Cozy Caves

A D&D Dungeon Generator

Room Generation Module

Written By
Abdulrahman Asfari

Reviewed By
Gavin Lim
Gideon Wilkins

Technical Layout - Rooms

Encapsulation

This process is vital to creating abstraction in this module, as well as preventing control over parts of an object that should not be changed outside of the module. Encapsulation will largely take form using the JS private access modifier. As there is no support for the protected modifier, abstracted objects should generally avoid using inheritance unless necessary.

Module Inputs

When the dungeon generation module attempts to access this module, there should be a high level of customizability when requesting a room in order to meet the generation algorithm's exact needs. Because of this requirement, there may be additional parameters added to the room generation inputs as development progresses, and for this reason I will be using the builder pattern. This not only makes using the module more maintainable due to not having to rewrite the constructor call, but also allows the caller to omit options they are not concerned with.

Allocated Size

Default: None - REQUIRED PARAMETER

Attempt Fill Size Allocation

Default: No Fill

Allow Non-Rect Shape

Default: Allow Non-Rect Shapes

Access Point Sides

Default: None

Each time this option is used in the builder it will add another access point to the room. If multiple access points are requested on one side when the selected room shape does not support this, then the unsupported access points will be ignored. Ideally, dungeon generation will not be strictly needing access point locations more specific than a chosen cardinal direction, and so the algorithm should be able to adapt to the returned room.

Optional Access Point Params

Access Point Width - Overrides default access point width.
Preferred Segment - Each side is split into X segments, this indicates a preferred but not guaranteed segment to place the access point in.

Default Access Point Width

Default: 1 Tile Wide

Populate with Items

Default: Do Not Populate with Items

Populate with Props
Default: Populate with Props

Tileset
Default: The First Tileset Created
This will have no functional use, but is more just future proofing the system. Considering tilesets this early allows them to be integrated with all other systems and so they will be seamless to add should we choose to include the option to have different tilesets.

Blacklisted Layouts
Default: None
If a room is generated and does not match the dungeon generation algorithm's needs, but there is no option to further fine-tune the room request, then this option exists to request a room, excluding a given layout from the pool of possible rooms.

Room Data Structure
Rooms will be returned in a format more complex than a 2D array. This is more intuitive but allows for a linear iteration method, whereas creating my own data structure allows for more utilities, whilst also providing anything a 2D array would.

Room
This will be the returned object. It will contain a collection of tiles as well as methods to extract useful information, such as access points, listing tiles sorted by depth for rendering purposes, or effective dimensions which will be different to grid dimensions due to inset walls. There might also be utility methods such as room rotators, based on the demands of the dungeon generation module.

Tile
A basic data structure containing:
- Tile Source
- Position
- Offsets
- Rotation
- Depth

Tile Source
Contains image source and dimension.

Access Point
Contains all positions the access point takes up, as well as general information such as its width.

Tile Type
Emulates an enum that dictates all possible tile types and their roles

Tileset

Each tileset maps each tile type to a tile source. It also contains metadata about the tileset that will affect room generation logic such as whether the room walls are inset or not, or if blending floors are used.

Generation Pipeline

Inputs - Process inputs from the room builder.
Choose Shape - Decide on room shape from a pool.
Generate Room - Creates the room using tiles, to represent the chosen shape.
Insert Access Points - Insert access points to the existing room, if any.
Populate Room - Send room to prop & item module, if the option is selected.

Hallways

Hallways will not be considered for this first iteration of room generation. This is because generating hallways will require radically different parameters that are highly dependent on how the dungeon generation module decides to approach hallways.

Room Shapes & Layouts

This iteration of room generation will only have a mocked up version of the shape and layout system. This system will be more fleshed out and customizable in future iterations but is not a focus for now.