Serialization and Atlan Mod Packager

Overview of serialization and Atlan Mod Packager

Overview

Doom Eternal's entitydefs, logic decls, and map entities files are stored natively in plaintext. What you see when editing them, is how they actually look when stored in the game files. At runtime, the game parses these text files and converts them into data.

Doom The Dark Ages changes everything. All of these files are pre-serialized into a binary format. Modding them now requires:

1. Deserializing them into a human-readable format. Atlan Resource Extractor does this for you.
2. Editing the files as desired. As a modder, you'll do this step.
3. Reserializing the files before distributing your mod. Atlan Mod Packager does this for you.
4. Installing the modded files using Atlan Mod Loader.

Those interested in the technical details can check out the source code for all these tools **here**!

Atlan Mod Packager

Download Atlan Mod Packager here!

Please read everything on the download page carefully! It tells you how to use Atlan Mod Packager, and how to optimize your workflow when developing your mods!

Serialized Files vs Regular Decls

This section highlights differences between regular Dark Ages decls and Atlan-Serialized files. You must keep all of these in consideration when editing these files.

Colors

Serialized files use *Linear RGB* for color values. Example:

```
127.5 / 255 = 0.5
my_idColor = {
	r = 0.5;
```

```
	g = 0.5;
	b = 0.5;
	a = 0.5;
}
```

File Paths

With normal decls, a reference to an entityDef file might look like this:

```
entity = "projectile_ent/my_projectile_entity"
```

Inside serialized files, this same filepath will look like:

```
entity = "entityDef/projectile_ent/my_projectile_entity"
```

The string before the first `/` is the file type. *To correctly reserialize the file, you MUST ensure the file type is there, and capitalized accurately.* See the end of this page for a list of correctly-capitalized file types.

Additional Example: Let's say we have an `idDeclAbility_Dash*` variable named `dashDecl`:

```
Full Asset Path: "rs_streamfile/generated/decls/ability_dash/modded/my_dash.decl

Regular Decl Files: dashDecl = "modded/my_dash";

Atlan-Serialized Files: dashDecl = "ability_Dash/modded/my_dash"
```

Type Info Object Pointers

If you've spent extensive time editing decl files, you'll have noticed variables with the following format:

```
some_variable = {
	className = "Class_Name_String";
	object = {
		<Object Properties>
		}
}
```

These are `idTypeInfoObjectPtr` variables. They are polymorphic object references whose exact class is determined by the `className` variable.

In Atlan-Serialized files, the syntax of these variables is simplified to:

```
some_variable "Class_Name_String" {
	<Object Properties>
}
```

The goal of this change is to simplify the syntax of these variables and remove a layer of indentation.

File Type Strings

List of all known, correctly capitalized file types. Others exist, but aren't encountered in the vanilla files and therefore aren't included here. If a type you need isn't listed here, you can likely guess the correct capitalization. They typically follow standard camel-casing rules. Obtaining a complete list of *every possible type* would require dumping data from the game executable.

```
ability_Dash
ability_ParrySwarm
ability_ShieldThrow
actorModifier
actorPopulation
advancedScreenViewShake
aiBehavior
aiBehaviorEvents
aiComponentList
aiDamageDeclCollection
aiDazeSettings
aiEnforcerAbilities
aiEvent
aiGlobalSettings
aiPoolNumbers
aiPositioningParms
aiUpgrades
aimAssist
ammo
anim
animCameraLensEffects
animWeb
attackgraph
audioLogStory
automap
automapProperties
collectibles
collisionShape
```

```
colorLUT
combatEncounterScoring
damage
damagemarkertype
destructible
devInvLoadout
dialog
ecozoneconfig
entityDamage
entityDef
env
equipmentLauncher
executions
explosion
extraLife
flare
footstepEvents
formationAttackParams
formationPositioningParams
fx
fxConditionLogicEvents
gameItem
geomcacheoptions
globalDataComponent
gorecontainer
gorewounds
gridSpaceData
handsBobCycle
highlights
hud
image
impactEffect
importance
interaction
inventoryItem
inventoryconversion
layer
lightrig
loadScreenBuckets
logicClass
logicEntity
```

```
logicFX
logicLibrary
logicObjectDescriptor
logicProfile
logicUIWidget
lootDrop
lootDropComponent
mapInfo
mapbrushes
material2
materialInteraction
model
modelSkinned
modelstream
notification
notification2
opacitymicromapdata
particle
perkGroups
perks
physicsObject
playableCharacter
playerMovement
playerProps
poi
projectile
propAttribs
propChargeBoost
propCoopRotateBob
propExtraLife
propHealth
propItem
propMoveable
propStatusEffect
propUse
questCategory
questDefs
renderParm
renderProgResource
rs_emb_sfile
rs_emb_sfilem
```

```
rumble
scoringRubric
sectorState
sectorremeshmodel
skeleton
soundPack
soundevent
soundrtpc
soundstate
soundswitch
statusEffect
strandsHaircg
string
swf
syncInteractions
table
targeting
throwable
transportControllerInfo
transportLayoutInfo
tutorialEvent
twitchPain
uiWidget
uiwalkthroughmenuargentcell
uiwalkthroughmenudossier
uiwalkthroughmenumodbot
vegetation
viewInfo_viewTuning
violenceEvent
visorSplatterEffect
waterThresholds
watergridconfig
weapon
weaponClass
windsource
```

Revision #6
Created 14 November 2025 16:11:03 by FlavorfulGecko5
Updated 28 July 2026 12:12:12 by FlavorfulGecko5