Dungeons and Dragons 5E Character Sheet Creator

Andrew Newbill and Logan Brown

Abstract

Andrew Newbill and Logan Brown are proposing a D&D 5E character sheet creator. This tool would use C# and WPF to provide an easy to use interface. Its main purpose is to create character sheets for the role-playing game Dungeons and Dragons which can be saved or loaded at will. We are targeting users who want an offline tool that is easy to use. This project aims to help new players through the creation of a character sheet(which defines the character in its entirety)so that they may begin playing without a Dungeon Master manually instructing them through the process. This project will pull from an existing database all of the items, spells, abilities, races, classes and other information necessary to define a character. Interactivity will be minimal although dice rolls for attacks and other actions are currently being targeted.

1. Introduction

The idea for a character sheet creator was borne from the need for a free tool for character creation that is fully usable offline. We are designing this tool with the aim of walking the user through the process of creating a character sheet for Dungeons and Dragons. The program would allow the user to fully define their character including: name, avatar, age, race, sex, statistics, origin, abilities, spells. One of the most common issues with playing Dungeons and Dragons is understanding how to properly create a character. Our project would alleviate this issue and allow users to start playing in the same session that they create their character. We are targeting those new to Dungeons and Dragons, since other more advanced tools exist but are often complex to the point of discouraging play. This tool will allow new users to invest nothing monetarily in order to begin playing. Simplicity is our overarching goal for this project in order to draw more players into the hobby. In this tool's final iteration it will need to allow fully creating a new character, and loading an existing character for viewing. We will also include exporting to image files so that the sheets may be printed and used for physical play.

1.1. Background

We decided upon this project quite quickly whilst brainstorming. We both have a passion for role-playing games and also recognized the need for this tool. Our goal is to get players into the hobby and we hope that this tool will facilitate that. Our experience playing the game will ease some of the research necessary in order to create this tool.

1.1.1. Terms To Know.

- Dungeons and Dragons - A pen and paper role-playing game featuring fantasy elements.
- Character Sheet - A physical or virtual document which defines your character.
- Abilities - actions a character can perform that are not basic attacks.
- Spells - magic that a character can cast.
- Statistics - a character's traits, such as Intelligence or Strength, that are represented numerically.
- Origin- background information for a character.
- Avatar - a picture which visually represents your character.

1.2. Impacts

This program is planned to be free when released, so that it is accessible to everyone who can find it. In this way, it will spread quickly, allowing many of people (who may be unfamiliar with Dungeons in Dragons) to have what they need to have fun. The impacts that this will have are:

- Widespread encouragement of problem solving skills that are not exercised often
- Practice with minor mental math that many people frequently struggle with
- A way to make new friends, or connect with existing ones
- A general happiness and relief due to the simplicity of the application

1.3. Challenges

Initial thoughts on challenges are many before one even starts coding. But here I think are three of the biggest, foreseeable issues that will come up during the project.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Add item to cart	Shopper	Med	1
2	Checkout	Shopper	Med	1

TABLE 1. SAMPLE USE CASE TABLE

1.3.1. The GUI. - Characters are complex, and need a lot of information in order to properly function in game. There is a lot of information, and the user has to be able to hold all of it in a GUI form. The KISS method will be our saving grace.

1.3.2. Information. - As mentioned before, there is a lot that goes into a character. With offline functionality being a basic goal, I can see a long time of just punching in the data.

1.3.3. User Input Error. - There will be a big mix of Integers, Strings, Doubles, and even more Strings in this project. Being sure that users are allowed to make mistakes with input *without* crashing the program is a big consideration. A possible error handling function for a few of the common data types might not hurt.

2. Scope

This project should meet multiple standards that are required in order to create and view a 5th Edition Dungeon and Dragons Character. A lot of these requirements are big and small, but are all equally important to the feeling of ease when creating your character.

2.1. Requirements

Here are the following minimum requirements, and under any of these can exist a stretch goal that under them improves upon the base idea:

- Have a Minimalist GUI for Character creation and viewing
  (Stretch Goal) Have an enhanced GUI that does not bore whoever uses it
- Full offline functionality
- Ability to create a 5th Edition Dungeon and Dragons Character (Handbook only)
  (Stretch Goal) User can add custom information, and use it how they like
- Need no outside tools to create a character (e.x. Dice, Character Sheet, Source Books, etc...)

2.1.1. Functional.

- User needs to have a private shopping cart – this cannot be shared between users, and needs to maintain state across subsequent visits to the site
- Users need to have website accounts – this will help track recent purchases, keep shopping cart records, etc.
- You'll need more than 2 of these...

2.1.2. Non-Functional.

- Security – user credentials must be encrypted on disk, users should be able to reset their passwords if forgotten
- you'll typically have fewer non-functional than functional requirements

2.2. Use Cases

This subsection is arguably part of how you define your project scope (why it is in the Scope section...). In a traditional Waterfall approach, as part of your requirements gathering phase (what does the product actually *need* to do?), you will typically sit down with a user to develop use cases.

You should have a table listing all use cases discussed in the document, the ID is just the order it is listed in, the name should be indicative of what should happen, the primary actor is typically most important in an application where you may have different levels of users (think admin vs normal user), complexity is a best-guess on your part as to how hard it should be. A lower number in priority indicates that it needs to happen sooner rather than later. A sample table, or Use Case Index can be seen in Table 1.

Use Case Number: 1
Use Case Name: Add item to cart

Description: A shopper on our site has identified an item they wish to buy. They will click on a "Add to Cart" button. This will kick off a process to add one instance of the item to their cart.

You will then go on to (minimally) discuss a basic flow for the process:

1) User navigates to page listing desired item
2) User left-clicks on "Add to Cart" button.
3) User cart is updated to reflect the new item, this also updates the current total.

Termination Outcome: The user now has a single instance of the item in their cart.

You may need to also add in any alternative flows:

Alternative: Item already exists in the cart

1) User navigates to page listing desired item
2) User left-clicks on "Add to Cart" button.
3) User cart is updated to reflect the new item, showing that one more instance of the existing item has been added. This also updates the current total.

Termination Outcome: The user now has multiple instances of the item in their cart.

You will often also need to include pictures or diagrams. It is quite common to see use-case diagrams in such write-ups. To properly reference an image, you will need to use the `figure` environment and will need to reference it in your text (via the `ref` command) (see Figure 1). NOTE: this is not a use case diagram, but a kitten.

After fully describing a use case, it is time to move on to the next use case:

Use Case Number: 2
Use Case Name: Checkout
Description: A shopper on our site has finished shopping. They will click on a "Checkout" button. This will kick off a process to calculate cart total, any taxes, shipping rates, and collect payment from the shopper.

You will then need to continue to flesh out all use cases you have identified for your project.

Figure 1. First picture, this is a kitten, not a use case diagram

2.3. Interface Mockups

At first, this will largely be completely made up, as you get further along in your project, and closer to a final product, this will typically become simple screenshots of your running application.

In this subsection, you will be showing what the screen should look like as the user moves through various use cases (make sure to tie the interface mockups back to the specific use cases they illustrate).

3. Project Timeline

Go back to your notes and look up a typical project development life cycle for the Waterfall approach. How will you follow this life cycle over the remainder of this semester? This will usually involve a chart showing your proposed timeline, with specific milestones plotted out. Make sure you have deliverable dates from the course schedule listed, with a plan to meet them (NOTE: these are generally optimistic deadlines).

4. Project Structure

At first, this will be a little empty (it will need to be filled in by the time you turn in your final report). This is your chance to discuss all of your design decisions (consider this the README's big brother).

4.1. UML Outline

Show the full structure of your program. Make sure to keep on updating this section as your project evolves (you often start out with one plan, but end up modifying things as you move along). As a note, while Dia fails miserably at generating pdfs (probably my fault), I have had much success with png files. Make sure to wrap your images in a `figure` environment, and to reference with the `ref` command. For example, see Figure 2.

Figure 2. Your figures should be in the *figure* environment, and have captions. Should also be of diagrams pertaining to your project, not random internet kittens

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.