BCI-RPG Character Conversion Documentation (In Progress)

Original Author: Doug McCord
Original Date: April 27, 2025

REVISIONS:

Version	Date of Change	Revised by
1.0.0	27-April-2025	Doug McCord

Table of Contents

Introduction

1. **Statement of Purpose**: This document is WIP notes for converting characters for the BCI-RPG game. At this time, that is primarily to move from an RPG game system (such as OpenD6 or Dungeons & Dragons) to our back-end system. The goal is to remain flexible on potential game systems in use, by taking in characters of a given system and converting them into our percentile-based back-end.

2. **Backend Character Reference**: Our backend character system used by the game is the BCI-RPG Generic Testing RPG System Mechanics. At the time of this writing that's version 20220925c, available in the project Github here: Phase2/Documentation/BCI-RPG-temporary-Simple-Reference-Implemntation-System-20220925c.pdf

3. **Conversion Process**: While all game mechanics are run using the backend percentile system, the expected use is that games will be run with a different game system abstraction layer (GAL). Players, as a result, will see their character displayed in-game, both via character sheet and by adding new or importing sheets, in the system they prefer. It is therefore necessary for the game to convert from provided characters into the backend system detailed below.

4. **Scope Limitations and Preconditions**:

    4.1. For the initial release of the game, in-game stats will not be displayed, aside from on the character sheet. Rolls and stats used will therefore not need to be converted regularly, primarily for character creation and importing.

    4.2. Initial game systems supported are OpenD6 and Dungeons&Dragons only.

Backend Percentile Character System

1. **Overview**: To allow for maximum flexibility, the backend character uses the abilities and rules referenced above, translated into Godot in the form of a PlayerCharacter. This instantiates the PlayerCharacterTemplate resource. Each character contains a few text fields (name, profession, and quote at this time), and an array of **capability** objects. All game mechanics are handled by individual capability objects which are meant to allow for maximum flexibility. Ability scores, hit points, spells, super powers, key equipment, health, skills are all examples of things that can be made out of individual capability objects.

2. **The Character Capability**: Aside from a few text fields that provide character notes and details, characters in the backend are made from a series of capability objects. Each contains:
    2.1. **name** (string)
        2.1.1. Purpose: Label the specific capability.
        2.1.2. Examples: APP, Light Spell, 9mm Handgun, Health
    2.2. **score** (integer)
        2.2.1.Purpose: Percentile number to reflect the core numerical value for the capability.

2.2.2. Example: 54

2.3. **attack** (Boolean)

2.3.1. Purpose: True/False flag for if this capability can be used to attack, in combat. If True, it will be included in list of attacking options for player during combat, should it also have uses above 0 or -1 (see below)

2.3.2. Example: True (for 9mm Handgun, Magic Missile, Martial Arts Skill), False (APP, Chainmail Shirt, Cooking Skill)

2.4. **defend** (Boolean)

2.4.1. Purpose: True/False flag for if this capability contributes to defense. The game polls capabilities for true defenses and tallies their scores and modifiers to set a character's current defense, for combat rounds.

2.4.2. Example: True (Chainmail Shirt, Martial Arts Defense, Force Field Generator), False (9mm Handgun, Light Spell, ME)

2.5. **use_range** (integer)

2.5.1. Purpose: Numerical value in feet for how far the capability extends.

2.5.2. Example: 75, 0

2.6. **duration** (integer)

2.6.1. Purpose: Numerical value for rounds that something lasts, or -1 for NA.

2.6.2. Example: 10, -1

2.7. **impact_target** (string)

2.7.1. Purpose: Capability in a target that is impacted by this capability. Expects text/string. Note that combat capabilities (attack=true) expect an opponent's health to impacted so this is not necessary to repeat here.

2.7.2. Example: ST (Vampiric Touch Spell)

2.8. **impact_amount** (integer)

2.8.1. Purpose: Numerical value for how this target capability is impacted. This is different from score, which could reflect on usage ability. For combat, this refers to damage.

2.8.2. Example: 25

2.9. **uses_max** (integer)

2.9.1. Purpose: Numerical value for the number of uses a capability can have at most. This can refer to spell slots, or ammunition, for example.

2.9.2. Example: 13

2.10. **uses_current** (integer)

2.10.1. Purpose: Numerical value for the current uses a capability has left, or -1 for NA. Note a 0 indicates the thing cannot be used until it gets more uses_current added (see below).

2.10.2. Example: -1, 10

2.11. **recharge** (Boolean)

2.11.1. Purpose: True/False flag for if a capability refills on its own, over time. Such as a spell, or power, or energy weapon that needs time to recharge.

2.11.2. Example: True

2.12. **reload** (Boolean)

2.12.1. Purpose: True/False flag for it a capability can be refilled by something being applied in-game, such as finding more bullets, for example, or a bottle of potion that is being drained.

2.12.2. Example: True

2.13. **modifier** (integer)

2.13.1. Purpose: Numerical value for the current impact to the score. This is used for fluctuations or temporary increases or deductions.

2.13.2. Example:10

3. **Starting/Default Character Capabilities**: The game assumes the following capabilities will be set by default, from the BCI-RPG Generic Testing RPG System Mechanics:
   3.1. source_backend_capabilities = ["AG","APP","CO","QU","MD","ST","CH","EM","IN","ME","MX","PR","RE","SD"]
   3.2. Each has a name (indicated above, “AG” for example) and a score (out of 100, percentile system).

4. **Calculated Default Character Capabilities**: In addition, the following capabilities are generated using starting defaults or from the above abilities:
   4.1. "Health","Defense","Vision"
      4.1.1. **Health**: The backend system for damage, this is set from the CO ability plus any provided modifiers at game start and used for damage taken. This has a name “Health” and a score (from “CO”). Modifier would track current boosts or losses to the score.
      4.1.2. **Defense**: The backend system to be used to defensive protection, this is generated at game start by polling all capabilities for a “true” in defense. It then adds all scores for these capabilities and their modifiers to set its own score for a character’s current defense. Depending on the game system, this could be armor, for example, plus dodging ability.
      4.1.3. **Vision**: A current placeholder for the ability to see, this capability could be modified by darkness, light sources, dark/blinding affects, etc.)

Example Uses

1. From the source PDF: picking a lock.
   - Name: Lockpicking
   - Score: 35
   - Attack: N
   - Defend: N
   - Range: 0
   - Duration: 0
   - Impact: 0
   - Target: 0
   - Uses_Current: -1
   - EXAMPLE: Average lock, as in PDF, has a 0 modifier. So to pick it, the character has to roll a 35 or lower on D%.

2. Casting a light spell.
   - Name: Light Spell
   - Score: -1
   - Attack: N
   - Defend: N
   - Range: 0
   - Duration: 10
   - Impact_Target: Vision
   - Impact_Amount: 50
   - Uses_Current: 1
   - Recharge: Y
   - Reload: N
   - EXAMPLE: In module, group falls through a trap into darkness. All Vision scores are modified to 0. Game polls to activate light sources, by checking for an impact_target of

vision and available uses. Character with current uses available can opt to use it. The score of -1 demonstrates automatic success, so once cast the impacted party member gains vision of +50. Uses are decremented, so in this example the player now has 0 and won't be able to use it again until a camp/rest (recharge of true).

3. A detective fires a pistol in combat.
    - Name: 9mm Pistol
    - Score: 54
    - Attack: Y
    - Defend: N
    - Range: 30
    - Duration: 0
    - Impact_Target: Health
    - Impact_Amount: 25
    - Uses_Max: 13
    - Uses_Current: 13
    - Recharge: N
    - Reload: Y
    - EXAMPLE:
        - In combat, since this has Attack=Y, it's presented to player on list of actions. Use Pistol? Selected, then as other combat, a target selected. (Range check may be out of scope, but otherwise range checked.) When fired, difficulty is set for defender's ability + defense.
        - This detective is firing at a Sentry with Athletics skill (Score: 20, Defend: Y) and an Armored Vest (Score: 5, Defend: Y), giving him a Defense score of 25.
        - The Sentry is at point blank range (+20), so the player needs to roll lower than a 49 on a percentile roll (Score: 54 + Modifier:0 – Sentry Defense (25) + Range (+20).
        - Player rolls a 34, a hit.
        - Now the Sentry takes damage, their Health reduced by 25 points.
        - One Uses_Current is removed from the capability.