So a while ago I was DMing for my friends and we were approaching the end of our 2-year long campaign – after a rough week at work I realized that we were playing in about half an hour and I had done zero prep for that session. I knew it was going to be a heavy combat session and I had finished scripting this really basic, sort of abstract generator to fuel my own improv-heavy DMing style. So I used it. And now I’m here to share with you all about how it changed the way I run my D&D games.

Hi! My name is Atty, and I’m an associate game designer at Improbable where I’m working on Redacted and doing cool Redacted things. I’ve also been DMing games for 3 years and I love designing within 5e’s rules structure… And this is what it looks like when I start making DM tools to empower myself as a DM: (pause)

Now you might be thinking – Atty, what the heck is this? Understandable, allow me to break the illusion of how this amazing tool works and then show you how genius \*cough\* basic the code actually is. (insert crazy board meme here)

Now immediately you might notice the level of abstraction I’ve chosen if you’re at all familiar with the 5e system of D&D. The TTRPG system offers basic rules for “what can a player do on their turn, what can a monster do on its turn” within the various books. Generally, on a turn players can do an action, bonus action, move, interact with an object – all in the span of 6 seconds of in-game time, that’s 1 round of combat for the players. 5e is a very homebrewable system, and that’s because the rules are just stiff enough that they can be flexed in different ways to create cool custom content. With the rules in place the next thing I needed was a framework or philosophy of design or gameplay that I wanted to steep my generator in.

So the code itself uses Matt Colville’s philosophy of action oriented monsters as a framework or syntax for generation, the design truth that player character design is asymmetrical to monster design, a truth that I think DMs should lean heavily into – I’ve been using this combat philosophy for 2 years now and it has worked very well for me so I decided to model the generator around it. The only real difference is that Colville recommends getting into a monster’s headspace, figuring out the themes and such before designing its abilities and my generator focuses on getting the numbers and semantics out of the way first and then *using* those semantics to create a story for the monster.

Ok so tiny code tour time – All I’m really doing is pulling from a bunch of enums and using pre-set difficulty probabilities to dictate stats, # of actions and the kind of dice they use

I have a bunch of enum lists with various game relevant data as well as meta-suggestions. For different difficulties I have different min and max ranges for determining universal integer modifiers for stuff like health, spell slots etc.

Actually

1. Combat generator details
   1. Action oriented combat as the framework/rules/syntax for generation
   2. Room for interpretation
2. Lessons
   1. Yes-and during the prep period for the DM
   2. Suited to your improv style – what would it look like for a rules heavy DM?
   3. Learning to embrace the chaos and having fun with it
      1. Fear in your players eyes
   4. What else could we generate in our TTRPGs?
      1. Encounters? NPCs? Dungeons?
      2. Focus less on the preparation and turn your attention to having fun in the session
         1. No more – “fuck I didn’t expect that”, “this is going off the rails and I’m anxious”
         2. Personally, not knowing what the session was going to look like I had an excuse for not having all the material prepped
3. I coded it over the course of a couple of weekends to brush up on my python after spending way too much time in blueprints