Atlan Assets Info

This section describes advanced settings available through your mod's configuration file. Please read **this page** to gain a basic understanding of the Atlan config. file before continuing here.

> These are advanced settings. Please take care when using them. Improper usage may cause instability or crashes.

PackageMapSpec Manipulation

Located in your game installation is the file `base/packagemapspec.json`

This file declares the resource and streamdb archives that are to be loaded in each level. By editing this file, you can change what resources a level loads. For example, if you wanted to use DLC AI in a base-campaign level, you can import the DLC level's archives to gain access to those assets.

The following is an example of a mod config. file that performs packagemapspec manipulation

```
modinfo = {
    name = "MapSpec Example"
    author = "FlavorfulGecko5"
    description = "Demonstrates how the MapSpec block works"
    version = "1"
    loadPriority = 1
    requiredVersion = 9
}
mapspec = {
"game/ripatorium/ripatorium$game/sp/m1_intro/m1_intro$.resources"
    "game/ripatorium/ripatorium$game/sp/m7_armada/m7_armada$.streamdb"
    "game/ripatorium/ripatorium$game/sp/m11_styx/m11_styx"
    "game/dlc/m15_hub/m15_hub$game/sp/m9_cosmic_a/cosmic_a$_patch1.resources"
}
```

First, let's introduce the format: `"FROM_MAP$TO_MAP$FILTER"`

- FROM_MAP is the map you want to import files from.
- TO_MAP is the map you want to import files into.
- FILTER is a filter string for specifying the exact files you want to import. This field is optional.

- $ characters separate the three fields.
- You can find all map names in the maps list, at the bottom of the `packagemapspec.json` file.

Now let's walk through each example in the above config. file:

`game/ripatorium/ripatorium$game/sp/m1_intro/m1_intro$.resources`

1. Imports all `.resources` archives loaded by the Ripatorium into Village of Khalim.

`game/ripatorium/ripatorium$game/sp/m7_armada/m7_armada$.streamdb`

2. Imports all `.streamdb` archives loaded by the Ripatorium into Sentinel Command Station

`game/ripatorium/ripatorium$game/sp/m11_styx/m11_styx`

3. This import statement has no filter string. Therefore, ALL resource and streamdb files loaded by Ripatorium get imported into Harbor of Souls

`game/dlc/m15_hub/m15_hub$game/sp/m9_cosmic_a/cosmic_a$_patch1.resources`

4. This imports `m15_hub_patch1.resources` into the first Cosmic Realm level. It's not recommended to use this feature this way versus importing all resource archives. However, if you really want just a patch archive imported for some reason, this is one way you can do it.

MapResources Injection

You can use the Config file to inject data into the game's MapResources files. To understand what MapResources files are, and why you'd want to edit them, see **this page**.

Atlan allows you to inject data into any level's MapResources file. This is accomplished using the mapresources and assets properties. See the following example:

```
modinfo = {
    name = "MapResources"
    author = "FlavorfulGecko5"
    description = "Demonstrates how the MapResources block works"
    version = "1"
    loadPriority = 1
    requiredVersion = 9
}
mapresources = {
```

```
"common" {
    my_asset_list
    }

    "game/sp/m1_intro/m1_intro" {
    LOAD_ALL
        ListTwo
        ListThree
    }
}
assets = {

    my_asset_list {
    "aiComponentList/custom/my_custom_component_list"
    }

    ListTwo {
    "damage/my_damage_decl"
        "projectile/my_projectile_decl"
        "entityDef/modded/cool_entity"
    }

    ListThree {
    aiComponentList/custom/another_component
    }
}
```

The workflow is straightforward:

1. Use the `asset_lists` block to define lists of game assets.

In your asset lists, the text before the first `/` character is the resource type. **The capitalization of the resource type matters, and must be correct!** Atlan Resource Extractor's extra scripts will provide a full list of correctly capitalized resource types for you.

This example config. defines three asset lists named "my_asset_list", "ListTwo" and "ListThree"

2. Use the `mapresources` block to list all maps whose MapResources files you want to edit. Then, add the names of your asset lists to their blocks.

LOAD_ALL is a special indicator. If encountered, Atlan will disable the layermask of all vanilla entries to ensure every resource in the file is always loaded. For Dark Ages, this shouldn't be necessary to set. For Eternal, you may want to set this when doing extensive entities modding and encountering stability

issues. Read **this page** to understand what all of this means.

This example config does several things:

- The asset list `my_asset_list` is injected into `common.mapresources`
- The asset lists ListTwo and ListThree are injected into `m1_intro.mapresources` and LOAD_ALL is enabled on it.

Practical Usage

For the majority of mods that add new game assets, the following setup should be enough to ensure your custom assets always load correctly. Adding assets to `common.mapresources` will cause them to be globally loaded.

```
mapresources {
    "common" {
        new
        }
}
assets {
    new {
        "Your Custom Assets"
        }
}
```

Edge Cases

There are some situations where you'll need to edit level-specific mapresources. For example, let's say you add a new asset that references a DLC AI file only loaded in the DLC levels. What happens if you put this new asset into common's MapResource file?

1. Your custom asset will be loaded on game startup
2. Since your asset has inaccessible, level-specific dependencies, they won't get loaded. This leaves your asset in a semi-broken state. Since it's loaded by the common MapResources, the game will never reload or refresh this asset during level transitions.
3. When it comes time to use this game asset, it won't work properly. Your game may have even crashed before this point.

The "correct" action in this situation is to edit the level-specific MapResources where this asset will be used, instead of the common MapResources file.

Modder's Resources

Atlan Resource Extractor provides information to help you correctly edit the MapResources files. By running the *extra scripts*, you receive:

- A list of all correctly capitalized resource types
- All vanilla MapResources files decoded into plaintext, allowing you to see what assets are globally loaded, and which are level-specific.

Revision #12
Created 9 August 2026 21:29:22 by FlavorfulGecko5
Updated 9 November 2026 15:14:16 by FlavorfulGecko5