Advanced MUD Project

Contents

[Overview 1](#_Toc189679135)

[The Nature of this Document 1](#_Toc189679136)

[Definitions 1](#_Toc189679137)

[Players 2](#_Toc189679138)

[Non-Player Characters 2](#_Toc189679139)

[Items 2](#_Toc189679140)

[Procedures 2](#_Toc189679141)

[Rooms 2](#_Toc189679142)

[Movement 2](#_Toc189679143)

[Zones 2](#_Toc189679144)

[The World 2](#_Toc189679145)

[Detailed Requirements 3](#_Toc189679146)

[Structural Requirements 3](#_Toc189679147)

[Rooms 3](#_Toc189679148)

# Overview

## The Nature of this Document

This document attempts to approach AMP’s design and requirements independent of implementation or presentation. So we strive to say what it must do and must be able to do, not how it does it or how it looks and feels.

## Definitions

|  |  |
| --- | --- |
| Name | A string no longer than several words. Some names must be exactly one word long. |
| Direction | North, south, east, west, up, or down. Dir is short for direction. |
| Message | Text sent to a group of characters. |
|  |  |
|  |  |
|  |  |

## Players

Players are real people subscribing to AMP. A player controls a character, his or her manifestation in the game world.

## Non-Player Characters

A Non-Player Character (NPC) or mobile (mob) is anything alive in the game world that isn’t a player.

## Items

An item is a physical object in the game world. Every item is either being carried by a player, carried by a mob, in a room, or in a container. A container is a special type of item that may contain other items.

## Procedures

Procs make the game come to life, and are usually content-specific (part of the “story”). A proc can be characterized as:

1. Attached to an entity (player, mob, item, room, exit of a room, or zone)
2. If something happens (involving that entity)
3. Then something else happens (usually involving that entity)

## Rooms

Rooms are characterized by a name, description, zero to six exits, zero or more procs, and exactly one containing zone. A room may contain objects, NPCs, and players. If room B is an exit of room C, a character may move from C to B.

“A Well-Kept Bakery”, “Before the Blood-Soaked Gate”, and “A Wide Spot on Holly Street” are examples of rooms.

## Movement

A character moves from room to room by going north, south, east, west, up, or down.

## Zones

A zone is a set of procs acting on a set of rooms. A zone specifies certain distinguished members of the room set to be entrances to the zone.

“Castle Dracula”, “Drulstrider Swamp”, and “The Volcavi Circus” are examples of zones.

## The World

AMP’s world is a collection of regions. Each region is a grid and contains wilderness tiles and zone tiles. All tiles are rooms; players may move between adjacent tiles. Certain types of wilderness tiles may require a condition to enter/exit, but all zone tiles are passable. Zone tiles are the entrances to zones.

Wilderness tiles tend to be more generic, as there are many of them. “Dense Jungle”, “An Overgrown Trail”, and “On a Clear River” are examples of wilderness tiles.

# Detailed Requirements

## Structural Requirements

### Rooms

#### Room Instances

A room instance is an instantiation (copy) of a room. Whereas rooms are ‘blueprints’ that do not physically exist, room instances are physical parts of the game world. A room instance belongs to exactly one zone. The term “room” in general refers to a room instance. This permits a room to be instantiated (copied) as many times as necessary.

#### Uniqueness

Any room is unique among all rooms. Any room instance is unique among all instances of the same room.

#### Room Attributes

* Room name. A room name may change at any time, most likely as the result of a proc.
* Room description. A room description may change at any time, most likely as the result of a proc.
* A room may have zero or more procs. The condition on each proc should involve the room somehow. Procs always act on room instances. Therefore a proc that would change a room name would change the name of a particular instance, not the ‘blueprint’ room.
* A room may have up to six exits.

#### Room Exits

A room may have up to six exits, one for each of north, south, east, west, up, and down.

* An exit leads to exactly one room.
* An exit has an “open” name, usually the name of the room it leads to.
* Default messages
  + Exist to notify when a character exits through or enters from an exit.
  + Exist to notify when a character attempts to exit through a closed exit.
* An exit may specify a custom exiting message, or a custom entering message *for the room the exit leads to*. These custom messages only apply when using this exit.
* Exits may be illusionary. Illusionary exits appear not to exist (are not displayed) but react normally when a character attempts to move in that dir.
* Exits may be hidden. Hidden exits appear not to exist (are not displayed) and also appear not to exist if a character attempts to move in that dir. Therefore, hidden exits may not be used unless they are unhidden by a proc.
* A room may specify an exit to be a wilderness exit. In this case the exit will lead to an unspecified wilderness tile, and the room’s zone is responsible for determining which tile.
* Exits may have procs and they may modify exit attributes.

#### Closed Exits (Doors)

* An exit may have a “closed” name and so may be closed.
* If an exit may be closed, the exit displays whether or not it is closed.
* If an exit may be closed, it has a “door” name.
* Default messages using the door name exist to notify when a character
  + Attempts to move through a closed exit.
  + Attempts to open/close a door.
  + Successfully opens/closes a door.
* A condition may exist to open the door (ie possessing a key).
* A potentially different condition may exist to close the door.

#### Room Zone

A room instance belongs to exactly one zone, and this zone may change at any time. A room (blueprint) belongs to no zone.

#### Initial State

A room’s initial state (inherited from the blueprint) is:

* The room’s name, description, exits, and procs.
* A list of objects to load.
  + Each object must be in the room itself or in a container.
  + For each object, a condition may exist to load the object.
* A list of mobiles to load; for each mobile, a condition may exist to load the mobile.

### Zones

#### Zone Instances

Similar to rooms, a zone instance is a copy of a zone. Zones are the blueprints that do not physically exist, and zone instances are physical collections of rooms in the game world. The term “zone” in general refers to a zone instance. This permits a zone to be instantiated (copied) as many times as necessary.

#### Uniqueness

Any zone is unique among all zones. Any zone instance is unique among all instances of the same zone.

#### Zone Attributes

* Zone Name. A zone name may change at any time, most likely as the result of a proc.
* Procs. A zone may have zero or more procs.