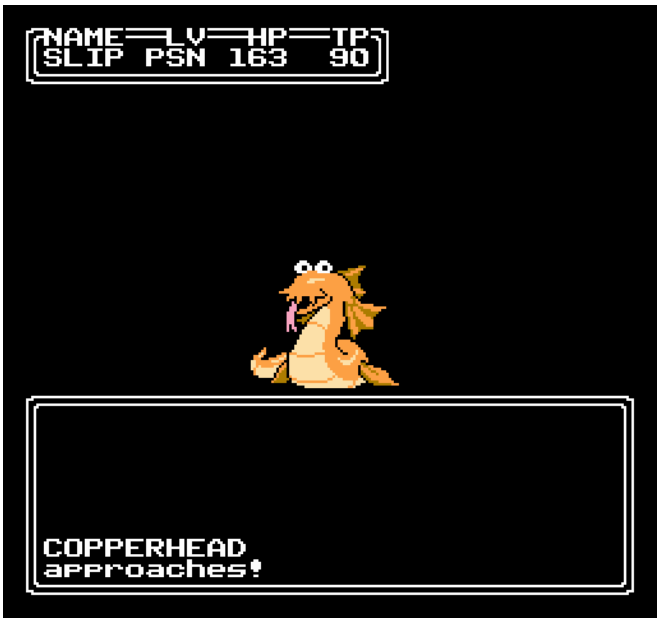
Achievements

During a dev stream, I was asked about GodotSteam and how difficult it had been to implement into Crop and Claw. The short answer was “it wasn’t” but the longer answer benefits from a brief instruction guide on how one can implement achievements into their project, not as Steam achievements, but as an abstracted concept.

There are a number of achievements in Crop and Claw, some with similar conditions (defeat a specific enemy), some with completely unique ones. Consequentially, this would mean having to drop Steam achievement triggering code in random places throughout the project, and that is always a pain. For example, perhaps you’d rather not run Steam all the time, as Crop and Claw does. If Steam code runs, it’ll just throw an error. You may even go as far as to build the game without GodotSteam linked to it, in which Steam code would throw compile errors without some dumb workaround being applied everywhere. Not ideal.



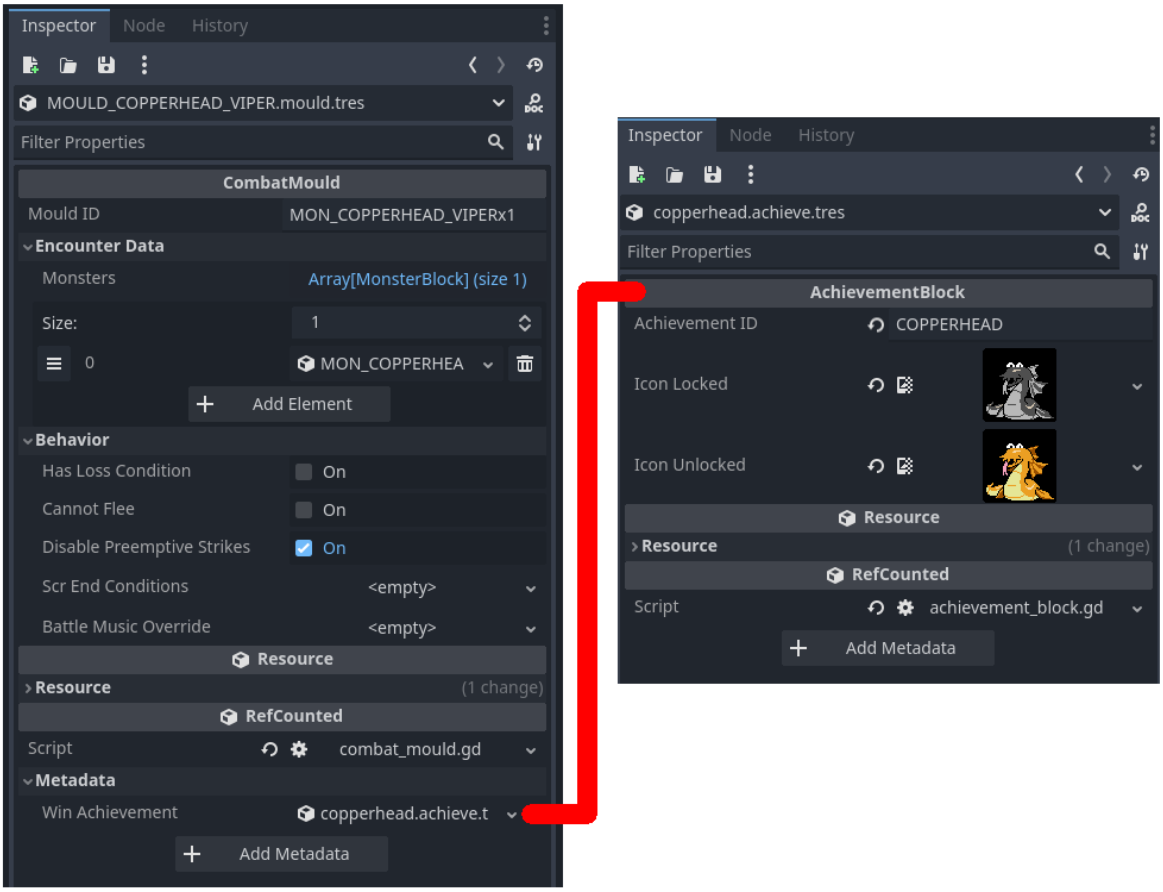
In Crop and Claw, one achievement is to defeat a rare encounter that serves a similar role as WarMech from Final Fantasy. That is to say, a creature that appears in a very specific area, with a very low encounter rate, and is deliberately overpowered compared to enemies around. Upon approaching death, they can even cast the strongest offensive magic in the game. Defeating a Copperhead is purely optional, but woe to those unfortunate enough to happen into them unexpectedly. You will not be able to submit CRT TV photos of encountering Copperheads for a chance to cameo in Link to the Past.

A naive implementation of this would be to just directly inject Steam calls into the combat end subroutine, in which we compare the encounter to the Copperhead id and flag the achievement. The

way CnC1 does it is not terribly complex, but far easier to expand upon. In fact, the only achievement code that's executed at combat end related to achievements of combat victory is this:

```
G_COMBAT.CombatResult.VICTORY:
    if mould.has_meta(WIN_ACHIEVEMENT_META):
        var win_achievement: AchievementBlock = mould.get_meta(WIN_ACHIEVEMENT_META)
        if win_achievement:
            G_ACHIEVEMENTS.unlock_achievement(win_achievement)
    await scr_combat_end.execute_rewards(self)
```

Achievement code was added very late into CnC1's development, and the majority of encounters wouldn't use them. Luckily I discovered metadata around this point, which is just a dictionary of Variants that Godot just tacks onto everything. Completely optional, but handy to have for edge-case stuff. This code just checks if an enemy mould has an achievement attached. Both of these are custom resources.



At bare minimum, all you need is this AchievementBlock resource which, when instructed to flag an achievement on Steam, does so. However, Crop and Claw does add a few extra steps that are worth bringing up. In the code above, the G_ACHIEVEMENTS global will perform more than just Steam achievements. Instead, you may store achievement progress locally, either as an ini, resource, or custom format, with optional encryption if you really wanted to. This allows you to cache if an achievement has been flagged or not, even if the game was not running in Steam mode (or any achievement tracking/analytics system you may want) which means, upon rebooting the game, it now

becomes trivial to check the achievements file for updates and flag anything set there. Yes, this means manipulation and injecting someone's completed list is possible, but the majority of players probably won't care, and you can work around that if you really wanted to. This also means you can have all your achievements stored in a database folder, able to be linked up to anything either via inspector and dynamic calls, or directly preloaded into specific scenario code. It also lets you change achievement ids should you have changed them on the server, or locally associate images to achievement sprites should you want achievements to be visible from in-game without Steam support for easy retrieval and display.

You may also notice that I've been writing about making Steam support optional. If you aren't using Steam as DRM but rather as an additive feature, you can configure the project to boot Steam systems only when an argument, such as --steam, is passed in on game launch. This is how Crop and Claw does it. The main difference between the Steam and itch builds is that Steam's launcher runs with --steam, and the game will branch to boot GodotSteam and flag the game in Steam mode. This way, achievement flagging from G_ACHIEVEMENTS will store both local achievements and Steam. This also means, if I need to completely remove GodotSteam, the only code that needs commenting out are centralized to two primary places: --steam argument initialization and Achievement handling. There is no Steam code scattered about Crop and Claw's source.

This is just a basic implementation. For Crop and Claw 2 and future projects, it is not unlikely I will be expanding on this to allow for more powerful control of what gets tracked and how. It doesn't take much, just having an internal goal of needing to track achievements both locally and on Steam means having better infrastructure, and thus can extend to other server-based integrations that might do achievements, even your own should you be inclined to link your games to a custom server.