Populous: The Beginning – Level File Format & Single Player Creation Guide

This document is written to be consumed by both humans and AI agents (e.g. Codex). It describes, in full technical detail, the on-disk level file formats used by Populous: The Beginning, along with a correct, editor-safe workflow for creating a working single-player level using Multiverse World Editor and Lua scripting.

1. Overview of Level Components

A playable Populous: The Beginning level is composed of the following files:

- **levlXXXX.dat** – Binary terrain + object data
- **levlXXXX.hdr** – Level header (players, spells, buildings)
- **levlXXXX.ver** – Editor-generated version/metadata (Multiverse only)
- **Lua script** – Single player logic (Populous 1.5+)

2. levlXXXX.dat – Binary Map File (FULL STRUCTURE)

File size: **192,137 bytes (0x2EE89)**
Endian: **Little-endian** for numeric values.

2.1 Heightmap Section

Offset: **0x00000 – 0x07FFF**
Structure: 128 × 128 grid of signed 16-bit integers (little-endian).
Total size: 32,768 bytes.

Height conventions used by stock campaign levels:

- 0 = Sea level
- ~200–260 = Shoreline
- ~256–700 = Playable land
- ~700–900 = High ground

IMPORTANT: Heights outside this range cause flooding, AI failures, or editor crashes.

2.2 Object Table Section

The object table begins at a template-dependent offset (commonly ~0x149EE). Each object record is exactly **55 bytes**.

Object Record Layout (first 8 bytes):

```
Byte 0: Model ID
Byte 1: Type ID
Byte 2: Owner (0=Blue, 1=Red, 2=Yellow, 3=Green, 255=Neutral)
Byte 3: Flags
Bytes 4–5: X coordinate (unsigned short, little-endian, 0–255)
Bytes 6–7: Y coordinate (unsigned short, little-endian, 0–255)
```

CRITICAL RULE: Multiverse World Editor requires that object tables either be created by the editor itself or be copied from a known-good template. Hand-constructed object tables will crash the editor during validation.

3. levlXXXX.hdr – Level Header

Size: **616 bytes**
Purpose: Stores player count, alliances, available spells, buildings, and UI metadata.

The header file can be edited cautiously, but Multiverse may rewrite it on save. Never attempt to hand-author a header without an editor-generated base.

4. levlXXXX.ver – Version File (Multiverse)

This file is **private to Multiverse World Editor**. Its format is undocumented and cannot be reliably generated programmatically.

Rule: Always let Multiverse create and manage the .ver file. Replacing or fabricating this file will result in 'Unknown level format (v0)'.

5. Correct Workflow to Create a Single Player Level

1. Create a new level in Multiverse World Editor and save it.
2. Use that level as the template for all future edits.
3. Modify terrain by replacing only the heightmap portion of levlXXXX.dat.
4. Add objects (shamans, huts, reinc sites) ONLY via the editor.
5. Use Lua scripts (Populous 1.5+) for AI and win/lose conditions.

6. Single Player Logic (Lua)

Single-player behaviour is implemented via Lua scripts loaded by the game (not by the editor). Typical responsibilities include AI setup and victory conditions.

Minimal example:

```
if IS_SHAMAN_DEAD(TRIBE_RED) ~= 0 then set_level_won() end
if IS_SHAMAN_DEAD(TRIBE_BLUE) ~= 0 then set_level_lost() end
```

7. Summary for Codex / AI Consumption

- Terrain data: safe to generate programmatically within standard ranges.
- Object data: must come from editor or a verified template.
- .ver file: editor-only, never fabricate.
- Lua: preferred mechanism for single-player logic.