

PingForce TD

Készítette Doxygen 1.9.4



<b>1. Névtérmutató</b>	<b>1</b>
1.1. Névtérlista	1
<b>2. Hierarchikus mutató</b>	<b>3</b>
2.1. Osztályhierarchia	3
<b>3. Adatszerkezet-mutató</b>	<b>5</b>
3.1. Adatszerkezetek	5
<b>4. Fájlmutató</b>	<b>7</b>
4.1. Fájllista	7
<b>5. Névterek dokumentációja</b>	<b>9</b>
5.1. pftd névtér-referencia	9
5.1.1. Típusdefiníciók dokumentációja	10
5.1.1.1. EndPoint	10
5.2. pftd::gr névtér-referencia	10
5.3. pftd::utils névtér-referencia	10
5.3.1. Típusdefiníciók dokumentációja	10
5.3.1.1. Vec2f	10
5.3.1.2. Vec2i	11
5.4. pftd::utils::parser névtér-referencia	11
<b>6. Adatszerkezetek dokumentációja</b>	<b>13</b>
6.1. pftd::App osztályreferencia	13
6.1.1. Részletes leírás	14
6.1.2. Konstruktorkok és destruktorkok dokumentációja	14
6.1.2.1. App() [1/3]	14
6.1.2.2. App() [2/3]	14
6.1.2.3. ~App()	14
6.1.2.4. App() [3/3]	14
6.1.3. Tagfüggvények dokumentációja	15
6.1.3.1. addScene()	15
6.1.3.2. changeScene()	15
6.1.3.3. create()	15
6.1.3.4. destroy()	16
6.1.3.5. getInstance()	16
6.1.3.6. getWindowHeight()	16
6.1.3.7. getWindowWidth()	17
6.1.3.8. isRunning()	17
6.1.3.9. operator=()	17
6.1.3.10. run()	17
6.1.4. Adatmezők dokumentációja	17
6.1.4.1. m_activeSceneID	17

6.1.4.2.	m_instance	18
6.1.4.3.	m_renderer	18
6.1.4.4.	m_running	18
6.1.4.5.	m_scenes	18
6.2.	pftd::gr::Button osztályreferencia	18
6.2.1.	Részletes leírás	19
6.2.2.	Konstruktorok és destruktorok dokumentációja	19
6.2.2.1.	Button()	19
6.2.2.2.	~Button()	20
6.2.3.	Tagfüggvények dokumentációja	20
6.2.3.1.	draw()	20
6.2.3.2.	getLabel()	20
6.2.3.3.	handleClick()	20
6.2.3.4.	setBackground()	21
6.2.3.5.	setSound()	21
6.2.4.	Adatmezők dokumentációja	21
6.2.4.1.	label	21
6.2.4.2.	m_background	21
6.2.4.3.	m_clickSound	22
6.2.4.4.	m_rect	22
6.3.	pftd::Clickable osztályreferencia	22
6.3.1.	Részletes leírás	23
6.3.2.	Konstruktorok és destruktorok dokumentációja	23
6.3.2.1.	Clickable() [1/2]	23
6.3.2.2.	Clickable() [2/2]	23
6.3.3.	Tagfüggvények dokumentációja	23
6.3.3.1.	handleClick()	23
6.3.3.2.	setCallback()	23
6.3.4.	Adatmezők dokumentációja	25
6.3.4.1.	isActive	25
6.3.4.2.	m_callback	25
6.4.	pftd::Object::Compare struktúrareferencia	25
6.4.1.	Részletes leírás	25
6.4.2.	Tagfüggvények dokumentációja	25
6.4.2.1.	operator() [1/2]	26
6.4.2.2.	operator() [2/2]	26
6.5.	pftd::utils::Container< T, C > osztálysablon-referencia	26
6.5.1.	Részletes leírás	26
6.5.2.	Konstruktorok és destruktorok dokumentációja	27
6.5.2.1.	Container()	27
6.5.2.2.	~Container()	27
6.5.3.	Tagfüggvények dokumentációja	27

6.5.3.1.	<code>append()</code>	27
6.5.3.2.	<code>getContainer()</code>	27
6.5.3.3.	<code>size()</code>	28
6.5.4.	Adatmezők dokumentációja	28
6.5.4.1.	<code>m_container</code>	28
6.6.	<code>pftd::Cub</code> osztályreferencia	28
6.6.1.	Részletes leírás	29
6.6.2.	Konstruktorok és destruktorok dokumentációja	29
6.6.2.1.	<code>Cub()</code>	29
6.6.3.	Tagfüggvények dokumentációja	29
6.6.3.1.	<code>clone()</code>	29
6.7.	<code>pftd::Entity</code> osztályreferencia	30
6.7.1.	Részletes leírás	31
6.7.2.	Konstruktorok és destruktorok dokumentációja	31
6.7.2.1.	<code>Entity()</code> [1/4]	31
6.7.2.2.	<code>Entity()</code> [2/4]	31
6.7.2.3.	<code>Entity()</code> [3/4]	31
6.7.2.4.	<code>Entity()</code> [4/4]	32
6.7.2.5.	<code>~Entity()</code>	32
6.7.3.	Tagfüggvények dokumentációja	32
6.7.3.1.	<code>advanceAnimationFrame()</code>	32
6.7.3.2.	<code>clone()</code>	32
6.7.3.3.	<code>draw()</code>	32
6.7.3.4.	<code>getPosition()</code>	33
6.7.3.5.	<code>getSprite()</code> [1/2]	33
6.7.3.6.	<code>getSprite()</code> [2/2]	33
6.7.3.7.	<code>setPosition()</code>	33
6.7.3.8.	<code>update()</code>	34
6.7.4.	Adatmezők dokumentációja	34
6.7.4.1.	<code>CELL_N</code>	34
6.7.4.2.	<code>cellSize</code>	34
6.7.4.3.	<code>currentCell</code>	34
6.7.4.4.	<code>currentSprite</code>	34
6.7.4.5.	<code>frameDurationSec</code>	35
6.7.4.6.	<code>isAnimated</code>	35
6.7.4.7.	<code>spriteSheet</code>	35
6.7.4.8.	<code>totalElapsedSec</code>	35
6.8.	<code>pftd::utils::parser::SaveFileParser::EntityInfo</code> struktúráreferencia	35
6.8.1.	Részletes leírás	36
6.8.2.	Adatmezők dokumentációja	36
6.8.2.1.		36
6.8.2.2.	<code>entityType</code>	36

6.8.2.3.	hp	36
6.8.2.4.	position	36
6.8.2.5.	sealID	36
6.8.2.6.	towerID	36
6.9.	pftd::FollowPath struktúrareferencia	37
6.9.1.	Részletes leírás	37
6.9.2.	Konstruktorok és destruktorok dokumentációja	37
6.9.2.1.	FollowPath()	37
6.10.	pftd::FZC osztályreferencia	37
6.10.1.	Részletes leírás	38
6.10.2.	Konstruktorok és destruktorok dokumentációja	38
6.10.2.1.	FZC()	38
6.10.3.	Tagfüggvények dokumentációja	38
6.10.3.1.	clone()	38
6.11.	pftd::GameScene osztályreferencia	39
6.11.1.	Részletes leírás	40
6.11.2.	Konstruktorok és destruktorok dokumentációja	40
6.11.2.1.	GameScene()	40
6.11.2.2.	~GameScene()	40
6.11.3.	Tagfüggvények dokumentációja	40
6.11.3.1.	onEvent()	40
6.11.3.2.	startGame()	40
6.11.3.3.	toggleActive()	41
6.11.3.4.	update()	41
6.11.3.5.	updateScore()	41
6.11.3.6.	updateWealth()	41
6.11.4.	Adatmezők dokumentációja	42
6.11.4.1.	m_gameoverSound	42
6.11.4.2.	m_hornSound	42
6.11.4.3.	m_inventory	42
6.11.4.4.	m_level	42
6.11.4.5.	m_moneyCounter	42
6.11.4.6.	m_saveButt	42
6.11.4.7.	m_scoreCounter	43
6.12.	pftd::IcicleStabber osztályreferencia	43
6.12.1.	Részletes leírás	43
6.12.2.	Konstruktorok és destruktorok dokumentációja	43
6.12.2.1.	IcicleStabber() [1/2]	44
6.12.2.2.	IcicleStabber() [2/2]	44
6.12.2.3.	~IcicleStabber()	44
6.12.3.	Tagfüggvények dokumentációja	44
6.12.3.1.	clone()	44

6.12.3.2. update()	44
6.13. pftd::GameScene::Inventory struktúráreferencia	45
6.13.1. Részletes leírás	45
6.13.2. Konstruktork és destruktorok dokumentációja	45
6.13.2.1. Inventory()	45
6.13.2.2. ~Inventory()	46
6.13.3. Tagfüggvények dokumentációja	46
6.13.3.1. draw()	46
6.13.4. Adatmezők dokumentációja	46
6.13.4.1. background	46
6.14. pftd::GameScene::InventoryItem struktúráreferencia	47
6.14.1. Részletes leírás	47
6.14.2. Konstruktork és destruktorok dokumentációja	47
6.14.2.1. InventoryItem()	47
6.14.2.2. ~InventoryItem()	48
6.14.3. Tagfüggvények dokumentációja	48
6.14.3.1. draw()	48
6.14.4. Adatmezők dokumentációja	48
6.14.4.1. frame	48
6.14.4.2. icon	48
6.14.4.3. priceLabel	48
6.14.4.4. towerToSpawn	49
6.15. pftd::gr::Label osztályreferencia	49
6.15.1. Részletes leírás	49
6.15.2. Konstruktork és destruktorok dokumentációja	50
6.15.2.1. Label() [1/2]	50
6.15.2.2. Label() [2/2]	50
6.15.3. Tagfüggvények dokumentációja	50
6.15.3.1. draw()	50
6.15.3.2. getText() [1/2]	51
6.15.3.3. getText() [2/2]	51
6.15.3.4. setLabel()	51
6.15.3.5. setOutline()	51
6.15.4. Adatmezők dokumentációja	52
6.15.4.1. m_text	52
6.16. pftd::Level osztályreferencia	52
6.16.1. Részletes leírás	53
6.16.2. Konstruktork és destruktorok dokumentációja	53
6.16.2.1. Level()	53
6.16.2.2. ~Level()	54
6.16.3. Tagfüggvények dokumentációja	54
6.16.3.1. deselectTower()	54

6.16.3.2. draw()	54
6.16.3.3. isGameOver()	54
6.16.3.4. loseHP()	55
6.16.3.5. placeTower()	55
6.16.3.6. selectTower()	55
6.16.3.7. spawnSeal()	55
6.16.3.8. update()	56
6.16.4. Adatmezők dokumentációja	56
6.16.4.1. config	56
6.16.4.2. followPath	56
6.16.4.3. m_accuTimeSpawnSec	56
6.16.4.4. nest	56
6.16.4.5. projectiles	56
6.16.4.6. seals	57
6.16.4.7. selectedTower	57
6.16.4.8. stats	57
6.16.4.9. towers	57
6.17. pftd::utils::parser::LevelConfigParser osztályreferencia	57
6.17.1. Részletes leírás	58
6.17.2. Konstruktorok és destruktorok dokumentációja	58
6.17.2.1. LevelConfigParser()	58
6.17.2.2. ~LevelConfigParser()	58
6.17.3. Tagfüggvények dokumentációja	58
6.17.3.1. _getAttribute()	58
6.17.3.2. getAttribute()	59
6.17.3.3. parse()	59
6.17.4. Adatmezők dokumentációja	59
6.17.4.1. m_attribs	59
6.17.4.2. m_lastAttribute	59
6.18. pftd::MenuScene osztályreferencia	59
6.18.1. Részletes leírás	60
6.18.2. Konstruktorok és destruktorok dokumentációja	60
6.18.2.1. MenuScene()	60
6.18.2.2. ~MenuScene()	60
6.18.3. Tagfüggvények dokumentációja	60
6.18.3.1. onEvent()	60
6.18.3.2. update()	61
6.18.4. Adatmezők dokumentációja	61
6.18.4.1. m_buttons	61
6.19. pftd::Level::Nest struktúráreferencia	61
6.19.1. Részletes leírás	62
6.19.2. Konstruktorok és destruktorok dokumentációja	62



6.19.2.1. Nest() [1/2]	62
6.19.2.2. Nest() [2/2]	62
6.19.2.3. ~Nest()	62
6.19.3. Adatmezők dokumentációja	63
6.19.3.1. radiusPixel	63
6.20. pftd::Object osztályreferencia	63
6.20.1. Részletes leírás	64
6.20.2. Konstruktork és destruktorok dokumentációja	64
6.20.2.1. Object() [1/4]	64
6.20.2.2. Object() [2/4]	64
6.20.2.3. Object() [3/4]	64
6.20.2.4. Object() [4/4]	64
6.20.2.5. ~Object()	64
6.20.3. Tagfüggvények dokumentációja	65
6.20.3.1. draw()	65
6.20.4. Adatmezők dokumentációja	65
6.20.4.1. position	65
6.20.4.2. size	65
6.20.4.3. zIndex	65
6.21. pftd::utils::parser::Parser osztályreferencia	66
6.21.1. Konstruktork és destruktorok dokumentációja	66
6.21.1.1. Parser()	66
6.21.1.2. ~Parser()	67
6.21.2. Tagfüggvények dokumentációja	67
6.21.2.1. _skipLine()	67
6.21.2.2. _skipWhitespace()	67
6.21.2.3. get()	67
6.21.2.4. isLabelValid()	67
6.21.2.5. parse()	68
6.21.2.6. peekAhead()	68
6.21.2.7. reset()	68
6.21.2.8. skip() [1/2]	68
6.21.2.9. skip() [2/2]	69
6.21.3. Adatmezők dokumentációja	69
6.21.3.1. commentDenoter	69
6.21.3.2. sourceStream	69
6.21.3.3. validLabel	69
6.22. pftd::Projectile osztályreferencia	70
6.22.1. Konstruktork és destruktorok dokumentációja	70
6.22.1.1. ~Projectile()	70
6.22.1.2. Projectile()	71
6.22.2. Tagfüggvények dokumentációja	71

6.22.2.1. update()	71
6.22.3. Adatmezők dokumentációja	71
6.22.3.1. angularVelocityRadPerSec	71
6.22.3.2. direction	72
6.22.3.3. linearSpeed	72
6.23. pftd::utils::Random osztályreferencia	72
6.23.1. Részletes leírás	72
6.23.2. Konstruktorkok és destruktorkok dokumentációja	72
6.23.2.1. Random()	73
6.23.3. Tagfüggvények dokumentációja	73
6.23.3.1. generate()	73
6.23.4. Adatmezők dokumentációja	73
6.23.4.1. randomEngine	73
6.24. pftd::RegularSeal osztályreferencia	74
6.24.1. Részletes leírás	74
6.24.2. Konstruktorkok és destruktorkok dokumentációja	74
6.24.2.1. RegularSeal()	74
6.24.3. Tagfüggvények dokumentációja	75
6.24.3.1. clone()	75
6.25. pftd::Renderer osztályreferencia	75
6.25.1. Részletes leírás	76
6.25.2. Konstruktorkok és destruktorkok dokumentációja	76
6.25.2.1. Renderer() [1/3]	76
6.25.2.2. Renderer() [2/3]	76
6.25.2.3. Renderer() [3/3]	76
6.25.2.4. ~Renderer()	76
6.25.3. Tagfüggvények dokumentációja	77
6.25.3.1. clear()	77
6.25.3.2. display()	77
6.25.3.3. getWindow()	77
6.25.3.4. pushQueue()	77
6.25.3.5. render()	78
6.25.4. Barát és kapcsolódó függvények dokumentációja	78
6.25.4.1. App	78
6.25.5. Adatmezők dokumentációja	78
6.25.5.1. m_height	78
6.25.5.2. m_queue	78
6.25.5.3. m_width	78
6.25.5.4. m_window	78
6.26. pftd::ResourceManager osztályreferencia	79
6.26.1. Részletes leírás	79
6.26.2. Konstruktorkok és destruktorkok dokumentációja	80

6.26.2.1. ResourceManager() [1/3]	80
6.26.2.2. ResourceManager() [2/3]	80
6.26.2.3. ~ResourceManager()	80
6.26.2.4. ResourceManager() [3/3]	80
6.26.3. Tagfüggvények dokumentációja	80
6.26.3.1. create()	80
6.26.3.2. destroy()	81
6.26.3.3. getDefaultFont()	81
6.26.3.4. getInstance()	81
6.26.3.5. getSound()	81
6.26.3.6. getTexture()	82
6.26.3.7. loadDefaultFont()	82
6.26.4. Adatmezők dokumentációja	82
6.26.4.1. m_defaultFont	82
6.26.4.2. m_instance	83
6.26.4.3. m_sounds	83
6.26.4.4. m_textures	83
6.27. pftd::utils::parser::SaveFileParser osztályreferencia	83
6.27.1. Részletes leírás	84
6.27.2. Enumeráció-tagok dokumentációja	84
6.27.2.1. EntityType	84
6.27.3. Konstruktorkok és destruktorkok dokumentációja	84
6.27.3.1. SaveFileParser()	84
6.27.3.2. ~SaveFileParser()	85
6.27.4. Tagfüggvények dokumentációja	85
6.27.4.1. getEntities()	85
6.27.4.2. getStats()	85
6.27.4.3. parse()	85
6.27.5. Adatmezők dokumentációja	85
6.27.5.1. m_entities	86
6.27.5.2. m_readStats	86
6.28. pftd::Scene osztályreferencia	86
6.28.1. Részletes leírás	87
6.28.2. Típusdefiníció-tagok dokumentációja	87
6.28.2.1. ObjPtrVec	87
6.28.3. Konstruktorkok és destruktorkok dokumentációja	87
6.28.3.1. Scene() [1/3]	87
6.28.3.2. Scene() [2/3]	87
6.28.3.3. Scene() [3/3]	87
6.28.3.4. ~Scene()	88
6.28.4. Tagfüggvények dokumentációja	88
6.28.4.1. getObjects() [1/2]	88

6.28.4.2. <a href="#">getObjects()</a> [2/2]	88
6.28.4.3. <a href="#">onEvent()</a>	88
6.28.4.4. <a href="#">setMusic()</a>	89
6.28.4.5. <a href="#">toggleActive()</a>	89
6.28.4.6. <a href="#">update()</a>	89
6.28.5. <a href="#">Adatmezők dokumentációja</a>	90
6.28.5.1. <a href="#">backgroundMusic</a>	90
6.28.5.2. <a href="#">isActive</a>	90
6.28.5.3. <a href="#">objects</a>	90
6.29. <a href="#">pftd::SceneError</a> struktúrareferencia	90
6.29.1. <a href="#">Részletes leírás</a>	90
6.30. <a href="#">pftd::Seal</a> osztályreferencia	91
6.30.1. <a href="#">Részletes leírás</a>	92
6.30.2. <a href="#">Konstruktorok és destruktorok dokumentációja</a>	92
6.30.2.1. <a href="#">~Seal()</a>	92
6.30.2.2. <a href="#">Seal()</a>	92
6.30.3. <a href="#">Tagfüggvények dokumentációja</a>	93
6.30.3.1. <a href="#">advanceAnimationFrame()</a>	93
6.30.3.2. <a href="#">clone()</a>	93
6.30.3.3. <a href="#">damage()</a>	93
6.30.3.4. <a href="#">hasCompletedPath()</a>	93
6.30.3.5. <a href="#">hasReachedNest()</a>	94
6.30.3.6. <a href="#">lerpPath()</a>	94
6.30.3.7. <a href="#">update()</a>	94
6.30.4. <a href="#">Adatmezők dokumentációja</a>	94
6.30.4.1. <a href="#">followPath</a>	94
6.30.4.2. <a href="#">hp</a>	95
6.30.4.3. <a href="#">isCurrentlyStealing</a>	95
6.30.4.4. <a href="#">lerpParam</a>	95
6.30.4.5. <a href="#">nextPoint</a>	95
6.30.4.6. <a href="#">reachedNest</a>	95
6.30.4.7. <a href="#">returned</a>	95
6.30.4.8. <a href="#">speed</a>	95
6.30.4.9. <a href="#">value</a>	96
6.31. <a href="#">pftd::utils::Serializable</a> osztályreferencia	96
6.31.1. <a href="#">Részletes leírás</a>	96
6.31.2. <a href="#">Tagfüggvények dokumentációja</a>	96
6.31.2.1. <a href="#">deserialize()</a>	96
6.31.2.2. <a href="#">serialize()</a>	97
6.32. <a href="#">pftd::Snowball</a> osztályreferencia	97
6.32.1. <a href="#">Részletes leírás</a>	98
6.32.2. <a href="#">Konstruktorok és destruktorok dokumentációja</a>	98

6.32.2.1. Snowball()	98
6.32.3. Tagfüggvények dokumentációja	98
6.32.3.1. clone()	98
6.33. pftd::Snowballer osztályreferencia	99
6.33.1. Részletes leírás	99
6.33.2. Konstruktorok és destruktorok dokumentációja	99
6.33.2.1. Snowballer() [1/2]	99
6.33.2.2. Snowballer() [2/2]	100
6.33.2.3. ~Snowballer()	100
6.33.3. Tagfüggvények dokumentációja	100
6.33.3.1. attack()	100
6.33.3.2. clone()	100
6.33.3.3. update()	100
6.34. pftd::gr::Sprite osztályreferencia	101
6.34.1. Részletes leírás	102
6.34.2. Konstruktorok és destruktorok dokumentációja	102
6.34.2.1. Sprite() [1/5]	102
6.34.2.2. Sprite() [2/5]	102
6.34.2.3. Sprite() [3/5]	102
6.34.2.4. Sprite() [4/5]	102
6.34.2.5. Sprite() [5/5]	103
6.34.2.6. ~Sprite()	103
6.34.3. Tagfüggvények dokumentációja	103
6.34.3.1. draw()	103
6.34.3.2. flipY()	103
6.34.3.3. modColor()	103
6.34.3.4. setSpriteRect()	104
6.34.4. Adatmezők dokumentációja	104
6.34.4.1. m_sprite	104
6.34.4.2. m_texture	104
6.35. pftd::Level::Stats struktúrareferencia	104
6.35.1. Részletes leírás	105
6.35.2. Konstruktorok és destruktorok dokumentációja	105
6.35.2.1. Stats() [1/2]	105
6.35.2.2. Stats() [2/2]	105
6.35.3. Tagfüggvények dokumentációja	105
6.35.3.1. deserialize()	105
6.35.3.2. serialize()	106
6.35.4. Adatmezők dokumentációja	106
6.35.4.1. hp	106
6.35.4.2. MAX_HP	106
6.35.4.3. money	106

6.35.4.4. score	107
6.36. pftd::utils::parser::SaveFileParser::StatsInfo struktúráreferencia	107
6.36.1. Részletes leírás	107
6.36.2. Adatmezők dokumentációja	107
6.36.2.1. hp	107
6.36.2.2. score	107
6.36.2.3. wealth	108
6.37. pftd::Tower osztályreferencia	108
6.37.1. Részletes leírás	109
6.37.2. Típusdefiníció-tagok dokumentációja	109
6.37.2.1. ProjSpawnFunc	109
6.37.3. Konstruktorkok és destruktorkok dokumentációja	109
6.37.3.1. Tower() [1/3]	110
6.37.3.2. Tower() [2/3]	110
6.37.3.3. Tower() [3/3]	110
6.37.3.4. ~Tower()	110
6.37.4. Tagfüggvények dokumentációja	110
6.37.4.1. advanceAnimationFrame()	110
6.37.4.2. attack()	111
6.37.4.3. clone()	111
6.37.4.4. lookForTarget()	111
6.37.4.5. setProjSpawnCb()	111
6.37.4.6. update()	112
6.37.5. Adatmezők dokumentációja	112
6.37.5.1. attackDamage	112
6.37.5.2. attackRangePixel	112
6.37.5.3. attackSpeedSec	112
6.37.5.4. attackTimerSec	113
6.37.5.5. instantAttack	113
6.37.5.6. price	113
6.37.5.7. radiusPixel	113
6.37.5.8. spawnProjectile	113
6.37.5.9. target	113
6.38. pftd::utils::Vec2< T > struktúrasablon-referencia	113
6.38.1. Részletes leírás	114
6.38.2. Konstruktorkok és destruktorkok dokumentációja	114
6.38.2.1. Vec2() [1/3]	114
6.38.2.2. Vec2() [2/3]	115
6.38.2.3. Vec2() [3/3]	115
6.38.3. Tagfüggvények dokumentációja	115
6.38.3.1. distance()	115
6.38.3.2. normalize()	115

6.38.3.3. operator!=(())	116
6.38.3.4. operator*()	116
6.38.3.5. operator+()	116
6.38.3.6. operator-()	116
6.38.3.7. operator/()	116
6.38.3.8. operator=()	116
6.38.3.9. operator==(())	117
6.38.4. Barát és kapcsolódó függvények dokumentációja	117
6.38.4.1. operator<<	117
6.38.5. Adatmezők dokumentációja	117
6.38.5.1. x	117
6.38.5.2. y	117
6.39. pftd::ZombieSeal osztályreferencia	118
6.39.1. Részletes leírás	118
6.39.2. Konstruktorkok és destruktorkok dokumentációja	118
6.39.2.1. ZombieSeal()	118
6.39.3. Tagfüggvények dokumentációja	119
6.39.3.1. clone()	119
<b>7. Fájlok dokumentációja</b>	<b>121</b>
7.1. include/all.hpp fájlreferencia	121
7.2. all.hpp	121
7.3. include/app.hpp fájlreferencia	122
7.4. app.hpp	122
7.5. include/game/level.hpp fájlreferencia	123
7.6. level.hpp	124
7.7. include/objects/clickable.hpp fájlreferencia	125
7.8. clickable.hpp	125
7.9. include/objects/entities/entity_base.hpp fájlreferencia	126
7.10. entity_base.hpp	126
7.11. include/objects/entities/projectiles/projectile_base.hpp fájlreferencia	127
7.12. projectile_base.hpp	127
7.13. include/objects/entities/projectiles/snowball.hpp fájlreferencia	127
7.14. snowball.hpp	128
7.15. include/objects/entities/seals/cub.hpp fájlreferencia	128
7.16. cub.hpp	128
7.17. include/objects/entities/seals/followpath.hpp fájlreferencia	129
7.18. followpath.hpp	129
7.19. include/objects/entities/seals/fortified_zombie_cub.hpp fájlreferencia	129
7.20. fortified_zombie_cub.hpp	130
7.21. include/objects/entities/seals/regular.hpp fájlreferencia	130
7.22. regular.hpp	130

7.23. include/objects/entities/seals/seal_base.hpp fájlreferencia . . . . .	130
7.24. seal_base.hpp . . . . .	131
7.25. include/objects/entities/seals/zombie.hpp fájlreferencia . . . . .	131
7.26. zombie.hpp . . . . .	132
7.27. include/objects/entities/towers/iclestabber.hpp fájlreferencia . . . . .	132
7.28. iclestabber.hpp . . . . .	132
7.29. include/objects/entities/towers/snowballer.hpp fájlreferencia . . . . .	133
7.30. snowballer.hpp . . . . .	133
7.31. include/objects/entities/towers/tower_base.hpp fájlreferencia . . . . .	133
7.32. tower_base.hpp . . . . .	134
7.33. include/objects/gui/button.hpp fájlreferencia . . . . .	134
7.34. button.hpp . . . . .	135
7.35. include/objects/gui/image.hpp fájlreferencia . . . . .	135
7.36. image.hpp . . . . .	136
7.37. include/objects/gui/label.hpp fájlreferencia . . . . .	136
7.38. label.hpp . . . . .	137
7.39. include/objects/object_base.hpp fájlreferencia . . . . .	137
7.40. object_base.hpp . . . . .	138
7.41. include/resources.hpp fájlreferencia . . . . .	138
7.42. resources.hpp . . . . .	139
7.43. include/scene.hpp fájlreferencia . . . . .	139
7.44. scene.hpp . . . . .	140
7.45. include/scenes/game.hpp fájlreferencia . . . . .	140
7.46. game.hpp . . . . .	141
7.47. include/scenes/menu.hpp fájlreferencia . . . . .	142
7.48. menu.hpp . . . . .	142
7.49. include/utls/hetero_collection.hpp fájlreferencia . . . . .	142
7.50. hetero_collection.hpp . . . . .	143
7.51. include/utls/logger.hpp fájlreferencia . . . . .	143
7.51.1. Makródefiníciók dokumentációja . . . . .	143
7.51.1.1. print . . . . .	143
7.51.1.2. where . . . . .	144
7.52. logger.hpp . . . . .	144
7.53. include/utls/parsers.hpp fájlreferencia . . . . .	144
7.54. parsers.hpp . . . . .	145
7.55. include/utls/random_gen.hpp fájlreferencia . . . . .	146
7.56. random_gen.hpp . . . . .	147
7.57. include/utls/serializable.hpp fájlreferencia . . . . .	147
7.58. serializable.hpp . . . . .	147
7.59. include/utls/substitute_types.hpp fájlreferencia . . . . .	148
7.60. substitute_types.hpp . . . . .	148



Tárgymutató

151



# 1. fejezet

## Névtérmutató

### 1.1. Névtérlista

Az összes névtér listája rövid leírásokkal:

<code>pftd</code>	9
<code>pftd::gr</code>	10
<code>pftd::utils</code>	10
<code>pftd::utils::parser</code>	11



## 2. fejezet

# Hierarchikus mutató

### 2.1. Osztályhierarchia

Majdnem (de nem teljesen) betűrendbe szedett leszármazási lista:

pftd::App . . . . .	13
pftd::Object::Compare . . . . .	25
pftd::utils::Container< T, C > . . . . .	26
pftd::utils::Container< EndPoint > . . . . .	26
pftd::FollowPath . . . . .	37
pftd::utils::Container< InventoryItem > . . . . .	26
pftd::GameScene::Inventory . . . . .	45
sf::Drawable	
pftd::Object . . . . .	63
pftd::Clickable . . . . .	22
pftd::GameScene::InventoryItem . . . . .	47
pftd::gr::Button . . . . .	18
pftd::Entity . . . . .	30
pftd::Level::Nest . . . . .	61
pftd::Projectile . . . . .	70
pftd::Snowball . . . . .	97
pftd::Seal . . . . .	91
pftd::Cub . . . . .	28
pftd::FZC . . . . .	37
pftd::RegularSeal . . . . .	74
pftd::ZombieSeal . . . . .	118
pftd::Tower . . . . .	108
pftd::IcicleStabber . . . . .	43
pftd::Snowballer . . . . .	99
pftd::GameScene::Inventory . . . . .	45
pftd::Level . . . . .	52
pftd::gr::Label . . . . .	49
pftd::gr::Sprite . . . . .	101
pftd::utils::parser::SaveFileParser::EntityInfo . . . . .	35
pftd::utils::parser::Parser . . . . .	66
pftd::utils::parser::LevelConfigParser . . . . .	57
pftd::utils::parser::SaveFileParser . . . . .	83
pftd::utils::Random . . . . .	72
pftd::Renderer . . . . .	75

pftd::ResourceManager . . . . .	79
std::runtime_error	
pftd::LoadError . . . . .	??
pftd::SceneError . . . . .	90
pftd::utils::parser::ParseError . . . . .	??
pftd::Scene . . . . .	86
pftd::GameScene . . . . .	39
pftd::MenuScene . . . . .	59
pftd::utils::Serializable . . . . .	96
pftd::Level::Stats . . . . .	104
pftd::Projectile . . . . .	70
pftd::Seal . . . . .	91
pftd::Tower . . . . .	108
pftd::utils::parser::SaveFileParser::StatsInfo . . . . .	107
pftd::Tower::TowerProperties . . . . .	??
pftd::utils::Vec2< T > . . . . .	113
pftd::utils::Vec2< float > . . . . .	113

## 3. fejezet

# Adatszerkezet-mutató

### 3.1. Adatszerkezetek

Az összes adatszerkezet listája rövid leírásokkal:

pftd::App	13
pftd::gr::Button	18
pftd::Clickable	22
pftd::Object::Compare	25
pftd::utils::Container< T, C >	26
pftd::Cub	28
pftd::Entity	30
pftd::utils::parser::SaveFileParser::EntityInfo	35
pftd::FollowPath	37
pftd::FZC	37
pftd::GameScene	39
pftd::IcicleStabber	43
pftd::GameScene::Inventory	45
pftd::GameScene::InventoryItem	47
pftd::gr::Label	49
pftd::Level	52
pftd::utils::parser::LevelConfigParser	57
pftd::LoadError	??
pftd::MenuScene	59
pftd::Level::Nest	61
pftd::Object	63
pftd::utils::parser::ParseError	??
pftd::utils::parser::Parser	66
pftd::Projectile	70
pftd::utils::Random	72
pftd::RegularSeal	74
pftd::Renderer	75
pftd::ResourceManager	79
pftd::utils::parser::SaveFileParser	83
pftd::Scene	86
pftd::SceneError	90
pftd::Seal	91
pftd::utils::Serializable	96
pftd::Snowball	97
pftd::Snowballer	99

<a href="#">pftd::gr::Sprite</a>	101
<a href="#">pftd::Level::Stats</a>	104
<a href="#">pftd::utils::parser::SaveFileParser::StatsInfo</a>	107
<a href="#">pftd::Tower</a>	108
<a href="#">pftd::Tower::TowerProperties</a>	??
<a href="#">pftd::utils::Vec2&lt; T &gt;</a>	113
<a href="#">pftd::ZombieSeal</a>	118



## 4. fejezet

# Fájlmutató

### 4.1. Fájllista

Az összes fájl listája rövid leírásokkal:

include/all.hpp	121
include/app.hpp	122
include/resources.hpp	138
include/scene.hpp	139
include/game/level.hpp	123
include/objects/clickable.hpp	125
include/objects/object_base.hpp	137
include/objects/entities/all_entities.hpp	??
include/objects/entities/entity_base.hpp	126
include/objects/entities/projectiles/projectile_base.hpp	127
include/objects/entities/projectiles/snowball.hpp	127
include/objects/entities/seals/cub.hpp	128
include/objects/entities/seals/followpath.hpp	129
include/objects/entities/seals/fortified_zombie_cub.hpp	129
include/objects/entities/seals/regular.hpp	130
include/objects/entities/seals/seal_base.hpp	130
include/objects/entities/seals/zombie.hpp	131
include/objects/entities/towers/iclestabber.hpp	132
include/objects/entities/towers/snowballer.hpp	133
include/objects/entities/towers/tower_base.hpp	133
include/objects/gui/button.hpp	134
include/objects/gui/image.hpp	135
include/objects/gui/label.hpp	136
include/scenes/game.hpp	140
include/scenes/menu.hpp	142
include/utis/hetero_collection.hpp	142
include/utis/logger.hpp	143
include/utis/parsers.hpp	144
include/utis/random_gen.hpp	146
include/utis/serializable.hpp	147
include/utis/substitute_types.hpp	148



## 5. fejezet

# Névterek dokumentációja

### 5.1. pftd névtér-referencia

#### Névterek

- namespace [gr](#)
- namespace [utils](#)

#### Adatszerkezetek

- class [App](#)
- class [Clickable](#)
- class [Cub](#)
- class [Entity](#)
- struct [FollowPath](#)
- class [FZC](#)
- class [GameScene](#)
- class [IcicleStabber](#)
- class [Level](#)
- struct [LoadError](#)
- class [MenuScene](#)
- class [Object](#)
- class [Projectile](#)
- class [RegularSeal](#)
- class [Renderer](#)
- class [ResourceManager](#)
- class [Scene](#)
- struct [SceneError](#)
- class [Seal](#)
- class [Snowball](#)
- class [Snowballer](#)
- class [Tower](#)
- class [ZombieSeal](#)

#### Típusdefiníciók

- using [EndPoint](#) = [utils::Vec2f](#)

## Enumerációk

- enum class `ProjectileID` { `SNOWBALL` = 0 }
- enum class `SealID` { `REGULAR` = 0 , `CUB` , `ZOMBIE` , `FZC` }
- enum class `TowerID` { `SNOWBALLER` = 0 , `ICICLE_STABBER` }

### 5.1.1. Típusdefiníciók dokumentációja

#### 5.1.1.1. EndPoint

```
using pftd::EndPoint = typedef utils::Vec2f
```

### 5.1.2. Enumerációk dokumentációja

#### 5.1.2.1. ProjectileID

```
enum class pftd::ProjectileID [strong]
```

##### Enumeráció-értékek

SNOWBALL	
----------	--

#### 5.1.2.2. SealID

```
enum class pftd::SealID [strong]
```

##### Enumeráció-értékek

REGULAR	
CUB	
ZOMBIE	
FZC	

#### 5.1.2.3. TowerID

```
enum class pftd::TowerID [strong]
```

## Enumeráció-értékek

SNOWBALLER	
ICICLE_STABBER	

## 5.2. pftd::gr névtér-referencia

### Adatszerkezetek

- class [Button](#)
- class [Label](#)
- class [Sprite](#)

## 5.3. pftd::utils névtér-referencia

### Névterek

- namespace [parser](#)

### Adatszerkezetek

- class [Container](#)
- class [Random](#)
- class [Serializable](#)
- struct [Vec2](#)

### Típusdefiníciók

- using [Vec2i](#) = [Vec2](#)< int >
- using [Vec2f](#) = [Vec2](#)< float >

### 5.3.1. Típusdefiníciók dokumentációja

#### 5.3.1.1. Vec2f

```
using pftd::utils::Vec2f = typedef Vec2<float>
```

#### 5.3.1.2. Vec2i

```
using pftd::utils::Vec2i = typedef Vec2<int>
```

## 5.4. pftd::utils::parser névtér-referencia

### Adatszerkezetek

- class [LevelConfigParser](#)
- struct [ParseError](#)
- class [Parser](#)
- class [SaveFileParser](#)



## 6. fejezet

# Adatszerkezetek dokumentációja

### 6.1. pftd::App osztályreferencia

```
#include <app.hpp>
```

#### Publikus tagfüggvények

- `App (App const &)=delete`
- `App (App &&)=delete`
- `~App ()`  
*Felszabadítja a kezelt erőforrásokat.*
- `void run ()`  
*A fő programciklus.*
- `void addScene (std::string id, Scene *scene, bool active=false)`  
*Hozzáad egy új nézetet.*
- `bool changeScene (std::string id, Scene::StateFlag flag=Scene::StateFlag::NONE)`  
*Nézetet vált (aktívvá tesz egy másikat).*
- `auto isRunning () const`  
*m\_running getter.*
- `int getWindowWidth () const`
- `int getWindowHeight () const`
- `App & operator= (App const &)=delete`

#### Statikus publikus tagfüggvények

- `static App * getInstance ()`  
*A singleton instance megszerzése.*
- `static App * create (unsigned int width, unsigned int height, std::string const &windowTitle)`  
*Inicializálja a különböző programelemeket.*
- `static void destroy ()`  
*Törli a singleton instance-t.*

#### Privát tagfüggvények

- `App ()=default`

## Privát attribútumok

- bool `m_running` = false
- `Renderer * m_renderer` = nullptr
- `std::map< std::string, Scene * >` `m_scenes`
- `std::string` `m_activeSceneID`

## Statikus privát attribútumok

- static `App * m_instance`

### 6.1.1. Részletes leírás

Az egész játékért felelő osztály.

### 6.1.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.1.2.1. `App()` [1/3]

```
pftd::App::App (  
    App const & ) [delete]
```

#### 6.1.2.2. `App()` [2/3]

```
pftd::App::App (  
    App && ) [delete]
```

#### 6.1.2.3. `~App()`

```
pftd::App::~~App ( )
```

Felszabadítja a kezelt erőforrásokat.

Törli a renderer-t, az erőforrás-kezelőt és a nézeteket.

#### 6.1.2.4. `App()` [3/3]

```
pftd::App::App ( ) [private], [default]
```



### 6.1.3. Tagfüggvények dokumentációja

#### 6.1.3.1. addScene()

```
void pftd::App::addScene (
    std::string id,
    Scene * scene,
    bool active = false )
```

Hozzáad egy új nézetet.

Ha az `active` igaz, akkor meghívja a `changeScene`-t a megadott `id`-ra. Megjegyzés: Ha már létezik ilyen nézet a megadott azonosítóval, akkor `SceneError`-t dob.

##### Paraméterek

<i>id</i>	A nézet azonosítója.
<i>scene</i>	Maga a nézet osztály egy példánya (pl.: <a href="#">MenuScene</a> , <a href="#">GameScene</a> ).
<i>active</i>	Ez legyen e az aktív nézet.

#### 6.1.3.2. changeScene()

```
bool pftd::App::changeScene (
    std::string id,
    Scene::StateFlag flag = Scene::StateFlag::NONE )
```

Nézetet vált (aktívvá tesz egy másikat).

Ha megegyezik az eddigi és a kért nézet, akkor nem csinál semmit. Egyébként meg az eddigi nézetet deaktiválja, és az újat pedig aktiválja. Megjegyzés: Ha már létezik ilyen nézet a megadott azonosítóval, akkor `SceneError`-t dob.

##### Paraméterek

<i>id</i>	Az új nézet azonosítója.
<i>flag</i>	Egyéb "kérést" közlő opció flag.

##### Visszatérési érték

Sikerült e váltani.

#### 6.1.3.3. create()

```
static App * pftd::App::create (
    unsigned int width,
    unsigned int height,
    std::string const & windowTitle ) [static]
```

Inicializálja a különböző programelemeket.

Megjegyzés: Meg kell hívni a `destroy()`-t, ha már nincs szükség rá!

##### Paraméterek

<i>width</i>	Az ablak szélessége.
<i>height</i>	Az ablak magassága.
<i>windowTitle</i>	Az ablak címe.

##### Visszatérési érték

Az osztályhoz tartozó singleton instance.

#### 6.1.3.4. destroy()

```
static void pftd::App::destroy ( ) [inline], [static]
```

Törli a singleton instance-t.

#### 6.1.3.5. getInstance()

```
static App * pftd::App::getInstance ( ) [inline], [static]
```

A singleton instance megszerzése.

##### Visszatérési érték

Az osztályhoz tartozó singleton instance.

#### 6.1.3.6. getWindowHeight()

```
int pftd::App::getWindowHeight ( ) const [inline]
```

##### Visszatérési érték

A programhoz tartozó ablak magassága.

#### 6.1.3.7. getWidth()

```
int pftd::App::getWidth ( ) const [inline]
```

##### Visszatérési érték

A programhoz tartozó ablak szélessége.

#### 6.1.3.8. isRunning()

```
auto pftd::App::isRunning ( ) const [inline]
```

m\_running getter.

##### Visszatérési érték

Fut e a program.

#### 6.1.3.9. operator=()

```
App & pftd::App::operator= (
    App const & ) [delete]
```

#### 6.1.3.10. run()

```
void pftd::App::run ( )
```

A fő programciklus.

### 6.1.4. Adatmezők dokumentációja

#### 6.1.4.1. m\_activeSceneID

```
std::string pftd::App::m_activeSceneID [private]
```

Az aktív nézet azonosítója.

#### 6.1.4.2. m\_instance

```
App* pftd::App::m_instance [static], [private]
```

A singleton instance.

#### 6.1.4.3. m\_renderer

```
Renderer* pftd::App::m_renderer = nullptr [private]
```

A renderer.

#### 6.1.4.4. m\_running

```
bool pftd::App::m_running = false [private]
```

Fut e a program.

#### 6.1.4.5. m\_scenes

```
std::map<std::string, Scene*> pftd::App::m_scenes [private]
```

A nézetek: [id, nézet].

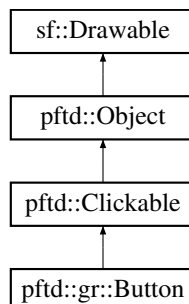
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/app.hpp

## 6.2. pftd::gr::Button osztályreferencia

```
#include <button.hpp>
```

A pftd::gr::Button osztály származási diagramja:



## Publikus tagfüggvények

- `Button` (`gr::Label` const &`label`, `utils::Vec2f` const &`position`, `utils::Vec2f` const &`size`, bool `active=true`, int `zIndex=0`)
- virtual `~Button` ()
- void `setSound` (std::string const &`src`)  
*Kattintás hangeffekt beállítása.*
- void `setBackground` (std::string const &`src`)  
*Háttérkép beállítása.*
- std::string `getLabel` () const  
*Ráírt szöveg.*
- virtual void `handleClick` (`utils::Vec2i` const &`clickCoords`) override  
*Kattintás kezelése.*
- virtual void `draw` (sf::RenderTarget &`target`, sf::RenderStates `states`) const override  
*Objektum megjelenítése.*

## Védett attribútumok

- `Label label`

## Privát attribútumok

- sf::FloatRect `m_rect`
- `Sprite * m_background = nullptr`
- sf::Sound `m_clickSound`

## További örökölt tagok

### 6.2.1. Részletes leírás

Gomb GUI elem.

### 6.2.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.2.2.1. Button()

```
pftd::gr::Button::Button (
    gr::Label const & label,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    bool active = true,
    int zIndex = 0 )
```

## Paraméterek

<i>label</i>	Szöveg objektum: maga a szöveg, betűtípus és betűméret.
<i>position</i>	Pozíció.
<i>size</i>	Méret.
<i>active</i>	Aktív e? Rá lehet e kattintani.
<i>zIndex</i>	Z koordináta.

## 6.2.2.2. ~Button()

```
virtual pftd::gr::Button::~~Button ( ) [virtual]
```

## 6.2.3. Tagfüggvények dokumentációja

## 6.2.3.1. draw()

```
virtual void pftd::gr::Button::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

## Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

## 6.2.3.2. getLabel()

```
std::string pftd::gr::Button::getLabel ( ) const [inline]
```

Ráírt szöveg.

## Visszatérési érték

A tárolt címkére írt szöveg.

### 6.2.3.3. handleClick()

```
virtual void pftd::gr::Button::handleClick (
    utils::Vec2i const & clickCoords ) [override], [virtual]
```

Kattintás kezelése.

Paraméterek

<i>clickCoords</i>	Kurzor koordinátái kattintáskor.
--------------------	----------------------------------

Újraimplementált ősök: [pftd::Clickable](#).

### 6.2.3.4. setBackground()

```
void pftd::gr::Button::setBackground (
    std::string const & src )
```

Háttérkép beállítása.

Paraméterek

<i>src</i>	Háttérkép elérési útvonala.
------------	-----------------------------

### 6.2.3.5. setSound()

```
void pftd::gr::Button::setSound (
    std::string const & src )
```

Kattintás hangeffekt beállítása.

Paraméterek

<i>src</i>	Hangfájl elérési útvonala.
------------	----------------------------

## 6.2.4. Adatmezők dokumentációja

### 6.2.4.1. label

`Label` pftd::gr::Button::label [protected]

Címke.

#### 6.2.4.2. m\_background

```
Sprite* pftd::gr::Button::m_background = nullptr [private]
```

Háttér.

#### 6.2.4.3. m\_clickSound

```
sf::Sound pftd::gr::Button::m_clickSound [private]
```

Kattintás hangeffekt.

#### 6.2.4.4. m\_rect

```
sf::FloatRect pftd::gr::Button::m_rect [private]
```

Minden gomb egy téglalap: ennek a pozíciója és mérete.

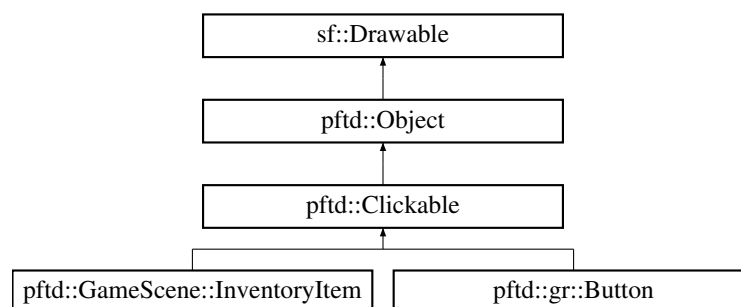
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/objects/gui/[button.hpp](#)

### 6.3. pftd::Clickable osztályreferencia

```
#include <clickable.hpp>
```

A pftd::Clickable osztály származási diagramja:



#### Publikus tagfüggvények

- [Clickable](#) (bool active=true)
- [Clickable](#) (utils::Vec2f const &position, utils::Vec2f const &size, int zIndex=0, bool active=true)
- virtual void [setCallback](#) (std::function< void()> callback)  
*Callback beállítása.*
- virtual void [handleClick](#) (utils::Vec2i const &clickCoords)  
*Kattintás kezelése.*



## Adatmezők

- bool `isActive`

## Védett attribútumok

- `std::function< void()>` `m_callback`

### 6.3.1. Részletes leírás

Kattintható (interaktálható) objektum.

### 6.3.2. Konstruktorkok és destruktorok dokumentációja

#### 6.3.2.1. Clickable() [1/2]

```
pftd::Clickable::Clickable (
    bool active = true )
```

#### 6.3.2.2. Clickable() [2/2]

```
pftd::Clickable::Clickable (
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0,
    bool active = true )
```

### 6.3.3. Tagfüggvények dokumentációja

#### 6.3.3.1. handleClick()

```
virtual void pftd::Clickable::handleClick (
    utils::Vec2i const & clickCoords ) [virtual]
```

Kattintás kezelése.

#### Paraméterek

<code>clickCoords</code>	Kurzor koordinátái kattintáskor.
--------------------------	----------------------------------

Újraimplementáló leszármazottak: [pftd::gr::Button](#).

### 6.3.3.2. setCallback()

```
virtual void pftd::Clickable::setCallback (
    std::function< void()> callback ) [virtual]
```

Callback beállítása.

#### Paraméterek

<i>callback</i>	A kattintásra lefuttatni kívánt függvény.
-----------------	---

## 6.3.4. Adatmezők dokumentációja

### 6.3.4.1. isActive

```
bool pftd::Clickable::isActive
```

Aktív e.

### 6.3.4.2. m\_callback

```
std::function<void()> pftd::Clickable::m_callback [protected]
```

Callback: akkor fut le, ha az objektumra rákattintunk.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/objects/[clickable.hpp](#)

## 6.4. pftd::Object::Compare struktúrareferencia

```
#include <object_base.hpp>
```

### Publikus tagfüggvények

- bool [operator\(\)](#) ([Object](#) const \*o1, [Object](#) const \*o2) const

### 6.4.1. Részletes leírás

Priority queue miatt szükséges segédosztály.

### 6.4.2. Tagfüggvények dokumentációja

#### 6.4.2.1. operator>()

```
bool pftd::Object::Compare::operator() (
    Object const * o1,
    Object const * o2 ) const [inline]
```

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/objects/object\_base.hpp

## 6.5. pftd::utils::Container< T, C > osztálysablon-referencia

```
#include <hetero_collection.hpp>
```

### Publikus tagfüggvények

- Container ()=default
- virtual ~Container ()
- T \*const append (T \*elem)  
*Új elem hozzáadása.*
- std::size\_t size () const  
*A tároló mérete.*
- C const & getContainer () const  
*Konstans m\_container getter.*

### Privát attribútumok

- C m\_container

#### 6.5.1. Részletes leírás

```
template<typename T, typename C = std::vector<T*>>
class pftd::utils::Container< T, C >
```

Olyan osztályok, amelyek tárolóként (is) használhatók. Fontos: nem egy tárolót tartalmaznak, hanem azok maguk a tárolók (öröklés, nem kompozíció)! Ilyen például: FollowPath, Inventory.

Megjegyzés: iterátorokat nem tartalmaz, de elérhető a getContainer().

## 6.5.2. Konstruktorkok és destruktorkok dokumentációja

### 6.5.2.1. Container()

```
template<typename T , typename C = std::vector<T*>>
pftd::utils::Container< T, C >::Container ( ) [default]
```

### 6.5.2.2. ~Container()

```
template<typename T , typename C = std::vector<T*>>
virtual pftd::utils::Container< T, C >::~~Container ( ) [inline], [virtual]
```

## 6.5.3. Tagfüggvények dokumentációja

### 6.5.3.1. append()

```
template<typename T , typename C = std::vector<T*>>
T *const pftd::utils::Container< T, C >::append (
    T * elem ) [inline]
```

Új elem hozzáadása.

#### Paraméterek

<i>elem</i>	A hozzáadni kívánt dolog.
-------------	---------------------------

#### Visszatérési érték

Az imént hozzáadott dolog.

### 6.5.3.2. getContainer()

```
template<typename T , typename C = std::vector<T*>>
C const & pftd::utils::Container< T, C >::getContainer ( ) const [inline]
```

Konstans `m_container` getter.

#### Visszatérési érték

A tároló.

### 6.5.3.3. size()

```
template<typename T , typename C = std::vector<T*>>
std::size_t pftd::utils::Container< T, C >::size ( ) const [inline]
```

A tároló mérete.

Visszatérési érték

A tároló elemeinek száma.

## 6.5.4. Adatmezők dokumentációja

### 6.5.4.1. m\_container

```
template<typename T , typename C = std::vector<T*>>
C pftd::utils::Container< T, C >::m_container [private]
```

A tároló.

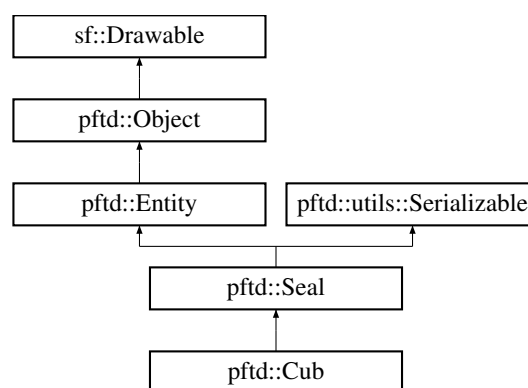
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/utills/hetero\\_collection.hpp](#)

## 6.6. pftd::Cub osztályreferencia

```
#include <cub.hpp>
```

A pftd::Cub osztály származási diagramja:



## Publikus tagfüggvények

- [Cub](#) ([FollowPath](#) const &followpath)  
A textúra hard code-olt, a kezdő pozíciót pedig a `followPath` határozza meg.
- [Seal](#) \* [clone](#) () const override  
Egy dinamikus memóriában foglalt másolatot készít az entitásról.

## További örökölt tagok

### 6.6.1. Részletes leírás

Kicsiny.

### 6.6.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.6.2.1. Cub()

```
pftd::Cub::Cub (
    FollowPath const & followpath )
```

A textúra hard code-olt, a kezdő pozíciót pedig a `followPath` határozza meg.

#### Paraméterek

<code>followpath</code>	Az útvonal amin végig kell menjen (majd pedig vissza).
-------------------------	--

### 6.6.3. Tagfüggvények dokumentációja

#### 6.6.3.1. clone()

```
Seal * pftd::Cub::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

#### Visszatérési érték

A másolat.

Megvalósítja a következőket: `pftd::Seal`.

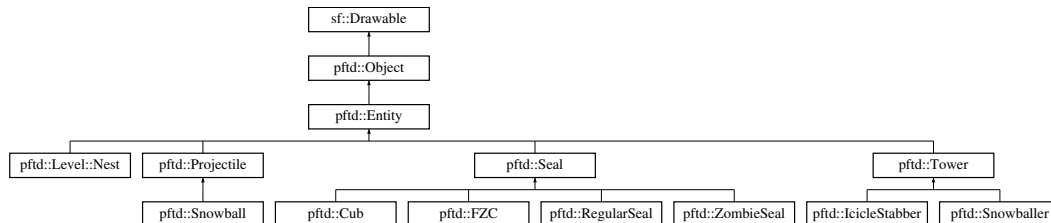
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- `include/objects/entities/seals/cub.hpp`

## 6.7. pftd::Entity osztályreferencia

```
#include <entity_base.hpp>
```

A pftd::Entity osztály származási diagramja:



### Publikus tagfüggvények

- **Entity** (std::string const &spriteSheetSrc, **utils::Vec2i** spriteSize, **utils::Vec2f** const &position, **utils::Vec2f** const &size, int zIndex=0)
- **Entity** (sf::Texture const &texture, **utils::Vec2i** spriteSize, **utils::Vec2f** const &position, **utils::Vec2f** const &size, int zIndex=0)
- **Entity** (std::string const &spriteSrc, **utils::Vec2f** const &position, **utils::Vec2f** const &size, int zIndex=0)
- **Entity** (**Entity** const &other)
- virtual **~Entity** ()
- virtual **Entity** \* **clone** () const =0

*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

- virtual void **update** (float dt)  
*Update.*
- virtual void **draw** (sf::RenderTarget &target, sf::RenderStates states) const override  
*Objektum megjelenítése.*
- void **setPosition** (**utils::Vec2f** position)  
*Pozíció beállítása.*
- auto **getPosition** () const  
*position getter.*
- **gr::Sprite** const \* **getSprite** () const  
*Konstans currentSprite getter.*
- **gr::Sprite** \* **getSprite** ()  
*currentSprite getter.*
- auto & **getSpriteSheet** () const  
*spriteSheet getter.*

### Adatmezők

- bool **isAnimated** = true

### Védett tagfüggvények

- virtual void **advanceAnimationFrame** ()  
*Animáció: képkocka léptetése.*
- virtual void **resetAnimation** ()  
*Visszaállítja a jelenlegi képkockát az elsőre, és nullázza az időt.*

## Védett attribútumok

- sf::Texture const & [spriteSheet](#)
- [utils::Vec2i](#) [cellSize](#)
- size\_t const [CELL\\_N](#)
- size\_t [currentCell](#) = 0U
- [gr::Sprite](#) \* [currentSprite](#) = nullptr
- float [frameDurationSec](#) = 1.0f
- float [totalElapsedSec](#) = 0.0f

### 6.7.1. Részletes leírás

Mindenféle "actor" őssztálya: tornyok, fólák és lövedékek.

### 6.7.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.7.2.1. Entity() [1/4]

```
pftd::Entity::Entity (
    std::string const & spriteSheetSrc,
    utils::Vec2i spriteSize,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0 )
```

#### 6.7.2.2. Entity() [2/4]

```
pftd::Entity::Entity (
    sf::Texture const & texture,
    utils::Vec2i spriteSize,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0 )
```

#### 6.7.2.3. Entity() [3/4]

```
pftd::Entity::Entity (
    std::string const & spriteSrc,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0 )
```



#### 6.7.2.4. Entity() [4/4]

```
pftd::Entity::Entity (
    Entity const & other )
```

#### 6.7.2.5. ~Entity()

```
virtual pftd::Entity::~~Entity ( ) [virtual]
```

### 6.7.3. Tagfüggvények dokumentációja

#### 6.7.3.1. advanceAnimationFrame()

```
virtual void pftd::Entity::advanceAnimationFrame ( ) [protected], [virtual]
```

Animáció: képkocka léptetése.

Újraimplementáló leszármazottak: [pftd::Seal](#) és [pftd::Tower](#).

#### 6.7.3.2. clone()

```
virtual Entity * pftd::Entity::clone ( ) const [pure virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

Visszatérési érték

A másolat.

Megvalósítják a következők: [pftd::Level::Nest](#), [pftd::Snowball](#), [pftd::Cub](#), [pftd::FZC](#), [pftd::RegularSeal](#), [pftd::ZombieSeal](#), [pftd::IcicleStabber](#), [pftd::Snowballer](#), [pftd::Seal](#) és [pftd::Tower](#).

#### 6.7.3.3. draw()

```
virtual void pftd::Entity::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

## Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

**6.7.3.4. getPosition()**

```
auto pftd::Entity::getPosition ( ) const [inline]
```

`position` getter.

**Visszatérési érték**

Az entitás jelenlegi pozíciója.

**6.7.3.5. getSprite() [1/2]**

```
gr::Sprite * pftd::Entity::getSprite ( ) [inline]
```

`currentSprite` getter.

**Visszatérési érték**

A jelenlegi sprite.

**6.7.3.6. getSprite() [2/2]**

```
gr::Sprite const * pftd::Entity::getSprite ( ) const [inline]
```

Konstans `currentSprite` getter.

**Visszatérési érték**

A jelenlegi sprite.

#### 6.7.3.7. getSpriteSheet()

```
auto & pftd::Entity::getSpriteSheet ( ) const [inline]
```

spriteSheet getter.

##### Visszatérési érték

A sprite sheet.

#### 6.7.3.8. resetAnimation()

```
virtual void pftd::Entity::resetAnimation ( ) [protected], [virtual]
```

Visszaállítja a jelenlegi képkockát az elsőre, és nullázza az időt.

#### 6.7.3.9. setPosition()

```
void pftd::Entity::setPosition (
    utils::Vec2f position )
```

Pozíció beállítása.

##### Paraméterek

<i>position</i>	Az új pozíció.	
-----------------	----------------	--

#### 6.7.3.10. update()

```
virtual void pftd::Entity::update (
    float dt ) [virtual]
```

Update.

##### Paraméterek

<i>dt</i>	Delta idő.	
-----------	------------	--

Újraimplementáló leszármazottak: [pftd::Projectile](#), [pftd::Seal](#), [pftd::IcicleStabber](#), [pftd::Snowballer](#) és [pftd::Tower](#).

## 6.7.4. Adatmezők dokumentációja

### 6.7.4.1. CELL\_N

```
size_t const pftd::Entity::CELL_N [protected]
```

Képkockák száma.

### 6.7.4.2. cellSize

```
utils::Vec2i pftd::Entity::cellSize [protected]
```

Képkockák mérete (fix).

### 6.7.4.3. currentCell

```
size_t pftd::Entity::currentCell = 0U [protected]
```

A sprite sheet aktív képkockája.

### 6.7.4.4. currentSprite

```
gr::Sprite* pftd::Entity::currentSprite = nullptr [protected]
```

A jelenlegi képkocka által meghatározott sprite.

### 6.7.4.5. frameDurationSec

```
float pftd::Entity::frameDurationSec = 1.0f [protected]
```

Ennyi ideig "tart" egy képkocka, ezután váltani kell.

### 6.7.4.6. isAnimated

```
bool pftd::Entity::isAnimated = true
```

Animált e?

### 6.7.4.7. spriteSheet

```
sf::Texture const& pftd::Entity::spriteSheet [protected]
```

A sprite sheet.

#### 6.7.4.8. totalElapsedSec

```
float pftd::Entity::totalElapsedSec = 0.0f [protected]
```

Eddig eltelt idő.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/entity\\_base.hpp](#)

## 6.8. pftd::utils::parser::SaveFileParser::EntityInfo struktúrareferencia

```
#include <parsers.hpp>
```

### Adatmezők

- [EntityType](#) `entityType`
- [utils::Vec2f](#) `position`
- struct {
  - unsigned int [sealID](#)
  - float [lerpParam](#)
  - bool [goingBackwards](#)
  - unsigned int [hp](#) } [seal](#)
- struct {
  - unsigned int [projID](#)
  - [utils::Vec2f](#) `direction`
  - float [speed](#) } [proj](#)
- unsigned int [towerID](#)

#### 6.8.1. Részletes leírás

Mentett játék betöltéséhez használt entitás infó.

#### 6.8.2. Adatmezők dokumentációja

##### 6.8.2.1. direction

```
utils::Vec2f pftd::utils::parser::SaveFileParser::EntityInfo::direction
```

#### 6.8.2.2. entityType

```
EntityType pftd::utils::parser::SaveFileParser::EntityInfo::entityType
```

Típusa.

#### 6.8.2.3. goingBackwards

```
bool pftd::utils::parser::SaveFileParser::EntityInfo::goingBackwards
```

#### 6.8.2.4. hp

```
unsigned int pftd::utils::parser::SaveFileParser::EntityInfo::hp
```

#### 6.8.2.5. lerpParam

```
float pftd::utils::parser::SaveFileParser::EntityInfo::lerpParam
```

#### 6.8.2.6. position

```
utils::Vec2f pftd::utils::parser::SaveFileParser::EntityInfo::position
```

Pozíciója.

#### 6.8.2.7.

```
struct { ... } pftd::utils::parser::SaveFileParser::EntityInfo::proj
```

Lövedék típusra valló adatok.

#### 6.8.2.8. projID

```
unsigned int pftd::utils::parser::SaveFileParser::EntityInfo::projID
```

**6.8.2.9.**

```
struct { ... } pftd::utils::parser::SaveFileParser::EntityInfo::seal
```

Típustól függő egyéb adat.

Ellenfél típusra valló adatok.

**6.8.2.10. sealID**

```
unsigned int pftd::utils::parser::SaveFileParser::EntityInfo::sealID
```

**6.8.2.11. speed**

```
float pftd::utils::parser::SaveFileParser::EntityInfo::speed
```

**6.8.2.12. towerID**

```
unsigned int pftd::utils::parser::SaveFileParser::EntityInfo::towerID
```

Torony típusra valló adatok.

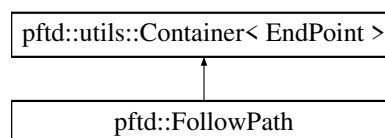
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/utils/parsers.hpp](#)

## 6.9. pftd::FollowPath struktúrareferencia

```
#include <followpath.hpp>
```

A pftd::FollowPath osztály származási diagramja:



### Publikus tagfüggvények

- [FollowPath\(\)](#)=default  
A *Level* - amihez tartozik - "építi fel."

### 6.9.1. Részletes leírás

Az ellenfelek által követett út.

## 6.9.2. Konstruktorkok és destruktorkok dokumentációja

### 6.9.2.1. FollowPath()

```
pftd::FollowPath::FollowPath ( ) [default]
```

A [Level](#) - amihez tartozik - "építi fel."

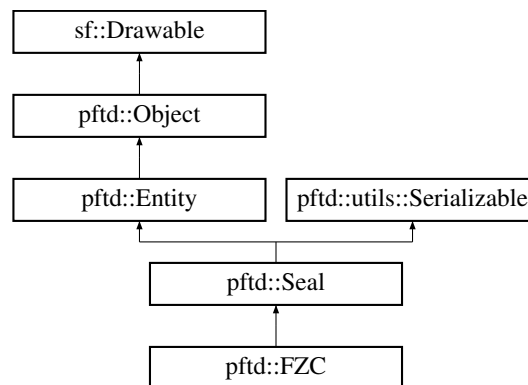
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/objects/entities/seals/followpath.hpp](#)

## 6.10. pftd::FZC osztályreferencia

```
#include <fortified_zombie_cub.hpp>
```

A pftd::FZC osztály származási diagramja:



### Publikus tagfüggvények

- [FZC](#) ([FollowPath](#) const &followpath)
- [Seal](#) \* [clone](#) () const override

*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

### További örökölt tagok

### 6.10.1. Részletes leírás

Páncélos zombi kicsiny.



## 6.10.2. Konstruktorok és destruktorok dokumentációja

### 6.10.2.1. FZC()

```
pftd::FZC::FZC (
    FollowPath const & followpath )
```

#### Paraméterek

<i>followpath</i>	Az útvonal amin végig kell menjen (majd pedig vissza).
-------------------	--

## 6.10.3. Tagfüggvények dokumentációja

### 6.10.3.1. clone()

```
Seal * pftd::FZC::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

#### Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Seal](#).

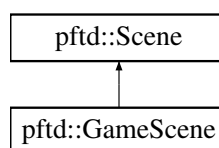
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/seals/fortified\\_zombie\\_cub.hpp](#)

## 6.11. pftd::GameScene osztályreferencia

```
#include <game.hpp>
```

A pftd::GameScene osztály származási diagramja:



## Adatszerkezetek

- struct [Inventory](#)
- struct [InventoryItem](#)

## Publikus tagfüggvények

- [GameScene](#) ()
- [~GameScene](#) ()
- void [onEvent](#) (sf::Event const &event) override  
*Delegált események kezelése.*
- void [update](#) (float dt) override  
*Update.*
- void [toggleActive](#) ([Scene::StateFlag](#) flag=[Scene::StateFlag::NONE](#)) override  
*Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.*
- void [startGame](#) ()  
*Játék elindítása.*
- void [updateScore](#) ()  
*Pontszám felirat frissítése.*
- void [updateWealth](#) ()  
*Pénz felirat frissítése.*

## Statikus publikus attribútumok

- static constexpr char const \* [SAVE\\_FILE\\_PATH](#) = "res/data/save.dat"

## Privát tagfüggvények

- void [\\_constructInventory](#) ()  
*Hozzáadja az `m_inventory`-hoz az `InventoryItem`-eket.*

## Privát attribútumok

- bool [m\\_shouldLoadSaved](#) = false
- [gr::Button](#) \* [m\\_saveButt](#) = nullptr
- [gr::Label](#) \* [m\\_moneyCounter](#) = nullptr
- [gr::Label](#) \* [m\\_scoreCounter](#) = nullptr
- sf::Sound [m\\_hornSound](#) = sf::Sound([ResourceManager::getInstance](#)()->getSound("res/audio/ready\_for\_↵ battle.mp3"))
- sf::Sound [m\\_gameoverSound](#) = sf::Sound([ResourceManager::getInstance](#)()->getSound("res/audio/gameover.↵ mp3"))
- [Inventory](#) \* [m\\_inventory](#) = nullptr
- [Level](#) \* [m\\_level](#) = nullptr

## További örökölt tagok

### 6.11.1. Részletes leírás

Játék nézet.

## 6.11.2. Konstruktorkok és destruktorkok dokumentációja

### 6.11.2.1. GameScene()

```
pftd::GameScene::GameScene ( )
```

### 6.11.2.2. ~GameScene()

```
pftd::GameScene::~~GameScene ( )
```

## 6.11.3. Tagfüggvények dokumentációja

### 6.11.3.1. \_constructInventory()

```
void pftd::GameScene::_constructInventory ( ) [private]
```

Hozzáadja az m\_inventory-hoz az [InventoryItem](#)-eket.

### 6.11.3.2. onEvent()

```
void pftd::GameScene::onEvent (
    sf::Event const & event ) [override], [virtual]
```

Delegált események kezelése.

#### Paraméterek

<i>event</i>	Kezelni kívánt esemény.
--------------	-------------------------

Megvalósítja a következőket: [pftd::Scene](#).

### 6.11.3.3. startGame()

```
void pftd::GameScene::startGame ( )
```

Játék elindítása.

Inicializálja a játék kezdetét: létrehozza a "szintet."

#### 6.11.3.4. toggleActive()

```
void pftd::GameScene::toggleActive (
    Scene::StateFlag flag = Scene::StateFlag::NONE ) [override], [virtual]
```

Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.

##### Paraméterek

<i>flag</i>	Egyéb "kérést" közlő opció flag.
-------------	----------------------------------

Újraimplementált ősök: [pftd::Scene](#).

#### 6.11.3.5. update()

```
void pftd::GameScene::update (
    float dt ) [override], [virtual]
```

Update.

##### Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Megvalósítja a következőket: [pftd::Scene](#).

#### 6.11.3.6. updateScore()

```
void pftd::GameScene::updateScore ( )
```

Pontszám felirat frissítése.

#### 6.11.3.7. updateWealth()

```
void pftd::GameScene::updateWealth ( )
```

Pénz felirat frissítése.

### 6.11.4. Adatmezők dokumentációja

#### 6.11.4.1. m\_gameoverSound

```
sf::Sound pftd::GameScene::m_gameoverSound = sf::Sound{ResourceManager::getInstance()->get<↵  
Sound("res/audio/gameover.mp3")} [private]
```

Játék vége hangeffekt.

#### 6.11.4.2. m\_hornSound

```
sf::Sound pftd::GameScene::m_hornSound = sf::Sound{ResourceManager::getInstance()->getSound("res/audio/ready↵  
_for_battle.mp3")} [private]
```

Kürt hang. Ez a játék elindításánál játszódik be.

#### 6.11.4.3. m\_inventory

```
Inventory* pftd::GameScene::m_inventory = nullptr [private]
```

Az "eszköztár."

#### 6.11.4.4. m\_level

```
Level* pftd::GameScene::m_level = nullptr [private]
```

Játéklogikát tartalmazó "szint."

#### 6.11.4.5. m\_moneyCounter

```
gr::Label* pftd::GameScene::m_moneyCounter = nullptr [private]
```

Pánzszámláló felirat.

#### 6.11.4.6. m\_saveButt

```
gr::Button* pftd::GameScene::m_saveButt = nullptr [private]
```

Mentés gomb.

#### 6.11.4.7. m\_scoreCounter

```
gr::Label* pftd::GameScene::m_scoreCounter = nullptr [private]
```

Pontszámláló felirat.

#### 6.11.4.8. m\_shouldLoadSaved

```
bool pftd::GameScene::m_shouldLoadSaved = false [private]
```

Be kell e tölteni a mentett játékállást.

#### 6.11.4.9. SAVE\_FILE\_PATH

```
constexpr char const* pftd::GameScene::SAVE_FILE_PATH = "res/data/save.dat" [static], [constexpr]
```

Mentett játékállás fájlja.

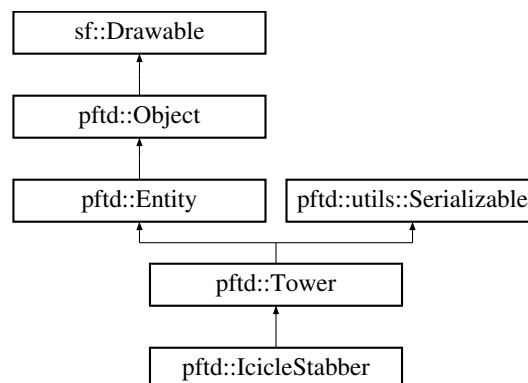
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/scenes/game.hpp](#)

### 6.12. pftd::IcicleStabber osztályreferencia

```
#include <iciclestabber.hpp>
```

A pftd::IcicleStabber osztály származási diagramja:



#### Publikus tagfüggvények

- [IcicleStabber](#) ([utils::Vec2f](#) const &[position](#), int [zIndex](#)=0)
- [IcicleStabber](#) ()
- [~IcicleStabber](#) ()=default
- [Tower](#) \* [clone](#) () const override  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*
- void [update](#) (float dt) override  
*Update.*

#### További örökölt tagok

##### 6.12.1. Részletes leírás

Jégcsap kardos.

## 6.12.2. Konstruktorkok és destruktorkok dokumentációja

### 6.12.2.1. IcicleStabber() [1/2]

```
pftd::IcicleStabber::IcicleStabber (
    utils::Vec2f const & position,
    int zIndex = 0 )
```

### 6.12.2.2. IcicleStabber() [2/2]

```
pftd::IcicleStabber::IcicleStabber ( )
```

### 6.12.2.3. ~IcicleStabber()

```
pftd::IcicleStabber::~~IcicleStabber ( ) [default]
```

## 6.12.3. Tagfüggvények dokumentációja

### 6.12.3.1. clone()

```
Tower * pftd::IcicleStabber::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Tower](#).

### 6.12.3.2. update()

```
void pftd::IcicleStabber::update (
    float dt ) [override], [virtual]
```

Update.

## Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Újrimplementált ősök: [pftd::Tower](#).

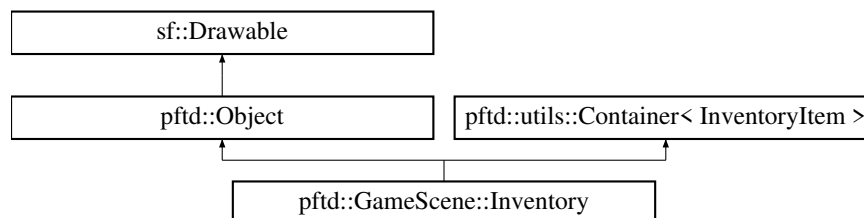
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- `include/objects/entities/towers/iclestabber.hpp`

## 6.13. pftd::GameScene::Inventory struktúrareferencia

```
#include <game.hpp>
```

A `pftd::GameScene::Inventory` osztály származási diagramja:



### Publikus tagfüggvények

- [Inventory](#) (`std::string const &backgroundImageSrc`)
- [~Inventory](#) ()=default
- void [draw](#) (`sf::RenderTarget &target, sf::RenderStates states`) const override  
*Objektum megjelenítése.*

### Adatmezők

- [gr::Sprite background](#)

### További örökölt tagok

#### 6.13.1. Részletes leírás

"Eszköztár": innen lehet megvenni a tornyokat.

#### 6.13.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.13.2.1. Inventory()

```
pftd::GameScene::Inventory::Inventory (
    std::string const & backgroundImageSrc )
```



## Paraméterek

<i>backgroundImageSrc</i>	Háttérkép elérési útvonala.
---------------------------	-----------------------------

**6.13.2.2. ~Inventory()**

```
pftd::GameScene::Inventory::~~Inventory ( ) [default]
```

**6.13.3. Tagfüggvények dokumentációja****6.13.3.1. draw()**

```
void pftd::GameScene::Inventory::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

## Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

**6.13.4. Adatmezők dokumentációja****6.13.4.1. background**

```
gr::Sprite pftd::GameScene::Inventory::background
```

Háttér.

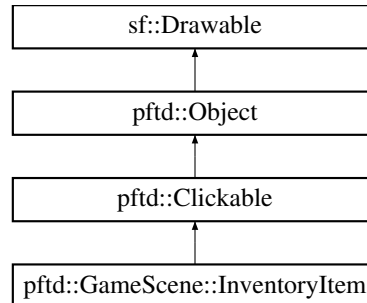
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/scenes/game.hpp](#)

## 6.14. pftd::GameScene::InventoryItem struktúrareferencia

```
#include <game.hpp>
```

A pftd::GameScene::InventoryItem osztály származási diagramja:



### Publikus tagfüggvények

- `InventoryItem (Tower *tower, Level *const level, utils::Vec2f const &position, utils::Vec2f const &size)`
- `~InventoryItem ()`
- `virtual void draw (sf::RenderTarget &target, sf::RenderStates states) const override`  
*Objektum megjelenítése.*

### Adatmezők

- `gr::Sprite frame`
- `gr::Sprite icon`
- `Tower * towerToSpawn = nullptr`
- `gr::Label priceLabel`

### További örökölt tagok

#### 6.14.1. Részletes leírás

Az "eszköztárban" tárolt torony.

#### 6.14.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.14.2.1. InventoryItem()

```

pftd::GameScene::InventoryItem::InventoryItem (
    Tower * tower,
    Level *const level,
    utils::Vec2f const & position,
    utils::Vec2f const & size )
  
```

### 6.14.2.2. ~InventoryItem()

```
pftd::GameScene::InventoryItem::~~InventoryItem ( )
```

## 6.14.3. Tagfüggvények dokumentációja

### 6.14.3.1. draw()

```
virtual void pftd::GameScene::InventoryItem::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

#### Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

## 6.14.4. Adatmezők dokumentációja

### 6.14.4.1. frame

```
gr::Sprite pftd::GameScene::InventoryItem::frame
```

Keret.

### 6.14.4.2. icon

```
gr::Sprite pftd::GameScene::InventoryItem::icon
```

Ikon.

### 6.14.4.3. priceLabel

```
gr::Label pftd::GameScene::InventoryItem::priceLabel
```

Árcímke (felirat).

#### 6.14.4.4. towerToSpawn

```
Tower* pftd::GameScene::InventoryItem::towerToSpawn = nullptr
```

Torony amit meg szeretnénk venni.

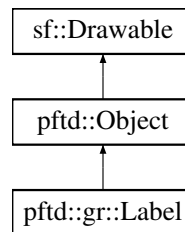
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/scenes/game.hpp

### 6.15. pftd::gr::Label osztályreferencia

```
#include <label.hpp>
```

A pftd::gr::Label osztály származási diagramja:



#### Publikus tagfüggvények

- [Label](#) (std::wstring const &label, sf::Font const &font, unsigned int cSize, sf::Color const &color=sf::Color::White)
- [Label](#) (std::wstring const &label, sf::Font const &font, unsigned int cSize, [utils::Vec2f](#) position, int zIndex=0, sf::Color const &color=sf::Color::White)
- void [setLabel](#) (std::string const &newLabel)  
*Új szöveg beállítása.*
- void [setOutline](#) (sf::Color color, float thickness=1.0f)  
*Szöveg outline beállítása.*
- sf::Text & [getText](#) ()  
*m\_text getter.*
- sf::Text const & [getText](#) () const  
*Konstans m\_text getter.*
- void [draw](#) (sf::RenderTarget &target, sf::RenderStates states) const override  
*Objektum megjelenítése.*

#### Privát attribútumok

- sf::Text [m\\_text](#)

## További örökölt tagok

### 6.15.1. Részletes leírás

Felirat GUI elem.

### 6.15.2. Konstruktorok és destruktorok dokumentációja

#### 6.15.2.1. Label() [1/2]

```
pftd::gr::Label::Label (
    std::wstring const & label,
    sf::Font const & font,
    unsigned int cSize,
    sf::Color const & color = sf::Color::White )
```

##### Paraméterek

<i>label</i>	Szöveg.
<i>font</i>	Betűtípus.
<i>cSize</i>	Betűméret.
<i>color</i>	Betűszín.

#### 6.15.2.2. Label() [2/2]

```
pftd::gr::Label::Label (
    std::wstring const & label,
    sf::Font const & font,
    unsigned int cSize,
    utils::Vec2f position,
    int zIndex = 0,
    sf::Color const & color = sf::Color::White )
```

##### Paraméterek

<i>label</i>	Szöveg.
<i>font</i>	Betűtípus.
<i>cSize</i>	Betűméret.
<i>position</i>	Pozíció.
<i>zIndex</i>	Z koordináta.
<i>color</i>	Betűszín.

### 6.15.3. Tagfüggvények dokumentációja

#### 6.15.3.1. draw()

```
void pftd::gr::Label::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [inline], [override], [virtual]
```

Objektum megjelenítése.

##### Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

#### 6.15.3.2. getText() [1/2]

```
sf::Text & pftd::gr::Label::getText ( ) [inline]
```

m\_text getter.

##### Visszatérési érték

A tárolt szöveg objektum.

#### 6.15.3.3. getText() [2/2]

```
sf::Text const & pftd::gr::Label::getText ( ) const [inline]
```

Konstans m\_text getter.

##### Visszatérési érték

A tárolt szöveg objektum.

#### 6.15.3.4. setLabel()

```
void pftd::gr::Label::setLabel (
    std::string const & newLabel ) [inline]
```

Új szöveg beállítása.

## Paraméterek

<i>newLabel</i>	Az új szöveg.
-----------------	---------------

## 6.15.3.5. setOutline()

```
void pftd::gr::Label::setOutline (
    sf::Color color,
    float thickness = 1.0f ) [inline]
```

Szöveg outline beállítása.

## Paraméterek

<i>color</i>	Színe.
<i>thickness</i>	Vastagsága.

## 6.15.4. Adatmezők dokumentációja

## 6.15.4.1. m\_text

```
sf::Text pftd::gr::Label::m_text [private]
```

Tárolt szöveg objektum.

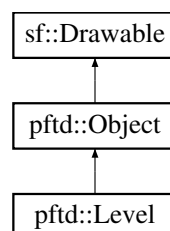
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/objects/gui/[label.hpp](#)

## 6.16. pftd::Level osztályreferencia

```
#include <level.hpp>
```

A pftd::Level osztály származási diagramja:



## Adatszerkezetek

- struct [Nest](#)
- struct [Stats](#)

## Publikus tagfüggvények

- [Level](#) (std::string const &[saveFile](#), [Stats stats](#))
- [Level](#) (std::string const &[saveFile](#))  
*Ilyenkor a játékállást is a mentésfájlból olvassa vissza.*
- virtual [~Level](#) ()
- virtual void [loseHP](#) (int hpLost=1)  
*HP vesztes.*
- virtual bool [placeTower](#) ()  
*Kiválasztott torony lehelyezése.*
- void [deselectTower](#) ()  
*Kiválasztott torony törlése.*
- void [selectTower](#) ([Tower](#) \*newTower)  
*Torony kiválasztása.*
- void [update](#) (float dt)  
*Update.*
- bool [isGameOver](#) () const  
*Vége van e játéknak?*
- void [save](#) () const  
*Játékállás mentése.*
- void [reset](#) ([Stats stats](#))  
*Játék újraindítása.*
- virtual void [draw](#) (sf::RenderTarget &target, sf::RenderStates states) const override  
*Objektum megjelenítése.*

## Adatmezők

- [Stats stats](#) = [Stats](#){}
- [Tower](#) \* [selectedTower](#) = nullptr

## Védett tagfüggvények

- void [spawnSeal](#) ()  
*Ellenfél spawnolás.*

## Védett attribútumok

- [Nest](#) \* [nest](#) = nullptr
- [FollowPath](#) [followPath](#)
- std::vector< [Tower](#) \* > [towers](#)
- std::vector< [Seal](#) \* > [seals](#)
- std::vector< [Projectile](#) \* > [projectiles](#)
- [utils::parser::LevelConfigParser](#) [config](#)
- std::string [saveFile](#)



## Privát tagfüggvények

- void `_spawnSeal` (float dt)  
*spawnSeal-t hívja meg, ha eltelt megfelelő mennyiségű idő.*
- void `_updateSeals` (float dt)  
*Kezeli az ellenfeleket.*
- void `_updateTowers` (float dt)  
*Kezeli a tornyokat.*
- void `_updateProjectiles` (float dt)  
*Kezeli a lövedékeket.*

## Privát attribútumok

- float `m_accuTimeSpawnSec` = 0.0f

### 6.16.1. Részletes leírás

Játéklogikát és entitásokat tartalmazó osztály.

### 6.16.2. Konstruktorok és destruktorok dokumentációja

#### 6.16.2.1. Level() [1/2]

```
pftd::Level::Level (
    std::string const & saveFile,
    Stats stats )
```

##### Paraméterek

<code>saveFile</code>	Mentésfájl elérési útvonala.
<code>stats</code>	Kezdeti játékállás.

#### 6.16.2.2. Level() [2/2]

```
pftd::Level::Level (
    std::string const & saveFile )
```

Ilyenkor a játékállást is a mentésfájlból olvassa vissza.

##### Paraméterek

<code>saveFile</code>	Mentésfájl elérési útvonala.
-----------------------	------------------------------

### 6.16.2.3. ~Level()

```
virtual pftd::Level::~~Level ( ) [virtual]
```

## 6.16.3. Tagfüggvények dokumentációja

### 6.16.3.1. \_spawnSeal()

```
void pftd::Level::_spawnSeal (
    float dt ) [private]
```

spawnSeal-t hívja meg, ha eltelt megfelelő mennyiségű idő.

### 6.16.3.2. \_updateProjectiles()

```
void pftd::Level::_updateProjectiles (
    float dt ) [private]
```

Kezeli a lövedékeket.

### 6.16.3.3. \_updateSeals()

```
void pftd::Level::_updateSeals (
    float dt ) [private]
```

Kezeli az ellenfeleket.

### 6.16.3.4. \_updateTowers()

```
void pftd::Level::_updateTowers (
    float dt ) [private]
```

Kezeli a tornyokat.

**6.16.3.5. deselectTower()**

```
void pftd::Level::deselectTower ( )
```

Kiválasztott torony törlése.

**6.16.3.6. draw()**

```
virtual void pftd::Level::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

## Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

**6.16.3.7. isGameOver()**

```
bool pftd::Level::isGameOver ( ) const [inline]
```

Vége van e játéknak?

## Visszatérési érték

Vesztett e a játékos?

**6.16.3.8. loseHP()**

```
virtual void pftd::Level::loseHP (
    int hpLost = 1 ) [virtual]
```

HP vesztes.

## Paraméterek

<i>hpLost</i>	Ennyi HP-t veszít a játékos.
---------------	------------------------------

**6.16.3.9. placeTower()**

```
virtual bool pftd::Level::placeTower ( ) [virtual]
```

Kiválasztott torny lehelyezése.

## Visszatérési érték

Sikerült e lehelyezni?

**6.16.3.10. reset()**

```
void pftd::Level::reset (
    Stats stats )
```

Játék újraindítása.

"Object Pool" nem használata miatt van.

**6.16.3.11. save()**

```
void pftd::Level::save ( ) const
```

Játékállás mentése.

`LoadError`-t dobhat, ha nem sikerül megnyitnia a fájlt / nincs engedélye hozzá.

**6.16.3.12. selectTower()**

```
void pftd::Level::selectTower (
    Tower * newTower )
```

Torony kiválasztása.

Ezt a játék nézet "eszköztárja" adja át.

**Paraméterek**

<code>newTower</code>	Lehelyezni kívánt torony.
-----------------------	---------------------------

**6.16.3.13. spawnSeal()**

```
void pftd::Level::spawnSeal ( ) [protected]
```

Ellenfél spawnolás.

**6.16.3.14. update()**

```
void pftd::Level::update (
    float dt )
```

Update.

## Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

## 6.16.4. Adatmezők dokumentációja

### 6.16.4.1. config

```
utils::parser::LevelConfigParser pftd::Level::config [protected]
```

Inicializáláshoz szükséges konfiguráció.

### 6.16.4.2. followPath

```
FollowPath pftd::Level::followPath [protected]
```

Ellenségek által követett útvonal.

### 6.16.4.3. m\_accuTimeSpawnSec

```
float pftd::Level::m_accuTimeSpawnSec = 0.0f [private]
```

Előző spawn óta eltelt idő.

### 6.16.4.4. nest

```
Nest* pftd::Level::nest = nullptr [protected]
```

A fészek (cél).

### 6.16.4.5. projectiles

```
std::vector<Projectile*> pftd::Level::projectiles [protected]
```

Lövedékek.

### 6.16.4.6. saveFile

```
std::string pftd::Level::saveFile [protected]
```

Mentésfájl.

**6.16.4.7. seals**

```
std::vector<Seal*> pftd::Level::seals [protected]
```

Élő ellenfelek.

**6.16.4.8. selectedTower**

```
Tower* pftd::Level::selectedTower = nullptr
```

Lehelyezni kívánt torony.

**6.16.4.9. stats**

```
Stats pftd::Level::stats = Stats{}
```

Játékos statisztika.

**6.16.4.10. towers**

```
std::vector<Tower*> pftd::Level::towers [protected]
```

Lehelyezett tornyok.

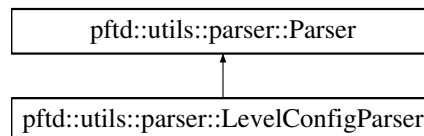
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/game/level.hpp

**6.17. pftd::utils::parser::LevelConfigParser osztályreferencia**

```
#include <parsers.hpp>
```

A pftd::utils::parser::LevelConfigParser osztály származási diagramja:

**Publikus tagfüggvények**

- [LevelConfigParser](#) (std::string const &sourceFile)
- [~LevelConfigParser](#) ()=default
- void [parse](#) () override  
A parse fűciklus.
- std::vector< [Vec2f](#) > [getAttribute](#) (std::string name) const  
Mentett attribútum értékének kiolvasása.

## Privát tagfüggvények

- `std::string _getAttribute ()`  
*Következő attribútum beolvasása a konfigurálból.*

## Privát attribútumok

- `std::unordered_map< std::string, std::vector< Vec2f > > m_attribs`

## További örökölt tagok

### 6.17.1. Részletes leírás

Egy szint alap beállításait tudjuk vele betölteni: az ellenfelek által követett útvonalat és a fészek helyzetét.

### 6.17.2. Konstruktorok és destruktorok dokumentációja

#### 6.17.2.1. LevelConfigParser()

```
pftd::utils::parser::LevelConfigParser::LevelConfigParser (
    std::string const & sourceFile )
```

##### Paraméterek

<i>sourceFile</i>	A fájlnek az elérési útvonala, ahonnan a konfigot betöltjük.
-------------------	--

#### 6.17.2.2. ~LevelConfigParser()

```
pftd::utils::parser::LevelConfigParser::~~LevelConfigParser ( ) [default]
```

### 6.17.3. Tagfüggvények dokumentációja

#### 6.17.3.1. \_getAttribute()

```
std::string pftd::utils::parser::LevelConfigParser::_getAttribute ( ) [private]
```

Következő attribútum beolvasása a konfigurálból.

##### Visszatérési érték

A kapott attribútum neve.



### 6.17.3.2. getAttribute()

```
std::vector< Vec2f > pftd::utils::parser::LevelConfigParser::getAttribute (
    std::string name ) const
```

Mentett attribútum értékének kiolvasása.

#### Paraméterek

<i>name</i>	Az attribútum neve.
-------------	---------------------

#### Visszatérési érték

Az attribútum értéke (mindig egy array).

### 6.17.3.3. parse()

```
void pftd::utils::parser::LevelConfigParser::parse ( ) [override], [virtual]
```

A parse főciklus.

Megvalósítja a következőket: [pftd::utils::parser::Parser](#).

## 6.17.4. Adatmezők dokumentációja

### 6.17.4.1. m\_attribs

```
std::unordered_map<std::string, std::vector<Vec2f> > pftd::utils::parser::LevelConfigParser↵
::m_attribs [private]
```

Mentett [attribútum, érték] párok.

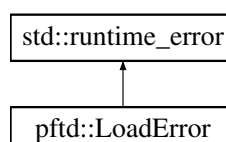
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/utills/parsers.hpp](#)

## 6.18. pftd::LoadError struktúráreferencia

```
#include <resources.hpp>
```

A pftd::LoadError osztály származási diagramja:



### 6.18.1. Részletes leírás

Erőforrás-betöltés hiba.

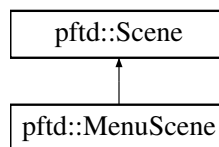
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/[resources.hpp](#)

## 6.19. pftd::MenuScene osztályreferencia

```
#include <menu.hpp>
```

A pftd::MenuScene osztály származási diagramja:



### Publikus tagfüggvények

- [MenuScene](#) ()
- [~MenuScene](#) ()=default
- void [onEvent](#) (sf::Event const &event) override  
*Delegált események kezelése.*
- void [update](#) (float dt) override  
*Update.*
- void [toggleActive](#) ([Scene::StateFlag](#) flag=[Scene::StateFlag::NONE](#)) override  
*Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.*

### Privát tagfüggvények

- bool [\\_isSaveFileAvailable](#) ()  
*Megnézi, hogy elérhető e a mentés fájl ([GameScene : : SAVE\\_FILE\\_PATH](#) alapján).*

### Privát attribútumok

- std::vector< [gr::Button](#) \* > [m\\_buttons](#)

### További örökölt tagok

#### 6.19.1. Részletes leírás

Menü nézet.

## 6.19.2. Konstruktorkok és destruktorkok dokumentációja

### 6.19.2.1. MenuScene()

```
pftd::MenuScene::MenuScene ( )
```

### 6.19.2.2. ~MenuScene()

```
pftd::MenuScene::~MenuScene ( ) [default]
```

## 6.19.3. Tagfüggvények dokumentációja

### 6.19.3.1. \_isSaveFileAvailable()

```
bool pftd::MenuScene::_isSaveFileAvailable ( ) [private]
```

Megnézi, hogy elérhető e a mentés fájl ([GameScene::SAVE\\_FILE\\_PATH](#) alapján).

#### Visszatérési érték

Elérhető e a fájl.

### 6.19.3.2. onEvent()

```
void pftd::MenuScene::onEvent (
    sf::Event const & event ) [override], [virtual]
```

Delegált események kezelése.

#### Paraméterek

<i>event</i>	Kezelni kívánt esemény.
--------------	-------------------------

Megvalósítja a következőket: [pftd::Scene](#).

### 6.19.3.3. toggleActive()

```
void pftd::MenuScene::toggleActive (
    Scene::StateFlag flag = Scene::StateFlag::NONE ) [override], [virtual]
```

Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.

#### Paraméterek

<i>flag</i>	Egyéb "kérést" közlő opció flag.
-------------	----------------------------------

Újraimplementált ősök: [pftd::Scene](#).

### 6.19.3.4. update()

```
void pftd::MenuScene::update (
    float dt ) [override], [virtual]
```

Update.

#### Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Megvalósítja a következőket: [pftd::Scene](#).

## 6.19.4. Adatmezők dokumentációja

### 6.19.4.1. m\_buttons

```
std::vector<gr::Button*> pftd::MenuScene::m_buttons [private]
```

Gombok.

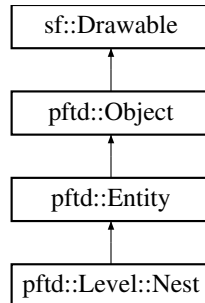
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/scenes/menu.hpp](#)

## 6.20. pftd::Level::Nest struktúrareferencia

```
#include <level.hpp>
```

A pftd::Level::Nest osztály származási diagramja:



### Publikus tagfüggvények

- `Nest (utils::Vec2f const &position)`  
*A pozíción kívül minden más hard code-olt.*
- `Nest (Nest const &other)=default`
- `~Nest ()=default`
- `Nest * clone () const override`  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

### Adatmezők

- `int const radiusPixel = 110`

### További örökölt tagok

#### 6.20.1. Részletes leírás

Fészek.

#### 6.20.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.20.2.1. Nest() [1/2]

```
pftd::Level::Nest::Nest (
    utils::Vec2f const & position ) [explicit]
```

A pozíción kívül minden más hard code-olt.

## Paraméterek

<i>position</i>	Pozíció.
-----------------	----------

**6.20.2.2. Nest()** [2/2]

```
pftd::Level::Nest::Nest (
    Nest const & other ) [default]
```

**6.20.2.3. ~Nest()**

```
pftd::Level::Nest::~~Nest ( ) [default]
```

**6.20.3. Tagfüggvények dokumentációja****6.20.3.1. clone()**

```
Nest * pftd::Level::Nest::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

**Visszatérési érték**

A másolat.

Megvalósítja a következőket: [pftd::Entity](#).

**6.20.4. Adatmezők dokumentációja****6.20.4.1. radiusPixel**

```
int const pftd::Level::Nest::radiusPixel = 110
```

Az a környezete, amelybe tornyot nem lehet tenni.

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/game/level.hpp](#)

## 6.21. pftd::Object osztályreferencia

```
#include <object_base.hpp>
```

A pftd::Object osztály származási diagramja:



### Adatszerkezetek

- struct [Compare](#)

### Publikus tagfüggvények

- [Object](#) ()=default
- [Object](#) (utils::Vec2f position, utils::Vec2f size, int zIndex=0)
- [Object](#) (Object const &)=default
- [Object](#) (Object &&) noexcept=default
- virtual ~[Object](#) ()=default
- virtual void [draw](#) (sf::RenderTarget &target, sf::RenderStates states) const =0

*Objektum megjelenítése.*

### Védett attribútumok

- int [zIndex](#) = 0
- utils::Vec2f [position](#)
- utils::Vec2f [size](#)

#### 6.21.1. Részletes leírás

Programelemek.

#### 6.21.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.21.2.1. Object() [1/4]

```
pftd::Object::Object ( ) [default]
```

##### 6.21.2.2. Object() [2/4]

```
pftd::Object::Object (
    utils::Vec2f position,
    utils::Vec2f size,
    int zIndex = 0 ) [inline]
```

## Paraméterek

<i>position</i>	Pozíció koordinátái.
<i>size</i>	Méret.
<i>zIndex</i>	Z koordináta: "mélységi index."

**6.21.2.3. Object() [3/4]**

```
pftd::Object::Object (
    Object const & ) [default]
```

**6.21.2.4. Object() [4/4]**

```
pftd::Object::Object (
    Object && ) [default], [noexcept]
```

**6.21.2.5. ~Object()**

```
virtual pftd::Object::~Object ( ) [virtual], [default]
```

**6.21.3. Tagfüggvények dokumentációja****6.21.3.1. draw()**

```
virtual void pftd::Object::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [pure virtual]
```

Objektum megjelenítése.

## Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítják a következők: [pftd::Level](#), [pftd::Entity](#), [pftd::gr::Button](#), [pftd::gr::Sprite](#), [pftd::gr::Label](#), [pftd::GameScene::InventoryItem](#) és [pftd::GameScene::Inventory](#).



## 6.21.4. Adatmezők dokumentációja

### 6.21.4.1. position

```
utils::Vec2f pftd::Object::position [protected]
```

Pozíció.

### 6.21.4.2. size

```
utils::Vec2f pftd::Object::size [protected]
```

Méret (szélesség x magasság).

### 6.21.4.3. zIndex

```
int pftd::Object::zIndex = 0 [protected]
```

Z koordináta.

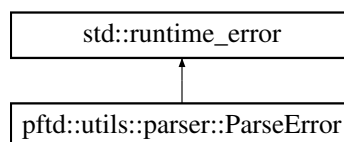
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/objects/[object\\_base.hpp](#)

## 6.22. pftd::utils::parser::ParseError struktúrareferencia

```
#include <parsers.hpp>
```

A pftd::utils::parser::ParseError osztály származási diagramja:



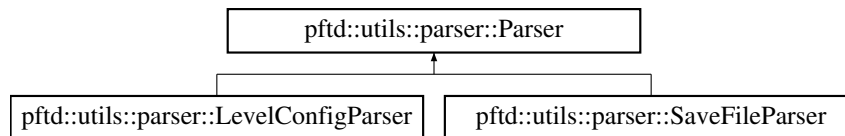
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/utils/[parsers.hpp](#)

## 6.23. pftd::utils::parser::Parser osztályreferencia

```
#include <parsers.hpp>
```

A pftd::utils::parser::Parser osztály származási diagramja:



### Publikus tagfüggvények

- `Parser` (`std::string const &sourceFile, std::string label=""`)
- virtual `~Parser` ()
- void `reset` ()  
*Reseteli az IO flageket, és a stream elejére mozgatja a kurzort.*
- bool `isLabelValid` (bool `skip=true`)  
*Megmondja, hogy tényleg a megadott karaktersorral kezdődik-e a stream.*
- void `skip` (`size_t howMany, char until='\n'`)  
*Ugorjon át valamennyi karaktert, de legfeljebb egy megadottig.*
- void `skip` (`char while_`)  
*Ugorjon át mindent amíg a következő karakter egy megadott.*
- template<typename T >  
  T `get` ()  
*Beolvas egy megadott típusú adatot.*
- char `peekAhead` ()  
*Megnézi, hogy milyen karakter van a kurzornál (lásd: megjegyzés).*
- virtual void `parse` ()=0  
*A parse főciklus.*

### Védett attribútumok

- std::ifstream `sourceStream`
- std::string `validLabel`
- char `commentDenoter` = '#'

### Privát tagfüggvények

- void `_skipLine` ()  
*Egész sor átugrása.*
- void `_skipWhitespace` ()  
*Whitespace-ek átugrása.*

## 6.23.1. Konstruktorkok és destruktorkok dokumentációja

### 6.23.1.1. Parser()

```
pftd::utils::parser::Parser::Parser (
    std::string const & sourceFile,
    std::string label = "" )
```

## Paraméterek

<i>sourceFile</i>	A fájlnek az elérési útvonala, ahonnan olvasni szeretnénk.
<i>label</i>	A karaktersor, amellyel kötelezően kezdődnie kell a fájlnek.

**6.23.1.2. ~Parser()**

```
virtual pftd::utils::parser::Parser::~~Parser ( ) [virtual]
```

**6.23.2. Tagfüggvények dokumentációja****6.23.2.1. \_skipLine()**

```
void pftd::utils::parser::Parser::_skipLine ( ) [private]
```

Egész sor átugrása.

**6.23.2.2. \_skipWhitespace()**

```
void pftd::utils::parser::Parser::_skipWhitespace ( ) [private]
```

Whitespace-ek átugrása.

**6.23.2.3. get()**

```
template<typename T >  
T pftd::utils::parser::Parser::get ( ) [inline]
```

Beolvas egy megadott típusú adatot.

Megjegyzés: a kommenteket átugorja.

**Visszatérési érték**

A beolvasott adat.

**6.23.2.4. isLabelValid()**

```
bool pftd::utils::parser::Parser::isLabelValid (  
    bool skip = true )
```

Megmondja, hogy tényleg a megadott karaktersorral kezdődik e a stream.

**Paraméterek**

<i>skip</i>	Ha beolvasta a karaktersort, akkor skippelje e, vagy visszamozogjon.
-------------	--

**Visszatérési érték**

Helyes e a karaktersor.

**6.23.2.5. parse()**

```
virtual void pftd::utils::parser::Parser::parse ( ) [pure virtual]
```

A parse főciklus.

Megvalósítják a következők: [pftd::utils::parser::LevelConfigParser](#) és [pftd::utils::parser::SaveFileParser](#).

**6.23.2.6. peekAhead()**

```
char pftd::utils::parser::Parser::peekAhead ( )
```

Megnézi, hogy milyen karakter van a kurzornál (lásd: megjegyzés).

Megjegyzés: átugorja a white space-eket és a kommenteket, és csak az azutáni karaktert nézi meg.

**Visszatérési érték**

A kurzornál levő karakter.

**6.23.2.7. reset()**

```
void pftd::utils::parser::Parser::reset ( )
```

Reseteli az IO flageket, és a stream elejére mozgatja a kurzort.

**6.23.2.8. skip() [1/2]**

```
void pftd::utils::parser::Parser::skip (
    char while_ )
```

Ugorjon át mindent amíg a következő karakter egy megadott.

## Paraméterek

<i>while</i> ↔	Addig ugor át mindent, amíg ez a következő karakter.
—	

**6.23.2.9. skip() [2/2]**

```
void pftd::utils::parser::Parser::skip (
    size_t howMany,
    char until = '\n' )
```

Ugorjon át valamennyi karaktert, de legfeljebb egy megadottig.

## Paraméterek

<i>howMany</i>	Ennyi karaktert ugrik át legfeljebb.
<i>until</i>	Eddig a karakterig ugrik át mindent, ha még nem ugrott át <i>howMany</i> számút (ezt is beleértve).

**6.23.3. Adatmezők dokumentációja****6.23.3.1. commentDenoter**

```
char pftd::utils::parser::Parser::commentDenoter = '#' [protected]
```

Kommentet jelölő karakter (ezt követően a sorban minden karakter a komment része).

**6.23.3.2. sourceStream**

```
std::ifstream pftd::utils::parser::Parser::sourceStream [protected]
```

Input stream.

**6.23.3.3. validLabel**

```
std::string pftd::utils::parser::Parser::validLabel [protected]
```

Elvart karaktersor, amivel a streamnek kezdődnie kell.

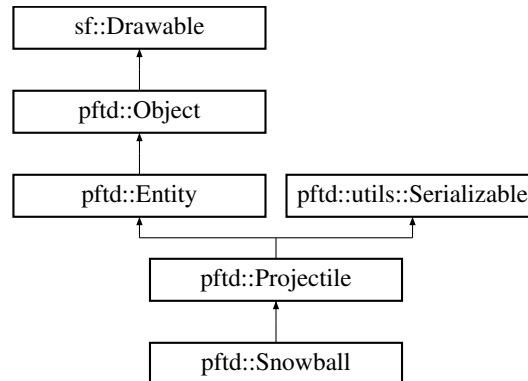
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/utils/[parsers.hpp](#)

## 6.24. pftd::Projectile osztályreferencia

```
#include <projectile_base.hpp>
```

A pftd::Projectile osztály származási diagramja:



### Publikus tagfüggvények

- virtual `~Projectile()`=default
- virtual void `update` (float dt) override  
*Update.*
- void `serialize` (std::ostream &out) const override  
*Kiír egy streamre.*

### Adatmezők

- `utils::Vec2f` direction
- float `linearSpeed`
- float `angularVelocityRadPerSec`

### Védett tagfüggvények

- `Projectile` (std::string const &spriteSrc, `utils::Vec2f` const &position, `utils::Vec2f` const &size, `utils::Vec2f` const &direction, float speed, float angularSpeed=0.0f, int zIndex=0)

### Védett attribútumok

- `ProjectileID` id

#### 6.24.1. Konstruktorkok és destruktorkok dokumentációja

### 6.24.1.1. ~Projectile()

```
virtual pftd::Projectile::~~Projectile ( ) [virtual], [default]
```

### 6.24.1.2. Projectile()

```
pftd::Projectile::Projectile (
    std::string const & spriteSrc,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    utils::Vec2f const & direction,
    float speed,
    float angularSpeed = 0.0f,
    int zIndex = 0 ) [protected]
```

#### Paraméterek

<i>spriteSrc</i>	Sprite eleérési útvonal.
<i>position</i>	Pozíció.
<i>size</i>	Méret.
<i>direction</i>	Írány.
<i>speed</i>	Lineáris gyorsaság.
<i>angularSpeed</i>	Forgási sebesség (radián / mp).
<i>zIndex</i>	Z koordináta.

## 6.24.2. Tagfüggvények dokumentációja

### 6.24.2.1. serialize()

```
void pftd::Projectile::serialize (
    std::ostream & out ) const [override], [virtual]
```

Kiír egy streamre.

#### Paraméterek

<i>out</i>	A stream.
------------	-----------

Megvalósítja a következőket: [pftd::utils::Serializable](#).

#### 6.24.2.2. update()

```
virtual void pftd::Projectile::update (
    float dt ) [override], [virtual]
```

Update.

##### Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Újraimplementált ősök: [pftd::Entity](#).

### 6.24.3. Adatmezők dokumentációja

#### 6.24.3.1. angularVelocityRadPerSec

```
float pftd::Projectile::angularVelocityRadPerSec
```

Forgási sebesség.

#### 6.24.3.2. direction

```
utils::Vec2f pftd::Projectile::direction
```

Írány (normalizált).

#### 6.24.3.3. id

```
ProjectileID pftd::Projectile::id [protected]
```

Mentéshez és betöltéshez használd azonosító.

#### 6.24.3.4. linearSpeed

```
float pftd::Projectile::linearSpeed
```

Lineáris gyorsaság.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/projectiles/projectile\\_base.hpp](#)



## 6.25. pftd::utils::Random osztályreferencia

```
#include <random_gen.hpp>
```

### Publikus tagfüggvények

- `Random()`=delete

### Statikus publikus tagfüggvények

- `template<typename RandomDist >`  
`static auto generate (RandomDist distribution)`  
*Random szám generálása megadott eloszlás szerint.*

### Statikus publikus attribútumok

- `static std::default_random_engine randomEngine`

#### 6.25.1. Részletes leírás

`Random` szám generáláshoz használt segédosztály. Ez egy teljesen statikus osztály.

#### 6.25.2. Konstruktorkok és destruktorok dokumentációja

##### 6.25.2.1. Random()

```
pftd::utils::Random::Random ( ) [delete]
```

#### 6.25.3. Tagfüggvények dokumentációja

##### 6.25.3.1. generate()

```
template<typename RandomDist >  
static auto pftd::utils::Random::generate (  
    RandomDist distribution ) [inline], [static]
```

`Random` szám generálása megadott eloszlás szerint.

## Paraméterek

<i>distribution</i>	Generálandó számok eloszlása.
---------------------	-------------------------------

## Visszatérési érték

Generált szám.

## 6.25.4. Adatmezők dokumentációja

### 6.25.4.1. randomEngine

```
std::default_random_engine pftd::utils::Random::randomEngine [static]
```

Felhasznált random engine. (Az alapértelmezett.)

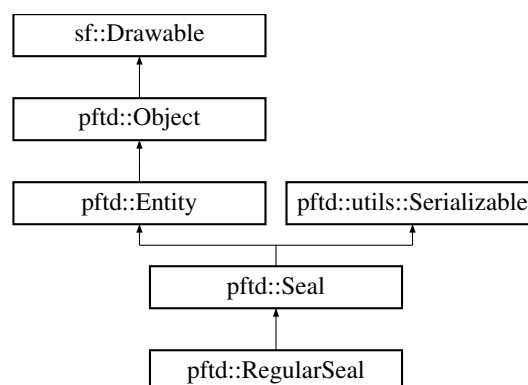
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/utils/[random\\_gen.hpp](#)

## 6.26. pftd::RegularSeal osztályreferencia

```
#include <regular.hpp>
```

A pftd::RegularSeal osztály származási diagramja:



### Publikus tagfüggvények

- [RegularSeal](#) ([FollowPath](#) const &followpath)
- [Seal](#) \* [clone](#) () const override

*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

## További örökölt tagok

### 6.26.1. Részletes leírás

"Normális" főka.

### 6.26.2. Konstruktorok és destruktorok dokumentációja

#### 6.26.2.1. RegularSeal()

```
pftd::RegularSeal::RegularSeal (
    FollowPath const & followpath )
```

##### Paraméterek

<i>followpath</i>	Az útvonal amin végig kell menjen (majd pedig vissza).
-------------------	--

### 6.26.3. Tagfüggvények dokumentációja

#### 6.26.3.1. clone()

```
Seal * pftd::RegularSeal::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

##### Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Seal](#).

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/seals/regular.hpp](#)

## 6.27. pftd::Renderer osztályreferencia

```
#include <app.hpp>
```

## Publikus tagfüggvények

- `Renderer` (unsigned int width, unsigned int height, std::string const &windowTitle)  
*Létrehozza az ablakot a rendering kontextussal együtt.*
- `Renderer` (`Renderer` const &)=delete
- `Renderer` (`Renderer` &&)=delete
- `~Renderer` ()  
*Felszabadítja az ablak erőforrásait.*
- void `render` ()  
*Átmásolja a frame buffer-re az `m_queue`-ban tárolt objektumokat.*
- void `clear` () const  
*Törli a frame buffer-t.*
- void `display` () const  
*Megjeleníti a frame buffer tartalmát.*
- void `pushQueue` (`Object` const \*o)  
*Hozzáad az `m_queue`-hoz egy megjelenítendő objektumot.*
- sf::RenderWindow \* `getWindow` () const  
*`m_window` getter.*

## Privát attribútumok

- sf::RenderWindow \* `m_window` = nullptr
- unsigned int `m_width`
- unsigned int `m_height`
- std::priority\_queue< `Object` const \*, std::vector< `Object` const \* >, `Object::Compare` > `m_queue`

## Barátok

- class `App`

### 6.27.1. Részletes leírás

A renderer.

### 6.27.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.27.2.1. `Renderer()` [1/3]

```
pftd::Renderer::Renderer (
    unsigned int width,
    unsigned int height,
    std::string const & windowTitle )
```

Létrehozza az ablakot a rendering kontextussal együtt.

## Paraméterek

<i>width</i>	Az ablak szélessége.
<i>height</i>	Az ablak magassága.
<i>windowTitle</i>	Az ablak címe.

**6.27.2.2. Renderer()** [2/3]

```
pftd::Renderer::Renderer (  
    Renderer const & ) [delete]
```

**6.27.2.3. Renderer()** [3/3]

```
pftd::Renderer::Renderer (  
    Renderer && ) [delete]
```

**6.27.2.4. ~Renderer()**

```
pftd::Renderer::~~Renderer ( )
```

Felszabadítja az ablak erőforrásait.

**6.27.3. Tagfüggvények dokumentációja****6.27.3.1. clear()**

```
void pftd::Renderer::clear ( ) const [inline]
```

Törli a frame buffer-t.

**6.27.3.2. display()**

```
void pftd::Renderer::display ( ) const [inline]
```

Megjeleníti a frame buffer tartalmát.

### 6.27.3.3. getWindow()

```
sf::RenderWindow * pftd::Renderer::getWindow ( ) const [inline]
```

m\_window getter.

#### Visszatérési érték

A kezelt ablak.

### 6.27.3.4. pushQueue()

```
void pftd::Renderer::pushQueue (
    Object const * o ) [inline]
```

Hozzáad az m\_queue-hoz egy megjelenítendő objektumot.

#### Paraméterek

<i>o</i>	Az objektum amit megjeleníteni szeretnénk.
----------	--

### 6.27.3.5. render()

```
void pftd::Renderer::render ( )
```

Átmásolja a frame buffer-re az m\_queue-ban tárolt objektumokat.

## 6.27.4. Barát és kapcsolódó függvények dokumentációja

### 6.27.4.1. App

```
friend class App [friend]
```

### 6.27.5. Adatmezők dokumentációja

### 6.27.5.1. m\_height

```
unsigned int pftd::Renderer::m_height [private]
```

Az ablak magassága.

### 6.27.5.2. m\_queue

```
std::priority_queue<Object const*, std::vector<Object const*>, Object::Compare> pftd::↵
Renderer::m_queue [private]
```

A render queue.

### 6.27.5.3. m\_width

```
unsigned int pftd::Renderer::m_width [private]
```

Az ablak szélessége.

### 6.27.5.4. m\_window

```
sf::RenderWindow* pftd::Renderer::m_window = nullptr [private]
```

A kezelt ablak.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/app.hpp](#)

## 6.28. pftd::ResourceManager osztályreferencia

```
#include <resources.hpp>
```

### Publikus tagfüggvények

- [ResourceManager](#) ([ResourceManager](#) const &)=delete
- [ResourceManager](#) ([ResourceManager](#) &&)=delete
- [~ResourceManager](#) ()=default  
*Felszabadítja a tárolt erőforrásokat.*
- void [loadDefaultFont](#) (std::string const &path)  
*Az alapértelmezett (valójában csak ez van) betűtípus betöltése.*
- sf::Texture const & [getTexture](#) (std::string const &source)  
*Új vagy tárolt textúra megszerzése.*
- sf::SoundBuffer const & [getSound](#) (std::string const &source)  
*Új vagy tárolt audio buffer megszerzése.*
- sf::Font const & [getDefaultFont](#) () const  
*m\_defaultFont getter.*

## Statikus publikus tagfüggvények

- static `ResourceManager * getInstance ()`  
*A singleton instance megszerzése.*
- static `ResourceManager * create ()`  
*Létrehozza a singleton instance-t.*
- static void `destroy ()`  
*Törli a singleton instance-t.*

## Privát tagfüggvények

- `ResourceManager ()=default`

## Privát attribútumok

- `sf::Font m_defaultFont`
- `std::unordered_map< std::string, sf::Texture > m_textures`
- `std::unordered_map< std::string, sf::SoundBuffer > m_sounds`

## Statikus privát attribútumok

- static `ResourceManager * m_instance`

### 6.28.1. Részletes leírás

Az erőforrás-kezelő.

### 6.28.2. Konstruktorok és destruktorok dokumentációja

#### 6.28.2.1. `ResourceManager()` [1/3]

```
pftd::ResourceManager::ResourceManager (  
    ResourceManager const & ) [delete]
```

#### 6.28.2.2. `ResourceManager()` [2/3]

```
pftd::ResourceManager::ResourceManager (  
    ResourceManager && ) [delete]
```



### 6.28.2.3. ~ResourceManager()

```
pftd::ResourceManager::~~ResourceManager ( ) [default]
```

Felszabadítja a tárolt erőforrásokat.

### 6.28.2.4. ResourceManager() [3/3]

```
pftd::ResourceManager::ResourceManager ( ) [private], [default]
```

## 6.28.3. Tagfüggvények dokumentációja

### 6.28.3.1. create()

```
static ResourceManager * pftd::ResourceManager::create ( ) [static]
```

Létrehozza a singleton instance-t.

Megjegyzés: Meg kell hívni a `destroy()`-t, ha már nincs szükség rá!

#### Visszatérési érték

Az osztályhoz tartozó singleton instance.

### 6.28.3.2. destroy()

```
static void pftd::ResourceManager::destroy ( ) [inline], [static]
```

Törli a singleton instance-t.

### 6.28.3.3. getDefaultFont()

```
sf::Font const & pftd::ResourceManager::getDefaultFont ( ) const [inline]
```

`m_defaultFont` getter.

#### Visszatérési érték

Az alapértelmezett betűtípus.

#### 6.28.3.4. getInstance()

```
static ResourceManager * pftd::ResourceManager::getInstance ( ) [inline], [static]
```

A singleton instance megszerzése.

##### Visszatérési érték

Az osztályhoz tartozó singleton instance.

#### 6.28.3.5. getSound()

```
sf::SoundBuffer const & pftd::ResourceManager::getSound (
    std::string const & source )
```

Új vagy tárolt audio buffer megszerzése.

Megjegyzés: `LoadError`-t dob, ha nem találja.

##### Paraméterek

<i>source</i>	A hangfájl elérési útvonala / azonosítója (megegyezik).
---------------	---

##### Visszatérési érték

A kért audio buffer.

#### 6.28.3.6. getTexture()

```
sf::Texture const & pftd::ResourceManager::getTexture (
    std::string const & source )
```

Új vagy tárolt textúra megszerzése.

Megjegyzés: `LoadError`-t dob, ha nem találja.

##### Paraméterek

<i>source</i>	A textúra elérési útvonala / azonosítója (megegyezik).
---------------	--

##### Visszatérési érték

A kért textúra.

### 6.28.3.7. loadDefaultFont()

```
void pftd::ResourceManager::loadDefaultFont (
    std::string const & path )
```

Az alapértelmezett (valójában csak ez van) betűtípus betöltése.

Megjegyzés: `LoadError`-t dob, ha nem tudja betölteni bármi miatt is.

#### Paraméterek

<i>path</i>	A betűtípus elérési útvonala.
-------------	-------------------------------

## 6.28.4. Adatmezők dokumentációja

### 6.28.4.1. m\_defaultFont

```
sf::Font pftd::ResourceManager::m_defaultFont [private]
```

Az alapértelmezett betűtípus.

### 6.28.4.2. m\_instance

```
ResourceManager* pftd::ResourceManager::m_instance [static], [private]
```

A singleton instance.

### 6.28.4.3. m\_sounds

```
std::unordered_map<std::string, sf::SoundBuffer> pftd::ResourceManager::m_sounds [private]
```

Audio: [source, hang buffer példány]

### 6.28.4.4. m\_textures

```
std::unordered_map<std::string, sf::Texture> pftd::ResourceManager::m_textures [private]
```

Textúrák: [source, textúra példány]

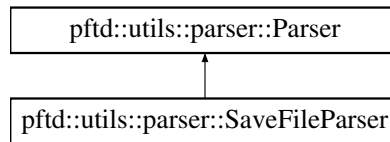
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/[resources.hpp](#)

## 6.29. pftd::utils::parser::SaveFileParser osztályreferencia

```
#include <parsers.hpp>
```

A pftd::utils::parser::SaveFileParser osztály származási diagramja:



### Adatszerkezetek

- struct [EntityInfo](#)
- struct [StatsInfo](#)

### Publikus típusok

- enum class [EntityType](#) { [TOWER](#) = 0U , [SEAL](#) , [PROJECTILE](#) }

### Publikus tagfüggvények

- [SaveFileParser](#) (std::string const &sourceFile)
- [~SaveFileParser](#) ()=default
- void [parse](#) () override  
*A parse fűciklus.*
- [StatsInfo](#) const & [getStats](#) () const  
*Mentett statisztika beolvasása: pont, pénz, HP.*
- std::vector< [EntityInfo](#) > const & [getEntities](#) () const  
*Mentett entitások beolvasása.*

### Privát tagfüggvények

- void [\\_getStats](#) ()  
*Statisztika beolvasása a fájlból.*
- void [\\_getEntity](#) ()  
*Következ entitás beolvasása a fájlból.*

### Privát attribútumok

- [StatsInfo](#) [m\\_readStats](#)
- std::vector< [EntityInfo](#) > [m\\_entities](#)

## További örökölt tagok

### 6.29.1. Részletes leírás

Mentett játékállás betöltésére szolgál.

### 6.29.2. Enumeráció-tagok dokumentációja

#### 6.29.2.1. EntityType

```
enum class pftd::utils::parser::SaveFileParser::EntityType [strong]
```

Mentett játék betöltéséhez használt entitás típusok.

Enumeráció-értékek

TOWER	
SEAL	
PROJECTILE	

### 6.29.3. Konstruktorok és destruktorok dokumentációja

#### 6.29.3.1. SaveFileParser()

```
pftd::utils::parser::SaveFileParser::SaveFileParser (  
    std::string const & sourceFile )
```

#### 6.29.3.2. ~SaveFileParser()

```
pftd::utils::parser::SaveFileParser::~SaveFileParser ( ) [default]
```

### 6.29.4. Tagfüggvények dokumentációja

#### 6.29.4.1. `_getEntity()`

```
void pftd::utils::parser::SaveFileParser::_getEntity ( ) [private]
```

Következ entitás beolvasása a fájlból.

#### 6.29.4.2. `_getStats()`

```
void pftd::utils::parser::SaveFileParser::_getStats ( ) [private]
```

Statisztika beolvasása a fájlból.

#### 6.29.4.3. `getEntities()`

```
std::vector< EntityInfo > const & pftd::utils::parser::SaveFileParser::getEntities ( ) const  
[inline]
```

Mentett entitások beolvasása.

##### Visszatérési érték

Ezeket az entitásokat (azoknak a lényeges információját) tartalmazó lista.

#### 6.29.4.4. `getStats()`

```
StatsInfo const & pftd::utils::parser::SaveFileParser::getStats ( ) const [inline]
```

Mentett statisztika beolvasása: pont, pénz, HP.

##### Visszatérési érték

A mentett statisztikát tartalmazó struktúra.

#### 6.29.4.5. `parse()`

```
void pftd::utils::parser::SaveFileParser::parse ( ) [override], [virtual]
```

A parse főciklus.

Megvalósítja a következőket: `pftd::utils::parser::Parser`.

## 6.29.5. Adatmezők dokumentációja

### 6.29.5.1. m\_entities

```
std::vector<EntityInfo> pftd::utils::parser::SaveFileParser::m_entities [private]
```

A beolvasott entitás infók.

### 6.29.5.2. m\_readStats

```
StatsInfo pftd::utils::parser::SaveFileParser::m_readStats [private]
```

A beolvasott statisztika.

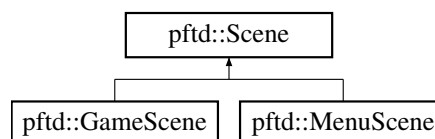
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/utils/parsers.hpp

## 6.30. pftd::Scene osztályreferencia

```
#include <scene.hpp>
```

A pftd::Scene osztály származási diagramja:



### Publikus típusok

- enum class StateFlag : uint8\_t { NONE = 0 , LOAD\_STATE = 1 << 0 }

### Publikus tagfüggvények

- Scene ()=default
- Scene (Scene const &)=delete
- Scene (Scene &&) noexcept=delete
- virtual ~Scene ()  
*Törli a kezelt objektumokat.*
- ObjPtrVec const & getObjects () const  
*objects getter (konstans).*
- ObjPtrVec & getObjects ()  
*objects getter.*
- void setMusic (std::string const &source, float volume=100.0f)  
*Zene beállítása (loopen megy amíg aktív a nézet).*
- virtual void toggleActive (StateFlag flag=StateFlag::NONE)  
*Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.*
- virtual void onEvent (sf::Event const &event)=0  
*Delegált események kezelése.*
- virtual void update (float dt)=0  
*Update.*

## Védett attribútumok

- bool `isActive` = false
- `ObjPtrVec` objects
- `sf::Music * backgroundMusic` = nullptr

## Privát típusok

- using `ObjPtrVec` = `std::vector< Object * >`

### 6.30.1. Részletes leírás

Nézet ősosztály.

### 6.30.2. Típusdefiníció-tagok dokumentációja

#### 6.30.2.1. ObjPtrVec

```
using pftd::Scene::ObjPtrVec = std::vector<Object*> [private]
```

### 6.30.3. Enumeráció-tagok dokumentációja

#### 6.30.3.1. StateFlag

```
enum class pftd::Scene::StateFlag : uint8_t [strong]
```

Plusz információt közlő opciók.

Enumeráció-értékek

NONE	
LOAD_STATE	

### 6.30.4. Konstruktorkok és destruktorkok dokumentációja



**6.30.4.1. Scene() [1/3]**

```
pftd::Scene::Scene ( ) [default]
```

**6.30.4.2. Scene() [2/3]**

```
pftd::Scene::Scene (
    Scene const & ) [delete]
```

**6.30.4.3. Scene() [3/3]**

```
pftd::Scene::Scene (
    Scene && ) [delete], [noexcept]
```

**6.30.4.4. ~Scene()**

```
virtual pftd::Scene::~Scene ( ) [virtual]
```

Törli a kezelt objektumokat.

**6.30.5. Tagfüggvények dokumentációja****6.30.5.1. getObjects() [1/2]**

```
ObjPtrVec & pftd::Scene::getObjects ( ) [inline]
```

objects getter.

**Visszatérési érték**

A kezelt elemek listája.

**6.30.5.2. getObjects() [2/2]**

```
ObjPtrVec const & pftd::Scene::getObjects ( ) const [inline]
```

objects getter (konstans).

**Visszatérési érték**

A kezelt elemek listája.

**6.30.5.3. onEvent()**

```
virtual void pftd::Scene::onEvent (
    sf::Event const & event ) [pure virtual]
```

Delegált események kezelése.

## Paraméterek

<i>event</i>	Kezelni kívánt esemény.
--------------	-------------------------

Megvalósítják a következők: [pftd::GameScene](#) és [pftd::MenuScene](#).

**6.30.5.4. setMusic()**

```
void pftd::Scene::setMusic (
    std::string const & source,
    float volume = 100.0f )
```

Zene beállítása (loopon megy amíg aktív a nézet).

## Paraméterek

<i>source</i>	A hangfájl elérési útvonala.
<i>volume</i>	A zene hangereje.

**6.30.5.5. toggleActive()**

```
virtual void pftd::Scene::toggleActive (
    StateFlag flag = StateFlag::NONE ) [virtual]
```

Nézet aktiválása/deaktiválása: aktív -> nem aktív VAGY nem aktív -> aktív.

## Paraméterek

<i>flag</i>	Egyéb "kérés" közlő opció flag.
-------------	---------------------------------

Újrimplementáló leszármazottak: [pftd::GameScene](#) és [pftd::MenuScene](#).

**6.30.5.6. update()**

```
virtual void pftd::Scene::update (
    float dt ) [pure virtual]
```

Update.

## Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Megvalósítják a következők: [pftd::GameScene](#) és [pftd::MenuScene](#).

## 6.30.6. Adatmezők dokumentációja

### 6.30.6.1. backgroundMusic

```
sf::Music* pftd::Scene::backgroundMusic = nullptr [protected]
```

Háttérzene.

### 6.30.6.2. isActive

```
bool pftd::Scene::isActive = false [protected]
```

Aktív e.

### 6.30.6.3. objects

```
ObjPtrVec pftd::Scene::objects [protected]
```

Kezelt objektumok.

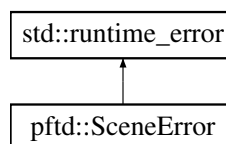
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/[scene.hpp](#)

## 6.31. pftd::SceneError struktúrareferencia

```
#include <app.hpp>
```

A pftd::SceneError osztály származási diagramja:



### 6.31.1. Részletes leírás

Nézethiba.

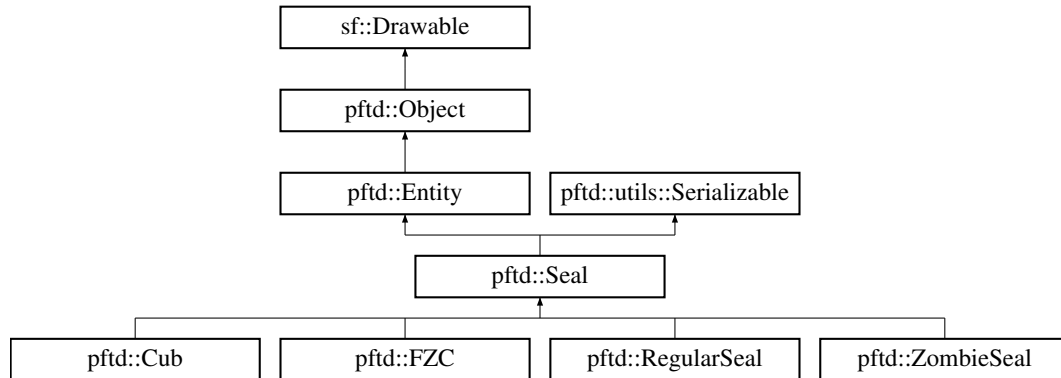
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- include/[app.hpp](#)

## 6.32. pftd::Seal osztályreferencia

```
#include <seal_base.hpp>
```

A pftd::Seal osztály származási diagramja:



### Publikus tagfüggvények

- virtual `~Seal()`=default
- void `lerpPath()`  
*LERP paraméter frissítése.*
- virtual `Seal * clone()` const override=0  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*
- void `update(float dt)` override  
*Update.*
- bool `hasCompletedPath()` const  
*returned getter.*
- bool `hasReachedNest()` const  
*reachedNest getter.*
- void `setLerpState(float param, bool backwards=false)`  
*Szükség van rá, amikor mentett állást töltünk.*
- void `damage(int hpLost=1)`  
*Lesebzés.*
- void `serialize(std::ostream &out)` const override  
*Kíír egy streamre.*

### Adatmezők

- bool `isCurrentlyStealing` = false
- int `hp`
- float `speed`
- unsigned int `value`

### Védett tagfüggvények

- `Seal(FollowPath const &followPath, std::string const &spriteSrc, utils::Vec2f const &size, int hp, float speed, unsigned int value, int zIndex=0)`

## Védett attribútumok

- [SealID](#) id
- [FollowPath](#) const & [followPath](#)
- float [lerpParam](#) = 0.0f
- bool [reachedNest](#) = false
- bool [returned](#) = false
- [EndPoint](#) nextPoint

## Privát tagfüggvények

- void [advanceAnimationFrame](#) () override  
*Animáció: képkocka léptetése.*

### 6.32.1. Részletes leírás

Fóka.

### 6.32.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.32.2.1. ~Seal()

```
virtual pftd::Seal::~Seal ( ) [virtual], [default]
```

#### 6.32.2.2. Seal()

```
pftd::Seal::Seal (
    FollowPath const & followPath,
    std::string const & spriteSrc,
    utils::Vec2f const & size,
    int hp,
    float speed,
    unsigned int value,
    int zIndex = 0 ) [protected]
```

#### Paraméterek

<i>followPath</i>	Az útvonal, amit követnie kell.
<i>spriteSrc</i>	A sprite-jának az elérési útvonala.
<i>size</i>	Méret.
<i>hp</i>	HP.
<i>speed</i>	Gyorsaság.
<i>value</i>	Érték: ennyi pénzt kap a játékos ha megöli.
<i>zIndex</i>	Z koordináta.

### 6.32.3. Tagfüggvények dokumentációja

#### 6.32.3.1. advanceAnimationFrame()

```
void pftd::Seal::advanceAnimationFrame ( ) [override], [private], [virtual]
```

Animáció: képkocka léptetése.

Újrimplementált ősök: [pftd::Entity](#).

#### 6.32.3.2. clone()

```
virtual Seal * pftd::Seal::clone ( ) const [override], [pure virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

##### Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Entity](#).

Megvalósítják a következők: [pftd::Cub](#), [pftd::FZC](#), [pftd::RegularSeal](#) és [pftd::ZombieSeal](#).

#### 6.32.3.3. damage()

```
void pftd::Seal::damage (
    int hpLost = 1 )
```

Lesebzés.

##### Paraméterek

<i>hpLost</i>	Ennyi HP-ja fog lemenni a fókának.
---------------	------------------------------------

#### 6.32.3.4. hasCompletedPath()

```
bool pftd::Seal::hasCompletedPath ( ) const [inline]
```

returned getter.

**Visszatérési érték**

Elment e a fészekig és vissza a kezdőhelyre.

**6.32.3.5. hasReachedNest()**

```
bool pftd::Seal::hasReachedNest ( ) const [inline]
```

reachedNest getter.

**Visszatérési érték**

Elment e a fészekig.

**6.32.3.6. lerpPath()**

```
void pftd::Seal::lerpPath ( )
```

LERP paraméter frissítése.

A followPath pontjai közt lineárisan interpolál.

**6.32.3.7. serialize()**

```
void pftd::Seal::serialize (
    std::ostream & out ) const [override], [virtual]
```

Kiír egy streamre.

**Paraméterek**

<i>out</i>	A stream.
------------	-----------

Megvalósítja a következőket: [pftd::utils::Serializable](#).

**6.32.3.8. setLerpState()**

```
void pftd::Seal::setLerpState (
    float param,
    bool backwards = false ) [inline]
```

Szükség van rá, amikor mentett állást töltünk.

#### 6.32.3.9. update()

```
void pftd::Seal::update (
    float dt ) [override], [virtual]
```

Update.

##### Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Újraimplementált ősök: [pftd::Entity](#).

### 6.32.4. Adatmezők dokumentációja

#### 6.32.4.1. followPath

```
FollowPath const& pftd::Seal::followPath [protected]
```

Az útvonal amin végigmegy.

#### 6.32.4.2. hp

```
int pftd::Seal::hp
```

HP.

#### 6.32.4.3. id

```
SealID pftd::Seal::id [protected]
```

#### 6.32.4.4. isCurrentlyStealing

```
bool pftd::Seal::isCurrentlyStealing = false
```

Éppen lopás közben van e?

#### 6.32.4.5. lerpParam

```
float pftd::Seal::lerpParam = 0.0f [protected]
```

0-tól 1-ig terjedő paraméter az interpolációhoz.



**6.32.4.6. nextPoint**

```
EndPoint pftd::Seal::nextPoint [protected]
```

A következő pont az útvonalon.

**6.32.4.7. reachedNest**

```
bool pftd::Seal::reachedNest = false [protected]
```

Elérte e a fészket.

**6.32.4.8. returned**

```
bool pftd::Seal::returned = false [protected]
```

Visszatért e a kezdő ponthoz.

**6.32.4.9. speed**

```
float pftd::Seal::speed
```

Gyorsaság.

**6.32.4.10. value**

```
unsigned int pftd::Seal::value
```

Pénz amit ad halál után.

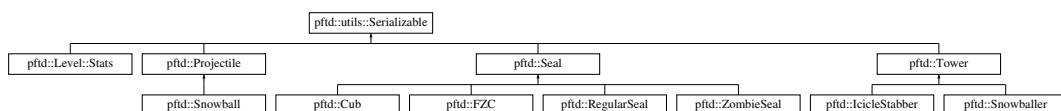
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/objects/entities/seals/[seal\\_base.hpp](#)

**6.33. pftd::utils::Serializable osztályreferencia**

```
#include <serializable.hpp>
```

A pftd::utils::Serializable osztály származási diagramja:



## Publikus tagfüggvények

- virtual void [serialize](#) (std::ostream &out) const =0  
*Kiír egy streamre.*

### 6.33.1. Részletes leírás

Szerializálható osztályok.

### 6.33.2. Tagfüggvények dokumentációja

#### 6.33.2.1. serialize()

```
virtual void pftd::utils::Serializable::serialize (
    std::ostream & out ) const [pure virtual]
```

Kiír egy streamre.

#### Paraméterek

<i>out</i>	A stream.
------------	-----------

Megvalósítják a következők: [pftd::Level::Stats](#), [pftd::Projectile](#), [pftd::Seal](#) és [pftd::Tower](#).

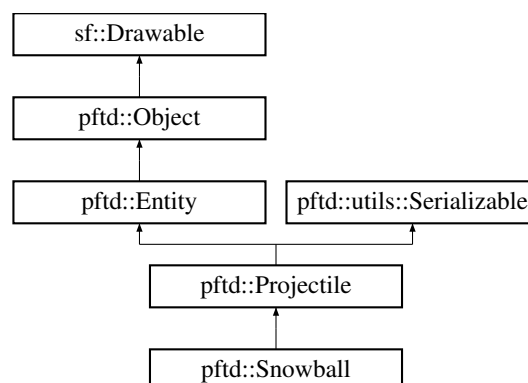
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- include/utils/[serializable.hpp](#)

## 6.34. pftd::Snowball osztályreferencia

```
#include <snowball.hpp>
```

A pftd::Snowball osztály származási diagramja:



## Publikus tagfüggvények

- `Snowball` (`utils::Vec2f` const & `position`, `utils::Vec2f` `direction`, float `speed`=50.0f)  
*A textúra hard code-olt.*
- `Snowball` \* `clone` () const override  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

## További örökölt tagok

### 6.34.1. Részletes leírás

Hógolyó lövedék.

### 6.34.2. Konstruktork és destruktork dokumentációja

#### 6.34.2.1. Snowball()

```
pftd::Snowball::Snowball (
    utils::Vec2f const & position,
    utils::Vec2f direction,
    float speed = 50.0f )
```

A textúra hard code-olt.

#### Paraméterek

<i>position</i>	Pozíció.
<i>direction</i>	Írány.
<i>speed</i>	Repülési gyorsaság.

### 6.34.3. Tagfüggvények dokumentációja

#### 6.34.3.1. clone()

```
Snowball * pftd::Snowball::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

**Visszatérési érték**

A másolat.

Megvalósítja a következőket: [pftd::Entity](#).

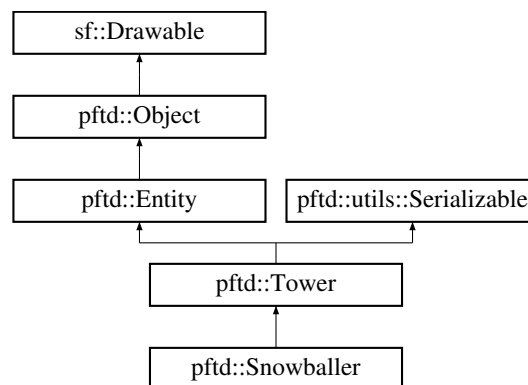
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/projectiles/snowball.hpp](#)

## 6.35. pftd::Snowballer osztályreferencia

```
#include <snowballer.hpp>
```

A pftd::Snowballer osztály származási diagramja:



### Publikus tagfüggvények

- [Snowballer](#) ([utils::Vec2f](#) const &[position](#), int [zIndex](#)=0)
- [Snowballer](#) ()
- [~Snowballer](#) ()=default
- [Tower](#) \* [clone](#) () const override  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*
- void [update](#) (float dt) override  
*Update.*
- void [attack](#) () override  
*Megtámadja a [Tower::target](#)-et, ha van.*

### További örökölt tagok

#### 6.35.1. Részletes leírás

Hógolyó dobáló.

## 6.35.2. Konstruktorkok és destruktorkok dokumentációja

### 6.35.2.1. Snowballer() [1/2]

```
pftd::Snowballer::Snowballer (
    utils::Vec2f const & position,
    int zIndex = 0 )
```

### 6.35.2.2. Snowballer() [2/2]

```
pftd::Snowballer::Snowballer ( )
```

### 6.35.2.3. ~Snowballer()

```
pftd::Snowballer::~~Snowballer ( ) [default]
```

## 6.35.3. Tagfüggvények dokumentációja

### 6.35.3.1. attack()

```
void pftd::Snowballer::attack ( ) [override], [virtual]
```

Megtámadja a `Tower::target`-et, ha van.

Újrimplementált ősök: `pftd::Tower`.

### 6.35.3.2. clone()

```
Tower * pftd::Snowballer::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

Visszatérési érték

A másolat.

Megvalósítja a következőket: `pftd::Tower`.

### 6.35.3.3. update()

```
void pftd::Snowballer::update (
    float dt ) [override], [virtual]
```

Update.

## Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Újraimplementált ősök: [pftd::Tower](#).

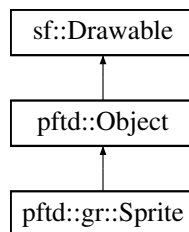
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- `include/objects/entities/towers/snowballer.hpp`

## 6.36. pftd::gr::Sprite osztályreferencia

```
#include <image.hpp>
```

A `pftd::gr::Sprite` osztály származási diagramja:



### Publikus tagfüggvények

- [Sprite](#) (`sf::Texture const &texture`, [utils::Vec2f](#) const &[position](#), [utils::Vec2f](#) const &[size](#), int [zIndex](#)==-1)
- [Sprite](#) (`sf::Texture const &texture`, `sf::IntRect const &textureRect`, [utils::Vec2f](#) const &[position](#), [utils::Vec2f](#) const &[size](#), int [zIndex](#)==-1)
- [Sprite](#) (`std::string const &imageSrc`, [utils::Vec2f](#) const &[position](#), [utils::Vec2f](#) const &[size](#), int [zIndex](#)==-1)
- [Sprite](#) ([Sprite](#) const &[other](#))
- [Sprite](#) ([Sprite](#) &&[other](#)) noexcept
- [~Sprite](#) ()=default
- void [modColor](#) (`sf::Color const &color`)  
*[Sprite](#) szín modulációjának beállítása.*
- void [flipY](#) ()  
*Y tengely mentén tükröz.*
- void [scale](#) ([utils::Vec2f](#) const &[factor](#))  
*Nyújt.*
- void [setSpriteRect](#) (`sf::IntRect const &textureRect`)  
*A textúra csak egy részletének használata Sprite-ként.*
- void [setPosition](#) ([utils::Vec2f](#) const &[newPos](#))  
*Beállítja az `m_sprite` pozícióját.*
- void [draw](#) (`sf::RenderTarget &target`, `sf::RenderStates states`) const override  
*Objektum megjelenítése.*

## Privát attribútumok

- sf::Texture const & [m\\_texture](#)
- sf::Sprite [m\\_sprite](#)

## További örökölt tagok

### 6.36.1. Részletes leírás

Saját sprite osztály. Használható kép GUI elemként.

### 6.36.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.36.2.1. Sprite() [1/5]

```
pftd::gr::Sprite::Sprite (  
    sf::Texture const & texture,  
    utils::Vec2f const & position,  
    utils::Vec2f const & size,  
    int zIndex = -1 )
```

#### 6.36.2.2. Sprite() [2/5]

```
pftd::gr::Sprite::Sprite (  
    sf::Texture const & texture,  
    sf::IntRect const & textureRect,  
    utils::Vec2f const & position,  
    utils::Vec2f const & size,  
    int zIndex = -1 )
```

#### 6.36.2.3. Sprite() [3/5]

```
pftd::gr::Sprite::Sprite (  
    std::string const & imageSrc,  
    utils::Vec2f const & position,  
    utils::Vec2f const & size,  
    int zIndex = -1 )
```

#### 6.36.2.4. Sprite() [4/5]

```
pftd::gr::Sprite::Sprite (
    Sprite const & other )
```

#### 6.36.2.5. Sprite() [5/5]

```
pftd::gr::Sprite::Sprite (
    Sprite && other ) [noexcept]
```

#### 6.36.2.6. ~Sprite()

```
pftd::gr::Sprite::~~Sprite ( ) [default]
```

### 6.36.3. Tagfüggvények dokumentációja

#### 6.36.3.1. draw()

```
void pftd::gr::Sprite::draw (
    sf::RenderTarget & target,
    sf::RenderStates states ) const [override], [virtual]
```

Objektum megjelenítése.

##### Paraméterek

<i>target</i>	Hol legyen megjelenítve? Ez általában egy ablak.
<i>states</i>	Render-eléshez szükséges egyéb állapotok.

Megvalósítja a következőket: [pftd::Object](#).

#### 6.36.3.2. flipY()

```
void pftd::gr::Sprite::flipY ( )
```

Y tengely mentén tükröz.

Nincs használva.



### 6.36.3.3. modColor()

```
void pftd::gr::Sprite::modColor (
    sf::Color const & color )
```

[Sprite](#) szín modulációjának beállítása.

Megjegyzés: Az SFML által biztosított modulációt használja.

#### Paraméterek

<i>color</i>	Szín, amivel szorzunk.
--------------	------------------------

### 6.36.3.4. scale()

```
void pftd::gr::Sprite::scale (
    utils::Vec2f const & factor )
```

Nyújt.

#### Paraméterek

<i>factor</i>	Ennyivel nyújt az X-Y tengelyek mentén.
---------------	---

### 6.36.3.5. setPosition()

```
void pftd::gr::Sprite::setPosition (
    utils::Vec2f const & newPos ) [inline]
```

Beállítja az `m_sprite` pozícióját.

#### Paraméterek

<i>newPos</i>	Az új pozíció.
---------------	----------------

### 6.36.3.6. setSpriteRect()

```
void pftd::gr::Sprite::setSpriteRect (
    sf::IntRect const & textureRect )
```

A textúra csak egy részletének használata Sprite-ként.

## Paraméterek

<code>textureRect</code>	A tárolt textúra része egy téglalappal megadva.
--------------------------	---

### 6.36.4. Adatmezők dokumentációja

#### 6.36.4.1. `m_sprite`

```
sf::Sprite pftd::gr::Sprite::m_sprite [private]
```

[Sprite](#).

#### 6.36.4.2. `m_texture`

```
sf::Texture const& pftd::gr::Sprite::m_texture [private]
```

Sprite-hoz tartozó textúra.

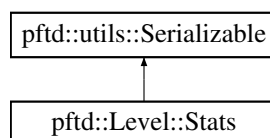
Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/gui/image.hpp](#)

## 6.37. `pftd::Level::Stats` struktúrareferencia

```
#include <level.hpp>
```

A `pftd::Level::Stats` osztály származási diagramja:



### Publikus tagfüggvények

- [Stats](#) ()=default
- [Stats](#) (int [maxHp](#), int currentHp, unsigned int [score](#), unsigned int wealth)
- void [serialize](#) (std::ostream &out) const override

*Kiír egy streamre.*

## Adatmezők

- int `maxHp` = 3
- int `hp` = `maxHp`
- unsigned int `score` = 0U
- unsigned int `money` = 100U

### 6.37.1. Részletes leírás

Játékállás / statisztika.

### 6.37.2. Konstruktorok és destruktorok dokumentációja

#### 6.37.2.1. Stats() [1/2]

```
pftd::Level::Stats::Stats ( ) [explicit], [default]
```

#### 6.37.2.2. Stats() [2/2]

```
pftd::Level::Stats::Stats (
    int maxHp,
    int currentHp,
    unsigned int score,
    unsigned int wealth ) [explicit]
```

### 6.37.3. Tagfüggvények dokumentációja

#### 6.37.3.1. serialize()

```
void pftd::Level::Stats::serialize (
    std::ostream & out ) const [override], [virtual]
```

Kiír egy streamre.

##### Paraméterek

<i>out</i>	A stream.
------------	-----------

Megvalósítja a következőket: [pftd::utils::Serializable](#).

### 6.37.4. Adatmezők dokumentációja

#### 6.37.4.1. hp

```
int pftd::Level::Stats::hp = maxHp
```

Jelenlegi HP.

#### 6.37.4.2. maxHp

```
int pftd::Level::Stats::maxHp = 3
```

A maximum HP.

#### 6.37.4.3. money

```
unsigned int pftd::Level::Stats::money = 100U
```

Pénz.

#### 6.37.4.4. score

```
unsigned int pftd::Level::Stats::score = 0U
```

Pontszám.

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- `include/game/level.hpp`

## 6.38. pftd::utils::parser::SaveFileParser::StatsInfo struktúrareferencia

```
#include <parsers.hpp>
```

### Adatmezők

- unsigned int `score`
- unsigned int `wealth`
- int `maxHp`
- int `hp`

### 6.38.1. Részletes leírás

Mentett játék betöltéséhez használt statisztika osztály.

### 6.38.2. Adatmezők dokumentációja

#### 6.38.2.1. hp

```
int pftd::utils::parser::SaveFileParser::StatsInfo::hp
```

#### 6.38.2.2. maxHp

```
int pftd::utils::parser::SaveFileParser::StatsInfo::maxHp
```

#### 6.38.2.3. score

```
unsigned int pftd::utils::parser::SaveFileParser::StatsInfo::score
```

#### 6.38.2.4. wealth

```
unsigned int pftd::utils::parser::SaveFileParser::StatsInfo::wealth
```

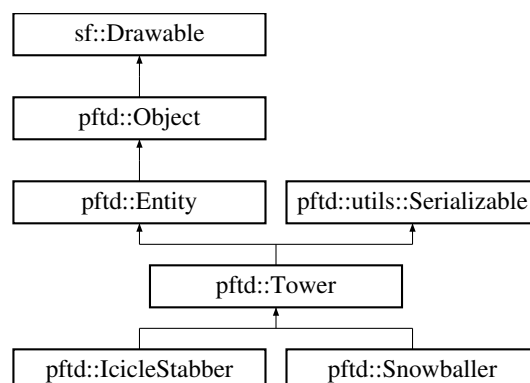
Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/utils/parsers.hpp](#)

## 6.39. pftd::Tower osztályreferencia

```
#include <tower_base.hpp>
```

A pftd::Tower osztály származási diagramja:



## Adatszerkezetek

- struct [TowerProperties](#)

## Publikus tagfüggvények

- [Tower](#) ([TowerProperties](#) const &props, std::string const &spriteSheetSrc, [utils::Vec2i](#) spriteSize, [utils::Vec2f](#) const &position, [utils::Vec2f](#) const &size, int zIndex=0)
- [Tower](#) ([TowerProperties](#) const &props, std::string const &spriteSrc, [utils::Vec2f](#) const &position, [utils::Vec2f](#) const &size, int zIndex=0)
- [Tower](#) ([Tower](#) const &other)
- virtual [~Tower](#) ()=default
- void [setProjSpawnCb](#) ([ProjSpawnFunc](#) callback)  
*Lövedék kilövéséhez használt callback függvény beállítása.*
- virtual [Tower](#) \* [clone](#) () const override=0  
*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*
- virtual void [attack](#) ()  
*Megtámadja a target-et, ha van.*
- virtual bool [lookForTarget](#) (std::vector< [Seal](#) \* > const &enemies)  
*Megfelelő célpontot keres.*
- virtual void [update](#) (float dt) override  
*Update.*
- void [serialize](#) (std::ostream &out) const override  
*Kiír egy streamre.*

## Adatmezők

- struct [pftd::Tower::TowerProperties](#) properties

## Védett tagfüggvények

- void [advanceAnimationFrame](#) () override  
*Animáció: képkocka léptetése.*

## Védett attribútumok

- [Seal](#) \* target = nullptr
- [ProjSpawnFunc](#) spawnProjectile
- float [attackTimerSec](#) = 0.0f

## Privát típusok

- using [ProjSpawnFunc](#) = std::function< void([Projectile](#) \*)>

### 6.39.1. Részletes leírás

Torony (pingvin).

## 6.39.2. Típusdefiníció-tagok dokumentációja

### 6.39.2.1. ProjSpawnFunc

```
using pftd::Tower::ProjSpawnFunc = std::function<void(Projectile*)> [private]
```

## 6.39.3. Konstruktorkok és destruktorkok dokumentációja

### 6.39.3.1. Tower() [1/3]

```
pftd::Tower::Tower (
    TowerProperties const & props,
    std::string const & spriteSheetSrc,
    utils::Vec2i spriteSize,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0 )
```

### 6.39.3.2. Tower() [2/3]

```
pftd::Tower::Tower (
    TowerProperties const & props,
    std::string const & spriteSrc,
    utils::Vec2f const & position,
    utils::Vec2f const & size,
    int zIndex = 0 )
```

### 6.39.3.3. Tower() [3/3]

```
pftd::Tower::Tower (
    Tower const & other )
```

### 6.39.3.4. ~Tower()

```
virtual pftd::Tower::~~Tower ( ) [virtual], [default]
```

## 6.39.4. Tagfüggvények dokumentációja

### 6.39.4.1. advanceAnimationFrame()

```
void pftd::Tower::advanceAnimationFrame ( ) [override], [protected], [virtual]
```

Animáció: képkocka léptetése.

Újrimplementált ősök: [pftd::Entity](#).

### 6.39.4.2. attack()

```
virtual void pftd::Tower::attack ( ) [virtual]
```

Megtámadja a `target`-et, ha van.

Újrimplementáló leszármazottak: [pftd::Snowballer](#).

### 6.39.4.3. clone()

```
virtual Tower * pftd::Tower::clone ( ) const [override], [pure virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

#### Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Entity](#).

Megvalósítják a következők: [pftd::IcicleStabber](#) és [pftd::Snowballer](#).

### 6.39.4.4. lookForTarget()

```
virtual bool pftd::Tower::lookForTarget (
    std::vector< Seal * > const & enemies ) [virtual]
```

Megfelelő célpontot keres.

Megjegyzés: csak akkor keres, ha a `target` egy nullptr.



## Paraméterek

<i>enemies</i>	Ellenfelek, amik közt célpontot keres.
----------------	--

**6.39.4.5. serialize()**

```
void pftd::Tower::serialize (
    std::ostream & out ) const [override], [virtual]
```

Kiír egy streamre.

## Paraméterek

<i>out</i>	A stream.
------------	-----------

Megvalósítja a következőket: [pftd::utils::Serializable](#).

**6.39.4.6. setProjSpawnCb()**

```
void pftd::Tower::setProjSpawnCb (
    ProjSpawnFunc callback )
```

Lövedék kilövéséhez használt callback függvény beállítása.

## Paraméterek

<i>callback</i>	A függvény.
-----------------	-------------

**6.39.4.7. update()**

```
virtual void pftd::Tower::update (
    float dt ) [override], [virtual]
```

Update.

## Paraméterek

<i>dt</i>	Delta idő.
-----------	------------

Újrimplementált ősök: [pftd::Entity](#).

Újrimplementáló leszármazottak: [pftd::IcicleStabber](#) és [pftd::Snowballer](#).

### 6.39.5. Adatmezők dokumentációja

#### 6.39.5.1. attackTimerSec

```
float pftd::Tower::attackTimerSec = 0.0f [protected]
```

Támadások közti idő mérésére szolgáló számláló.

#### 6.39.5.2. properties

```
struct pftd::Tower::TowerProperties pftd::Tower::properties
```

#### 6.39.5.3. spawnProjectile

```
ProjSpawnFunc pftd::Tower::spawnProjectile [protected]
```

Callback függvény, ha a torony távolsági (valamit kilő).

#### 6.39.5.4. target

```
Seal* pftd::Tower::target = nullptr [protected]
```

Célpont.

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/towers/tower\\_base.hpp](#)

## 6.40. pftd::Tower::TowerProperties struktúrareferencia

```
#include <tower_base.hpp>
```

### Publikus tagfüggvények

- [TowerProperties](#) ([TowerID](#) id, float radius=120.0f, float attackRange=100.0f, float attackSpeed=1.0f, unsigned int [attackDamage](#)=1U, bool instant=false, unsigned int [price](#)=0U)

## Adatmezők

- TowerID id
- float radiusPixel = 120.0f
- float attackRangePixel = 100.0f
- float attackSpeedSec = 1.0f
- unsigned int attackDamage = 1U
- bool instantAttack = false
- unsigned int price = 0U

### 6.40.1. Konstruktorok és destruktorok dokumentációja

#### 6.40.1.1. TowerProperties()

```
pftd::Tower::TowerProperties::TowerProperties (
    TowerID id,
    float radius = 120.0f,
    float attackRange = 100.0f,
    float attackSpeed = 1.0f,
    unsigned int attackDamage = 1U,
    bool instant = false,
    unsigned int price = 0U )
```

### 6.40.2. Adatmezők dokumentációja

#### 6.40.2.1. attackDamage

```
unsigned int pftd::Tower::TowerProperties::attackDamage = 1U
```

Támadás ereje: mennyi HP-t visz le.

#### 6.40.2.2. attackRangePixel

```
float pftd::Tower::TowerProperties::attackRangePixel = 100.0f
```

Az a környezete, amiben célpontot keres.

#### 6.40.2.3. attackSpeedSec

```
float pftd::Tower::TowerProperties::attackSpeedSec = 1.0f
```

Támadás gyorsasága.

#### 6.40.2.4. id

```
TowerID pftd::Tower::TowerProperties::id
```

#### 6.40.2.5. instantAttack

```
bool pftd::Tower::TowerProperties::instantAttack = false
```

Instant támadó (true)? Vagy lövedékkel (false)?

#### 6.40.2.6. price

```
unsigned int pftd::Tower::TowerProperties::price = 0U
```

Az ára, amikor az "eszköztárban" van.

#### 6.40.2.7. radiusPixel

```
float pftd::Tower::TowerProperties::radiusPixel = 120.0f
```

Az a környezete, amibe másik tornyot nem lehet lehelyezni.

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

- [include/objects/entities/towers/tower\\_base.hpp](#)

### 6.41. pftd::utils::Vec2< T > struktúrasablon-referencia

```
#include <substitute_types.hpp>
```

#### Publikus tagfüggvények

- [Vec2](#) ()
- [Vec2](#) (T x, T y)
- [Vec2](#) ([Vec2](#) const &right)
- [Vec2](#) [normalize](#) () const  
*Vektor normalizálása.*
- [Vec2](#) [operator+](#) ([Vec2](#) const &right) const
- [Vec2](#) [operator-](#) ([Vec2](#) const &right) const
- [Vec2](#) [operator/](#) (T scalar) const
- [Vec2](#) [operator\\*](#) (T scalar) const
- bool [operator==](#) ([Vec2](#) other) const
- bool [operator!=](#) ([Vec2](#) other) const
- [Vec2](#) & [operator=](#) ([Vec2](#) const &right)

## Statikus publikus tagfüggvények

- static float `distance` (`Vec2` const &v1, `Vec2` const &v2)  
*Két vektor végpontja közti euklideszi távolság.*

## Adatmezők

- T `x`
- T `y`

## Barátok

- std::ostream & `operator<<` (std::ostream &out, `Vec2` const &vec)
- std::istream & `operator>>` (std::istream &in, `Vec2` &vec)

### 6.41.1. Részletes leírás

```
template<typename T>
struct pftd::utils::Vec2< T >
```

sf::Vector2 helyett használt matematikai vektor implementáció.

### 6.41.2. Konstruktorkok és destruktorkok dokumentációja

#### 6.41.2.1. Vec2() [1/3]

```
template<typename T >
pftd::utils::Vec2< T >::Vec2 ( ) [inline]
```

#### 6.41.2.2. Vec2() [2/3]

```
template<typename T >
pftd::utils::Vec2< T >::Vec2 (
    T x,
    T y ) [inline]
```

#### 6.41.2.3. Vec2() [3/3]

```
template<typename T >
pftd::utils::Vec2< T >::Vec2 (
    Vec2< T > const & right ) [inline]
```

### 6.41.3. Tagfüggvények dokumentációja

#### 6.41.3.1. distance()

```
template<typename T >
static float pftd::utils::Vec2< T >::distance (
    Vec2< T > const & v1,
    Vec2< T > const & v2 ) [inline], [static]
```

Két vektor végpontja közti euklideszi távolság.

##### Paraméterek

v1	Egyik vektor.
v2	Másik vektor.

##### Visszatérési érték

A távolság.

#### 6.41.3.2. normalize()

```
template<typename T >
Vec2 pftd::utils::Vec2< T >::normalize ( ) const [inline]
```

Vektor normalizálása.

Ez a metódus nem módosítja a Vec2-t, hanem egy újat hoz létre belőle.

##### Visszatérési érték

Egy új, normalizált Vec2.

#### 6.41.3.3. operator"!=()"

```
template<typename T >
bool pftd::utils::Vec2< T >::operator!= (
    Vec2< T > other ) const [inline]
```

#### 6.41.3.4. operator\*()

```
template<typename T >
Vec2 pftd::utils::Vec2< T >::operator* (
    T scalar ) const [inline]
```

#### 6.41.3.5. operator+()

```
template<typename T >
Vec2 pftd::utils::Vec2< T >::operator+ (
    Vec2< T > const & right ) const [inline]
```

#### 6.41.3.6. operator-()

```
template<typename T >
Vec2 pftd::utils::Vec2< T >::operator- (
    Vec2< T > const & right ) const [inline]
```

#### 6.41.3.7. operator/()

```
template<typename T >
Vec2 pftd::utils::Vec2< T >::operator/ (
    T scalar ) const [inline]
```

#### 6.41.3.8. operator=()

```
template<typename T >
Vec2 & pftd::utils::Vec2< T >::operator= (
    Vec2< T > const & right ) [inline]
```

#### 6.41.3.9. operator==( )

```
template<typename T >
bool pftd::utils::Vec2< T >::operator== (
    Vec2< T > other ) const [inline]
```

## 6.41.4. Barát és kapcsolódó függvények dokumentációja

### 6.41.4.1. operator<<

```
template<typename T >
std::ostream & operator<< (
    std::ostream & out,
    Vec2< T > const & vec ) [friend]
```

### 6.41.4.2. operator>>

```
template<typename T >
std::istream & operator>> (
    std::istream & in,
    Vec2< T > & vec ) [friend]
```

## 6.41.5. Adatmezők dokumentációja

### 6.41.5.1. x

```
template<typename T >
T pftd::utils::Vec2< T >::x
```

X koordináta.

### 6.41.5.2. y

```
template<typename T >
T pftd::utils::Vec2< T >::y
```

Y koordináta.

Ez a dokumentáció a struktúráról a következő fájl alapján készült:

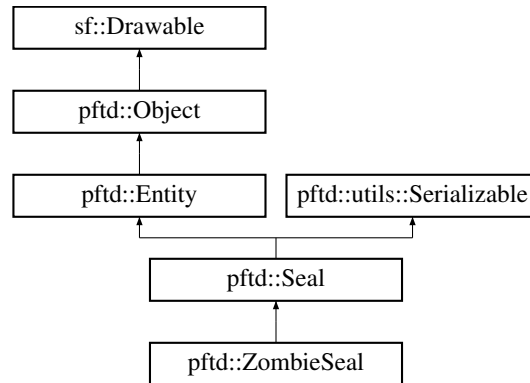
- [include/utils/substitute\\_types.hpp](#)



## 6.42. pftd::ZombieSeal osztályreferencia

```
#include <zombie.hpp>
```

A pftd::ZombieSeal osztály származási diagramja:



### Publikus tagfüggvények

- [ZombieSeal](#) ([FollowPath](#) const &followpath)
- [Seal](#) \* [clone](#) () const override

*Egy dinamikus memóriában foglalt másolatot készít az entitásról.*

### További örökölt tagok

#### 6.42.1. Részletes leírás

Zombi fóka.

#### 6.42.2. Konstruktorkok és destruktorkok dokumentációja

##### 6.42.2.1. ZombieSeal()

```
pftd::ZombieSeal::ZombieSeal (
    FollowPath const & followpath )
```

##### Paraméterek

<i>followpath</i>	Az útvonal amin végig kell menjen (majd pedig vissza).
-------------------	--

### 6.42.3. Tagfüggvények dokumentációja

#### 6.42.3.1. clone()

```
Seal * pftd::ZombieSeal::clone ( ) const [override], [virtual]
```

Egy dinamikus memóriában foglalt másolatot készít az entitásról.

##### Visszatérési érték

A másolat.

Megvalósítja a következőket: [pftd::Seal](#).

Ez a dokumentáció az osztályról a következő fájl alapján készült:

- [include/objects/entities/seals/zombie.hpp](#)

## 7. fejezet

# Fájlok dokumentációja

### 7.1. include/all.hpp fájlreferencia

```
#include <cmath>
#include <vector>
#include <functional>
#include <optional>
#include <utility>
#include <unordered_map>
#include <map>
#include <queue>
#include <algorithm>
#include <ctime>
#include <ios>
#include <limits>
#include <stdexcept>
#include <random>
#include <fstream>
#include "SFML/Graphics.hpp"
#include "SFML/Audio.hpp"
#include "utils/logger.hpp"
#include "utils/substitute_types.hpp"
```

### 7.2. all.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 // Ez egy precompiled header fájl.
4
5 // STL
6 #include <cmath>
7 #include <vector>
8 #include <functional>
9 #include <optional>
10 #include <utility>
11 #include <unordered_map>
12 #include <map>
13 #include <queue>
14 #include <algorithm>
15 #include <ctime>
16 #include <ios>
```

```
17 #include <limits>
18 #include <stdexcept>
19 #include <random>
20 #include <fstream>
21
22 // SFML
23 #if not defined(_PFTD_TEST) && not defined(CPORTA)
24 #include "SFML/Graphics.hpp"
25 #include "SFML/Audio.hpp"
26 #endif
27
28 // Own
29 #include "utils/logger.hpp"
30 #include "utils/substitute_types.hpp"
```

## 7.3. include/app.hpp fájlreferencia

```
#include "objects/object_base.hpp"
#include "scene.hpp"
```

### Adatszerkezetek

- class `pftd::Renderer`
- class `pftd::App`
- struct `pftd::SceneError`

### Névterek

- namespace `pftd`

## 7.4. app.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/object_base.hpp"
6 #include "scene.hpp"
7
8 namespace pftd {
9
10 class Renderer final
11 {
12     friend class App;
13 public:
14     Renderer(unsigned int width, unsigned int height, std::string const& windowTitle);
15
16     Renderer(Renderer const&) = delete;
17     Renderer(Renderer&&) = delete;
18
19     ~Renderer();
20
21     void render();
22
23     void clear() const { m_window->clear(); }
24
25     void display() const { m_window->display(); }
26
27     void pushQueue(Object const* o) { m_queue.push(o); }
28
29     sf::RenderWindow* getWindow() const { return m_window; }
30
31 private:
```

```

63     sf::RenderWindow* m_window = nullptr;
64
66     unsigned int m_width;
67
69     unsigned int m_height;
70
72     std::priority_queue<Object const*, std::vector<Object const*>, Object::Compare> m_queue;
73
74 };
75
77 class App final
78 {
79 public:
80     App(App const&) = delete;
81     App(App&&) = delete;
82
88     ~App();
89
93     void run();
94
105     void addScene(std::string id, Scene* scene, bool active = false);
106
117     bool changeScene(std::string id, Scene::StateFlag flag = Scene::StateFlag::NONE);
118
124     auto isRunning()const { return this->m_running; }
125
131     int getWindowWidth()const { return this->m_renderer->m_width; }
132
138     int getWindowHeight()const { return this->m_renderer->m_height; }
139
140     App& operator=(App const&) = delete;
141
147     static App* getInstance() { return m_instance; }
148
159     static App* create(unsigned int width, unsigned int height, std::string const& windowTitle);
160
164     static void destroy() { delete m_instance; }
165
166 private:
167     // Az 'm_instance' létrehozásához van használva.
168     App() = default;
169
171     static App* m_instance;
172
174     bool m_running = false;
175
177     Renderer* m_renderer = nullptr;
178
180     std::map<std::string, Scene*> m_scenes;
181
183     std::string m_activeSceneID;
184
185 };
186
188 struct SceneError : public std::runtime_error
189 {
190     using std::runtime_error::runtime_error;
191 };
192
193 }
194
195 #endif

```

## 7.5. include/game/level.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/entities/projectiles/projectile_base.hpp"
#include "objects/entities/seals/seal_base.hpp"
#include "objects/entities/towers/tower_base.hpp"
#include "utils/serializable.hpp"
#include "utils/parsers.hpp"

```

### Adatszerkezetek

- class `pftd::Level`

- struct [pftd::Level::Nest](#)
- struct [pftd::Level::Stats](#)

## Névterek

- namespace [pftd](#)

## 7.6. level.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/entities/projectiles/projectile_base.hpp"
8 #include "objects/entities/seals/seal_base.hpp"
9 #include "objects/entities/towers/tower_base.hpp"
10 #include "utils/serializable.hpp"
11 #include "utils/parsers.hpp"
12
13 namespace pftd {
14
15 class Level : public Object
16 {
17 public:
18     struct Nest : public Entity
19     {
20         int const radiusPixel = 110;
21
22         explicit Nest(utils::Vec2f const& position);
23         Nest(Nest const& other) = default;
24
25         ~Nest() = default;
26
27         Nest* clone() const override;
28     };
29
30     struct Stats : public utils::Serializable
31     {
32         int maxHp = 3;
33
34         int hp = maxHp;
35
36         unsigned int score = 0U;
37
38         unsigned int money = 100U;
39
40         explicit Stats() = default;
41         explicit Stats(int maxHp, int currentHp, unsigned int score, unsigned int wealth);
42
43         void serialize(std::ostream& out) const override;
44     };
45
46     Stats stats = Stats{};
47
48     Tower* selectedTower = nullptr;
49
50     Level(std::string const& saveFile, Stats stats);
51
52     Level(std::string const& saveFile);
53
54     virtual ~Level();
55
56     virtual void loseHP(int hpLost = 1);
57
58     virtual bool placeTower();
59
60     void deselectTower();
61
62     void selectTower(Tower* newTower);
63
64     void update(float dt);
65
66     bool isGameOver()const { return stats.hp <= 0; };

```

```

125
131     void save() const;
132
138     void reset(Stats stats);
139
140     virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
141
142 protected:
144     Nest* nest = nullptr;
145
147     FollowPath followPath;
148
150     std::vector<Tower*> towers;
151
153     std::vector<Seal*> seals;
154
156     std::vector<Projectile*> projectiles;
157
159     utils::parser::LevelConfigParser config;
160
162     std::string saveFile;
163
167     void spawnSeal();
168
169 private:
171     float m_accuTimeSpawnSec = 0.0f;
172
176     void _spawnSeal(float dt);
177
181     void _updateSeals(float dt);
182
186     void _updateTowers(float dt);
187
191     void _updateProjectiles(float dt);
192
193 };
194
195 }
196
197 #endif

```

## 7.7. include/objects/clickable.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/object_base.hpp"

```

### Adatszerkezetek

- class [pftd::Clickable](#)

### Névterek

- namespace [pftd](#)

## 7.8. clickable.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/object_base.hpp"
8

```

```

9 namespace pftd {
10
12 class Clickable : public Object
13 {
14 public:
15     bool isActive;
16
17     Clickable(bool active = true);
18     Clickable(utils::Vec2f const& position, utils::Vec2f const& size, int zIndex = 0, bool active =
19         true);
20
26     virtual void setCallback(std::function<void()> callback);
27
33     virtual void handleClick(utils::Vec2i const& clickCoords);
34
35 protected:
36     std::function<void()> m_callback;
37
38
39 };
40
41 }
42
43 #endif

```

## 7.9. include/objects/entities/all\_entities.hpp fájlreferencia

```

#include "objects/entities/entity_base.hpp"
#include "objects/entities/projectiles/projectile_base.hpp"
#include "objects/entities/projectiles/snowball.hpp"
#include "objects/entities/seals/seal_base.hpp"
#include "objects/entities/seals/cub.hpp"
#include "objects/entities/seals/fortified_zombie_cub.hpp"
#include "objects/entities/seals/regular.hpp"
#include "objects/entities/seals/zombie.hpp"
#include "objects/entities/towers/tower_base.hpp"
#include "objects/entities/towers/iciclestabber.hpp"
#include "objects/entities/towers/snowballer.hpp"

```

## 7.10. all\_entities.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #include "objects/entities/entity_base.hpp"
4
5 #include "objects/entities/projectiles/projectile_base.hpp"
6 #include "objects/entities/projectiles/snowball.hpp"
7
8 #include "objects/entities/seals/seal_base.hpp"
9 #include "objects/entities/seals/cub.hpp"
10 #include "objects/entities/seals/fortified_zombie_cub.hpp"
11 #include "objects/entities/seals/regular.hpp"
12 #include "objects/entities/seals/zombie.hpp"
13
14 #include "objects/entities/towers/tower_base.hpp"
15 #include "objects/entities/towers/iciclestabber.hpp"
16 #include "objects/entities/towers/snowballer.hpp"

```

## 7.11. include/objects/entities/entity\_base.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/gui/image.hpp"
#include "objects/object_base.hpp"

```



## Adatszerkezetek

- class `pftd::Entity`

## Névterek

- namespace `pftd`

## 7.12. entity\_base.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/gui/image.hpp"
8 #include "objects/object_base.hpp"
9
10 namespace pftd {
11
12     class Entity : public Object
13     {
14     public:
15         bool isAnimated = true;
16
17         Entity(std::string const& spriteSheetSrc, utils::Vec2i spriteSize, utils::Vec2f const& position,
18             utils::Vec2f const& size, int zIndex = 0);
19         Entity(sf::Texture const& texture, utils::Vec2i spriteSize, utils::Vec2f const& position,
20             utils::Vec2f const& size, int zIndex = 0);
21         Entity(std::string const& spriteSrc, utils::Vec2f const& position, utils::Vec2f const& size, int
22             zIndex = 0);
23         Entity(Entity const& other);
24         virtual ~Entity();
25
26         virtual Entity* clone() const = 0;
27
28         virtual void update(float dt);
29
30         virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
31
32         void setPosition(utils::Vec2f position);
33
34         auto getPosition()const { return position; }
35
36         gr::Sprite const* getSprite()const { return currentSprite; }
37
38         gr::Sprite* getSprite() { return currentSprite; }
39
40         auto& getSpriteSheet()const { return spriteSheet; }
41
42     protected:
43         sf::Texture const& spriteSheet; // Feltételezzük, hogy egy sorban vannak a sprite-ok
44
45         utils::Vec2i cellSize;
46
47         size_t const CELL_N;
48
49         size_t currentCell = 0U;
50
51         gr::Sprite* currentSprite = nullptr;
52
53         float frameDurationSec = 1.0f;
54
55         float totalElapsedSec = 0.0f; // Animációhoz van használva elsősorban.
56
57         virtual void advanceAnimationFrame();
58         virtual void resetAnimation();
59
60     };
61
62 }
63 #endif

```

## 7.13. include/objects/entities/projectiles/projectile\_base.hpp fájlreferencia

```
#include "objects/entities/entity_base.hpp"
#include "utils/serializable.hpp"
#include "utils/substitute_types.hpp"
```

### Adatszerkezetek

- class `pftd::Projectile`

### Névterek

- namespace `pftd`

### Enumerációk

- enum class `pftd::ProjectileID` { `pftd::SNOWBALL` = 0 }

## 7.14. projectile\_base.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/entity_base.hpp"
6 #include "utils/serializable.hpp"
7 #include "utils/substitute_types.hpp"
8
9 namespace pftd {
10
11 enum class ProjectileID
12 {
13     SNOWBALL = 0,
14 };
15
16 class Projectile : public Entity, public utils::Serializable
17 {
18 public:
19     utils::Vec2f direction;
20     float linearSpeed;
21     float angularVelocityRadPerSec;
22     virtual ~Projectile() = default;
23     virtual void update(float dt) override;
24     void serialize(std::ostream& out) const override;
25
26 protected:
27     ProjectileID id;
28
29     Projectile(std::string const& spriteSrc, utils::Vec2f const& position,
30         utils::Vec2f const& size, utils::Vec2f const& direction, float speed, float angularSpeed = 0.0f,
31         int zIndex = 0);
32
33 };
34
35 #endif
```

## 7.15. include/objects/entities/projectiles/snowball.hpp fájlreferencia

```
#include "objects/entities/projectiles/projectile_base.hpp"
```

### Adatszerkezetek

- class `pftd::Snowball`

### Névterek

- namespace `pftd`

## 7.16. snowball.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/projectiles/projectile_base.hpp"
6
7 namespace pftd {
8
9     class Snowball : public Projectile
10     {
11     public:
12         Snowball(utils::Vec2f const& position, utils::Vec2f direction, float speed = 50.0f);
13         Snowball* clone() const override;
14     };
15
16 }
17
18 #endif
```

## 7.17. include/objects/entities/seals/cub.hpp fájlreferencia

```
#include "objects/entities/seals/seal_base.hpp"
```

### Adatszerkezetek

- class `pftd::Cub`

### Névterek

- namespace `pftd`

## 7.18. cub.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/seals/seal_base.hpp"
6
7 namespace pftd {
8
9     class Cub : public Seal
10     {
11     public:
12         Cub(FollowPath const& followpath);
13         Seal* clone() const override;
14     };
15 }
16 #endif
```

## 7.19. include/objects/entities/seals/followpath.hpp fájlreferencia

```
#include "utils/hetero_collection.hpp"
```

### Adatszerkezetek

- struct [pftd::FollowPath](#)

### Névterek

- namespace [pftd](#)

### Típusdefiníciók

- using [pftd::EndPoint](#) = [utils::Vec2f](#)

## 7.20. followpath.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #include "utils/hetero_collection.hpp"
4
5 namespace pftd {
6
7     using EndPoint = utils::Vec2f;
8
9     struct FollowPath : public utils::Container<EndPoint>
10     {
11     public:
12         FollowPath() = default;
13     };
14 }
15 }
```

## 7.21. include/objects/entities/seals/fortified\_zombie\_cub.hpp fájlreferencia

```
#include "objects/entities/seals/seal_base.hpp"
```

### Adatszerkezetek

- class [pftd::FZC](#)

### Névterek

- namespace [pftd](#)

## 7.22. fortified\_zombie\_cub.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/seals/seal_base.hpp"
6
7 namespace pftd {
8
9     class FZC : public Seal
10     {
11     public:
12         FZC(FollowPath const& followpath);
13
14         Seal* clone() const override;
15
16     };
17
18 }
19
20 #endif
```

## 7.23. include/objects/entities/seals/regular.hpp fájlreferencia

```
#include "objects/entities/seals/seal_base.hpp"
```

### Adatszerkezetek

- class [pftd::RegularSeal](#)

### Névterek

- namespace [pftd](#)

## 7.24. regular.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/seals/seal_base.hpp"
6
7 namespace pftd {
8
9 class RegularSeal : public Seal
10 {
11 public:
12     RegularSeal(FollowPath const& followpath);
13
14     Seal* clone() const override;
15
16 };
17
18 }
19
20 #endif
```

## 7.25. include/objects/entities/seals/seal\_base.hpp fájlreferencia

```
#include "objects/entities/entity_base.hpp"
#include "objects/entities/seals/followpath.hpp"
#include "utils/serializable.hpp"
#include "utils/substitute_types.hpp"
```

### Adatszerkezetek

- class [pftd::Seal](#)

### Névterek

- namespace [pftd](#)

### Enumerációk

- enum class [pftd::SealID](#) { [pftd::REGULAR](#) = 0 , [pftd::CUB](#) , [pftd::ZOMBIE](#) , [pftd::FZC](#) }

## 7.26. seal\_base.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/entity_base.hpp"
6 #include "objects/entities/seals/followpath.hpp"
7 #include "utils/serializable.hpp"
8 #include "utils/substitute_types.hpp"
9
10 namespace pftd {
11
12 enum class SealID
```

```

13 {
14     REGULAR = 0,
15     CUB,
16     ZOMBIE,
17     FZC
18 };
19
20
21 class Seal : public Entity, public utils::Serializable
22 {
23 public:
24     bool isCurrentlyStealing = false;
25
26
27     int hp;
28
29     float speed;
30
31     unsigned int value;
32
33     virtual ~Seal() = default;
34
35     void lerpPath();
36
37     virtual Seal* clone() const override = 0;
38
39     void update(float dt) override;
40
41     bool hasCompletedPath()const { return returned; }
42
43     bool hasReachedNest()const { return reachedNest; }
44
45     void setLerpState(float param, bool backwards = false)
46     {
47         lerpParam = std::min(1.0f, std::max(param, 0.0f));
48         reachedNest = backwards;
49     }
50
51     void damage(int hpLost = 1);
52
53     void serialize(std::ostream& out) const override;
54
55 protected:
56     SealID id;
57
58     Seal(FollowPath const& followPath, std::string const& spriteSrc, utils::Vec2f const& size, int hp,
59         float speed, unsigned int value, int zIndex = 0);
60
61     FollowPath const& followPath;
62
63     // Interpolációhoz
64
65     float lerpParam = 0.0f;
66
67     bool reachedNest = false;
68
69     bool returned = false;
70
71     Endpoint nextPoint;
72
73 private:
74     void advanceAnimationFrame() override;
75
76 };
77
78 }
79
80 #endif

```

## 7.27. include/objects/entities/seals/zombie.hpp fájlreferencia

```
#include "objects/entities/seals/seal_base.hpp"
```

### Adatszerkezetek

- class `pftd::ZombieSeal`

## Névterek

- namespace [pftd](#)

## 7.28. zombie.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/seals/seal_base.hpp"
6
7 namespace pftd {
8
9     class ZombieSeal : public Seal
10     {
11     public:
12         ZombieSeal(FollowPath const& followpath);
13         Seal* clone() const override;
14     };
15 }
16 #endif
```

## 7.29. include/objects/entities/towers/iciclestabber.hpp fájlreferencia

```
#include "objects/entities/towers/tower_base.hpp"
```

## Adatszerkezetek

- class [pftd::IcicleStabber](#)

## Névterek

- namespace [pftd](#)

## 7.30. iciclestabber.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/towers/tower_base.hpp"
6
7 namespace pftd {
8
9     class IcicleStabber final : public Tower
10     {
11     public:
12         IcicleStabber(utils::Vec2f const& position, int zIndex = 0);
13         IcicleStabber();
14         ~IcicleStabber() = default;
15         Tower* clone() const override;
16         void update(float dt) override;
17     };
18 }
19 #endif
```



## 7.31. include/objects/entities/towers/snowballer.hpp fájlreferencia

```
#include "objects/entities/towers/tower_base.hpp"
```

### Adatszerkezetek

- class [pftd::Snowballer](#)

### Névterek

- namespace [pftd](#)

## 7.32. snowballer.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/towers/tower_base.hpp"
6
7 namespace pftd {
8
9     class Snowballer final : public Tower
10     {
11     public:
12         Snowballer(utils::Vec2f const& position, int zIndex = 0);
13         Snowballer();
14         ~Snowballer() = default;
15
16         Tower* clone() const override;
17
18         void update(float dt) override;
19
20         void attack() override;
21
22     };
23
24 }
25
26 #endif
```

## 7.33. include/objects/entities/towers/tower\_base.hpp fájlreferencia

```
#include "objects/entities/entity_base.hpp"
#include "objects/entities/projectiles/projectile_base.hpp"
#include "objects/entities/seals/seal_base.hpp"
#include "utils/serializable.hpp"
```

### Adatszerkezetek

- class [pftd::Tower](#)
- struct [pftd::Tower::TowerProperties](#)

## Névterek

- namespace `pftd`

## Enumerációk

- enum class `pftd::TowerID` { `pftd::SNOWBALLER` = 0 , `pftd::ICICLE_STABBER` }

## 7.34. tower\_base.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/entities/entity_base.hpp"
6 #include "objects/entities/projectiles/projectile_base.hpp"
7 #include "objects/entities/seals/seal_base.hpp"
8 #include "utils/serializable.hpp"
9
10 namespace pftd {
11
12 enum class TowerID
13 {
14     SNOWBALLER = 0,
15     ICICLE_STABBER
16 };
17
18 class Tower : public Entity, public utils::Serializable
19 {
20 public:
21     using ProjSpawnFunc = std::function<void(Projectile*)>;
22     struct TowerProperties
23     {
24         TowerID id;
25
26         float radiusPixel = 120.0f;
27
28         float attackRangePixel = 100.0f;
29
30         float attackSpeedSec = 1.0f;
31
32         unsigned int attackDamage = 1U;
33
34         bool instantAttack = false;
35
36         unsigned int price = 0U;
37
38         TowerProperties(TowerID id, float radius = 120.0f, float attackRange = 100.0f, float attackSpeed
39 = 1.0f,
40         unsigned int attackDamage = 1U, bool instant = false, unsigned int price = 0U);
41     } properties;
42
43     Tower(TowerProperties const& props, std::string const& spriteSheetSrc, utils::Vec2i spriteSize,
44     utils::Vec2f const& position, utils::Vec2f const& size, int zIndex = 0);
45     Tower(TowerProperties const& props, std::string const& spriteSrc, utils::Vec2f const& position,
46     utils::Vec2f const& size, int zIndex = 0);
47     Tower(Tower const& other);
48     virtual ~Tower() = default;
49
50     void setProjSpawnCb(ProjSpawnFunc callback);
51
52     virtual Tower* clone() const override = 0;
53
54     virtual void attack();
55
56     virtual bool lookForTarget(std::vector<Seal*> const& enemies);
57
58     virtual void update(float dt) override;
59
60     void serialize(std::ostream& out) const override;
61
62 protected:
63     Seal* target = nullptr;
64
65 
```

```

87     ProjSpawnFunc spawnProjectile;
88
89     float attackTimerSec = 0.0f;
90
91     void advanceAnimationFrame() override;
92 };
93
94
95 }
96
97 #endif

```

## 7.35. include/objects/gui/button.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/clickable.hpp"
#include "objects/gui/image.hpp"
#include "objects/gui/label.hpp"

```

### Adatszerkezetek

- class [pftd::gr::Button](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::gr](#)

## 7.36. button.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/clickable.hpp"
8 #include "objects/gui/image.hpp"
9 #include "objects/gui/label.hpp"
10
11 namespace pftd {
12 namespace gr {
13
14 class Button : public Clickable
15 {
16 public:
17     Button(gr::Label const& label, utils::Vec2f const& position, utils::Vec2f const& size, bool active =
18         true, int zIndex = 0);
19     virtual ~Button();
20
21     void setSound(std::string const& src);
22
23     void setBackground(std::string const& src);
24
25     std::string getLabel() const { return static_cast<std::string>(label.getText().getString()); };
26
27     virtual void handleClick(utils::Vec2i const& clickCoords) override;
28
29     virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
30
31 protected:
32     Label label;
33
34 }
35
36 }
37
38 #endif

```

```

59 private:
60     sf::FloatRect m_rect;
61
62     Sprite* m_background = nullptr;
63
64     sf::Sound m_clickSound;
65
66 };
67
68 }
69
70 #endif

```

## 7.37. include/objects/gui/image.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/object_base.hpp"

```

### Adatszerkezetek

- class [pftd::gr::Sprite](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::gr](#)

## 7.38. image.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/object_base.hpp"
8
9 namespace pftd {
10 namespace gr {
11
12 class Sprite final : public Object
13 {
14 public:
15     Sprite(sf::Texture const& texture, utils::Vec2f const& position, utils::Vec2f const& size, int zIndex
        = -1);
16     Sprite(sf::Texture const& texture, sf::IntRect const& textureRect, utils::Vec2f const& position,
        utils::Vec2f const& size, int zIndex = -1);
17     Sprite(std::string const& imageSrc, utils::Vec2f const& position, utils::Vec2f const& size, int
        zIndex = -1);
18
19     Sprite(Sprite const& other);
20     Sprite(Sprite&& other) noexcept;
21
22     ~Sprite() = default;
23
24     void modColor(sf::Color const& color);
25
26     void flipY();
27
28     void scale(utils::Vec2f const& factor);
29
30     void setSpriteRect(sf::IntRect const& textureRect);
31
32

```

```

54
60     void setPosition(utils::Vec2f const& newPos) { m_sprite.setPosition({newPos.x, newPos.y}); }
61
62     void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
63
64 private:
65     sf::Texture const& m_texture;
66
67     sf::Sprite m_sprite;
68
69 };
70
71 };
72
73 }
74 }
75
76 #endif

```

## 7.39. include/objects/gui/label.hpp fájlreferencia

```

#include "all.hpp"
#include "objects/object_base.hpp"

```

### Adatszerkezetek

- class [pftd::gr::Label](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::gr](#)

## 7.40. label.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6
7 #include "objects/object_base.hpp"
8
9 namespace pftd {
10 namespace gr {
11
12 class Label final : public Object
13 {
14 public:
15     Label(std::wstring const& label, sf::Font const& font, unsigned int cSize, sf::Color const& color =
16         sf::Color::White);
17
18     Label(std::wstring const& label, sf::Font const& font, unsigned int cSize, utils::Vec2f position, int
19         zIndex = 0, sf::Color const& color = sf::Color::White);
20
21     void setLabel(std::string const& newLabel) { m_text.setString(newLabel); }
22
23     void setOutline(sf::Color color, float thickness = 1.0f) { m_text.setOutlineColor(color);
24         m_text.setOutlineThickness(thickness); }
25
26     sf::Text& getText() { return m_text; }
27
28     sf::Text const& getText() const { return m_text; }
29
30     void draw(sf::RenderTarget& target, sf::RenderStates states) const override

```

```

68 {
69     target.draw(m_text, states);
70 }
71
72 private:
73     sf::Text m_text;
74 };
75
76 };
77
78 }
79 }
80
81 #endif

```

## 7.41. include/objects/object\_base.hpp fájlreferencia

```
#include "all.hpp"
```

### Adatszerkezetek

- class [pftd::Object](#)
- struct [pftd::Object::Compare](#)

### Névterek

- namespace [pftd](#)

## 7.42. object\_base.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1  #pragma once
2
3  #ifndef CPORTA
4
5  #include "all.hpp"
6
7  namespace pftd {
8
9  class Object : public sf::Drawable
10 {
11 public:
12     Object() = default;
13
14     Object(utils::Vec2f position, utils::Vec2f size, int zIndex = 0):
15         zIndex{zIndex}, position{position}, size{size}
16     {}
17
18     Object(Object const&) = default;
19     Object(Object&&) noexcept = default;
20
21     virtual ~Object() = default;
22
23     virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const = 0;
24
25     struct Compare
26     {
27         // bool operator() (Object const& o1, Object const& o2) const
28         // {
29         //     return o1.zIndex > o2.zIndex;
30         // };
31
32         bool operator() (Object const* o1, Object const* o2) const
33     {
34         if(!o1 || !o2) return false;
35         return o1->zIndex > o2->zIndex;
36     }
37 }

```

```

51         //return this->operator() (*o1, *o2);
52     };
53 };
54
55 protected:
57     int zIndex = 0;
58
59     utils::Vec2f position;
60
61     utils::Vec2f size;
62
63 };
64
65 #endif

```

## 7.43. include/resources.hpp fájlreferencia

```

#include "SFML/Graphics.hpp"
#include "SFML/Audio.hpp"
#include "objects/gui/image.hpp"

```

### Adatszerkezetek

- class [pftd::ResourceManager](#)
- struct [pftd::LoadError](#)

### Névterek

- namespace [pftd](#)

## 7.44. resources.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "SFML/Graphics.hpp"
6 #include "SFML/Audio.hpp"
7 #include "objects/gui/image.hpp"
8
9 namespace pftd {
10
11     class ResourceManager final
12     {
13     public:
14         ResourceManager(ResourceManager const&) = delete;
15         ResourceManager(ResourceManager&&) = delete;
16
17         ~ResourceManager() = default;
18
19         void loadDefaultFont(std::string const& path);
20
21         sf::Texture const& getTexture(std::string const& source);
22
23         sf::SoundBuffer const& getSound(std::string const& source);
24
25         sf::Font const& getDefaultFont() const { return m_defaultFont; }
26
27         static ResourceManager* getInstance() { return m_instance; }
28
29         static ResourceManager* create();
30

```

```

74
75     static void destroy() { delete m_instance; }
76
77 private:
78     // Az 'm_instance' létrehozásához van használva.
79     ResourceManager() = default;
80
81     static ResourceManager* m_instance;
82
83     sf::Font m_defaultFont;
84
85     std::unordered_map<std::string, sf::Texture> m_textures;
86
87     std::unordered_map<std::string, sf::SoundBuffer> m_sounds;
88
89 };
90
91 struct LoadError : public std::runtime_error
92 {
93     using std::runtime_error::runtime_error;
94 };
95
96 }
97
98 #endif

```

## 7.45. include/scene.hpp fájlreferencia

```

#include "all.hpp"
#include "SFML/Window/Event.hpp"
#include "objects/object_base.hpp"

```

### Adatszerkezetek

- class [pftd::Scene](#)

### Névterek

- namespace [pftd](#)

## 7.46. scene.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "all.hpp"
6 #include "SFML/Window/Event.hpp"
7 #include "objects/object_base.hpp"
8
9 namespace pftd {
10
11 class Scene
12 {
13     using ObjPtrVec = std::vector<Object*>;
14 public:
15     enum class StateFlag : uint8_t
16     {
17         NONE = 0,
18         LOAD_STATE = 1 << 0
19     };
20
21     Scene() = default;
22
23 };
24

```



```

25     Scene(Scene const&) = delete;
26     Scene(Scene&&) noexcept = delete;
27
31     virtual ~Scene();
32
38     ObjPtrVec const& getObjects()const { return objects; }
39
45     ObjPtrVec& getObjects() { return objects; }
46
53     void setMusic(std::string const& source, float volume = 100.0f);
54
60     virtual void toggleActive(StateFlag flag = StateFlag::NONE);
61
67     virtual void onEvent(sf::Event const& event) = 0;
68
74     virtual void update(float dt) = 0;
75
76 protected:
77     bool isActive = false;
78
81     ObjPtrVec objects;
82
84     sf::Music* backgroundMusic = nullptr;
85
86 };
87
88 }
89
90 #endif

```

## 7.47. include/scenes/game.hpp fájlreferencia

```

#include "objects/gui/image.hpp"
#include "objects/gui/button.hpp"
#include "utils/hetero_collection.hpp"
#include "game/level.hpp"
#include "resources.hpp"
#include "scene.hpp"

```

### Adatszerkezetek

- class [pftd::GameScene](#)
- struct [pftd::GameScene::InventoryItem](#)
- struct [pftd::GameScene::Inventory](#)

### Névterek

- namespace [pftd](#)

## 7.48. game.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/gui/image.hpp"
6 #include "objects/gui/button.hpp"
7 #include "utils/hetero_collection.hpp"
8 #include "game/level.hpp"
9 #include "resources.hpp"
10 #include "scene.hpp"

```

```

11
12 namespace pftd {
13
14 class GameScene final : public Scene
15 {
16 public:
17     static constexpr char const* SAVE_FILE_PATH = "res/data/save.dat";
18
19     struct InventoryItem : public Clickable
20     {
21         gr::Sprite frame;
22
23         gr::Sprite icon;
24
25         Tower* towerToSpawn = nullptr;
26
27         gr::Label priceLabel;
28
29         InventoryItem(Tower* tower, Level * const level, utils::Vec2f const& position, utils::Vec2f
30 const& size);
31 ~InventoryItem();
32
33         virtual void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
34     };
35
36     struct Inventory final : public Object, public utils::Container<InventoryItem>
37     {
38         gr::Sprite background;
39
40         Inventory(std::string const& backgroundImageSrc);
41         ~Inventory() = default;
42
43         void draw(sf::RenderTarget& target, sf::RenderStates states) const override;
44     };
45
46     GameScene();
47     ~GameScene();
48
49     void onEvent(sf::Event const& event) override;
50
51     void update(float dt) override;
52
53     void toggleActive(Scene::StateFlag flag = Scene::StateFlag::NONE) override;
54
55     void startGame();
56
57     void updateScore();
58
59     void updateWealth();
60
61 private:
62     bool m_shouldLoadSaved = false;
63
64     gr::Button* m_saveButt = nullptr;
65
66     gr::Label* m_moneyCounter = nullptr;
67
68     gr::Label* m_scoreCounter = nullptr;
69
70     sf::Sound m_hornSound =
71 sf::Sound{ResourceManager::getInstance()->getSound("res/audio/ready_for_battle.mp3")};
72
73     sf::Sound m_gameoverSound =
74 sf::Sound{ResourceManager::getInstance()->getSound("res/audio/gameover.mp3")};
75
76     Inventory* m_inventory = nullptr;
77
78     Level* m_level = nullptr;
79
80     void _constructInventory();
81 };
82 }
83 #endif

```

## 7.49. include/scenes/menu.hpp fájlreferencia

```

#include "objects/gui/button.hpp"
#include "scene.hpp"

```

## Adatszerkezetek

- class [pftd::MenuScene](#)

## Névterek

- namespace [pftd](#)

## 7.50. menu.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #ifndef CPORTA
4
5 #include "objects/gui/button.hpp"
6 #include "scene.hpp"
7
8 namespace pftd {
9
10 class MenuScene final : public Scene
11 {
12 public:
13     MenuScene();
14     ~MenuScene() = default;
15
16     void onEvent(sf::Event const& event) override;
17
18     void update(float dt) override;
19
20     void toggleActive(Scene::StateFlag flag = Scene::StateFlag::NONE) override;
21
22 private:
23     std::vector<gr::Button*> m_buttons;
24
25     bool _isSaveFileAvailable();
26
27 };
28
29 #endif
```

## 7.51. include/utils/hetero\_collection.hpp fájlreferencia

```
#include "all.hpp"
```

## Adatszerkezetek

- class [pftd::utils::Container< T, C >](#)

## Névterek

- namespace [pftd](#)
- namespace [pftd::utils](#)

## 7.52. hetero\_collection.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #include "all.hpp"
4
5 namespace pftd {
6 namespace utils {
7
14 template<typename T, typename C = std::vector<T*>
15 class Container
16 {
17 public:
18     Container() = default;
19     virtual ~Container()
20     {
21         for(auto& elem : m_container) {
22             delete elem;
23         }
24     }
25
32     /*virtual*/ T* const append(T* elem)
33     {
34         m_container.push_back(elem);
35         return elem;
36     }
37
43     std::size_t size()const { return m_container.size(); }
44
50     C const& getContainer()const { return m_container; }
51
52 private:
54     C m_container;
55
56 };
57
58 }
59 }
```

## 7.53. include/utils/logger.hpp fájlreferencia

### Makródefiníciók

- #define `print(x)`
- #define `where()`

### 7.53.1. Makródefiníciók dokumentációja

#### 7.53.1.1. print

```
#define print(  
    x )
```

#### 7.53.1.2. where

```
#define where( )
```

## 7.54. logger.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 // Zseniálisan egyszerű logger makrókkal.
4
5 #ifdef _PFTD_DEBUG
6
7 #include <iostream>
8
9 #define print(x) std::cout << x << '\n'
10 #define where() print("[CALLED IN] " << __PRETTY_FUNCTION__)
11
12 #else
13
14 #define print(x)
15 #define where()
16
17 #endif
```

## 7.55. include/utils/parsers.hpp fájlreferencia

```
#include "all.hpp"
```

### Adatszerkezetek

- struct [pftd::utils::parser::ParseError](#)
- class [pftd::utils::parser::Parser](#)
- class [pftd::utils::parser::LevelConfigParser](#)
- class [pftd::utils::parser::SaveFileParser](#)
- struct [pftd::utils::parser::SaveFileParser::StatsInfo](#)
- struct [pftd::utils::parser::SaveFileParser::EntityInfo](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::utils](#)
- namespace [pftd::utils::parser](#)

## 7.56. parsers.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #include "all.hpp"
4
5 namespace pftd {
6 namespace utils {
7 namespace parser {
8
9 struct ParseError : std::runtime_error
10 {
11     using std::runtime_error::runtime_error;
12 };
13
14 class Parser
15 {
16 public:
```

```

23     Parser(std::string const& sourceFile, std::string label = "");
24
25     virtual ~Parser();
26
27     void reset();
28
29     bool isLabelValid(bool skip = true);
30
31     void skip(size_t howMany, char until = '\n');
32
33     void skip(char while_);
34
35     template<typename T> T get()
36     {
37         this->_skipWhitespace();
38         while(sourceStream.peek() == commentDenoter) {
39             this->_skipLine();
40             this->_skipWhitespace();
41         }
42
43         T got;
44         sourceStream >> got;
45         if (sourceStream.fail() || sourceStream.bad()) {
46             throw ParseError("(" + validLabel + ") Nem sikerült beolvasni a következő tokent!");
47         }
48         return got;
49     }
50
51     char peekAhead();
52
53     virtual void parse() = 0;
54
55 protected:
56     std::ifstream sourceStream;
57
58     std::string validLabel;
59
60     char commentDenoter = '#';
61
62     //virtual bool validate() const = 0;
63
64 private:
65     void _skipLine();
66
67     void _skipWhitespace();
68
69 };
70
71 class LevelConfigParser final : public Parser
72 {
73 public:
74     LevelConfigParser(std::string const& sourceFile);
75     ~LevelConfigParser() = default;
76
77     void parse() override;
78
79     std::vector<Vec2f> getAttribute(std::string name) const;
80
81 private:
82     std::unordered_map<std::string, std::vector<Vec2f>> m_attribs;
83
84     std::string _getAttribute();
85
86 };
87
88 class SaveFileParser final : public Parser
89 {
90 public:
91     struct StatsInfo
92     {
93         unsigned int score, wealth;
94         int maxHp, hp;
95     };
96
97     enum class EntityType
98     {
99         TOWER = 0U,
100         SEAL,
101         PROJECTILE
102     };
103
104     struct EntityInfo
105     {
106         EntityType entityType;

```

```

183         utils::Vec2f position;
184
185         //union {
186         struct {
187             unsigned int sealID;
188             float lerpParam;
189             bool goingBackwards;
190             unsigned int hp;
191         } seal;
192
193         struct {
194             unsigned int projID;
195             utils::Vec2f direction;
196             float speed;
197         } proj;
198
199         unsigned int towerID;
200     };
201
202     //};
203 };
204
205 //public:
206 SaveFileParser(std::string const& sourceFile);
207 ~SaveFileParser() = default;
208
209 void parse() override;
210
211 StatsInfo const& getStats()const { return m_readStats; };
212
213 std::vector<EntityInfo> const& getEntities()const { return m_entities; };
214
215 private:
216 StatsInfo m_readStats;
217
218 std::vector<EntityInfo> m_entities;
219
220 void _getStats();
221
222 void _getEntity();
223 };
224
225 }
226
227 }
228
229 }
230
231 }
232
233 }
234
235 }
236
237 }
238
239 }
240
241 }
242
243 }
244
245 }
246
247 }

```

## 7.57. include/utils/random\_gen.hpp fájlreferencia

```
#include "all.hpp"
```

### Adatszerkezetek

- class [pftd::utils::Random](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::utils](#)

## 7.58. random\_gen.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #include "all.hpp"
4
5 namespace pftd {

```

```
6 namespace utils {
7
9 class Random
10 {
11 public:
12     Random() = delete;
13
15     static std::default_random_engine randomEngine;
16
23     template<typename RandomDist>
24     static auto generate(RandomDist distribution)
25     {
26         return distribution(randomEngine);
27     }
28 };
29
30 }
31 }
```

## 7.59. include/utils/serializable.hpp fájlreferencia

```
#include "all.hpp"
```

### Adatszerkezetek

- class [pftd::utils::Serializable](#)

### Névterek

- namespace [pftd](#)
- namespace [pftd::utils](#)

## 7.60. serializable.hpp

[Ugrás a fájl dokumentációjához.](#)

```
1 #pragma once
2
3 #include "all.hpp"
4
5 namespace pftd {
6 namespace utils {
7
9 class Serializable
10 {
11 public:
17     virtual void serialize(std::ostream& out) const = 0;
18
26     //virtual void deserialize(std::istream& in) = 0;
27 };
28
29 }
30 }
```

## 7.61. include/utils/substitute\_types.hpp fájlreferencia

```
#include <cmath>
#include <iostream>
```



## Adatszerkezetek

- struct `pftd::utils::Vec2< T >`

## Névterek

- namespace `pftd`
- namespace `pftd::utils`

## Típusdefiníciók

- using `pftd::utils::Vec2i` = `Vec2< int >`
- using `pftd::utils::Vec2f` = `Vec2< float >`

## 7.62. substitute\_types.hpp

[Ugrás a fájl dokumentációjához.](#)

```

1 #pragma once
2
3 #include <cmath>
4 #include <iostream>
5
6 namespace pftd {
7     namespace utils {
8
9         template<typename T>
10         struct Vec2
11         {
12             T x;
13             T y;
14
15             Vec2():
16                 x{0}, y{0}
17             {}
18             Vec2(T x, T y):
19                 x{x}, y{y}
20             {}
21             Vec2(Vec2 const& right):
22                 x{right.x}, y{right.y}
23             {}
24
25             static float distance(Vec2 const& v1, Vec2 const& v2)
26             {
27                 return std::sqrt((v2.x-v1.x)*(v2.x-v1.x) + (v2.y-v1.y)*(v2.y-v1.y));
28             }
29
30             Vec2 normalize()const
31             {
32                 return *this / Vec2::distance(*this, {0, 0});
33             }
34
35             Vec2 operator+(Vec2 const& right)const
36             {
37                 return {this->x + right.x, this->y + right.y};
38             }
39
40             Vec2 operator-(Vec2 const& right)const
41             {
42                 return {this->x - right.x, this->y - right.y};
43             }
44
45             Vec2 operator/(T scalar)const
46             {
47                 return {this->x / scalar, this->y / scalar};
48             }
49
50             Vec2 operator*(T scalar)const
51             {
52                 return {this->x * scalar, this->y * scalar};
53             }
54
55             bool operator==(Vec2 other)const

```

```
72 {
73     return this->x == other.x && this->y == other.y;
74 }
75 bool operator!=(Vec2 other) const
76 {
77     return !(this == other);
78 }
79
80 Vec2& operator=(Vec2 const& right)
81 {
82     if(&right != this) {
83         this->x = right.x;
84         this->y = right.y;
85     }
86     return *this;
87 }
88
89 friend std::ostream& operator<<(std::ostream& out, Vec2 const& vec)
90 {
91     out << vec.x << " " << vec.y;
92     return out;
93 }
94
95 friend std::istream& operator>>(std::istream& in, Vec2& vec)
96 {
97     in >> vec.x >> vec.y;
98     return in;
99 }
100 };
101
102 // Az SFML is ilyet csinál.
103 using Vec2i = Vec2<int>;
104 using Vec2f = Vec2<float>;
105
106 }
107 }
```

# Tárgymutató

- [\\_getAttribute](#)  
    [pftd::utils::parser::LevelConfigParser, 58](#)
  - [\\_skipLine](#)  
    [pftd::utils::parser::Parser, 67](#)
  - [\\_skipWhitespace](#)  
    [pftd::utils::parser::Parser, 67](#)
- [~App](#)  
    [pftd::App, 14](#)
- [~Button](#)  
    [pftd::gr::Button, 20](#)
- [~Container](#)  
    [pftd::utils::Container< T, C >, 27](#)
- [~Entity](#)  
    [pftd::Entity, 32](#)
- [~GameScene](#)  
    [pftd::GameScene, 40](#)
- [~IcicleStabber](#)  
    [pftd::IcicleStabber, 44](#)
- [~Inventory](#)  
    [pftd::GameScene::Inventory, 46](#)
- [~InventoryItem](#)  
    [pftd::GameScene::InventoryItem, 47](#)
- [~Level](#)  
    [pftd::Level, 54](#)
- [~LevelConfigParser](#)  
    [pftd::utils::parser::LevelConfigParser, 58](#)
- [~MenuScene](#)  
    [pftd::MenuScene, 60](#)
- [~Nest](#)  
    [pftd::Level::Nest, 62](#)
- [~Object](#)  
    [pftd::Object, 64](#)
- [~Parser](#)  
    [pftd::utils::parser::Parser, 67](#)
- [~Projectile](#)  
    [pftd::Projectile, 70](#)
- [~Renderer](#)  
    [pftd::Renderer, 76](#)
- [~ResourceManager](#)  
    [pftd::ResourceManager, 80](#)
- [~SaveFileParser](#)  
    [pftd::utils::parser::SaveFileParser, 85](#)
- [~Scene](#)  
    [pftd::Scene, 87](#)
- [~Seal](#)  
    [pftd::Seal, 92](#)
- [~Snowballer](#)  
    [pftd::Snowballer, 100](#)
- [~Sprite](#)  
    [pftd::gr::Sprite, 103](#)
- [~Tower](#)  
    [pftd::Tower, 110](#)
- [addScene](#)  
    [pftd::App, 15](#)
- [advanceAnimationFrame](#)  
    [pftd::Entity, 32](#)  
    [pftd::Seal, 93](#)  
    [pftd::Tower, 110](#)
- [angularVelocityRadPerSec](#)  
    [pftd::Projectile, 71](#)
- [App](#)  
    [pftd::App, 14](#)  
    [pftd::Renderer, 78](#)
- [append](#)  
    [pftd::utils::Container< T, C >, 27](#)
- [attack](#)  
    [pftd::Snowballer, 100](#)  
    [pftd::Tower, 110](#)
- [attackDamage](#)  
    [pftd::Tower, 112](#)
- [attackRangePixel](#)  
    [pftd::Tower, 112](#)
- [attackSpeedSec](#)  
    [pftd::Tower, 112](#)
- [attackTimerSec](#)  
    [pftd::Tower, 112](#)
- [background](#)  
    [pftd::GameScene::Inventory, 46](#)
- [backgroundMusic](#)  
    [pftd::Scene, 90](#)
- [Button](#)  
    [pftd::gr::Button, 19](#)
- [CELL\\_N](#)  
    [pftd::Entity, 34](#)
- [cellSize](#)  
    [pftd::Entity, 34](#)
- [changeScene](#)  
    [pftd::App, 15](#)
- [clear](#)  
    [pftd::Renderer, 77](#)
- [Clickable](#)  
    [pftd::Clickable, 23](#)
- [clone](#)  
    [pftd::Cub, 29](#)  
    [pftd::Entity, 32](#)  
    [pftd::FZC, 38](#)

pftd::IcicleStabber, [44](#)  
 pftd::RegularSeal, [75](#)  
 pftd::Seal, [93](#)  
 pftd::Snowball, [98](#)  
 pftd::Snowballer, [100](#)  
 pftd::Tower, [111](#)  
 pftd::ZombieSeal, [119](#)  
 commentDenoter  
   pftd::utils::parser::Parser, [69](#)  
 config  
   pftd::Level, [56](#)  
 Container  
   pftd::utils::Container< T, C >, [27](#)  
 create  
   pftd::App, [15](#)  
   pftd::ResourceManager, [80](#)  
 Cub  
   pftd::Cub, [29](#)  
 currentCell  
   pftd::Entity, [34](#)  
 currentSprite  
   pftd::Entity, [34](#)  
  
 damage  
   pftd::Seal, [93](#)  
 deselectTower  
   pftd::Level, [54](#)  
 deserialize  
   pftd::Level::Stats, [105](#)  
   pftd::utils::Serializable, [96](#)  
 destroy  
   pftd::App, [16](#)  
   pftd::ResourceManager, [80](#)  
 direction  
   pftd::Projectile, [71](#)  
 display  
   pftd::Renderer, [77](#)  
 distance  
   pftd::utils::Vec2< T >, [115](#)  
 draw  
   pftd::Entity, [32](#)  
   pftd::GameScene::Inventory, [46](#)  
   pftd::GameScene::InventoryItem, [48](#)  
   pftd::gr::Button, [20](#)  
   pftd::gr::Label, [50](#)  
   pftd::gr::Sprite, [103](#)  
   pftd::Level, [54](#)  
   pftd::Object, [65](#)  
  
 EndPoint  
   pftd, [10](#)  
 Entity  
   pftd::Entity, [31](#)  
 EntityType  
   pftd::utils::parser::SaveFileParser, [84](#)  
 entityType  
   pftd::utils::parser::SaveFileParser::EntityInfo, [36](#)  
  
 flipY  
  
   pftd::gr::Sprite, [103](#)  
 FollowPath  
   pftd::FollowPath, [37](#)  
 followPath  
   pftd::Level, [56](#)  
   pftd::Seal, [94](#)  
 frame  
   pftd::GameScene::InventoryItem, [48](#)  
 frameDurationSec  
   pftd::Entity, [34](#)  
 FZC  
   pftd::FZC, [38](#)  
  
 GameScene  
   pftd::GameScene, [40](#)  
 generate  
   pftd::utils::Random, [73](#)  
 get  
   pftd::utils::parser::Parser, [67](#)  
 getAttribute  
   pftd::utils::parser::LevelConfigParser, [58](#)  
 getContainer  
   pftd::utils::Container< T, C >, [27](#)  
 getDefaultFont  
   pftd::ResourceManager, [81](#)  
 getEntities  
   pftd::utils::parser::SaveFileParser, [85](#)  
 getInstance  
   pftd::App, [16](#)  
   pftd::ResourceManager, [81](#)  
 getLabel  
   pftd::gr::Button, [20](#)  
 getObjects  
   pftd::Scene, [88](#)  
 getPosition  
   pftd::Entity, [33](#)  
 getSound  
   pftd::ResourceManager, [81](#)  
 getSprite  
   pftd::Entity, [33](#)  
 getStats  
   pftd::utils::parser::SaveFileParser, [85](#)  
 getText  
   pftd::gr::Label, [51](#)  
 getTexture  
   pftd::ResourceManager, [82](#)  
 getWindow  
   pftd::Renderer, [77](#)  
 getWindowHeight  
   pftd::App, [16](#)  
 getWindowWidth  
   pftd::App, [16](#)  
  
 handleClick  
   pftd::Clickable, [23](#)  
   pftd::gr::Button, [20](#)  
 hasCompletedPath  
   pftd::Seal, [93](#)  
 hasReachedNest

- pftd::Seal, [94](#)
- hp
  - pftd::Level::Stats, [106](#)
  - pftd::Seal, [94](#)
  - pftd::utils::parser::SaveFileParser::EntityInfo, [36](#)
  - pftd::utils::parser::SaveFileParser::StatsInfo, [107](#)
- IcicleStabber
  - pftd::IcicleStabber, [43, 44](#)
- icon
  - pftd::GameScene::InventoryItem, [48](#)
- include/all.hpp, [121](#)
- include/app.hpp, [122](#)
- include/game/level.hpp, [123, 124](#)
- include/objects/clickable.hpp, [125](#)
- include/objects/entities/entity\_base.hpp, [126](#)
- include/objects/entities/projectiles/projectile\_base.hpp, [127](#)
- include/objects/entities/projectiles/snowball.hpp, [127, 128](#)
- include/objects/entities/seals/cub.hpp, [128](#)
- include/objects/entities/seals/followpath.hpp, [129](#)
- include/objects/entities/seals/fortified\_zombie\_cub.hpp, [129, 130](#)
- include/objects/entities/seals/regular.hpp, [130](#)
- include/objects/entities/seals/seal\_base.hpp, [130, 131](#)
- include/objects/entities/seals/zombie.hpp, [131, 132](#)
- include/objects/entities/towers/iciclestabber.hpp, [132](#)
- include/objects/entities/towers/snowballer.hpp, [133](#)
- include/objects/entities/towers/tower\_base.hpp, [133, 134](#)
- include/objects/gui/button.hpp, [134, 135](#)
- include/objects/gui/image.hpp, [135, 136](#)
- include/objects/gui/label.hpp, [136, 137](#)
- include/objects/object\_base.hpp, [137, 138](#)
- include/resources.hpp, [138, 139](#)
- include/scene.hpp, [139, 140](#)
- include/scenes/game.hpp, [140, 141](#)
- include/scenes/menu.hpp, [142](#)
- include/utils/hetero\_collection.hpp, [142, 143](#)
- include/utils/logger.hpp, [143, 144](#)
- include/utils/parsers.hpp, [144, 145](#)
- include/utils/random\_gen.hpp, [146, 147](#)
- include/utils/serializable.hpp, [147](#)
- include/utils/substitute\_types.hpp, [148](#)
- instantAttack
  - pftd::Tower, [113](#)
- Inventory
  - pftd::GameScene::Inventory, [45](#)
- InventoryItem
  - pftd::GameScene::InventoryItem, [47](#)
- isActive
  - pftd::Clickable, [25](#)
  - pftd::Scene, [90](#)
- isAnimated
  - pftd::Entity, [35](#)
- isCurrentlyStealing
  - pftd::Seal, [95](#)
- isGameOver
  - pftd::Level, [54](#)
- isLabelValid
  - pftd::utils::parser::Parser, [67](#)
- isRunning
  - pftd::App, [17](#)
- Label
  - pftd::gr::Label, [50](#)
- label
  - pftd::gr::Button, [21](#)
- lerpParam
  - pftd::Seal, [95](#)
- lerpPath
  - pftd::Seal, [94](#)
- Level
  - pftd::Level, [53](#)
- LevelConfigParser
  - pftd::utils::parser::LevelConfigParser, [58](#)
- linearSpeed
  - pftd::Projectile, [72](#)
- loadDefaultFont
  - pftd::ResourceManager, [82](#)
- logger.hpp
  - print, [143](#)
  - where, [143](#)
- lookForTarget
  - pftd::Tower, [111](#)
- loseHP
  - pftd::Level, [54](#)
- m\_accuTimeSpawnSec
  - pftd::Level, [56](#)
- m\_activeSceneID
  - pftd::App, [17](#)
- m\_attribs
  - pftd::utils::parser::LevelConfigParser, [59](#)
- m\_background
  - pftd::gr::Button, [21](#)
- m\_buttons
  - pftd::MenuScene, [61](#)
- m\_callback
  - pftd::Clickable, [25](#)
- m\_clickSound
  - pftd::gr::Button, [21](#)
- m\_container
  - pftd::utils::Container< T, C >, [28](#)
- m\_defaultFont
  - pftd::ResourceManager, [82](#)
- m\_entities
  - pftd::utils::parser::SaveFileParser, [85](#)
- m\_gameoverSound
  - pftd::GameScene, [42](#)
- m\_height
  - pftd::Renderer, [78](#)
- m\_hornSound
  - pftd::GameScene, [42](#)
- m\_instance
  - pftd::App, [17](#)
  - pftd::ResourceManager, [82](#)

- m\_inventory
  - pftd::GameScene, 42
- m\_lastAttribute
  - pftd::utils::parser::LevelConfigParser, 59
- m\_level
  - pftd::GameScene, 42
- m\_moneyCounter
  - pftd::GameScene, 42
- m\_queue
  - pftd::Renderer, 78
- m\_readStats
  - pftd::utils::parser::SaveFileParser, 86
- m\_rect
  - pftd::gr::Button, 22
- m\_renderer
  - pftd::App, 18
- m\_running
  - pftd::App, 18
- m\_saveButt
  - pftd::GameScene, 42
- m\_scenes
  - pftd::App, 18
- m\_scoreCounter
  - pftd::GameScene, 42
- m\_sounds
  - pftd::ResourceManager, 83
- m\_sprite
  - pftd::gr::Sprite, 104
- m\_text
  - pftd::gr::Label, 52
- m\_texture
  - pftd::gr::Sprite, 104
- m\_textures
  - pftd::ResourceManager, 83
- m\_width
  - pftd::Renderer, 78
- m\_window
  - pftd::Renderer, 78
- MAX\_HP
  - pftd::Level::Stats, 106
- MenuScene
  - pftd::MenuScene, 60
- modColor
  - pftd::gr::Sprite, 103
- money
  - pftd::Level::Stats, 106
- Nest
  - pftd::Level::Nest, 62
- nest
  - pftd::Level, 56
- nextPoint
  - pftd::Seal, 95
- normalize
  - pftd::utils::Vec2< T >, 115
- Object
  - pftd::Object, 64
- objects
  - pftd::Scene, 90
- ObjPtrVec
  - pftd::Scene, 87
- onEvent
  - pftd::GameScene, 40
  - pftd::MenuScene, 60
  - pftd::Scene, 88
- operator!=
  - pftd::utils::Vec2< T >, 115
- operator<<
  - pftd::utils::Vec2< T >, 117
- operator\*
  - pftd::utils::Vec2< T >, 116
- operator()
  - pftd::Object::Compare, 25, 26
- operator+
  - pftd::utils::Vec2< T >, 116
- operator-
  - pftd::utils::Vec2< T >, 116
- operator/
  - pftd::utils::Vec2< T >, 116
- operator=
  - pftd::App, 17
  - pftd::utils::Vec2< T >, 116
- operator==
  - pftd::utils::Vec2< T >, 116
- parse
  - pftd::utils::parser::LevelConfigParser, 59
  - pftd::utils::parser::Parser, 68
  - pftd::utils::parser::SaveFileParser, 85
- Parser
  - pftd::utils::parser::Parser, 66
- peekAhead
  - pftd::utils::parser::Parser, 68
- pftd, 9
  - EndPoint, 10
- pftd::App, 13
  - ~App, 14
  - addScene, 15
  - App, 14
  - changeScene, 15
  - create, 15
  - destroy, 16
  - getInstance, 16
  - getWindowHeight, 16
  - getWindowWidth, 16
  - isRunning, 17
  - m\_activeSceneID, 17
  - m\_instance, 17
  - m\_renderer, 18
  - m\_running, 18
  - m\_scenes, 18
  - operator=, 17
  - run, 17
- pftd::Clickable, 22
  - Clickable, 23
  - handleClick, 23
  - isActive, 25

- m\_callback, 25
  - setCallback, 23
- pftd::Cub, 28
  - clone, 29
  - Cub, 29
- pftd::Entity, 30
  - ~Entity, 32
  - advanceAnimationFrame, 32
  - CELL\_N, 34
  - cellSize, 34
  - clone, 32
  - currentCell, 34
  - currentSprite, 34
  - draw, 32
  - Entity, 31
  - frameDurationSec, 34
  - getPosition, 33
  - getSprite, 33
  - isAnimated, 35
  - setPosition, 33
  - spriteSheet, 35
  - totalElapsedSec, 35
  - update, 34
- pftd::FollowPath, 37
  - FollowPath, 37
- pftd::FZC, 37
  - clone, 38
  - FZC, 38
- pftd::GameScene, 39
  - ~GameScene, 40
  - GameScene, 40
  - m\_gameoverSound, 42
  - m\_hornSound, 42
  - m\_inventory, 42
  - m\_level, 42
  - m\_moneyCounter, 42
  - m\_saveButt, 42
  - m\_scoreCounter, 42
  - onEvent, 40
  - startGame, 40
  - toggleActive, 40
  - update, 41
  - updateScore, 41
  - updateWealth, 41
- pftd::GameScene::Inventory, 45
  - ~Inventory, 46
  - background, 46
  - draw, 46
  - Inventory, 45
- pftd::GameScene::InventoryItem, 47
  - ~InventoryItem, 47
  - draw, 48
  - frame, 48
  - icon, 48
  - InventoryItem, 47
  - priceLabel, 48
  - towerToSpawn, 48
- pftd::gr, 10
  - m\_callback, 25
  - setCallback, 23
- pftd::gr::Button, 18
  - ~Button, 20
  - Button, 19
  - draw, 20
  - getLabel, 20
  - handleClick, 20
  - label, 21
  - m\_background, 21
  - m\_clickSound, 21
  - m\_rect, 22
  - setBackground, 21
  - setSound, 21
- pftd::gr::Label, 49
  - draw, 50
  - getText, 51
  - Label, 50
  - m\_text, 52
  - setLabel, 51
  - setOutline, 51
- pftd::gr::Sprite, 101
  - ~Sprite, 103
  - draw, 103
  - flipY, 103
  - m\_sprite, 104
  - m\_texture, 104
  - modColor, 103
  - setSpriteRect, 104
  - Sprite, 102
- pftd::IcicleStabber, 43
  - ~IcicleStabber, 44
  - clone, 44
  - IcicleStabber, 43, 44
  - update, 44
- pftd::Level, 52
  - ~Level, 54
  - config, 56
  - deselectTower, 54
  - draw, 54
  - followPath, 56
  - isGameOver, 54
  - Level, 53
  - loseHP, 54
  - m\_accuTimeSpawnSec, 56
  - nest, 56
  - placeTower, 55
  - projectiles, 56
  - seals, 56
  - selectedTower, 57
  - selectTower, 55
  - spawnSeal, 55
  - stats, 57
  - towers, 57
  - update, 55
- pftd::Level::Nest, 61
  - ~Nest, 62
  - Nest, 62
  - radiusPixel, 63
- pftd::Level::Stats, 104

- deserialize, 105
- hp, 106
- MAX\_HP, 106
- money, 106
- score, 106
- serialize, 106
- Stats, 105
- pftd::MenuScene, 59
  - ~MenuScene, 60
  - m\_buttons, 61
  - MenuScene, 60
  - onEvent, 60
  - update, 61
- pftd::Object, 63
  - ~Object, 64
  - draw, 65
  - Object, 64
  - position, 65
  - size, 65
  - zIndex, 65
- pftd::Object::Compare, 25
  - operator(), 25, 26
- pftd::Projectile, 70
  - ~Projectile, 70
  - angularVelocityRadPerSec, 71
  - direction, 71
  - linearSpeed, 72
  - Projectile, 70
  - update, 71
- pftd::RegularSeal, 74
  - clone, 75
  - RegularSeal, 74
- pftd::Renderer, 75
  - ~Renderer, 76
  - App, 78
  - clear, 77
  - display, 77
  - getWindow, 77
  - m\_height, 78
  - m\_queue, 78
  - m\_width, 78
  - m\_window, 78
  - pushQueue, 77
  - render, 77
  - Renderer, 76
- pftd::ResourceManager, 79
  - ~ResourceManager, 80
  - create, 80
  - destroy, 80
  - getDefaultFont, 81
  - getInstance, 81
  - getSound, 81
  - getTexture, 82
  - loadDefaultFont, 82
  - m\_defaultFont, 82
  - m\_instance, 82
  - m\_sounds, 83
  - m\_textures, 83
  - ResourceManager, 80
- pftd::Scene, 86
  - ~Scene, 87
  - backgroundMusic, 90
  - getObjects, 88
  - isActive, 90
  - objects, 90
  - ObjPtrVec, 87
  - onEvent, 88
  - Scene, 87
  - setMusic, 89
  - toggleActive, 89
  - update, 89
- pftd::SceneError, 90
- pftd::Seal, 91
  - ~Seal, 92
  - advanceAnimationFrame, 93
  - clone, 93
  - damage, 93
  - followPath, 94
  - hasCompletedPath, 93
  - hasReachedNest, 94
  - hp, 94
  - isCurrentlyStealing, 95
  - lerpParam, 95
  - lerpPath, 94
  - nextPoint, 95
  - reachedNest, 95
  - returned, 95
  - Seal, 92
  - speed, 95
  - update, 94
  - value, 95
- pftd::Snowball, 97
  - clone, 98
  - Snowball, 98
- pftd::Snowballer, 99
  - ~Snowballer, 100
  - attack, 100
  - clone, 100
  - Snowballer, 99
  - update, 100
- pftd::Tower, 108
  - ~Tower, 110
  - advanceAnimationFrame, 110
  - attack, 110
  - attackDamage, 112
  - attackRangePixel, 112
  - attackSpeedSec, 112
  - attackTimerSec, 112
  - clone, 111
  - instantAttack, 113
  - lookForTarget, 111
  - price, 113
  - ProjSpawnFunc, 109
  - radiusPixel, 113
  - setProjSpawnCb, 111
  - spawnProjectile, 113



- target, 113
- Tower, 109, 110
- update, 112
- pftd::utils, 10
  - Vec2f, 10
  - Vec2i, 10
- pftd::utils::Container< T, C >, 26
  - ~Container, 27
  - append, 27
  - Container, 27
  - getContainer, 27
  - m\_container, 28
  - size, 27
- pftd::utils::parser, 11
- pftd::utils::parser::LevelConfigParser, 57
  - \_getAttribute, 58
  - ~LevelConfigParser, 58
  - getAttribute, 58
  - LevelConfigParser, 58
  - m\_attribs, 59
  - m\_lastAttribute, 59
  - parse, 59
- pftd::utils::parser::Parser, 66
  - \_skipLine, 67
  - \_skipWhitespace, 67
  - ~Parser, 67
  - commentDenoter, 69
  - get, 67
  - isLabelValid, 67
  - parse, 68
  - Parser, 66
  - peekAhead, 68
  - reset, 68
  - skip, 68, 69
  - sourceStream, 69
  - validLabel, 69
- pftd::utils::parser::SaveFileParser, 83
  - ~SaveFileParser, 85
  - EntityType, 84
  - getEntities, 85
  - getStats, 85
  - m\_entities, 85
  - m\_readStats, 86
  - parse, 85
  - SaveFileParser, 84
  - SEAL, 84
  - TOWER, 84
- pftd::utils::parser::SaveFileParser::EntityInfo, 35
  - entityType, 36
  - hp, 36
  - position, 36
  - sealID, 36
  - towerID, 36
- pftd::utils::parser::SaveFileParser::StatsInfo, 107
  - hp, 107
  - score, 107
  - wealth, 107
- pftd::utils::Random, 72
  - generate, 73
  - Random, 72
  - randomEngine, 73
- pftd::utils::Serializable, 96
  - deserialize, 96
  - serialize, 97
- pftd::utils::Vec2< T >, 113
  - distance, 115
  - normalize, 115
  - operator!=, 115
  - operator<<, 117
  - operator\*, 116
  - operator+, 116
  - operator-, 116
  - operator/, 116
  - operator=, 116
  - operator==, 116
  - Vec2, 114, 115
  - x, 117
  - y, 117
- pftd::ZombieSeal, 118
  - clone, 119
  - ZombieSeal, 118
- placeTower
  - pftd::Level, 55
- position
  - pftd::Object, 65
  - pftd::utils::parser::SaveFileParser::EntityInfo, 36
- price
  - pftd::Tower, 113
- priceLabel
  - pftd::GameScene::InventoryItem, 48
- print
  - logger.hpp, 143
- Projectile
  - pftd::Projectile, 70
- projectiles
  - pftd::Level, 56
- ProjSpawnFunc
  - pftd::Tower, 109
- pushQueue
  - pftd::Renderer, 77
- radiusPixel
  - pftd::Level::Nest, 63
  - pftd::Tower, 113
- Random
  - pftd::utils::Random, 72
- randomEngine
  - pftd::utils::Random, 73
- reachedNest
  - pftd::Seal, 95
- RegularSeal
  - pftd::RegularSeal, 74
- render
  - pftd::Renderer, 77
- Renderer
  - pftd::Renderer, 76
- reset

- pftd::utils::parser::Parser, 68
- ResourceManager
  - pftd::ResourceManager, 80
- returned
  - pftd::Seal, 95
- run
  - pftd::App, 17
- SaveFileParser
  - pftd::utils::parser::SaveFileParser, 84
- Scene
  - pftd::Scene, 87
- score
  - pftd::Level::Stats, 106
  - pftd::utils::parser::SaveFileParser::StatsInfo, 107
- SEAL
  - pftd::utils::parser::SaveFileParser, 84
- Seal
  - pftd::Seal, 92
- sealID
  - pftd::utils::parser::SaveFileParser::EntityInfo, 36
- seals
  - pftd::Level, 56
- selectedTower
  - pftd::Level, 57
- selectTower
  - pftd::Level, 55
- serialize
  - pftd::Level::Stats, 106
  - pftd::utils::Serializable, 97
- setBackground
  - pftd::gr::Button, 21
- setCallback
  - pftd::Clickable, 23
- setLabel
  - pftd::gr::Label, 51
- setMusic
  - pftd::Scene, 89
- setOutline
  - pftd::gr::Label, 51
- setPosition
  - pftd::Entity, 33
- setProjSpawnCb
  - pftd::Tower, 111
- setSound
  - pftd::gr::Button, 21
- setSpriteRect
  - pftd::gr::Sprite, 104
- size
  - pftd::Object, 65
  - pftd::utils::Container< T, C >, 27
- skip
  - pftd::utils::parser::Parser, 68, 69
- Snowball
  - pftd::Snowball, 98
- Snowballer
  - pftd::Snowballer, 99
- sourceStream
  - pftd::utils::parser::Parser, 69
- spawnProjectile
  - pftd::Tower, 113
- spawnSeal
  - pftd::Level, 55
- speed
  - pftd::Seal, 95
- Sprite
  - pftd::gr::Sprite, 102
- spriteSheet
  - pftd::Entity, 35
- startGame
  - pftd::GameScene, 40
- Stats
  - pftd::Level::Stats, 105
- stats
  - pftd::Level, 57
- target
  - pftd::Tower, 113
- toggleActive
  - pftd::GameScene, 40
  - pftd::Scene, 89
- totalElapsedSec
  - pftd::Entity, 35
- TOWER
  - pftd::utils::parser::SaveFileParser, 84
- Tower
  - pftd::Tower, 109, 110
- towerID
  - pftd::utils::parser::SaveFileParser::EntityInfo, 36
- towers
  - pftd::Level, 57
- towerToSpawn
  - pftd::GameScene::InventoryItem, 48
- update
  - pftd::Entity, 34
  - pftd::GameScene, 41
  - pftd::IcicleStabber, 44
  - pftd::Level, 55
  - pftd::MenuScene, 61
  - pftd::Projectile, 71
  - pftd::Scene, 89
  - pftd::Seal, 94
  - pftd::Snowballer, 100
  - pftd::Tower, 112
- updateScore
  - pftd::GameScene, 41
- updateWealth
  - pftd::GameScene, 41
- validLabel
  - pftd::utils::parser::Parser, 69
- value
  - pftd::Seal, 95
- Vec2
  - pftd::utils::Vec2< T >, 114, 115
- Vec2f
  - pftd::utils, 10

Vec2i

pftd::utils, [10](#)

wealth

pftd::utils::parser::SaveFileParser::StatsInfo, [107](#)

where

logger.hpp, [143](#)

x

pftd::utils::Vec2< T >, [117](#)

y

pftd::utils::Vec2< T >, [117](#)

zIndex

pftd::Object, [65](#)

ZombieSeal

pftd::ZombieSeal, [118](#)