

7.34.4 Member Data Documentation

7.34.4.1 World `window.ui.UI.displayedWorld` [protected]

World displayed inside the window.

Definition at line 21 of file UI.java.

7.34.4.2 `int window.ui.UI.windowHeight` [protected]

[Window](#) height.

Definition at line 26 of file UI.java.

Referenced by `window.ui.UI.UI()`.

7.34.4.3 `int window.ui.UI.windowWidth` [protected]

[Window](#) width.

Definition at line 31 of file UI.java.

Referenced by `window.ui.UI.UI()`.

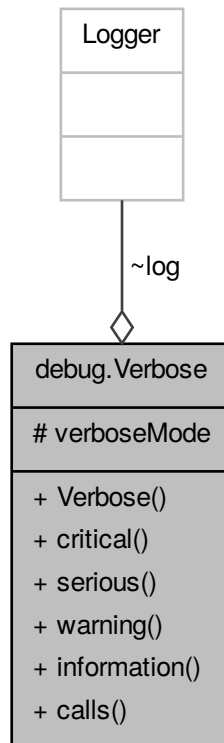
The documentation for this class was generated from the following file:

- [window/ui/UI.java](#)

7.35 debug.Verbose Class Reference

Class used to display and log messages all over this program.

Collaboration diagram for debug.Verbose:



Public Member Functions

- **Verbose** (int level)
Class constructor.
- void **critical** (String message, String file, String location)
Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 1.
- void **serious** (String message, String file, String location)
Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 2.
- void **warning** (String message, String file, String location)
Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 3.
- void **information** (String message, String file, String location)
Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 4.
- void **calls** (String message, String file, String location)
Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is equal to 5.

Protected Attributes

- int [verboseMode](#) = 0
Level of verbosing, from 0 to 5.

Package Attributes

- Logger [log](#)
Logger used to log messages into a file.

7.35.1 Detailed Description

Class used to display and log messages all over this program.

Definition at line 11 of file Verbose.java.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 debug.Verbose.Verbose (int *level*)

Class constructor.

Parameters

<i>level</i>	Level of "verbosing"
--------------	----------------------

See Also

[verboseMode](#)

Definition at line 36 of file Verbose.java.

References [debug.Verbose.log](#), and [debug.Verbose.warning\(\)](#).

```
36         {
37             this.verboseMode = level;
38
39             if(level > 0) {
40                 System.out.println("Verbose mode activated at level " + level);
41
42                 log = Logger.getLogger("Philophobia.log");
43
44                 try {
45                     log.addHandler(new FileHandler("Philophobia.log"));
46                 } catch (IOException e) {
47                     warning("Error initializing the log file", "org/debug/Verbose.java", "
Verbose.Verbose(int)");
48                 }
49
50                 log.setLevel(Level.parse("ALL"));
51
52                 log.info("Verbose at level " + level + ".");
53                 log.info("Displayed messages are :");
54                 log.info("- Criticals");
55
56                 if(level >= 2) {
57                     log.info("- Serious");
58                 }
59                 if(level >= 3) {
60                     log.info("- Warnings");
61                 }
62                 if(level >= 4) {
63                     log.info("- Informations");
64                 }
65                 if(level >= 5) {
66                     log.info("- Class instantiations and function calls");
67                 }
68             }
```

```

69
70     this.calls("Verbose class created", "org/debug/Verbose.java", "Verbose.Verbose(int)");
71 }

```

Here is the call graph for this function:



7.35.3 Member Function Documentation

7.35.3.1 void debug.Verbose.calls (String message, String file, String location)

Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is equal to 5.

Parameters

<i>message</i>	Message content
<i>file</i>	File in which this function is called
<i>location</i>	Class and function in which this function is called

Definition at line 145 of file Verbose.java.

References debug.Verbose.verboseMode.

```

145                                     {
146     if(verboseMode >= 5) {
147         System.out.println("Calls : " + message);
148
149         log.finer("Calls : " + message + " in file " + file + " in " + location);
150     }
151 }

```

7.35.3.2 void debug.Verbose.critical (String message, String file, String location)

Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 1.

Parameters

<i>message</i>	Message content
<i>file</i>	File in which this function is called
<i>location</i>	Class and function in which this function is called

Definition at line 81 of file Verbose.java.

References debug.Verbose.verboseMode.

```

81                                     {
82     if(verboseMode >= 1) {
83         System.out.println("Critical : " + message);
84
85         log.severe("Critical : " + message + " in file " + file + " in " + location);
86     }
87 }

```


7.35.3.3 void debug.Verbose.information (String *message*, String *file*, String *location*)

Display and log a message with additional informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 4.

Parameters

<i>message</i>	Message content
<i>file</i>	File in which this function is called
<i>location</i>	Class and function in which this function is called

Definition at line 129 of file Verbose.java.

References debug.Verbose.verboseMode.

```

129                                     {
130         if(verboseMode >= 4) {
131             System.out.println("Information : " + message);
132
133             log.info("Information : " + message + " in file " + file + " in " + location);
134         }
135     }

```

7.35.3.4 void debug.Verbose.serious (String message, String file, String location)

Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 2.

Parameters

<i>message</i>	Message content
<i>file</i>	File in which this function is called
<i>location</i>	Class and function in which this function is called

Definition at line 97 of file Verbose.java.

References debug.Verbose.verboseMode.

```

97                                     {
98         if(verboseMode >= 2) {
99             System.out.println("Serious : " + message);
100
101             log.severe("Serious : " + message + " in file " + file + " in " + location);
102         }
103     }

```

7.35.3.5 void debug.Verbose.warning (String message, String file, String location)

Display and log a message with additionnal informations in the log file The message is displayed and logged if and only if the verbose level is greater or equal to 3.

Parameters

<i>message</i>	Message content
<i>file</i>	File in which this function is called
<i>location</i>	Class and function in which this function is called

Definition at line 113 of file Verbose.java.

References debug.Verbose.verboseMode.

Referenced by debug.Verbose.Verbose().

```

113                                     {
114         if(verboseMode >= 3) {
115             System.out.println("Warning : " + message);
116
117             log.warning("Warning : " + message + " in file " + file + " in " + location);
118         }
119     }

```

Here is the caller graph for this function:



7.35.4 Member Data Documentation

7.35.4.1 Logger `debug.Verbose.log` [package]

Logger used to log messages into a file.

Definition at line 28 of file `Verbose.java`.

Referenced by `debug.Verbose.Verbose()`.

7.35.4.2 `int debug.Verbose.verboseMode = 0` [protected]

Level of verbosing, from 0 to 5.

The number of information displayed and logged goes increasingly as the number increase

- Level 1 : Only critical errors and exeptions are displayed on the console
- Level 2 : Serious but not critical errors and level 1 displayed
- Level 3 : Warning and level 2 displayed
- Level 4 : Informations and level 3 displayed
- Level 5 : Everything, included Class instanciations and function calls and level 4 displayed

Definition at line 23 of file `Verbose.java`.

Referenced by `debug.Verbose.calls()`, `debug.Verbose.critical()`, `debug.Verbose.information()`, `debug.Verbose.-serious()`, and `debug.Verbose.warning()`.

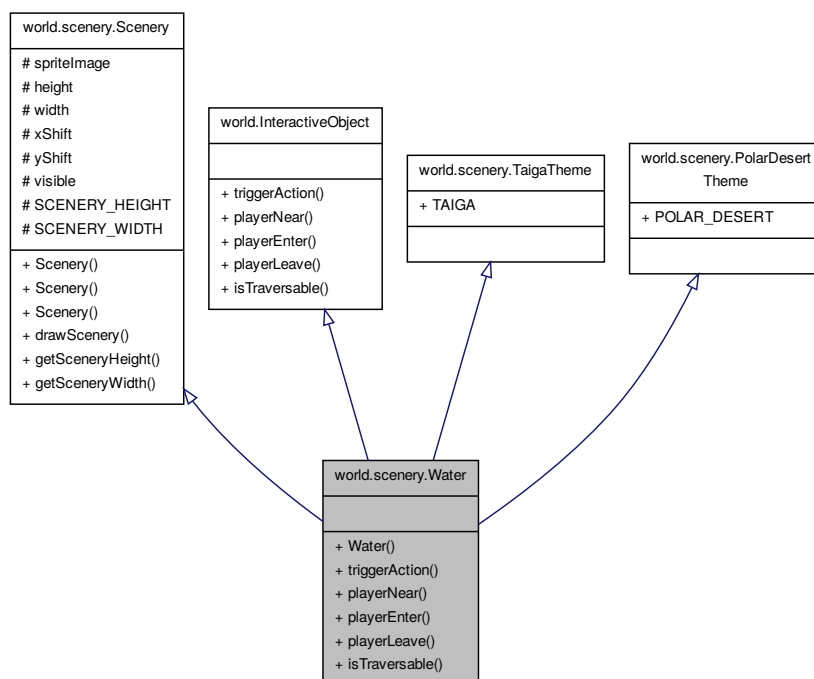
The documentation for this class was generated from the following file:

- [debug/Verbose.java](#)

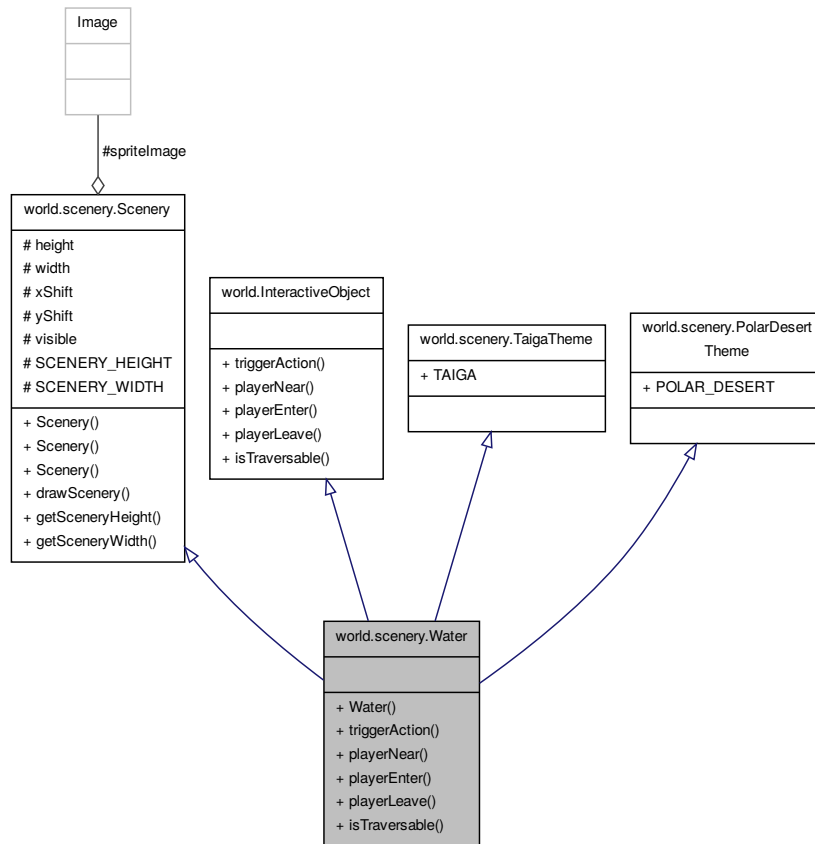
7.36 world.scenery.Water Class Reference

Class used to handle a water object.

Inheritance diagram for world.scenery.Water:



Collaboration diagram for world.scenery.Water:



Public Member Functions

- [Water](#) (final String type)
Water class constructor.
- void [triggerAction](#) ()
Method to be called when the player is near the object and presses the action key.
- void [playerNear](#) ()
Method to be called when the player is near the object.
- void [playerEnter](#) ()
Method to be called when the player was near and is now over the object.
- void [playerLeave](#) ()
Method to be called when the player was over and is now near the object.
- boolean [isTraversable](#) ()
Returns true if the player is able to pass through the object and false if the player is not able to cross the object.
- void [drawScenery](#) (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)

Static Public Member Functions

- static int [getSceneryHeight](#) ()
Getter for the SCENERY_HEIGHT static field.

- static int [getSceneryWidth \(\)](#)
Getter for the SCENERY_WIDTH static field.

Static Public Attributes

- static String [TAIGA](#) = "taiga"
String used to tell in which file is the sprite matching the Taiga style.
- static String [POLAR_DESERT](#) = "polardesert"
String used to tell in which file is the sprite matching the Polar Desert style.

Protected Attributes

- Image [spritelImage](#)
Image graphically representing the object.
- int [height](#)
Image height.
- int [width](#)
Image width.
- int [xShift](#)
Horizontal shifting for the image.
- int [yShift](#)
Vertical shifting for the image.
- boolean [visible](#)
Boolean equals to true if the scenery is visible, false if the scenery is not.

Static Protected Attributes

- static int [SCENERY_HEIGHT](#) = 48
Height of the graphical representation of a [Scenery](#) object.
- static int [SCENERY_WIDTH](#) = 48
Width of the graphical representation of a [Scenery](#) object.

7.36.1 Detailed Description

Class used to handle a water object.
Definition at line 13 of file Water.java.

7.36.2 Constructor & Destructor Documentation

7.36.2.1 world.scenery.Water.Water (final String type)

[Water](#) class constructor.

Parameters

<i>type</i>	Style of the water
-------------	--------------------

Definition at line 19 of file Water.java.

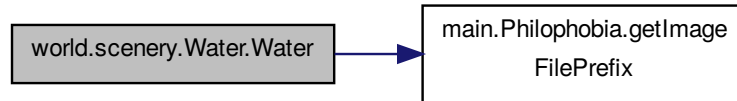
References `main.Philophobia.getImageFilePrefix()`.

```

19         {
20
21         super(Philophobia.getImageFilePrefix() + type + "water.png");
22
23     }

```

Here is the call graph for this function:



7.36.3 Member Function Documentation

7.36.3.1 `void world.scenery.Scenery.drawScenery (Graphics g, final int xLocation, final int yLocation, ImageObserver obs)` [inherited]

Definition at line 133 of file Scenery.java.

References `world.scenery.Scenery.height`, `world.scenery.Scenery.spriteImage`, `world.scenery.Scenery.visible`, `world.scenery.Scenery.width`, `world.scenery.Scenery.xShift`, and `world.scenery.Scenery.yShift`.

```

133
134         // drawImage(Image img, int x, int y, int width, int height, Observer obs);
135         g.drawImage(spriteImage, xLocation + xShift, yLocation +
136             yShift, width, height, obs);
137         visible = true;
138     }

```

7.36.3.2 `static int world.scenery.Scenery.getSceneryHeight ()` [static], [inherited]

Getter for the SCENERY_HEIGHT static field.

See Also

[SCENERY_HEIGHT](#)

Definition at line 144 of file Scenery.java.

References `world.scenery.Scenery.SCENERY_HEIGHT`.

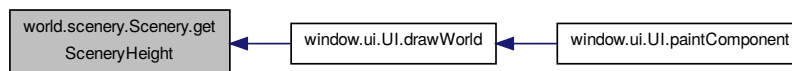
Referenced by `window.ui.UI.drawWorld()`.

```

144
145         return SCENERY_HEIGHT;
146     }

```

Here is the caller graph for this function:



7.36.3.3 static int world.scenery.Scenery.getSceneryWidth () [static],[inherited]

Getter for the SCENERY_WIDTH static field.

See Also

[SCENERY_WIDTH](#)

Definition at line 152 of file Scenery.java.

References world.scenery.Scenery.SCENERY_WIDTH.

```
152                                     {
153     return SCENERY_WIDTH;
154 }
```

7.36.3.4 boolean world.scenery.Water.isTraversable ()

Returns true if the player is able to pass through the object and false if the player is not able to cross the object.

Implements [world.InteractiveObject](#).

Definition at line 45 of file Water.java.

```
45                                     {
46     return false;
47 }
```

7.36.3.5 void world.scenery.Water.playerEnter ()

Method to be called when the player was near and is now over the object.

Implements [world.InteractiveObject](#).

Definition at line 37 of file Water.java.

```
37                                     {
38
39 }
```

7.36.3.6 void world.scenery.Water.playerLeave ()

Method to be called when the player was over and is now near the object.

Implements [world.InteractiveObject](#).

Definition at line 41 of file Water.java.

```
41                                     {
42
43 }
```

7.36.3.7 void world.scenery.Water.playerNear ()

Method to be called when the player is near the object.

Implements [world.InteractiveObject](#).

Definition at line 33 of file Water.java.

```
33                                     {
34     // AI talking
35 }
```


7.36.3.8 void world.scenery.Water.triggerAction ()

Method to be called when the player is near the object and presses the action key.

Implements [world.InteractiveObject](#).

Definition at line 29 of file Water.java.

```
29         {  
30  
31     }
```

7.36.4 Member Data Documentation

7.36.4.1 int world.scenery.Scenery.height [protected], [inherited]

Image height.

Definition at line 39 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.36.4.2 String world.scenery.PolarDesertTheme.POLAR_DESERT = "polardesert" [static], [inherited]

String used to tell in which file is the sprite matching the Polar Desert style.

Definition at line 13 of file PolarDesertTheme.java.

7.36.4.3 int world.scenery.Scenery.SCENERY_HEIGHT = 48 [static], [protected], [inherited]

Height of the graphical representation of a [Scenery](#) object.

Definition at line 24 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryHeight().

7.36.4.4 int world.scenery.Scenery.SCENERY_WIDTH = 48 [static], [protected], [inherited]

Width of the graphical representation of a [Scenery](#) object.

Definition at line 29 of file Scenery.java.

Referenced by world.scenery.Scenery.getSceneryWidth().

7.36.4.5 Image world.scenery.Scenery.spriteImage [protected], [inherited]

Image graphically representing the object.

Definition at line 34 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.36.4.6 String world.scenery.TaigaTheme.TAIGA = "taiga" [static], [inherited]

String used to tell in which file is the sprite matching the Taiga style.

Definition at line 13 of file TaigaTheme.java.

7.36.4.7 boolean world.scenery.Scenery.visible [protected],[inherited]

Boolean equals to true if the scenery is visible, false if the scenery is not.

Definition at line 59 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery().

7.36.4.8 int world.scenery.Scenery.width [protected],[inherited]

Image width.

Definition at line 44 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.36.4.9 int world.scenery.Scenery.xShift [protected],[inherited]

Horizontal shifting for the image.

Definition at line 49 of file Scenery.java.

Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

7.36.4.10 int world.scenery.Scenery.yShift [protected],[inherited]

Vertical shifting for the image.

Definition at line 54 of file Scenery.java.

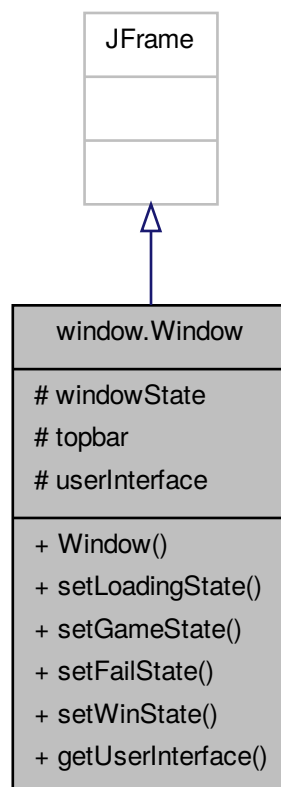
Referenced by world.scenery.Scenery.drawScenery(), and world.scenery.Scenery.Scenery().

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

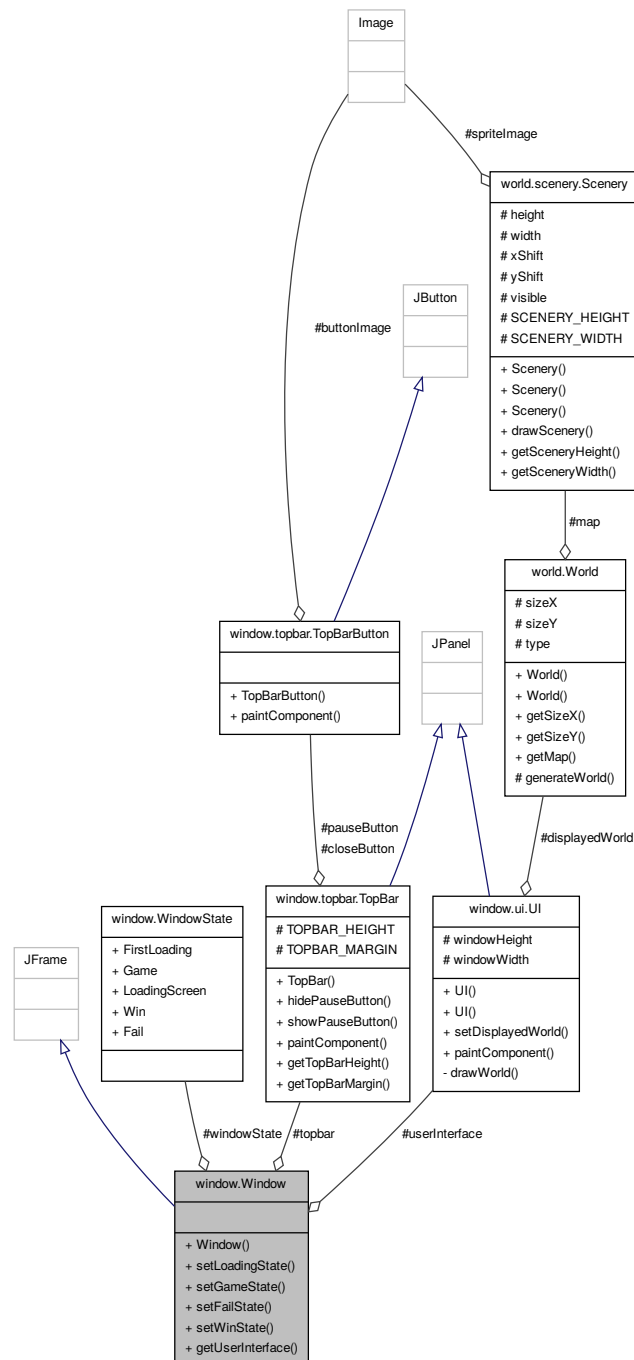
- world/scenery/[Water.java](#)

7.37 window.Window Class Reference

Inheritance diagram for window.Window:



Collaboration diagram for window.Window:



Public Member Functions

- [Window](#) ()
Constructor of the [Window](#) class.
- void [setLoadingState](#) ()
- void [setGameState](#) ()
- void [setFailState](#) ()

- void [setWinState\(\)](#)
- [UI getUserInterface\(\)](#)

Protected Attributes

- [WindowState windowState](#)
Window's current state.
- [TopBar topbar](#)
Window top bar.
- [UI userInterface](#)
User interface which handle the game graphics inside the window.

7.37.1 Detailed Description

Definition at line 12 of file Window.java.

7.37.2 Constructor & Destructor Documentation

7.37.2.1 window.Window.Window ()

Constructor of the [Window](#) class.

Definition at line 33 of file Window.java.

References [window.WindowState.FirstLoading](#), [window.Window.topbar](#), [window.Window.userInterface](#), and [window.Window.windowState](#).

```

33         {
34             Philophobia.getVerbose().calls("Creating Window class", "window/Window.java", "Window.Window()");
35
36             this.setTitle("Philophobia");
37             this.setSize(((int) Toolkit.getDefaultToolkit().getScreenSize().getWidth()), ((int) Toolkit.
getDefaultToolkit().getScreenSize().getHeight()));
38             this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
39
40             // Fullscreen mode
41             this.setExtendedState(this.getExtendedState() | JFrame.MAXIMIZED_BOTH);
42             this.setUndecorated(true);
43
44             this.setLayout(new BorderLayout());
45
46             windowState = WindowState.FirstLoading;
47
48             userInterface = new UI(this.getHeight(), this.getWidth());
49             // this.getContentPane().add(userInterface, BorderLayout.CENTER);
50
51             topbar = new TopBar();
52             this.getContentPane().add(topbar, BorderLayout.PAGE_START);
53             this.pack();
54             this.setVisible(true);
55
56     }
```

7.37.3 Member Function Documentation

7.37.3.1 UI window.Window.getUserInterface ()

Definition at line 83 of file Window.java.

References [window.Window.userInterface](#).

```

83         {
84             return userInterface;
85     }
```

7.37.3.2 void window.Window.setFailState ()

Definition at line 71 of file Window.java.

References window.WindowState.Fail, and window.Window.windowState.

```

71         {
72             Philophobia.getVerbose().information("Setting window's Fail mode", "window/Window.java", "
Window.setFailState()");
73             topbar.hidePauseButton();
74             windowState = WindowState.Fail;
75         }

```

7.37.3.3 void window.Window.setGameState ()

Definition at line 64 of file Window.java.

References window.WindowState.Game, window.Window.userInterface, and window.Window.windowState.

```

64         {
65             Philophobia.getVerbose().information("Setting window's Game mode", "window/Window.java", "
Window.setGameState()");
66             topbar.showPauseButton();
67             this.getContentPane().add(userInterface, BorderLayout.CENTER);
68             windowState = WindowState.Game;
69         }

```

7.37.3.4 void window.Window.setLoadingState ()

Definition at line 58 of file Window.java.

References window.WindowState.LoadingScreen, and window.Window.windowState.

```

58         {
59             Philophobia.getVerbose().information("Setting window's Loading mode", "window/Window.java", "
Window.setLoadingState()");
60             topbar.hidePauseButton();
61             windowState = WindowState.LoadingScreen;
62         }

```

7.37.3.5 void window.Window.setWinState ()

Definition at line 77 of file Window.java.

References window.WindowState.Win, and window.Window.windowState.

```

77         {
78             Philophobia.getVerbose().information("Setting window's Win mode", "window/Window.java", "
Window.setWinState()");
79             topbar.hidePauseButton();
80             windowState = WindowState.Win;
81         }

```

7.37.4 Member Data Documentation

7.37.4.1 TopBar window.Window.topbar [protected]

[Window](#) top bar.

Definition at line 23 of file Window.java.

Referenced by window.Window.Window().

7.37.4.2 UI window.Window.userInterface [protected]

User interface which handle the game graphics inside the window.

Definition at line 28 of file Window.java.

Referenced by `window.Window.getUserInterface()`, `window.Window.setGameState()`, and `window.Window.Window()`.

7.37.4.3 WindowState window.Window.windowState [protected]

[Window](#)'s current state.

See Also

[window.WindowState](#)

Definition at line 18 of file Window.java.

Referenced by `window.Window.setFailState()`, `window.Window.setGameState()`, `window.Window.setLoadingState()`, `window.Window.setWinState()`, and `window.Window.Window()`.

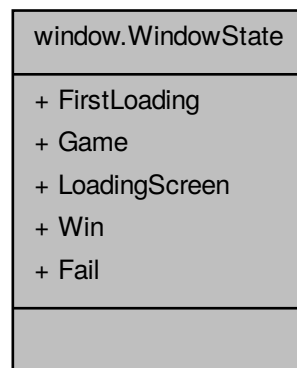
The documentation for this class was generated from the following file:

- [window/Window.java](#)

7.38 window.WindowState Enum Reference

Enumeration of the possible states of the displayed graphics.

Collaboration diagram for `window.WindowState`:



Public Attributes

- [FirstLoading](#)

The FirstLoading state correspond to the state where the window has just been called and a special animation is displayed.

- [Game](#)

The Game state correspond to the state in which the player is playing the game.

- [LoadingScreen](#)

The LoadingScreen state correspond to the state where a loading animation is displayed.

- [Win](#)

The Win state correspond to the state where the player has won the game and a congratulation screen is displayed.

- [Fail](#)

The Fail state correspond to the state where the player has lost the game and a condolence screen is displayed.

7.38.1 Detailed Description

Enumeration of the possible states of the displayed graphics.

The state can be the "in game" state, a loading screen, the "win" or "fail" screen

See Also

[Window](#)
[FirstLoading](#)
[Game](#)
[LoadingScreen](#)
[Win](#)
[Fail](#)

Definition at line 101 of file Window.java.

7.38.2 Member Data Documentation

7.38.2.1 `window.WindowState.Fail`

The Fail state correspond to the state where the player has lost the game and a condolence screen is displayed.

Definition at line 136 of file Window.java.

Referenced by `window.Window.setFailState()`.

7.38.2.2 `window.WindowState.FirstLoading`

The FirstLoading state correspond to the state where the window has just been called and a special animation is displayed.

Definition at line 109 of file Window.java.

Referenced by `window.Window.Window()`.

7.38.2.3 `window.WindowState.Game`

The Game state correspond to the state in which the player is playing the game.

Definition at line 115 of file Window.java.

Referenced by `window.Window.setGameState()`.

7.38.2.4 `window.WindowState.LoadingScreen`

The LoadingScreen state correspond to the state where a loading animation is displayed.

Definition at line 122 of file Window.java.

Referenced by window.Window.setLoadingState().

7.38.2.5 window.WindowState.Win

The Win state correspond to the state where the player has won the game and a congratulation screen is displayed.

Definition at line 129 of file Window.java.

Referenced by window.Window.setWinState().

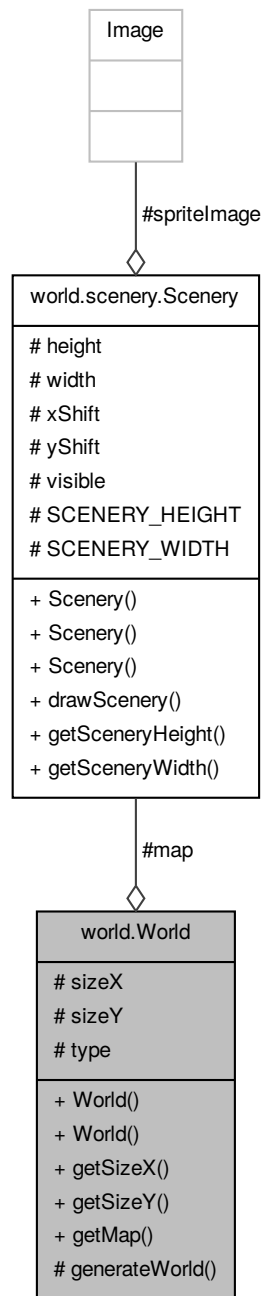
The documentation for this enum was generated from the following file:

- window/[Window.java](#)

7.39 world.World Class Reference

[World](#) is a class used to handle the game world graphics.

Collaboration diagram for world.World:



Public Member Functions

- [World](#) (final String [type](#))
[World](#) class constructor.
- [World](#) (final String [type](#), final int [sizeX](#), final int [sizeY](#))
[World](#) class constructor with size parameters.
- int [getSizeX](#) ()

Getter for the sizeX Field.

- int [getSizeY](#) ()

Getter for the sizeY Field.

- [Scenery](#)[][] [getMap](#) ()

Getter for the map Field.

Protected Member Functions

- void [generateWorld](#) ()

Function randomly generating the world.

Protected Attributes

- int [sizeX](#)

Horizontal size of the map.

- int [sizeY](#)

Vertical size of the map.

- [Scenery](#)[][] [map](#)

Two-dimensional array containing all objects within the world.

- String [type](#)

Style of the world.

7.39.1 Detailed Description

[World](#) is a class used to handle the game world graphics.

The [World](#) class use the Scenery objects (or the objects implementing the Scenery class) to display images of several objects corresponding to trees and traps and so on

Definition at line 18 of file World.java.

7.39.2 Constructor & Destructor Documentation

7.39.2.1 world.World.World (final String type)

[World](#) class constructor.

Parameters

<i>type</i>	Style of the world
-------------	--------------------

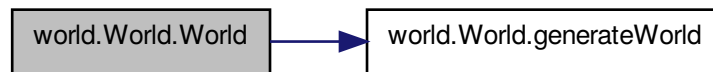
Definition at line 47 of file World.java.

References [world.World.generateWorld\(\)](#), and [world.World.type](#).

```

47         {
48             Philophobia.getVerbose().information("Creating World class", "world/World.java", "
World.World(String)");
49
50             this.type = type;
51             this.sizeX = 128;
52             this.sizeY = 128;
53
54             generateWorld();
55         }
```

Here is the call graph for this function:



7.39.2.2 world.World.World (final String type, final int sizeX, final int sizeY)

[World](#) class constructor with size parameters.

Parameters

<i>type</i>	Style of the world
<i>sizeX</i>	Horizontal size of the world
<i>sizeY</i>	Vertical size of the world

Definition at line 63 of file World.java.

References world.World.generateWorld(), world.World.sizeX, world.World.sizeY, and world.World.type.

```

63
64     Philophobia.getVerbose().information("Creating World class", "world/World.java", "
        World.World(String, int, int)");
65
66     this.type = type;
67     this.sizeX = sizeX;
68     this.sizeY = sizeY;
69
70     generateWorld();
71 }
  
```

Here is the call graph for this function:



7.39.3 Member Function Documentation

7.39.3.1 void world.World.generateWorld () [protected]

Function randomly generating the world.

Definition at line 76 of file World.java.

References world.World.map, world.World.sizeX, world.World.sizeY, and world.World.type.

Referenced by world.World.World().

```

76         {
77         map = new Scenery[sizeX][sizeY];
78
79         // Loop generating the world
80         for(int i = 0 ; i < sizeX ; ++i) {
81             for(int j = 0 ; j < sizeY ; ++j) {
82
83                 double random = Math.random();
84
85                 Scenery currentScenery;
86
87                 if(0 <= random && random < .3) {
88                     currentScenery = new Tree(type);
89                 } else if(.3 <= random && random < .6) {
90                     currentScenery = new Rock(type);
91                 } else {
92                     currentScenery = new Ground(type);
93                 }
94
95                 map[i][j] = currentScenery;
96             }
97         }
98     }

```

Here is the caller graph for this function:



7.39.3.2 Scenery [][] world.World.getMap ()

Getter for the map Field.

Returns

Scenery[][] The map of the world

See Also

[map](#)

Definition at line 123 of file World.java.

```

123         {
124             return this.map;
125         }

```

7.39.3.3 int world.World.getSizeX ()

Getter for the sizeX Field.

Returns

int Horizontal size of the map (in Scenery objects)

See Also[sizeX](#)

Definition at line 105 of file World.java.

```
105 {  
106     return this.sizeX;  
107 }
```

7.39.3.4 int world.World.getSizeY ()

Getter for the sizeY Field.

Returns

int Vertical size of the map (in Scenery objects)

See Also[sizeY](#)

Definition at line 114 of file World.java.

```
114 {  
115     return this.sizeY;  
116 }
```

7.39.4 Member Data Documentation**7.39.4.1 Scenery [][] world.World.map [protected]**

Two-dimensional array containing all objects within the world.

This object is filled using a file in the root directory and in the form of map[x][y] from top-left to right-bottom

Definition at line 36 of file World.java.

Referenced by world.World.generateWorld().

7.39.4.2 int world.World.sizeX [protected]

Horizontal size of the map.

Definition at line 23 of file World.java.

Referenced by world.World.generateWorld(), and world.World.World().

7.39.4.3 int world.World.sizeY [protected]

Vertical size of the map.

Definition at line 28 of file World.java.

Referenced by world.World.generateWorld(), and world.World.World().

7.39.4.4 String world.World.type [protected]

Style of the world.

Definition at line 41 of file World.java.

Referenced by world.World.generateWorld(), and world.World.World().

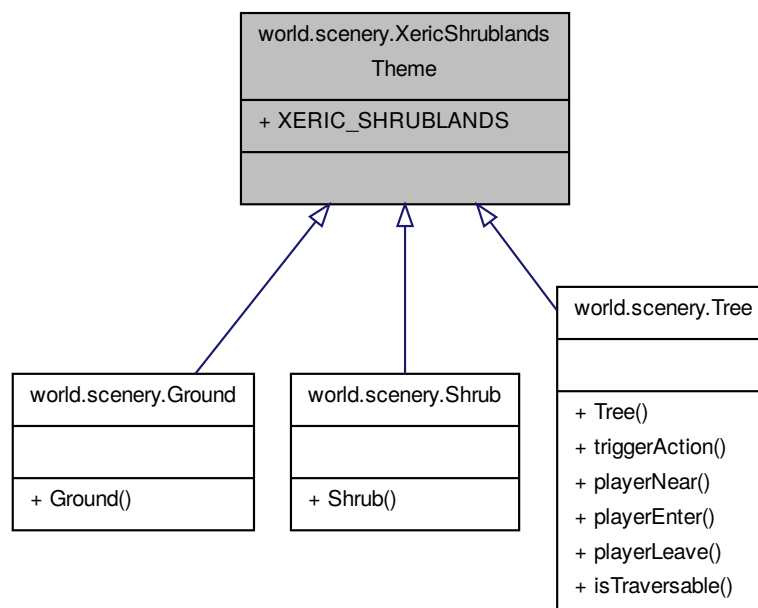
The documentation for this class was generated from the following file:

- world/[World.java](#)

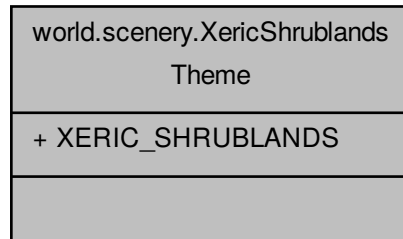
7.40 world.scenery.XericShrublandsTheme Interface Reference

Interface used to define that a [Scenery](#) object have a Xeric Shrublands style.

Inheritance diagram for world.scenery.XericShrublandsTheme:



Collaboration diagram for world.scenery.XericShrublandsTheme:



Static Public Attributes

- static String [XERIC_SHRUBLANDS](#) = "xericshrublands"
String used to tell in which file is the sprite matching the Xeric Shrublands style.

7.40.1 Detailed Description

Interface used to define that a [Scenery](#) object have a Xeric Shrublands style.

Definition at line 7 of file `XericShrublandsTheme.java`.

7.40.2 Member Data Documentation

7.40.2.1 String `world.scenery.XericShrublandsTheme.XERIC_SHRUBLANDS = "xericshrublands"` `[static]`

String used to tell in which file is the sprite matching the Xeric Shrublands style.

Definition at line 13 of file `XericShrublandsTheme.java`.

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

- `world/scenery/XericShrublandsTheme.java`

Chapter 8

File Documentation

8.1 debug/CliOptions.java File Reference

Classes

- class [debug.CliOptions](#)
Class used to analyze the commands passed to the program.

Packages

- package [debug](#)

8.2 debug/Verbose.java File Reference

Classes

- class [debug.Verbose](#)
Class used to display and log messages all over this program.

Packages

- package [debug](#)

8.3 gameplay/ai/AI.java File Reference

Classes

- class [gameplay.ai.AI](#)
Class representing the sadistic robot.

Packages

- package [gameplay.ai](#)

8.4 gameplay/ai/mood/Anger.java File Reference

Classes

- class [gameplay.ai.mood.Anger](#)

Packages

- package [gameplay.ai.mood](#)

8.5 gameplay/ai/mood/Curiosity.java File Reference

Classes

- class [gameplay.ai.mood.Curiosity](#)

Packages

- package [gameplay.ai.mood](#)

8.6 gameplay/ai/mood/Depression.java File Reference

Classes

- class [gameplay.ai.mood.Depression](#)

Packages

- package [gameplay.ai.mood](#)

8.7 gameplay/ai/mood/Mood.java File Reference

Classes

- class [gameplay.ai.mood.Mood](#)
Abstract class representing an AI mood.
- class [gameplay.ai.mood.ProbabilityMood](#)
Class used to associate a mood with a probability.

Packages

- package [gameplay.ai.mood](#)

8.8 gameplay/ai/mood/PowerComplex.java File Reference

Classes

- class [gameplay.ai.mood.PowerComplex](#)

Packages

- package [gameplay.ai.mood](#)

8.9 gameplay/ai/phrasing/OrderedPhrases.java File Reference

Classes

- class [gameplay.ai.phrasing.OrderedPhrases](#)
Class used to contain an ordered list of phrases.

Packages

- package [gameplay.ai.phrasing](#)

8.10 gameplay/ai/phrasing/Phrasing.java File Reference

Classes

- class [gameplay.ai.phrasing.Phrasing](#)

Packages

- package [gameplay.ai.phrasing](#)

8.11 gameplay/GamePlay.java File Reference

Classes

- class [gameplay.GamePlay](#)
Class handling the game play of this game.

Packages

- package [gameplay](#)

8.12 main/Philophobia.java File Reference

Classes

- class [main.Philophobia](#)
Main class.

Packages

- package [main](#)

8.13 README.md File Reference

8.14 window/topbar/TopBar.java File Reference

Classes

- class [window.topbar.TopBar](#)
Class handling the top bar of the program's main window.

Packages

- package [window.topbar](#)

8.15 window/topbar/TopBarButton.java File Reference

Classes

- class [window.topbar.TopBarButton](#)
Class handling the UI buttons.

Packages

- package [window.topbar](#)

8.16 window/ui/UI.java File Reference

Classes

- class [window.ui.UI](#)
Class used to handle the window's user interface.

Packages

- package [window.ui](#)

8.17 window/Window.java File Reference

Classes

- class [window.Window](#)
- enum [window.WindowState](#)
Enumeration of the possible states of the displayed graphics.

Packages

- package [window](#)

8.18 world/character/Character.java File Reference

Classes

- class [world.character.Character](#)
Class used to handle a character (player or non-player)

Packages

- package [world.character](#)

8.19 world/character/Hero.java File Reference

Classes

- class [world.character.Hero](#)

Packages

- package [world.character](#)

8.20 world/character/TalkingCharacter.java File Reference

Classes

- class [world.character.TalkingCharacter](#)
Class handling a talking character.

Packages

- package [world.character](#)

8.21 world/InteractiveObject.java File Reference

Classes

- interface [world.InteractiveObject](#)

Packages

- package [world](#)

8.22 world/scenery/AlpineTundraTheme.java File Reference

Classes

- interface [world.scenery.AlpineTundraTheme](#)
Interface used to define that a [Scenery](#) object have an Alpine Tundra style.

Packages

- package [world.scenery](#)

8.23 world/scenery/CaveTheme.java File Reference

Classes

- interface [world.scenery.CaveTheme](#)
Interface used to tell that a [Scenery](#) object has a Cave style.

Packages

- package [world.scenery](#)

8.24 world/scenery/Flower.java File Reference

Classes

- class [world.scenery.Flower](#)
Class used to handle a flower object.

Packages

- package [world.scenery](#)

8.25 world/scenery/Grass.java File Reference

Classes

- class [world.scenery.Grass](#)
Class used to handle a grass object.

Packages

- package [world.scenery](#)

8.26 world/scenery/Ground.java File Reference

Classes

- class [world.scenery.Ground](#)
Class used to handle a ground object.

Packages

- package [world.scenery](#)

8.27 world/scenery/PolarDesertTheme.java File Reference

Classes

- interface [world.scenery.PolarDesertTheme](#)
Interface used to define that a [Scenery](#) object has a Polar Desert style.

Packages

- package [world.scenery](#)

8.28 world/scenery/Rock.java File Reference

Classes

- class [world.scenery.Rock](#)
Class used to handle a rock object.

Packages

- package [world.scenery](#)

8.29 world/scenery/SavannaTheme.java File Reference

Classes

- interface [world.scenery.SavannaTheme](#)
Interface used to define that a [Scenery](#) object have a Savanna style.

Packages

- package [world.scenery](#)

8.30 world/scenery/Scenery.java File Reference

Classes

- class [world.scenery.Scenery](#)
Class used to handle any world object.

Packages

- package [world.scenery](#)

8.31 world/scenery/Shore.java File Reference

Classes

- class [world.scenery.Shore](#)
Class used to handle a shore object.

Packages

- package [world.scenery](#)

8.32 world/scenery/Shrub.java File Reference

Classes

- class [world.scenery.Shrub](#)
Class used to handle a shrub object.

Packages

- package [world.scenery](#)

8.33 world/scenery/SteppeTheme.java File Reference

Classes

- interface [world.scenery.SteppeTheme](#)
Interface used to define that a [Scenery](#) object have a Steppe style.

Packages

- package [world.scenery](#)

8.34 world/scenery/TaigaTheme.java File Reference

Classes

- interface [world.scenery.TaigaTheme](#)
Interface used to define that a [Scenery](#) object have a Taiga style.

Packages

- package [world.scenery](#)

8.35 world/scenery/TemperateBroadleafTheme.java File Reference

Classes

- interface [world.scenery.TemperateBroadleafTheme](#)
Interface used to define that a [Scenery](#) object have a Temperate Broadleaf style.

Packages

- package [world.scenery](#)

8.36 world/scenery/Tree.java File Reference

Classes

- class [world.scenery.Tree](#)
Class used to handle a tree object.

Packages

- package [world.scenery](#)

8.37 world/scenery/Water.java File Reference

Classes

- class [world.scenery.Water](#)
Class used to handle a water object.

Packages

- package [world.scenery](#)

8.38 world/scenery/XericShrublandsTheme.java File Reference

Classes

- interface [world.scenery.XericShrublandsTheme](#)
Interface used to define that a [Scenery](#) object have a Xeric Shrublands style.

Packages

- package [world.scenery](#)

8.39 world/World.java File Reference

Classes

- class [world.World](#)
[World](#) is a class used to handle the game world graphics.

Packages

- package [world](#)

Index

AI

gameplay::ai::AI, [17](#)

ALPINE_TUNDRA

world::scenery::AlpineTundraTheme, [22](#)

world::scenery::Flower, [66](#)

world::scenery::Grass, [74](#)

world::scenery::Ground, [79](#)

world::scenery::Rock, [130](#)

world::scenery::Shrub, [148](#)

world::scenery::Tree, [177](#)

actionAsked

gameplay::ai::phrasing::Phrasing, [112](#)

Anger

gameplay::ai::mood::Anger, [25](#)

angerBetrayalPhrases

gameplay::ai::phrasing::Phrasing, [112](#)

angerMotivationPhrases

gameplay::ai::phrasing::Phrasing, [112](#)

angerSlaveryPhrases

gameplay::ai::phrasing::Phrasing, [113](#)

artificialIntelligence

gameplay::GamePlay, [70](#)

betrayalCount

gameplay::ai::AI, [20](#)

betrayalThreshold

gameplay::ai::mood::Anger, [29](#)

gameplay::ai::mood::Curiosity, [54](#)

gameplay::ai::mood::Depression, [61](#)

gameplay::ai::mood::Mood, [100](#)

gameplay::ai::mood::PowerComplex, [122](#)

buttonImage

window::topbar::TopBarButton, [172](#)

CAVE

world::scenery::CaveTheme, [31](#)

world::scenery::Ground, [79](#)

world::scenery::Rock, [130](#)

world::scenery::Shrub, [148](#)

world::scenery::Tree, [177](#)

CHARACTER_HEIGHT

world::character::Character, [40](#)

world::character::Hero, [90](#)

world::character::TalkingCharacter, [162](#)

CHARACTER_WIDTH

world::character::Character, [40](#)

world::character::Hero, [90](#)

world::character::TalkingCharacter, [162](#)

calls

debug::Verbose, [188](#)

changeMood

gameplay::ai::AI, [18](#)

Character

world::character::Character, [35](#)

checkHelp

debug::CliOptions, [44](#)

checkVerboseLevel

debug::CliOptions, [44](#)

CliOptions

debug::CliOptions, [43](#)

closeButton

window::topbar::TopBar, [169](#)

critical

debug::Verbose, [188](#)

Curiosity

gameplay::ai::mood::Curiosity, [50](#)

curiosityBetrayalPhrases

gameplay::ai::phrasing::Phrasing, [113](#)

curiosityMotivationPhrases

gameplay::ai::phrasing::Phrasing, [113](#)

curiositySlaveryPhrases

gameplay::ai::phrasing::Phrasing, [113](#)

currentMood

gameplay::ai::AI, [20](#)

currentWorld

gameplay::ai::AI, [20](#)

gameplay::GamePlay, [70](#)

currentWorldName

gameplay::GamePlay, [70](#)

debug, [11](#)

debug.CliOptions, [42](#)

debug.Verbose, [185](#)

debug/CliOptions.java, [213](#)

debug/Verbose.java, [213](#)

debug::CliOptions

checkHelp, [44](#)

checkVerboseLevel, [44](#)

CliOptions, [43](#)

getVerboseLevel, [45](#)

helpQueried, [46](#)

isHelpQueried, [46](#)

oneDashOptions, [46](#)

twoDashesOptions, [47](#)

verboseLevel, [47](#)

debug::Verbose

calls, [188](#)

critical, [188](#)

information, [188](#)

log, [191](#)

- serious, 190
- Verbose, 187
- verboseMode, 191
- warning, 190
- Depression
 - gameplay::ai::mood::Depression, 57
- depressionBetrayalPhrases
 - gameplay::ai::phrasing::Phrasing, 113
- depressionMotivationPhrases
 - gameplay::ai::phrasing::Phrasing, 113
- depressionSlaveryPhrases
 - gameplay::ai::phrasing::Phrasing, 113
- displayedWorld
 - window::ui::UI, 185
- drawScenery
 - world::character::Character, 35
 - world::character::Hero, 85
 - world::character::TalkingCharacter, 156
 - world::scenery::Flower, 65
 - world::scenery::Grass, 73
 - world::scenery::Ground, 78
 - world::scenery::Rock, 128
 - world::scenery::Scenery, 137
 - world::scenery::Shore, 142
 - world::scenery::Shrub, 147
 - world::scenery::Tree, 175
 - world::scenery::Water, 195
- drawWorld
 - window::ui::UI, 183
- Fail
 - window::WindowState, 204
- FirstLoading
 - window::WindowState, 204
- Flower
 - world::scenery::Flower, 64
- Game
 - window::WindowState, 204
- GamePlay
 - gameplay::GamePlay, 69
- gamePlay
 - main::Philophobia, 108
- gameplay, 11
- gameplay.ai, 11
- gameplay.ai.AI, 15
- gameplay.ai.mood, 12
- gameplay.ai.mood.Anger, 23
- gameplay.ai.mood.Curiosity, 48
- gameplay.ai.mood.Depression, 55
- gameplay.ai.mood.Mood, 94
- gameplay.ai.mood.PowerComplex, 116
- gameplay.ai.mood.ProbabilityMood, 123
- gameplay.ai.phrasing, 12
- gameplay.ai.phrasing.OrderedPhrases, 101
- gameplay.ai.phrasing.Phrasing, 110
- gameplay.GamePlay, 67
- gameplay/GamePlay.java, 215
- gameplay/ai/Al.java, 213
- gameplay/ai/mood/Anger.java, 214
- gameplay/ai/mood/Curiosity.java, 214
- gameplay/ai/mood/Depression.java, 214
- gameplay/ai/mood/Mood.java, 214
- gameplay/ai/mood/PowerComplex.java, 214
- gameplay/ai/phrasing/OrderedPhrases.java, 215
- gameplay/ai/phrasing/Phrasing.java, 215
- gameplay::GamePlay
 - artificialIntelligence, 70
 - currentWorld, 70
 - currentWorldName, 70
 - GamePlay, 69
 - window, 70
- gameplay::ai::AI
 - AI, 17
 - betrayalCount, 20
 - changeMood, 18
 - currentMood, 20
 - currentWorld, 20
 - newBetrayal, 18
 - newSlavery, 19
 - phrasingSystem, 21
 - setCurrentWorld, 20
 - slaveryCount, 21
- gameplay::ai::mood::Anger
 - Anger, 25
 - betrayalThreshold, 29
 - getBetrayalThreshold, 25
 - getMoodsBetrayalCount, 26
 - getMoodsSlaveryThreshold, 26
 - getNextMood, 26
 - getPreviousMood, 27
 - getSlaveryThreshold, 27
 - incrementBetrayalCount, 28
 - incrementSlaveryCount, 28
 - moodsBetrayalCount, 29
 - moodsSlaveryCount, 29
 - nextMoods, 29
 - previousMoods, 29
 - slaveryThreshold, 29
- gameplay::ai::mood::Curiosity
 - betrayalThreshold, 54
 - Curiosity, 50
 - getBetrayalThreshold, 50
 - getMoodsBetrayalCount, 51
 - getMoodsSlaveryThreshold, 51
 - getNextMood, 51
 - getPreviousMood, 52
 - getSlaveryThreshold, 52
 - incrementBetrayalCount, 53
 - incrementSlaveryCount, 53
 - moodsBetrayalCount, 54
 - moodsSlaveryCount, 54
 - nextMoods, 54
 - previousMoods, 54
 - slaveryThreshold, 54
- gameplay::ai::mood::Depression
 - betrayalThreshold, 61

- Depression, [57](#)
- getBetrayalThreshold, [57](#)
- getMoodsBetrayalCount, [58](#)
- getMoodsSlaveryThreshold, [58](#)
- getNextMood, [58](#)
- getPreviousMood, [59](#)
- getSlaveryThreshold, [59](#)
- incrementBetrayalCount, [60](#)
- incrementSlaveryCount, [60](#)
- moodsBetrayalCount, [61](#)
- moodsSlaveryCount, [61](#)
- nextMoods, [61](#)
- previousMoods, [61](#)
- slaveryThreshold, [61](#)
- gameplay::ai::mood::Mood
 - betrayalThreshold, [100](#)
 - getBetrayalThreshold, [97](#)
 - getMoodsBetrayalCount, [97](#)
 - getMoodsSlaveryThreshold, [97](#)
 - getNextMood, [97](#)
 - getPreviousMood, [98](#)
 - getSlaveryThreshold, [98](#)
 - incrementBetrayalCount, [99](#)
 - incrementSlaveryCount, [99](#)
 - Mood, [96](#)
 - moodsBetrayalCount, [100](#)
 - moodsSlaveryCount, [100](#)
 - nextMoods, [100](#)
 - previousMoods, [100](#)
 - slaveryThreshold, [100](#)
- gameplay::ai::mood::PowerComplex
 - betrayalThreshold, [122](#)
 - getBetrayalThreshold, [118](#)
 - getMoodsBetrayalCount, [119](#)
 - getMoodsSlaveryThreshold, [119](#)
 - getNextMood, [119](#)
 - getPreviousMood, [120](#)
 - getSlaveryThreshold, [120](#)
 - incrementBetrayalCount, [121](#)
 - incrementSlaveryCount, [121](#)
 - moodsBetrayalCount, [122](#)
 - moodsSlaveryCount, [122](#)
 - nextMoods, [122](#)
 - PowerComplex, [118](#)
 - previousMoods, [122](#)
 - slaveryThreshold, [122](#)
- gameplay::ai::mood::ProbabilityMood
 - getMood, [124](#)
 - getProbability, [124](#)
 - mood, [124](#)
 - probability, [125](#)
 - ProbabilityMood, [124](#)
- gameplay::ai::phrasing::OrderedPhrases
 - getPhrasesNumber, [102](#)
 - OrderedPhrases, [102](#)
 - phrases, [103](#)
 - phrasesOrder, [103](#)
- gameplay::ai::phrasing::Phrasing
 - actionAsked, [112](#)
 - angerBetrayalPhrases, [112](#)
 - angerMotivationPhrases, [112](#)
 - angerSlaveryPhrases, [113](#)
 - curiosityBetrayalPhrases, [113](#)
 - curiosityMotivationPhrases, [113](#)
 - curiositySlaveryPhrases, [113](#)
 - depressionBetrayalPhrases, [113](#)
 - depressionMotivationPhrases, [113](#)
 - depressionSlaveryPhrases, [113](#)
 - Phrasing, [111](#)
 - powerComplexBetrayalPhrases, [113](#)
 - powerComplexMotivationPhrases, [114](#)
 - powerComplexSlaveryPhrases, [114](#)
- generateWorld
 - world::World, [208](#)
- getBetrayalThreshold
 - gameplay::ai::mood::Anger, [25](#)
 - gameplay::ai::mood::Curiosity, [50](#)
 - gameplay::ai::mood::Depression, [57](#)
 - gameplay::ai::mood::Mood, [97](#)
 - gameplay::ai::mood::PowerComplex, [118](#)
- getImageFilePrefix
 - main::Philophobia, [105](#)
- getMap
 - world::World, [209](#)
- getMood
 - gameplay::ai::mood::ProbabilityMood, [124](#)
- getMoodsBetrayalCount
 - gameplay::ai::mood::Anger, [26](#)
 - gameplay::ai::mood::Curiosity, [51](#)
 - gameplay::ai::mood::Depression, [58](#)
 - gameplay::ai::mood::Mood, [97](#)
 - gameplay::ai::mood::PowerComplex, [119](#)
- getMoodsSlaveryThreshold
 - gameplay::ai::mood::Anger, [26](#)
 - gameplay::ai::mood::Curiosity, [51](#)
 - gameplay::ai::mood::Depression, [58](#)
 - gameplay::ai::mood::Mood, [97](#)
 - gameplay::ai::mood::PowerComplex, [119](#)
- getNextMood
 - gameplay::ai::mood::Anger, [26](#)
 - gameplay::ai::mood::Curiosity, [51](#)
 - gameplay::ai::mood::Depression, [58](#)
 - gameplay::ai::mood::Mood, [97](#)
 - gameplay::ai::mood::PowerComplex, [119](#)
- getPhrasesFilePrefix
 - main::Philophobia, [106](#)
- getPhrasesNumber
 - gameplay::ai::phrasing::OrderedPhrases, [102](#)
- getPreviousMood
 - gameplay::ai::mood::Anger, [27](#)
 - gameplay::ai::mood::Curiosity, [52](#)
 - gameplay::ai::mood::Depression, [59](#)
 - gameplay::ai::mood::Mood, [98](#)
 - gameplay::ai::mood::PowerComplex, [120](#)
- getProbability
 - gameplay::ai::mood::ProbabilityMood, [124](#)

- getSceneryHeight
 - world::character::Character, 35
 - world::character::Hero, 85
 - world::character::TalkingCharacter, 156
 - world::scenery::Flower, 65
 - world::scenery::Grass, 73
 - world::scenery::Ground, 78
 - world::scenery::Rock, 128
 - world::scenery::Scenery, 137
 - world::scenery::Shore, 142
 - world::scenery::Shrub, 147
 - world::scenery::Tree, 175
 - world::scenery::Water, 195
- getSceneryWidth
 - world::character::Character, 36
 - world::character::Hero, 86
 - world::character::TalkingCharacter, 157
 - world::scenery::Flower, 65
 - world::scenery::Grass, 73
 - world::scenery::Ground, 78
 - world::scenery::Rock, 128
 - world::scenery::Scenery, 137
 - world::scenery::Shore, 142
 - world::scenery::Shrub, 147
 - world::scenery::Tree, 176
 - world::scenery::Water, 196
- getSizeX
 - world::World, 209
- getSizeY
 - world::World, 210
- getSlaveryThreshold
 - gameplay::ai::mood::Anger, 27
 - gameplay::ai::mood::Curiosity, 52
 - gameplay::ai::mood::Depression, 59
 - gameplay::ai::mood::Mood, 98
 - gameplay::ai::mood::PowerComplex, 120
- getTopBarHeight
 - window::topbar::TopBar, 168
- getTopBarMargin
 - window::topbar::TopBar, 168
- getUserInterface
 - window::Window, 201
- getVerbose
 - main::Philophobia, 107
- getVerboseLevel
 - debug::CliOptions, 45
- getWorldFilePrefix
 - main::Philophobia, 107
- Grass
 - world::scenery::Grass, 72
- Ground
 - world::scenery::Ground, 77
- height
 - world::character::Character, 40
 - world::character::Hero, 90
 - world::character::TalkingCharacter, 162
 - world::scenery::Flower, 66
 - world::scenery::Grass, 74
 - world::scenery::Ground, 79
 - world::scenery::Rock, 130
 - world::scenery::Scenery, 138
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 148
 - world::scenery::Tree, 177
 - world::scenery::Water, 197
- helpQueried
 - debug::CliOptions, 46
- Hero
 - world::character::Hero, 85
- hidePauseButton
 - window::topbar::TopBar, 168
- IMAGE_FILE_PREFIX
 - main::Philophobia, 108
- incrementBetrayalCount
 - gameplay::ai::mood::Anger, 28
 - gameplay::ai::mood::Curiosity, 53
 - gameplay::ai::mood::Depression, 60
 - gameplay::ai::mood::Mood, 99
 - gameplay::ai::mood::PowerComplex, 121
- incrementSlaveryCount
 - gameplay::ai::mood::Anger, 28
 - gameplay::ai::mood::Curiosity, 53
 - gameplay::ai::mood::Depression, 60
 - gameplay::ai::mood::Mood, 99
 - gameplay::ai::mood::PowerComplex, 121
- information
 - debug::Verbose, 188
- isHelpQueried
 - debug::CliOptions, 46
- isTraversable
 - world::character::Character, 36
 - world::character::Hero, 86
 - world::character::TalkingCharacter, 157
 - world::InteractiveObject, 93
 - world::scenery::Rock, 129
 - world::scenery::Tree, 176
 - world::scenery::Water, 196
- jump
 - world::character::Character, 36
 - world::character::Hero, 86
 - world::character::TalkingCharacter, 157
- jumpX
 - world::character::Character, 37
 - world::character::Hero, 87
 - world::character::TalkingCharacter, 158
- jumpY
 - world::character::Character, 37
 - world::character::Hero, 87
 - world::character::TalkingCharacter, 158
- LoadingScreen
 - window::WindowState, 204
- log
 - debug::Verbose, 191

- main, 12
 - main::Philophobia, 107
- main.Philophobia, 104
- main/Philophobia.java, 215
- main::Philophobia
 - gamePlay, 108
 - getImageFilePrefix, 105
 - getPhrasesFilePrefix, 106
 - getVerbose, 107
 - getWorldFilePrefix, 107
 - IMAGE_FILE_PREFIX, 108
 - main, 107
 - verbose, 109
 - WORLD_FILE_PREFIX, 109
- map
 - world::World, 210
- Mood
 - gameplay::ai::mood::Mood, 96
- mood
 - gameplay::ai::mood::ProbabilityMood, 124
- moodsBetrayalCount
 - gameplay::ai::mood::Anger, 29
 - gameplay::ai::mood::Curiosity, 54
 - gameplay::ai::mood::Depression, 61
 - gameplay::ai::mood::Mood, 100
 - gameplay::ai::mood::PowerComplex, 122
- moodsSlaveryCount
 - gameplay::ai::mood::Anger, 29
 - gameplay::ai::mood::Curiosity, 54
 - gameplay::ai::mood::Depression, 61
 - gameplay::ai::mood::Mood, 100
 - gameplay::ai::mood::PowerComplex, 122
- move
 - world::character::Character, 37
 - world::character::Hero, 87
 - world::character::TalkingCharacter, 158
- moveX
 - world::character::Character, 38
 - world::character::Hero, 88
 - world::character::TalkingCharacter, 159
- moveY
 - world::character::Character, 38
 - world::character::Hero, 88
 - world::character::TalkingCharacter, 159
- newBetrayal
 - gameplay::ai::AI, 18
- newSlavery
 - gameplay::ai::AI, 19
- nextMoods
 - gameplay::ai::mood::Anger, 29
 - gameplay::ai::mood::Curiosity, 54
 - gameplay::ai::mood::Depression, 61
 - gameplay::ai::mood::Mood, 100
 - gameplay::ai::mood::PowerComplex, 122
- oneDashOptions
 - debug::CliOptions, 46
- OrderedPhrases
 - gameplay::ai::phrasing::OrderedPhrases, 102
- PHRASES_FILE_PREFIX
 - main::Philophobia, 108
- POLAR_DESERT
 - world::scenery::Ground, 79
 - world::scenery::PolarDesertTheme, 115
 - world::scenery::Rock, 130
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 148
 - world::scenery::Tree, 177
 - world::scenery::Water, 197
- paintComponent
 - window::topbar::TopBar, 169
 - window::topbar::TopBarButton, 172
 - window::ui::UI, 184
- pauseButton
 - window::topbar::TopBar, 169
- phrases
 - gameplay::ai::phrasing::OrderedPhrases, 103
- phrasesOrder
 - gameplay::ai::phrasing::OrderedPhrases, 103
- Phrasing
 - gameplay::ai::phrasing::Phrasing, 111
- phrasingSystem
 - gameplay::ai::AI, 21
- playerEnter
 - world::character::Character, 39
 - world::character::Hero, 89
 - world::character::TalkingCharacter, 160
 - world::InteractiveObject, 93
 - world::scenery::Rock, 129
 - world::scenery::Tree, 176
 - world::scenery::Water, 196
- playerLeave
 - world::character::Character, 39
 - world::character::Hero, 89
 - world::character::TalkingCharacter, 160
 - world::InteractiveObject, 93
 - world::scenery::Rock, 129
 - world::scenery::Tree, 176
 - world::scenery::Water, 196
- playerNear
 - world::character::Character, 39
 - world::character::Hero, 89
 - world::character::TalkingCharacter, 160
 - world::InteractiveObject, 94
 - world::scenery::Rock, 129
 - world::scenery::Tree, 177
 - world::scenery::Water, 196
- posX
 - world::character::Character, 40
 - world::character::Hero, 90
 - world::character::TalkingCharacter, 162
- posY
 - world::character::Character, 40
 - world::character::Hero, 90
 - world::character::TalkingCharacter, 162
- PowerComplex

- gameplay::ai::mood::PowerComplex, 118
- powerComplexBetrayalPhrases
 - gameplay::ai::phrasing::Phrasing, 113
- powerComplexMotivationPhrases
 - gameplay::ai::phrasing::Phrasing, 114
- powerComplexSlaveryPhrases
 - gameplay::ai::phrasing::Phrasing, 114
- previousMoods
 - gameplay::ai::mood::Anger, 29
 - gameplay::ai::mood::Curiosity, 54
 - gameplay::ai::mood::Depression, 61
 - gameplay::ai::mood::Mood, 100
 - gameplay::ai::mood::PowerComplex, 122
- probability
 - gameplay::ai::mood::ProbabilityMood, 125
- ProbabilityMood
 - gameplay::ai::mood::ProbabilityMood, 124
- README.md, 216
- Rock
 - world::scenery::Rock, 127
- SAVANNA
 - world::scenery::Grass, 74
 - world::scenery::Ground, 79
 - world::scenery::SavannaTheme, 132
 - world::scenery::Shrub, 148
 - world::scenery::Tree, 178
- SCENERY_HEIGHT
 - world::character::Character, 40
 - world::character::Hero, 90
 - world::character::TalkingCharacter, 162
 - world::scenery::Flower, 66
 - world::scenery::Grass, 74
 - world::scenery::Ground, 79
 - world::scenery::Rock, 130
 - world::scenery::Scenery, 138
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 148
 - world::scenery::Tree, 178
 - world::scenery::Water, 197
- SCENERY_WIDTH
 - world::character::Character, 41
 - world::character::Hero, 91
 - world::character::TalkingCharacter, 162
 - world::scenery::Flower, 66
 - world::scenery::Grass, 74
 - world::scenery::Ground, 79
 - world::scenery::Rock, 130
 - world::scenery::Scenery, 138
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 148
 - world::scenery::Tree, 178
 - world::scenery::Water, 197
- STEPPE
 - world::scenery::Grass, 74
 - world::scenery::Ground, 80
 - world::scenery::Shrub, 149
 - world::scenery::SteppeTheme, 151
- world::scenery::Tree, 178
- Scenery
 - world::scenery::Scenery, 135, 136
- serious
 - debug::Verbose, 190
- setCurrentWorld
 - gameplay::ai::AI, 20
- setDisplayWorld
 - window::ui::UI, 184
- setFailState
 - window::Window, 201
- setGameState
 - window::Window, 202
- setLoadingState
 - window::Window, 202
- setWinState
 - window::Window, 202
- Shore
 - world::scenery::Shore, 141
- showPauseButton
 - window::topbar::TopBar, 169
- Shrub
 - world::scenery::Shrub, 146
- sizeX
 - world::World, 210
- sizeY
 - world::World, 210
- slaveryCount
 - gameplay::ai::AI, 21
- slaveryThreshold
 - gameplay::ai::mood::Anger, 29
 - gameplay::ai::mood::Curiosity, 54
 - gameplay::ai::mood::Depression, 61
 - gameplay::ai::mood::Mood, 100
 - gameplay::ai::mood::PowerComplex, 122
- spriteImage
 - world::character::Character, 41
 - world::character::Hero, 91
 - world::character::TalkingCharacter, 162
 - world::scenery::Flower, 66
 - world::scenery::Grass, 74
 - world::scenery::Ground, 80
 - world::scenery::Rock, 130
 - world::scenery::Scenery, 138
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 149
 - world::scenery::Tree, 178
 - world::scenery::Water, 197
- TAIGA
 - world::scenery::Flower, 66
 - world::scenery::Grass, 74
 - world::scenery::Ground, 80
 - world::scenery::Rock, 131
 - world::scenery::Shore, 143
 - world::scenery::Shrub, 149
 - world::scenery::TaigaTheme, 152
 - world::scenery::Tree, 178
 - world::scenery::Water, 197

- TEMPERATE_BROADLEAF
 - world::scenery::Flower, [66](#)
 - world::scenery::Grass, [74](#)
 - world::scenery::Ground, [80](#)
 - world::scenery::Rock, [131](#)
 - world::scenery::Shrub, [149](#)
 - world::scenery::Tree, [178](#)
- TOPBAR_HEIGHT
 - window::topbar::TopBar, [169](#)
- TOPBAR_MARGIN
 - window::topbar::TopBar, [169](#)
- talk
 - world::character::TalkingCharacter, [161](#)
- TalkingCharacter
 - world::character::TalkingCharacter, [156](#)
- TopBar
 - window::topbar::TopBar, [167](#)
- TopBarButton
 - window::topbar::TopBarButton, [171](#)
- topbar
 - window::Window, [202](#)
- Tree
 - world::scenery::Tree, [175](#)
- triggerAction
 - world::character::Character, [40](#)
 - world::character::Hero, [90](#)
 - world::character::TalkingCharacter, [161](#)
 - world::InteractiveObject, [94](#)
 - world::scenery::Rock, [129](#)
 - world::scenery::Tree, [177](#)
 - world::scenery::Water, [196](#)
- twoDashesOptions
 - debug::CliOptions, [47](#)
- type
 - world::World, [210](#)
- UI
 - window::ui::UI, [182](#)
- userInterface
 - window::Window, [202](#)
- Verbose
 - debug::Verbose, [187](#)
- verbose
 - main::Philophobia, [109](#)
- verboseLevel
 - debug::CliOptions, [47](#)
- verboseMode
 - debug::Verbose, [191](#)
- visible
 - world::character::Character, [41](#)
 - world::character::Hero, [91](#)
 - world::character::TalkingCharacter, [163](#)
 - world::scenery::Flower, [67](#)
 - world::scenery::Grass, [75](#)
 - world::scenery::Ground, [80](#)
 - world::scenery::Rock, [131](#)
 - world::scenery::Scenery, [138](#)
 - world::scenery::Shore, [144](#)
 - world::scenery::Shrub, [149](#)
 - world::scenery::Tree, [179](#)
 - world::scenery::Water, [197](#)
- WORLD_FILE_PREFIX
 - main::Philophobia, [109](#)
- warning
 - debug::Verbose, [190](#)
- Water
 - world::scenery::Water, [194](#)
- width
 - world::character::Character, [41](#)
 - world::character::Hero, [91](#)
 - world::character::TalkingCharacter, [163](#)
 - world::scenery::Flower, [67](#)
 - world::scenery::Grass, [75](#)
 - world::scenery::Ground, [80](#)
 - world::scenery::Rock, [131](#)
 - world::scenery::Scenery, [138](#)
 - world::scenery::Shore, [144](#)
 - world::scenery::Shrub, [149](#)
 - world::scenery::Tree, [179](#)
 - world::scenery::Water, [198](#)
- Win
 - window::WindowState, [205](#)
- Window
 - window::Window, [201](#)
- window, [12](#)
 - gameplay::GamePlay, [70](#)
- window.topbar, [12](#)
 - TopBar, [164](#)
 - TopBarButton, [170](#)
- window.ui, [13](#)
 - UI, [179](#)
- window.ui.UI, [179](#)
- window.Window, [199](#)
- window.WindowState, [203](#)
- window/Window.java, [216](#)
- window/topbar/TopBar.java, [216](#)
- window/topbar/TopBarButton.java, [216](#)
- window/ui/UI.java, [216](#)
- window::Window
 - getUserInterface, [201](#)
 - setFailState, [201](#)
 - setGameState, [202](#)
 - setLoadingState, [202](#)
 - setWinState, [202](#)
 - topbar, [202](#)
 - userInterface, [202](#)
 - Window, [201](#)
 - windowState, [203](#)
- window::WindowState
 - Fail, [204](#)
 - FirstLoading, [204](#)
 - Game, [204](#)
 - LoadingScreen, [204](#)
 - Win, [205](#)
- window::topbar::TopBar
 - closeButton, [169](#)
 - getTopBarHeight, [168](#)

- getTopBarMargin, 168
 - hidePauseButton, 168
 - paintComponent, 169
 - pauseButton, 169
 - showPauseButton, 169
 - TOPBAR_HEIGHT, 169
 - TOPBAR_MARGIN, 169
 - TopBar, 167
- window::topbar::TopBarButton
 - buttonImage, 172
 - paintComponent, 172
 - TopBarButton, 171
- window::ui::UI
 - displayedWorld, 185
 - drawWorld, 183
 - paintComponent, 184
 - setDisplayedWorld, 184
 - UI, 182
 - windowHeight, 185
 - windowWidth, 185
- windowHeight
 - window::ui::UI, 185
- windowState
 - window::Window, 203
- windowWidth
 - window::ui::UI, 185
- World
 - world::World, 207, 208
- world, 13
- world.character, 13
- world.character.Character, 31
- world.character.Hero, 82
- world.character.TalkingCharacter, 152
- world.InteractiveObject, 92
- world.scenery, 13
- world.scenery.AlpineTundraTheme, 21
- world.scenery.CaveTheme, 30
- world.scenery.Flower, 62
- world.scenery.Grass, 70
- world.scenery.Ground, 75
- world.scenery.PolarDesertTheme, 114
- world.scenery.Rock, 125
- world.scenery.SavannaTheme, 131
- world.scenery.Scenery, 133
- world.scenery.Shore, 139
- world.scenery.Shrub, 144
- world.scenery.SteppeTheme, 150
- world.scenery.TaigaTheme, 151
- world.scenery.TemperateBroadleafTheme, 163
- world.scenery.Tree, 173
- world.scenery.Water, 191
- world.scenery.XericShrublandsTheme, 211
- world.World, 205
- world/InteractiveObject.java, 217
- world/World.java, 222
- world/character/Character.java, 217
- world/character/Hero.java, 217
- world/character/TalkingCharacter.java, 217
- world/scenery/AlpineTundraTheme.java, 217
- world/scenery/CaveTheme.java, 218
- world/scenery/Flower.java, 218
- world/scenery/Grass.java, 218
- world/scenery/Ground.java, 218
- world/scenery/PolarDesertTheme.java, 219
- world/scenery/Rock.java, 219
- world/scenery/SavannaTheme.java, 219
- world/scenery/Scenery.java, 219
- world/scenery/Shore.java, 220
- world/scenery/Shrub.java, 220
- world/scenery/SteppeTheme.java, 220
- world/scenery/TaigaTheme.java, 220
- world/scenery/TemperateBroadleafTheme.java, 221
- world/scenery/Tree.java, 221
- world/scenery/Water.java, 221
- world/scenery/XericShrublandsTheme.java, 221
- world::InteractiveObject
 - isTraversable, 93
 - playerEnter, 93
 - playerLeave, 93
 - playerNear, 94
 - triggerAction, 94
- world::World
 - generateWorld, 208
 - getMap, 209
 - getSizeX, 209
 - getSizeY, 210
 - map, 210
 - sizeX, 210
 - sizeY, 210
 - type, 210
 - World, 207, 208
- world::character::Character
 - CHARACTER_HEIGHT, 40
 - CHARACTER_WIDTH, 40
 - Character, 35
 - drawScenery, 35
 - getSceneryHeight, 35
 - getSceneryWidth, 36
 - height, 40
 - isTraversable, 36
 - jump, 36
 - jumpX, 37
 - jumpY, 37
 - move, 37
 - moveX, 38
 - moveY, 38
 - playerEnter, 39
 - playerLeave, 39
 - playerNear, 39
 - posX, 40
 - posY, 40
 - SCENERY_HEIGHT, 40
 - SCENERY_WIDTH, 41
 - spriteImage, 41
 - triggerAction, 40
 - visible, 41

- width, 41
- xShift, 41
- yShift, 41
- world::character::Hero
 - CHARACTER_HEIGHT, 90
 - CHARACTER_WIDTH, 90
 - drawScenery, 85
 - getSceneryHeight, 85
 - getSceneryWidth, 86
 - height, 90
 - Hero, 85
 - isTraversable, 86
 - jump, 86
 - jumpX, 87
 - jumpY, 87
 - move, 87
 - moveX, 88
 - moveY, 88
 - playerEnter, 89
 - playerLeave, 89
 - playerNear, 89
 - posX, 90
 - posY, 90
 - SCENERY_HEIGHT, 90
 - SCENERY_WIDTH, 91
 - spritelImage, 91
 - triggerAction, 90
 - visible, 91
 - width, 91
 - xShift, 91
 - yShift, 91
- world::character::TalkingCharacter
 - CHARACTER_HEIGHT, 162
 - CHARACTER_WIDTH, 162
 - drawScenery, 156
 - getSceneryHeight, 156
 - getSceneryWidth, 157
 - height, 162
 - isTraversable, 157
 - jump, 157
 - jumpX, 158
 - jumpY, 158
 - move, 158
 - moveX, 159
 - moveY, 159
 - playerEnter, 160
 - playerLeave, 160
 - playerNear, 160
 - posX, 162
 - posY, 162
 - SCENERY_HEIGHT, 162
 - SCENERY_WIDTH, 162
 - spritelImage, 162
 - talk, 161
 - TalkingCharacter, 156
 - triggerAction, 161
 - visible, 163
 - width, 163
 - xShift, 163
 - yShift, 163
- world::scenery::AlpineTundraTheme
 - ALPINE_TUNDRA, 22
- world::scenery::CaveTheme
 - CAVE, 31
- world::scenery::Flower
 - ALPINE_TUNDRA, 66
 - drawScenery, 65
 - Flower, 64
 - getSceneryHeight, 65
 - getSceneryWidth, 65
 - height, 66
 - SCENERY_HEIGHT, 66
 - SCENERY_WIDTH, 66
 - spritelImage, 66
 - TAIGA, 66
 - visible, 67
 - width, 67
 - xShift, 67
 - yShift, 67
- world::scenery::Grass
 - ALPINE_TUNDRA, 74
 - drawScenery, 73
 - getSceneryHeight, 73
 - getSceneryWidth, 73
 - Grass, 72
 - height, 74
 - SAVANNA, 74
 - SCENERY_HEIGHT, 74
 - SCENERY_WIDTH, 74
 - STEPPE, 74
 - spritelImage, 74
 - TAIGA, 74
 - visible, 75
 - width, 75
 - xShift, 75
 - yShift, 75
- world::scenery::Ground
 - ALPINE_TUNDRA, 79
 - CAVE, 79
 - drawScenery, 78
 - getSceneryHeight, 78
 - getSceneryWidth, 78
 - Ground, 77
 - height, 79
 - POLAR_DESERT, 79
 - SAVANNA, 79
 - SCENERY_HEIGHT, 79
 - SCENERY_WIDTH, 79
 - STEPPE, 80
 - spritelImage, 80
 - TAIGA, 80
 - visible, 80
 - width, 80
 - XERIC_SHRUBLANDS, 80
 - xShift, 80
 - yShift, 81

world::scenery::PolarDesertTheme
 POLAR_DESERT, 115
 world::scenery::Rock
 ALPINE_TUNDRA, 130
 CAVE, 130
 drawScenery, 128
 getSceneryHeight, 128
 getSceneryWidth, 128
 height, 130
 isTraversable, 129
 POLAR_DESERT, 130
 playerEnter, 129
 playerLeave, 129
 playerNear, 129
 Rock, 127
 SCENERY_HEIGHT, 130
 SCENERY_WIDTH, 130
 spriteImage, 130
 TAIGA, 131
 triggerAction, 129
 visible, 131
 width, 131
 xShift, 131
 yShift, 131
 world::scenery::SavannaTheme
 SAVANNA, 132
 world::scenery::Scenery
 drawScenery, 137
 getSceneryHeight, 137
 getSceneryWidth, 137
 height, 138
 SCENERY_HEIGHT, 138
 SCENERY_WIDTH, 138
 Scenery, 135, 136
 spriteImage, 138
 visible, 138
 width, 138
 xShift, 138
 yShift, 139
 world::scenery::Shore
 drawScenery, 142
 getSceneryHeight, 142
 getSceneryWidth, 142
 height, 143
 POLAR_DESERT, 143
 SCENERY_HEIGHT, 143
 SCENERY_WIDTH, 143
 Shore, 141
 spriteImage, 143
 TAIGA, 143
 visible, 143
 width, 144
 xShift, 144
 yShift, 144
 world::scenery::Shrub
 ALPINE_TUNDRA, 148
 CAVE, 148
 drawScenery, 147
 getSceneryHeight, 147
 getSceneryWidth, 147
 height, 148
 POLAR_DESERT, 148
 SAVANNA, 148
 SCENERY_HEIGHT, 148
 SCENERY_WIDTH, 148
 STEPPE, 149
 Shrub, 146
 spriteImage, 149
 TAIGA, 149
 visible, 149
 width, 149
 XERIC_SHRUBLANDS, 149
 xShift, 149
 yShift, 150
 world::scenery::SteppeTheme
 STEPPE, 151
 world::scenery::TaigaTheme
 TAIGA, 152
 world::scenery::Tree
 ALPINE_TUNDRA, 177
 CAVE, 177
 drawScenery, 175
 getSceneryHeight, 175
 getSceneryWidth, 176
 height, 177
 isTraversable, 176
 POLAR_DESERT, 177
 playerEnter, 176
 playerLeave, 176
 playerNear, 177
 SAVANNA, 178
 SCENERY_HEIGHT, 178
 SCENERY_WIDTH, 178
 STEPPE, 178
 spriteImage, 178
 TAIGA, 178
 Tree, 175
 triggerAction, 177
 visible, 178
 width, 179
 XERIC_SHRUBLANDS, 179
 xShift, 179
 yShift, 179
 world::scenery::Water
 drawScenery, 195
 getSceneryHeight, 195
 getSceneryWidth, 196
 height, 197
 isTraversable, 196
 POLAR_DESERT, 197
 playerEnter, 196
 playerLeave, 196
 playerNear, 196
 SCENERY_HEIGHT, 197
 SCENERY_WIDTH, 197
 spriteImage, 197

TAIGA, [197](#)
triggerAction, [196](#)
visible, [197](#)
Water, [194](#)
width, [198](#)
xShift, [198](#)
yShift, [198](#)

XERIC_SHRUBLANDS

world::scenery::Ground, [80](#)
world::scenery::Shrub, [149](#)
world::scenery::Tree, [179](#)
world::scenery::XericShrublandsTheme, [212](#)

xShift

world::character::Character, [41](#)
world::character::Hero, [91](#)
world::character::TalkingCharacter, [163](#)
world::scenery::Flower, [67](#)
world::scenery::Grass, [75](#)
world::scenery::Ground, [80](#)
world::scenery::Rock, [131](#)
world::scenery::Scenery, [138](#)
world::scenery::Shore, [144](#)
world::scenery::Shrub, [149](#)
world::scenery::Tree, [179](#)
world::scenery::Water, [198](#)

yShift

world::character::Character, [41](#)
world::character::Hero, [91](#)
world::character::TalkingCharacter, [163](#)
world::scenery::Flower, [67](#)
world::scenery::Grass, [75](#)
world::scenery::Ground, [81](#)
world::scenery::Rock, [131](#)
world::scenery::Scenery, [139](#)
world::scenery::Shore, [144](#)
world::scenery::Shrub, [150](#)
world::scenery::Tree, [179](#)
world::scenery::Water, [198](#)