**Purple Slime's 5th Edition Character Menu Requirements Documentation**

**Requirements Development Team:**

Kerry Uchtorff

Alex Hodges

Kaleb Jacobs

**1. Introduction**

*1.1 Purpose of the requirements document*

*1.2 Scope of the product*

*Projected project:*

Purple Slime’s Character Menu which is a character generator for Dungeons and Dragons for 5th edition.

*Description***:**

This program will allow the user to choose between a standard array of values, randomly assigned values, or reassign the values from either the standard array or random value generation to create a character for the tabletop role playing game Dungeons and Dragons using the 5th Edition rules. This will allow for input of a character’s name and the selection of alignment, background, class and race. Then the program will use generated numbers, manual choice selection, and/or preset information as per the selected class, background or race to fill in other fields needed to create character. The program will generate the correct values for different parts of the character based on the provided ability points. The character can then be leveled up to level 20 which includes choosing subclasses, adding ability points and making the appropriate choices as needed. As a character is leveled, the correct features, spells, and calculated values of the class and subclass will be added for the user to view available due to the character’s class and subclass. The character can be saved by means of a text file saved on the computer in the user's chosen location. Saved characters can be reloaded to the program to be updated and saved back to the program. Saved characters can also be deleted from the program when a user is done with them.

*Structure of our team:*

We have chosen to be a democratic decentralized structured team. As a result, the majority will win in discussion. In cases where we are all unsure, we will set a time/day to have their decision recorded, and if not, then whoever has actually decided/voted will be recorded and a decision reached. If it comes down to it, then the person most familiar with that portion of the project will be referred to if different opinions are tied.

*Scope Statement Questions***:**

• *What type of platform (Windows, Macintosh, UNIX, etc.) must the software work with?*

This project will be made to run on a Windows computer where Java version 8 Update 181 has been installed.

• *Will the software function as a standalone application on a given computer, or will it function over a network connection?*

This project will be a stand alone project on a given computer with no network connection required.

*• What other software, if any, must the software interact with? For example, you might be building a subsystem component that will be integrated into a larger system. In such a case, it’s important that you don’t duplicate functionality provided by existing subsystems.*

This program will use the installed Java Runtime environment. It will not modify or interfere with the installed Java Runtime environment.

• *If you are building a game, how many players will be supported? Will there be a computer player? If so, will it use artificial intelligence?*

This program will be for one user using one character at a time, but the program itself will allow the user to save multiple characters to the system. In order to use a different character, the current character will need to be saved before loading another. However, if you do not save before closing, any changes to the character will not be saved.

• *What programming language will be used for the project?*

This project will be created with Java.

*• Will the software use a graphical interface or a command line interface? Note that these are also part of the software requirements, and should therefore be included in your requirements document.*

This project will have a graphical interface so that the user can see their character’s information across multiple tabs.

1.3 Definitions, acronyms, and abbreviations

*Ability Scores*- These are the base of the character calculations. They represent the different aspects of the character. Ability scores are Strength, Dexterity, Constitution, Intelligence, Wisdom, and Charisma.

*Ability Score Mods* - This is the modifier a character gets for each ability score. This is used in multiple ways for the character. The ability score mod is calculated by the value of the ability score a character has minus 10 and divided by 2, rounding down.

*Backgrounds* - This is sourced from the Player’s Handbook and is supposed to represent a character’s history and equipment prior to the first game. The list of options for backgrounds is in Appendix C - Backgrounds.

*Classes* - This is a character’s occupation or training and can have a subclass or specialty. Again, this program only uses the options found in D&D 5E. A full list of classes is found in Appendix B - Classes.

*D&D 5E* - The Dungeons and Dragons 5th Edition ruleset as can be found in the 5th Edition Player's Handbook or System Resource Document.

*GUI or UI* - Graphical User Interface or User Interface, respectively. This is the part of the program the user interacts with.

*Lvl or Lvls*- This stands for level or levels, respectively. Mostly shortened for easy coding.

*Mod(s)* - Short-term for “modifier(s)”.

*Proficiency Bonus* **-** The proficiency bonus is a number used to represent how the character has advanced over the course of the game. The higher a player’s level, the higher the proficiency bonus.

*Skills* **-** Skills are the different things a character may know. A list of all skills with a quick definition is in Appendix D - Skills. Not all classes can learn all the skills, so each class has a list of available skills to learn.

*Skill Mods*- Skill mods are the bonus a character receives from the character’s knowledge of a skill. The skill mods are calculated by taking the ability score mod tied to the skill, plus the proficiency bonus multiplied by the level of skill the character.

*SRD*- TheSystem Resource Document, which can be found here <https://5thsrd.org/>

*Races* - The race is the species or ancestry of the character. Any race of character including their subraces that is sourced from the D&D 5E Player’s Handbook. List of races is found in Appendix A - Races.

1.4 References

Dungeons & Dragons: 5th Edition Player's Handbook by Wizards of the Coast.

System Resource Document - <https://5thsrd.org/>

1.5 Overview of the remainder of this document

**2. General description**

2.1 Product perspective - give a little background about why the software is being built and why it will be useful; one good paragraph will do

Dungeons & Dragons can be a bit overwhelming for a new player to begin playing and a bit tedious for veteran players when doing the calculations for their character or maintaining their character information. The Purple Slime Character Menu works as a supplement to the 5th Edition Player's Handbook and allows a user to gather most of the information they need for a new character with a few simple selections, ensuring a streamlined process that ensures new players won't miss features and that experienced players can speed through character creation. The Purple Slime Character Menu takes this information and makes the multiple calculations needed to create a character from scratch. The program also provides easy support for leveling up characters with the appropriate calculations and features. The character or multiple characters can be saved and reloaded for the user’s ease.

2.2 Product functions

The Purple Slime Character Menu generates attribute scores, modifiers, and calculates a character’s hit points based on criteria such as class, background, and race. It also asks the user to make choices for how they want their character set up and prints out the appropriate features provided by a character’s class, race, and background. As the character is leveled, the program provides an accurate list of features and calculations for the character as the character advances when they level up. A user is able to save, load, and delete saved characters from the program. Finally, the user can save and quit the program by selecting Save and Exit from the File menu.

2.3 User characteristics

The Purple Slime Character Menu will be used by players of the tabletop game Dungeons & Dragons: 5th Edition. These players may be on their first game or have played for years. Players of other tabletop games that have a similar system to D&D may also use this software as long as they understand its limitations since the system is built specifically for Dungeons & Dragons: 5th Edition.

2.4 General constraints

At this time, the user is responsible for entering their weapons and armor into the system. In some situations, the user is able to pick a weapon from a group of weapons. This is to allow non-standard items. The user can only have one character loaded at a time, so the Dungeon Master cannot use this system to easily run an entire group.

2.5 Assumptions and dependencies - examples of these would be: assumes end user has QuickTime installed; requires Visual Basic runtime to be installed on end user’s computer

This software assumes that the user has Java installed on their computer and is running Windows. We also assume that the user either has access to the Dungeon & Dragons: 5th Edition Player's Handbook, System Resource Document, or a Dungeon Master (group leader) who is willing to help them.

**3. Specific requirements**

Purple Slime Character Menu

1. Graphical Interface

1.1. Program begins with opening a graphical user interface.

1.1.1. The program launches when user clicks on the .Jar file icon

1.1.2. Program opens a new window with all fields blank

1.1.2.1. All fields will display various parts of the character

1.1.2.1.1. See Appendix E - Character Information

2. User Decided Ability Score Assignment

2.1. User clicks button that looks like a cogwheel

2.1.1. Message box appears with a drop down to allow user to pick one of the ways to assign ability scores and hits ok

2.2. User clicks button that looks like a dice

2.2.1. If Standard Array was selected (See 2.1.1)

2.2.1.1. The Program loads the scores {15, 14, 13, 12, 10, 8}

2.2.1.2. Roll Assignment box pops up to allow user to assign the values

2.2.1.3. User assigns values in the pop up and clicks ok

2.2.1.4. The program assigns new values to the ability scores

2.2.1.5. If character exists, rolls are assigned to correct ability score on character and applies any modifiers

2.2.2. If Rolled Auto Assigned was selected (See 2.1.1)

2.2.2.1. Program will roll 6 values; one for each ability score

2.2.2.2 Program will assign the rolls in the order they are made

2.2.2.3. If character exists, rolls are assigned to correct ability score on character and applies any modifiers

2.2.3. If Rolled User Assigned was selected (See 2.1.1)

2.2.3.1. Program will roll 6 values; one for each ability score

2.2.3.2. Roll Assignment box pops up to allow user to assign rolls

2.2.3.3. User assigns values in the pop up and hits ok

2.2.3.4. The program assigns new values to the ability scores

2.2.3.5. If character exists, rolls are assigned to correct ability score on character and applies any modifiers

2.3. User clicks button that looks like a refresh symbol

2.3.1. If Standard Array was selected (See 2.1.1)

2.3.1.1. Roll Assignment box pops up to allow user to reassign the values

2.3.1.2. User reassigns values in the pop up and clicks ok

2.3.1.3. The program reassigns new values to the ability scores

2.3.1.4. If character exists, rolls are assigned to correct ability score on character and applies any modifiers

2.3.2. If Rolled Auto Assigned was selected (See 2.1.1)

2.3.2.1. A message box will pop up if the user tries to reassign the rolls to inform the user that it is not possible to reassign these rolls

2.3.3. If Rolled User Assigned was selected (See 2.1.1)

2.3.3.1. Roll Assignment box pops up to allow user to reassign roll values

2.3.3.2. User reassigns values in the pop up and hits ok

2.3.3.3. The program reassigns new values to the ability scores

2.3.3.4. If character exists, rolls are assigned to correct ability score on character and applies any modifiers

3. User Driven Character Creation

3.1. User shall select a race from a pre-populated list a race for the character

3.1.1. (Non-functional) All races shall be a Java class with certain races being parent races to character subrace child classes

3.1.1.1. (Non-functional) All races are a child class of parent Character class.

3.1.1.2. (Non-functional) Races breakdown in Appendix A – Races

3.2. User shall select a class from a pre-populated list a class for the character

3.2.1. (Non-functional) All classes shall be a Java class and extend char\_class

3.2.1.1. (Non-functional) Classes breakdown in Appendix B – Class

3.3. User shall select a background from a pre-populated list a background for the character

3.3.1. (Non-functional) All backgrounds options shall be in one Java class

3.3.1.1. (Non-functional) Background breakdown in Appendix C – Backgrounds

3.4. User shall enter a name for the character

3.4.1. A clickable button with a pen on it will be provided to save the character name

3.5. User can roll their ability scores

3.5.1. See requirement 2.1 User Decided Ability Score Assignments

3.5.2. User can apply ability scores before initial generation of character

3.5.2.1. User can decide to wait to apply ability scores until after initial generation of character

3.5.2.2. User can decide to reroll their ability scores after generation of character

3.5.2.2.1. When scores are applied after initial generation, class, and racial bonuses to the ability scores will be reapplied.

3.6. User shall activate a clickable button named “Apply” to generate the character as a level 1 character.

3.6.1. The program shall check that a race, class, background and name have been chosen for the character

3.6.1.1. If not, a message box will pop up to tell the user the choice they did not make.

3.6.1.2. (Non-Functional) The newly generated character will be encapsulated in Java subclass of Character which is named for the race the character chose

3.6.1.2.1. If the chosen race has a parent race, the parent race will be applied first

3.6.1.2.1.1. All of the parent race’s racial bonuses, features, , ability modifiers and spells will be applied

3.6.1.2.1.2. All of the chosen race’s racial bonuses, features, ability modifiers and spells will be applied

3.6.2. (Non-Functional) The newly generated character will be used as a parameter and variable of the chosen classes Java object.

3.6.2.1. Class features including hit point calculations and saving throws are added

3.6.3. (Non-Functional) The newly generated character will be used as a parameter and variable of the chosen background method.

3.6.3.1. The background features, gold and equipment will be applied to the character

3.7. Choice pop up boxes will appear to allow the user to make the appropriate choices for their character

3.7.1. Number and types of choice pop ups will depend on the race, background, class and subclass of character.

3.7.1.1. Possible Choice pop up boxes

3.7.1.1.1. Choose Languages

3.7.1.1.1.1. The language pop up box will be pre-populated with only languages the character does not know.

3.7.1.1.1.1.1. This list will be formed from a list of possible languages where applicable to the race or class

3.7.1.1.1.2. The chosen language will be added to the characters known languages list

3.7.1.1.1.3. The next language pop up box will not have the language previously chosen

3.7.1.1.2. Choose Cantrip Spells

3.7.1.1.2.1. The cantrip choice pop up box will be pre-populated with only cantrips spells the character does not know.

3.7.1.1.2.1.1. This list will be formed from a list of possible cantrip spells where applicable to the race or class

3.7.1.1.2.2. The chosen cantrip will be added to the characters known cantrip spells list

3.7.1.1.2.3. The next cantrip pop up box will not have the cantrip previously chosen

3.7.1.1.3. Choose Skills

3.7.1.1.3.1. The skill choice pop up box will be pre-populated with ONLY skills the character does not know

3.7.1.1.3.2. This list will be formed from a list of possible skills where applicable to the race or class

3.7.1.1.3.3. The chosen skills will have one added to the character’s skill variable and weapons and armor proficiency list

3.7.1.1.3.4. The next skills pop up box will not have the skills previously chosen

3.7.1.1.4. Choose Equipment

3.7.1.1.4.1. The equipment choice pop up box will be pre-populated with ONLY equipment available to the character

3.7.1.1.4.1.1. This list will be formed from a list of possible equipment where applicable class

3.7.1.1.4.1.2. The pop up box will display all available equipment at the same time to applicable class

3.7.1.1.4.2. The chosen equipment will be added to the characters known equipment list

3.7.1.1.5. Choose a Subclass

3.7.1.1.5.1. The subclass choice pop up box will be pre-populated with ONLY possible subclasses available to the class

3.7.1.1.5.2. The chosen subclass will be added to the characters subclass variable

3.7.1.1.6. Subclass choices

3.7.1.1.6.1. The subclass choice pop up box will be pre-populated with ONLY information that pertains to a certain subclass and the character does not know.

3.7.1.1.6.1.1. This list will be formed from a list of possible subclass values.

3.7.1.1.6.2. The chosen subclass choice will be added to the correct location in the character.

3.7.1.1.6.2.1. If applicable, the next subclass pop up box will not have the values previously chosen

3.7.1.1.7. Choose Expert Skills

3.7.1.1.7.1. The expert skill choice pop up box will be pre-populated with ONLY skills the character already knows and is not an expert in

3.7.1.1.7.1.1. This list will be formed from a list of possible skills where applicable to the class that the character already knows

3.7.1.1.7.1.2. This pop up box will only appear to classes that have the expert skill available

3.7.1.1.7.1.2.1. Bard

3.7.1.1.7.1.2.2. Rogue

3.7.1.1.7.2. The chosen skills will

3.7.1.1.7.2.1. have one added to the character’s skill variable

3.7.1.1.7.2.2. remove the previous skill from weapons and armor proficiency list

3.7.1.1.7.2.3. weapons and armor proficiency list with an expert tag

3.7.1.1.7.3. The next expert skills pop up box will not have the expert skills previously chosen

3.8. The program will calculate all values for the character

3.8.1. The program will only calculate the values based on the correct formula for class, subclass and ability scores

3.8.2. The value will be placed in the correct variable

3.8.2.1. See Appendix E - Character Information

3.9. Graphical User interface is refreshed.

3.9.1. See requirements section 8. Refresh Display

4. Saving A Character

4.1. The user chooses “Save” from the file menu to initiate saving the character to a file.

4.1.1. Program checks if a character exists

4.1.1.1. If no character exists, the program will display a message box asking the user to create a character before saving.

4.1.2. Program will check if the save location has been declared

4.1.2.1. If no save location has been declared, the user will be asked to determine one.

4.1.3. Program will create text file of character information to save character

4.1.3.1. If the text file doesn’t exist, a new one will be created

4.1.4. Program will write the character's name and filepath in the index file.

4.2. A message box will appear to confirm the character was saved.

5. Deleting A Character

5.1. The user chooses “Delete” from the file menu to initiate deleting the character.

5.1.1. Program will check if the save location has been declared

5.1.1.1. If no save location has been declared, the user will be asked to determine one.

5.1.2 Index file is read in to check for all saved characters

5.1.2.1. If there are no saved characters, a message box will appear to inform the user there are no saved characters to load.

5.1.3. A message box will appear with a pre-populated list of saved characters

5.1.4. User selects character to delete

5.1.5. The character’s file is deleted from declared location folder

5.1.6. The Character is removed from the index file.

5.2. A message box will appear to confirm the character was deleted.

6. Loading a Character

6.1. The user chooses “Load” from the file menu to initiate loading the character to a file.

6.1.1. Program will check if the save location has been declared

6.1.1.1. If no save location has been declared, the user will be asked to determine one.

6.1.2. Index file is read in to check for all saved characters

6.1.2.1. If there are no saved characters in the index, a message box will appear to inform the user there are no saved characters to load.

6.1.3. A message box will appear with a pre-populated list of saved characters

6.1.4. User selects character to load

6.1.4.1. Program reads for file path in the index

6.1.4.2. Program reads in text file from appropriate file

6.1.4.2.1. Program creates character from text file information

6.1.4.2.1.1. Program loads all values in for appropriate variables.

6.1.4.2.2. Program creates class from text file information

6.1.4.2.2.1. Graphical User interface is refreshed.

6.1.4.2.2.1.1. See requirements section 8. Refresh Display

7. Leveling A Character

7.1. User clicks on the pre-populated drop down box of levels

7.1.1. Program checks if a character exists

7.1.1.1. If no character exists, the program will display a message box asking the user to create a character before leveling.

7.1.1.2. If character exists

7.1.1.2.1. Characters experience set

7.1.1.2.2. The program will check at the new level for any choices that need made based on the characters class

7.1.1.2.2.1. Possible Choices

7.1.1.2.2.1.1. Ability Point Increases

7.1.1.2.2.1.2. Choosing a subclass

7.1.1.2.2.1.3. Subclass specific ability choices

7.1.1.2.2.1.4. Class specific feature choices

7.1.1.2.3. Character has the level checked

7.1.1.2.3.1. Character level is set to new level

7.1.1.2.3.2. Character receives race specific leveling information

7.1.1.2.3.3. Character receives class specific leveling information

7.1.1.2.3.3.1. Hit points calculated differently above level 1

7.1.1.2.3.4. Character receives subclass specific leveling information

7.1.1.2.3.5. The character is saved

7.1.1.2.3.5.1. See requirements section 4.1 Saving a Character

7.1.1.2.3.6. Graphical User interface is refreshed.

7.1.1.2.3.6.1. See requirements section 8. Refresh Display

8. Refresh Display

8.1. User clicks the refresh button

8.1.1. Program checks if a character exists

8.1.1.1. If no character exists, the program will display a message box asking the user to create a character before refreshing

8.1.1.2. If character exists

8.1.1.2.1. All fields in the user interface are cleared

8.1.1.2.2. Character values calculated

8.1.1.2.3. Recharge values that can be depleted to max value

8.1.1.2.4. Display most current values for character to the user interface

9. Save and Exit

9.1. The user chooses “Save and Exit” from the file menu to save the character to a file and close the program

9.1.1. See requirement 4. Saving a Character for save information

9.1.2. The program will end and the graphical user interface will disappear.

10. Exit

10.1.The user chooses “Exit” from the file menu to close the program

10.1.1. The program will end and the graphical user interface will disappear.

11. User Entered Data

11.1.User has ability to enter data for various fields and have it be added to character

11.1.1. User enters hair color in hair text box

11.1.1.1. Text from hair text box is saved to character

11.1.1.2. Text will be saved if character is saved or leveled

11.1.1.2.1. Hair color remains after refresh

11.1.1.2.2. Hair color is displayed after load if saved

11.1.2. User enters eye color in eyes text box

11.1.2.1. Text from eyes text box is saved to character

11.1.2.2. Text will be saved if character is saved or leveled

11.1.2.2.1. Eyes color remains after refresh

11.1.2.2.2. Eyes color is displayed after load if saved

11.1.3. User enters height in height text field

11.1.3.1. Text from height text box is saved to character

11.1.3.2. Text will be saved if character is saved or leveled

11.1.3.2.1. Height remains after refresh

11.1.3.2.2. Height is displayed after load if saved

11.1.4. User enters weight in weight text field

11.1.4.1. Text from weight text box is saved to character

11.1.4.2. Text will be saved if character is saved or leveled

11.1.4.2.1. Weight remains after refresh

11.1.4.2.2. Weight is displayed after load if saved

11.1.5. User enters age in age text field

11.1.5.1. Text from age text box is saved to character as integer

11.1.5.2. Text will be saved if character is saved or leveled

11.1.5.2.1. Age remains after refresh

11.1.5.2.2. Age is displayed after load if saved

11.1.6. User has ability to enter data for 3 weapons

11.1.6.1. Each weapons information will be kept separately

11.1.6.1.1. Weapon name will be entered as text

11.1.6.1.2. Weapon attack bonus is an integer

11.1.6.1.3. Weapon damage will be entered as text

11.1.6.1.4. Weapon type will be entered as text

11.1.6.2. Text will be saved if character is saved or leveled

11.1.6.2.1. Weapon information remains after refresh

11.1.6.2.2. Weapon information is displayed after load if saved

11.1.7. User has ability to enter data for one set of armor

11.1.7.1. The armor’s information will be kept separately

11.1.7.1.1. Armor name will be entered as text

11.1.7.1.2. Armor AC is an integer

11.1.7.1.2.1. Armor AC will be calculated into players AC

11.1.7.1.3. Armor bonus entered as an integer

11.1.7.2. Text will be saved if character is saved or leveled

11.1.7.2.1. Armor information remains after refresh

11.1.7.2.2. Armor information is displayed after load if saved

11.1.8. User has ability to enter notes into the Info box

11.1.8.1. User types information into the Info box

11.1.8.2. User presses Save Info

11.1.8.2.1. Text from Info box will be saved to character

11.1.8.2.2. Text from Info box remains after refresh

11.1.8.2.3. Text from Info box is displayed after load if saved

11.1.9. User has ability to enter or remove equipment

11.1.9.1. User enters equipment information

11.1.9.1.1. User enters information about equipment in text field at bottom of box

11.1.9.1.2. User clicks + button

11.1.9.1.2.1. Information the user enters to equipment text field is added to the equipment box

11.1.9.1.2.2. Text added from equipment box remains after refresh

11.1.9.1.2.3. Text added from equipment box is displayed after load if saved

11.1.9.2. User removes equipment information

11.1.9.2.1. User enters information about equipment in text field at bottom of box

11.1.9.2.1.1. Information about equipment must match exactly information in equipment field

11.1.9.2.2. User clicks - button

11.1.9.2.2.1. Information the user enters to equipment text field is removed from the equipment box

11.1.9.2.2.2. Text removed from equipment box remains after refresh

11.1.9.2.2.3. Text removed from equipment box is displayed after load if saved

11.1.10. User has ability to enter or remove languages

11.1.10.1. User enters languages information

11.1.10.1.1. User enters information about languages in text field at bottom of box

11.1.10.1.2. User clicks + button

11.1.10.1.2.1. Information the user enters to languages text field is added to the languages box

11.1.10.1.2.2. Text added from languages box remains after refresh

11.1.10.1.2.3. Text added from languages box is displayed after load if saved

11.1.10.2. User removes languages information

11.1.10.2.1. User enters information about languages in text field at bottom of box

11.1.10.2.1.1. Information about languages must match exactly information in languages field

11.1.10.2.2. User clicks - button

11.1.10.2.2.1. Information the user enters to languages text field is removed from the languages box

11.1.10.2.2.2. Text removed from languages box remains after refresh

11.1.10.2.2.3. Text removed from languages box is displayed after load if saved

11.1.11. User has ability to enter or remove proficiencies

11.1.11.1. User enters proficiencies information

11.1.11.1.1. User enters information about proficiencies in text field at bottom of box

11.1.11.1.2. User clicks + button

11.1.11.1.2.1. Information the user enters to proficiencies text field is added to the proficiencies box

11.1.11.1.2.2. Text added from proficiencies box remains after refresh

11.1.11.1.2.3. Text added from proficiencies box is displayed after load if saved

11.1.11.2. User removes proficiencies information

11.1.11.2.1. User enters information about proficiencies in text field at bottom of box

11.1.11.2.1.1. Information about proficiencies must match exactly information in proficiencies field

11.1.11.2.2. User clicks - button

11.1.11.2.2.1. Information the user enters to proficiencies text field is removed from the proficiencies box

11.1.11.2.2.2. Text removed from proficiencies box remains after refresh

11.1.11.2.2.3. Text removed from proficiencies box is displayed after load if saved

11.1.12. User has ability to enter or remove cantrips

11.1.12.1. User enters cantrips information

11.1.12.1.1. User enters information about cantrips in text field at bottom of box

11.1.12.1.2. User clicks + button

11.1.12.1.2.1. Information the user enters to cantrips text field is added to the cantrips box

11.1.12.1.2.2. Text added from cantrips box remains after refresh

11.1.12.1.2.3. Text added from cantrips box is displayed after load if saved

11.1.12.2. User removes cantrips information

11.1.12.2.1. User enters information about cantrips in text field at bottom of box

11.1.12.2.1.1. Information about cantrips must match exactly information in cantrips field

11.1.12.2.2. User clicks - button

11.1.12.2.2.1. Information the user enters to cantrips text field is removed from the cantrips box

11.1.12.2.2.2. Text removed from cantrips box remains after refresh

11.1.12.2.2.3. Text removed from cantrips box is displayed after load if saved

11.1.13. User has ability to enter or remove level 1 spells

11.1.13.1. User enters level 1 spells information

11.1.13.1.1. User enters information about level 1 spells in text field at bottom of box

11.1.13.1.2. User clicks + button

11.1.13.1.2.1. Information the user enters to level 1 spells text field is added to the level 1 spells box

11.1.13.1.2.2. Text added from level 1 spells box remains after refresh

11.1.13.1.2.3. Text added from level 1 spells box is displayed after load if saved

11.1.13.2. User removes level 1 spells information

11.1.13.2.1. User enters information about level 1 spells in text field at bottom of box

11.1.13.2.1.1. Information about level 1 spells must match exactly information in level 1 spells field

11.1.13.2.2. User clicks - button

11.1.13.2.2.1. Information the user enters to level 1 spells text field is removed from the level 1 spells box

11.1.13.2.2.2. Text removed from level 1 spells box remains after refresh

11.1.13.2.2.3. Text removed from level 1 spells box is displayed after load if saved

11.1.14. User has ability to enter or remove level 2 spells

11.1.14.1. User enters level 2 spells information

11.1.14.1.1. User enters information about level 2 spells in text field at bottom of box

11.1.14.1.2. User clicks + button

11.1.14.1.2.1. Information the user enters to level 2 spells text field is added to the level 2 spells box

11.1.14.1.2.2. Text added from level 2 spells box remains after refresh

11.1.14.1.2.3. Text added from level 2 spells box is displayed after load if saved

11.1.14.2. User removes level 2 spells information

11.1.14.2.1. User enters information about level 2 spells in text field at bottom of box

11.1.14.2.1.1. Information about level 2 spells must match exactly information in level 2 spells field

11.1.14.2.2. User clicks - button

11.1.14.2.2.1. Information the user enters to level 2 spells text field is removed from the level 2 spells box

11.1.14.2.2.2. Text removed from level 2 spells box remains after refresh

11.1.14.2.2.3. Text removed from level 2 spells box is displayed after load if saved

11.1.15. User has ability to enter or remove level 3 spells

11.1.15.1. User enters level 3 spells information

11.1.15.1.1. User enters information about level 3 spells in text field at bottom of box

11.1.15.1.2. User clicks + button

11.1.15.1.2.1. Information the user enters to level 3 spells text field is added to the level 3 spells box

11.1.15.1.2.2. Text added from level 3 spells box remains after refresh

11.1.15.1.2.3. Text added from level 3 spells box is displayed after load if saved

11.1.15.2. User removes level 3 spells information

11.1.15.2.1. User enters information about level 3 spells in text field at bottom of box

11.1.15.2.1.1. Information about level 3 spells must match exactly information in level 3 spells field

11.1.15.2.2. User clicks - button

11.1.15.2.2.1. Information the user enters to level 3 spells text field is removed from the level 3 spells box

11.1.15.2.2.2. Text removed from level 3 spells box remains after refresh

11.1.15.2.2.3. Text removed from level 3 spells box is displayed after load if saved

11.1.16. User has ability to enter or remove level 4 spells

11.1.16.1. User enters level 4 spells information

11.1.16.1.1. User enters information about level 4 spells in text field at bottom of box

11.1.16.1.2. User clicks + button

11.1.16.1.2.1. Information the user enters to level 4 spells text field is added to the level 4 spells box

11.1.16.1.2.2. Text added from level 4 spells box remains after refresh

11.1.16.1.2.3. Text added from level 4 spells box is displayed after load if saved

11.1.16.2. User removes level 4 spells information

11.1.16.2.1. User enters information about level 4 spells in text field at bottom of box

11.1.16.2.1.1. Information about level 4 spells must match exactly information in level 4 spells field

11.1.16.2.2. User clicks - button

11.1.16.2.2.1. Information the user enters to level 4 spells text field is removed from the level 4 spells box

11.1.16.2.2.2. Text removed from level 4 spells box remains after refresh

11.1.16.2.2.3. Text removed from level 4 spells box is displayed after load if saved

11.1.17. User has ability to enter or remove level 5 spells

11.1.17.1. User enters level 5 spells information

11.1.17.1.1. User enters information about level 5 spells in text field at bottom of box

11.1.17.1.2. User clicks + button

11.1.17.1.2.1. Information the user enters to level 5 spells text field is added to the level 5 spells box

11.1.17.1.2.2. Text added from level 5 spells box remains after refresh

11.1.17.1.2.3. Text added from level 5 spells box is displayed after load if saved

11.1.17.2. User removes level 5 spells information

11.1.17.2.1. User enters information about level 5 spells in text field at bottom of box

11.1.17.2.1.1. Information about level 5 spells must match exactly information in level 5 spells field

11.1.17.2.2. User clicks - button

11.1.17.2.2.1. Information the user enters to level 5 spells text field is removed from the level 5 spells box

11.1.17.2.2.2. Text removed from level 5 spells box remains after refresh

11.1.17.2.2.3. Text removed from level 5 spells box is displayed after load if saved

11.1.18. User has ability to enter or remove level 6 spells

11.1.18.1. User enters level 6 spells information

11.1.18.1.1. User enters information about level 6 spells in text field at bottom of box

11.1.18.1.2. User clicks + button

11.1.18.1.2.1. Information the user enters to level 6 spells text field is added to the level 6 spells box

11.1.18.1.2.2. Text added from level 6 spells box remains after refresh

11.1.18.1.2.3. Text added from level 6 spells box is displayed after load if saved

11.1.18.2. User removes level 6 spells information

11.1.18.2.1. User enters information about level 6 spells in text field at bottom of box

11.1.18.2.1.1. Information about level 6 spells must match exactly information in level 6 spells field

11.1.18.2.2. User clicks - button

11.1.18.2.2.1. Information the user enters to level 6 spells text field is removed from the level 6 spells box

11.1.18.2.2.2. Text removed from level 6 spells box remains after refresh

11.1.18.2.2.3. Text removed from level 6 spells box is displayed after load if saved

11.1.19. User has ability to enter or remove level 7 spells

11.1.19.1. User enters level 7 spells information

11.1.19.1.1. User enters information about level 7 spells in text field at bottom of box

11.1.19.1.2. User clicks + button

11.1.19.1.2.1. Information the user enters to level 7 spells text field is added to the level 7 spells box

11.1.19.1.2.2. Text added from level 7 spells box remains after refresh

11.1.19.1.2.3. Text added from level 7 spells box is displayed after load if saved

11.1.19.2. User removes level 7 spells information

11.1.19.2.1. User enters information about level 7 spells in text field at bottom of box

11.1.19.2.1.1. Information about level 7 spells must match exactly information in level 7 spells field

11.1.19.2.2. User clicks - button

11.1.19.2.2.1. Information the user enters to level 7 spells text field is removed from the level 7 spells box

11.1.19.2.2.2. Text removed from level 7 spells box remains after refresh

11.1.19.2.2.3. Text removed from level 7 spells box is displayed after load if saved

11.1.20. User has ability to enter or remove level 8 spells

11.1.20.1. User enters level 8 spells information

11.1.20.1.1. User enters information about level 8 spells in text field at bottom of box

11.1.20.1.2. User clicks + button

11.1.20.1.2.1. Information the user enters to level 8 spells text field is added to the level 8 spells box

11.1.20.1.2.2. Text added from level 8 spells box remains after refresh

11.1.20.1.2.3. Text added from level 8 spells box is displayed after load if saved

11.1.20.2. User removes level 8 spells information

11.1.20.2.1. User enters information about level 8 spells in text field at bottom of box

11.1.20.2.1.1. Information about level 8 spells must match exactly information in level 8 spells field

11.1.20.2.2. User clicks - button

11.1.20.2.2.1. Information the user enters to level 8 spells text field is removed from the level 8 spells box

11.1.20.2.2.2. Text removed from level 8 spells box remains after refresh

11.1.20.2.2.3. Text removed from level 8 spells box is displayed after load if saved

11.1.21. User has ability to enter or remove level 9 spells

11.1.21.1. User enters level 9 spells information

11.1.21.1.1. User enters information about level 9 spells in text field at bottom of box

11.1.21.1.2. User clicks + button

11.1.21.1.2.1. Information the user enters to level 9 spells text field is added to the level 9 spells box

11.1.21.1.2.2. Text added from level 9 spells box remains after refresh

11.1.21.1.2.3. Text added from level 9 spells box is displayed after load if saved

11.1.21.2. User removes level 9 spells information

11.1.21.2.1. User enters information about level 9 spells in text field at bottom of box

11.1.21.2.1.1. Information about level 9 spells must match exactly information in level 9 spells field

11.1.21.2.2. User clicks - button

11.1.21.2.2.1. Information the user enters to level 9 spells text field is removed from the level 9 spells box

11.1.21.2.2.2. Text removed from level 9 spells box remains after refresh

11.1.21.2.2.3. Text removed from level 9 spells box is displayed after load if saved

11.1.22. User has ability to enter or remove race and class spells

11.1.22.1. User enters race and class spells information

11.1.22.1.1. User enters information about race and class spells in text field at bottom of box

11.1.22.1.2. User clicks + button

11.1.22.1.2.1. Information the user enters to race and class spells text field is added to the race and class spells box

11.1.22.1.2.2. Text added from race and class spells box remains after refresh

11.1.22.1.2.3. Text added from race and class spells box is displayed after load if saved

11.1.22.2. User removes race and class spells information

11.1.22.2.1. User enters information about race and class spells in text field at bottom of box

11.1.22.2.1.1. Information about race and class spells must match exactly information in race and class spells field

11.1.22.2.2. User clicks - button

11.1.22.2.2.1. Information the user enters to race and class spells text field is removed from the race and class spells box

11.1.22.2.2.2. Text removed from race and class spells box remains after refresh

11.1.22.2.2.3. Text removed from race and class spells box is displayed after load if saved

**4. Appendices**

*Appendix A - Races*

There are multiple races in D&D 5th Edition. Anytime this document refers to all races, these are the races it will refer to.

* Dwarf
  + Hill Dwarf
  + Mountain Dwarf
* Elf
  + High Elf
  + Dark Elf (Drow)
  + Wood Elf
* Halfling
  + Lightfoot Halfling
  + Stout Halfling
* Human
* Dragonborn
* Gnome
  + Forest Gnome
  + Rock Gnome
* Half Elf
* Half Orc
* Tiefling

*Appendix B - Classes*

There are multiple classes in D&D 5th Edition. Anytime this document refers to all classes, this is the list of classes and subclasses it is referring to.

* Barbarian
  + Path of the Berserker
  + Path of the Totem Warrior
* Bard
  + College of Lore
  + College of Valor
* Cleric
  + Knowledge Domain
  + Life Domain
  + Light Domain
  + Nature Domain
  + Tempest Domain
  + Trickery Domain
  + War Domain
* Druid
  + Circle of Land
  + Circle of Moon
* Fighter
  + Champion
  + Battle Master
  + Eldritch Knight
* Monk
  + Way of the Open Hand
  + Way of the Four Elements
* Paladin
  + Oath of Devotion
  + Oath of the Ancients
  + Oath of Vengeance
* Ranger
  + Hunter
  + Beast Master
* Rogue
  + Thief
  + Arcane Trickster
  + Assassin
* Sorcerer
  + Draconic Bloodline
  + Wild Magic
* Warlock
  + The Archfey
  + The Fiend
  + The Great Old One
* Wizard
  + School of Abjuration
  + School of Conjuration
  + School of Divination
  + School of Enchantment
  + School of Evocation
  + School of Illusion
  + School of Necromancy
  + School of Transmutation

*Appendix C - Background*

There are multiple backgrounds in D&D 5th Edition. Anytime this document refers to all backgrounds, this is the list of what it refers to.

* Acolyte
* Charlatan
* Criminal
* Entertainer
* Folk Hero
* Guild Artisan
* Hermit
* Noble
* Outlander
* Sage
* Sailor
* Soldier
* Urchin

*Appendix D - Skills*

Skills are the things a character knows or can do. These skills are all linked to an ability score.

* Acrobatics -
  + Covers when you try to stay on your toes, literally.
  + Tied to a character's Dexterity.
* Animal Handling -
  + Checks if you can calm an animal, and how you take care of animals.
  + Tied to a character’s Wisdom.
* Arcana
  + Checks the ability to remember lore about magic
  + Tied to a character's Intelligence.
* Athletics
  + Covers difficult situations while climbing, jumping, or swimming, deciding if you are able to surpass the danger, like pulling off a stunt in a midair jump.
  + Tied to a character's Strength.
* Deception
  + Check if you can be able to con a NPC, or fast-talk a guard and the like.
  + Tied to a character's Charisma
* History
  + Checks the ability to remember history.
  + Tied to a character's Intelligence
* Insight
  + Checks whether or not you can determine the truth about a person’s words.
  + Tied to a character's Wisdom.
* Intimidation
  + Check if you can be able to coerce a NPC to either give you information, for example.
  + Tied to a character's Charisma
* Investigation
  + Checks the ability to look closely at the scene of a crime, or to closely inspect things.
  + Tied to a character's Intelligence
* Medicine
  + Checks to see if you can help a person be stabilized if dying, and diagnose illness.
  + Tied to a character's Wisdom
* Nature
  + Checks the ability to recall things about the local terrain, and what creatures, plants and the weather of the local area.
  + Tied to a character's Intelligence
* Perception
  + Determines if you see, sense, hear or otherwise locate something that can be key in a scenario.
  + Tied to a character's Wisdom
* Performance
  + Determines how well you can delight an audience with a play, or storytelling, and the like.
  + Tied to a character's Charisma
* Persuasion
  + Determines if you are skilled with words, rather than blades.
  + Tied to a character's Charisma
* Religion
  + Checks the ability to remember things about certain religions, like when you investigate a local cult worshipping a demon, whether or not that demon has been part of the local religion.
  + Tied to a character's Intelligence
* Sleight of Hand
  + Covers when you try to either pick a person’s pocket, or try to stick something in said pocket.
  + Tied to a character's Dexterity
* Stealth
  + Covers when you try to pass by opponents or NPCs without trying to raise an alarm.
  + Tied to a character's Dexterity
* Survival
  + Determines if you can find, hunt, and follow tracks of wild game, through rough terrain, and the like.
  + Tied to a character's Wisdom.

Appendix E - Character Information

A character is a very complex thing in D&D 5e much like an actual person. As a result, there are many variables in the character class. Each variable represents something, so this will give a list of things the character may need, where it may be used and the type of information stored. We have 126 variables to save.

* Name -
  + The character’s name.
  + It is used to save the file and for the other characters to refer to this character by.
  + String
* Race -
  + The race of the character.
  + This has different benefits and drawbacks based on the race
  + String
* CharClass
  + The character’s martial class or occupation
  + This is what you advance in the longer you play
  + String
* Subclass
  + This is the portion of their class that the character chooses to focus on.
  + Each subclass has its own benefits and drawbacks compared to other subclasses
  + String
* Background
  + The characters' origin story, what happened to them before the game started.
  + Each background has its own benefits and drawbacks compared to other backgrounds
  + String
* Alignment
  + This describes the character behavior
  + This gears the player in how they handle social interactions and generally behave
  + String
* Age
  + This is how old the character is
  + Mostly used for behavior and how others perceive the character
  + Integer value
* Height
  + This is how tall the character is
  + Mostly used how others perceive the character and description
  + String
* Weight
  + This is how heavy the character is
  + Mostly used how others perceive the character and description
  + String
* Size
  + This is how big the character is and is determined by race
  + Mostly used how others perceive the character and description
  + String
* Eyes
  + The color of the character's eyes.
  + Mostly used for description
  + String;
* Hair
  + The color of the character's eyes.
  + Mostly used for description
  + String
* Strength
  + Ability score
  + Determines how strong the character is
  + Used to calculate multiple values
  + Integer
* Dexterity
  + Ability score
  + Determines how flexible and agile a character is.
  + Used to calculate multiple values
  + Integer
* Constitution
  + Ability score
  + Determines stamina, health and how tough the character is
  + Primary use is hit points, but also used in multiple places
  + Integer
* Intelligence
  + Ability score
  + Determines ability to reason and recall ability
  + Used to calculate multiple values
  + Integer
* Wisdom
  + Ability score
  + Perception and intuition of the world around you
  + Use to calculate multiple values
  + Integer
* Charisma
  + Ability score
  + The characters people skills and how effectively they interact with others
  + Used to calculate multiple values
  + Integer
* Acrobatics
  + Please see Appendix D - Skills
* AnimalHandle
  + Please see Appendix D - Skills
* Arcana
  + Please see Appendix D - Skills
* Athletics
  + Please see Appendix D - Skills
* Deception
  + Please see Appendix D - Skills
* History
  + Please see Appendix D - Skills
* Insight
  + Please see Appendix D - Skills
* Intimidate
  + Please see Appendix D - Skills
* Investigation
  + Please see Appendix D - Skills
* Medicine
  + Please see Appendix D - Skills
* Nature
  + Please see Appendix D - Skills
* Perception
  + Please see Appendix D - Skills
* Performance
  + Please see Appendix D - Skills
* Persuasion
  + Please see Appendix D - Skills
* Religion
  + Please see Appendix D - Skills
* SleightHand
  + Please see Appendix D - Skills
* Stealth
  + Please see Appendix D - Skills
* Survival
  + Please see Appendix D - Skills
* Acromod
  + The mod value of Acrobatics
  + Calculated as dexterity mod + Proficiency Bonus \* Acro Skill level
* Animalmod
  + The mod value of Animal Handling
  + Calculated as wisdom mod + Proficiency Bonus \* Animal Handling Skill level
* Arcanamod
  + The mod value of Arcana
  + Calculated as intelligence mod + Proficiency Bonus \* Arcana Skill level
* Athlmod
  + The mod value of Athletics
  + Calculated as strength mod + Proficiency Bonus \* Athletics Skill level
* Decmod
  + The mod value of Deception
  + Calculated as Dexterity mod + Proficiency Bonus \* Deception Skill level
* Hismod
  + The mod value of History
  + Calculated as intelligence mod + Proficiency Bonus \* History Skill level
* Insimod
  + The mod value of Insight
  + Calculated as wisdom mod + Proficiency Bonus \* Insight Skill level
* Intimod
  + The mod value of Intimidation
  + Calculated as charisma mod + Proficiency Bonus \* Intimidate Skill level
* Invemod
  + The mod value of Investigation
  + Calculated as wisdom mod + Proficiency Bonus \* Investigation Skill level
* Medmod
  + The mod value of Medicine
  + Calculated as wisdom mod + Proficiency Bonus \* Medicine Skill level
* Natmod
  + The mod value of Nature
  + Calculated as intelligence mod + Proficiency Bonus \* Nature Skill level
* Percmod
  + The mod value of Perception
  + Calculated as wisdom mod + Proficiency Bonus \* Perception Skill level
* Permod
  + The mod value of Performance
  + Calculated as charisma mod + Proficiency Bonus \* Performance Skill level
* Persmod
  + The mod value of Persuasion
  + Calculated as charisma mod + Proficiency Bonus \* Persuasion Skill level
* Relmod
  + The mod value of Religion
  + Calculated as intelligence mod + Proficiency Bonus \* Religion Skill level
* Slemod
  + The mod value of Sleight of Hand
  + Calculated as dexterity mod + Proficiency Bonus \* Sleight of Hand Skill level
* Stealmod
  + The mod value of Steal
  + Calculated as dexterity mod + Proficiency Bonus \* Stealth Skill level;
* Survmod
  + The mod value of Survival
  + Calculated as wisdom mod + Proficiency Bonus \* Survival Skill level
* expert
  + The Integer of skills the character has as Expertise
  + Set to 0 for all classes that do not allow Expertise
  + Integer
* Hitpoints
  + The max amount of damage a character can take
  + There are different formulas for character at level 1 and levels 2 to 20..
  + Please see resources for more information
  + Integer
* currenthitpoints
  + The current amount of damage a character can take
  + To refresh current hit points, the value is set to max hit points
  + Please see resources for more information
  + Integer
* hitdice
  + This is the amount of damage a character can cause
  + The formula for this is player level hitdice variable
  + Each class has a different hitdice
  + String
* Level
  + This is the level the character currently is
  + Changed by level drop down
  + Level for character is set in the checklvl method and based on the characters experience
  + Integer
* Pastlvls
  + ArrayList<String> of past levels the character has reached
  + Prevents players from going backwards in the leveling system to gain extra bonuses or ability points
  + As a character is leveled to a new level, the old is put in the Pastlvls arraylist
  + String
* Exp
  + Experience the character has gained from playing
  + This is currently set by the level drop down
  + In future versions, this can be altered directly
  + Integer
* Armorclass
  + The amount of armor a character gets from the character itself
  + Integer
* noarmorAc
  + Some characters get armor when not wearing Armor.
  + This is calculated per race or class
  + Integer
* totalArmorClass
  + Total of all armors bonuses
  + Calculated by armorclass + Armorarmorclass + shieldarmorclass
  + Integer
* ArmorArmorClassbonus
  + Armor class bonus from armor filled out by user
  + Integer
* ShieldArmorClassbonus
  + Armor class bonus if charater has shield
  + Calculated by looking through equipment list, if there is a shield, it sets the value to +2, otherwise value = 0
  + Integer
* Numcantripsknown
  + Cantrips are basically free spells magic users can cast anytime.
  + This is how many are known by the character
  + Set by each class
  + Integer
* Racecantripsknown
  + Cantrips are basically free spells magic users can cast anytime.
  + This is how many are known by the character’s race for level 1 set up
  + Set by each race
  + Integer
* profiencyBonus
  + Set by character’s level
  + Integer
* SpellSaveDC
  + This is a saving throw for magic users
  + Calculated by each class differently
  + Integer
* Spellattackmod
  + This is a bonus to spell attacks
  + Calculated by each class differently
  + Integer
* SpellcasterAbility
  + This the source of a characters magic
  + Set by each class differently
  + String
* FighterBattlemastersave
  + This is a choice for Fighters with the battle master save
  + Determines if a character uses Strength or Dexterity for their ManeuversSaveDC
  + No other class uses
  + String
* inspirations
  + Number used in various classes for class specific features
  + Used to remember number last used for removal of old feature to add new feature with new calc
  + Integer
* kipoints
  + Used by Monks for Ki points
  + Class specific
  + Set by Monk class at certain levels
  + Integer
* kipointsleft
  + Used by Monks for Ki points
  + Class specific
  + Tracks Kipoints used by class
  + Integer
* sneakattack
  + Used by Rogues
  + Class specific
  + Used for SneakAttackDamage calculations
  + String
* sorcerypoints
  + Used by Sorcerers for Spells
  + Class specific
  + Set by Sorcerers class at certain levels
  + Integer
* sorcerypointsleft
  + Used by Sorcerers for Spells
  + Class specific
  + Tracks sorcery points left for Sorcerers
  + Integer
* gold
  + Number of gold pieces character owns
  + Money
  + Integer
* Copper
  + Number of copper pieces character owns
  + Money
  + Integer
* Silver
  + Number of Silver pieces character owns
  + Money
  + Integer
* EP
  + Number of Electrum pieces character owns
  + Money
  + Integer
* PP
  + Number of Platinum pieces character owns
  + Money
  + Integer
* Numberofchoosenlang
  + Number of languages the user can choose for their character
  + Calculated by adding to it for every class, race and background
  + Integer
* NumberofChoosenskills
  + Number of skills the user can choose for their character
  + Calculated by adding to it for every class, race and background
  + Integer
* Abilityincrease
  + Number of ability point increases that a user can choose for their character
  + Calculated by adding to it for every class, race and background
  + Integer
* Speed
  + How fast the character can travel in a round
  + Determined by race
  + Integer
* Initiative
  + Determines turn order in combat
  + Calculated by dexterity mod
  + Integer
* CurrentArmor
  + The armor the character is currently wearing
  + Filled in by user
  + Uses Armor class that has following variables
    - Armorname - String
    - ArmorAC - Integer
    - ArmorBonus - Integer
* Shield
  + This is if the character has a shield in its equipment list
  + Calculated with recalcShieldandArmorAC
    - If a shield is found then set to true
    - Else false
  + Boolean
* Equipment\_counter
  + Count of how many items are in equipment
  + Calculated by adding items to equipment list
  + Integer
* feat\_counter
  + Count of how many items are in features list
  + Calculated by adding items to feats list
  + Integer
* subclasscounter
  + Count of how many items are in subclass abilities list
  + Calculated by adding items to subclass abilities list
  + Integer
* Domain
  + Used for deciding domain of clerics
  + Set by UI
  + Integer
* feats
  + List of all class, race and background features the character has
  + Does not include subclassabilities
  + Added to by CheckandaddFeat
  + Removed from by removeFeats
  + Arraylist of String values
* equipment
  + List of all equipment the character has
  + Added to by addequip
  + Removed from by removeequip
  + Arraylist of String values
* languages
  + List of all languages the character has
  + Used to compare against for list of new possible languages
  + Added to by Checkandaddlang
  + Removed from by removelang
  + Arraylist of String values
* ArmorandWeaponsProf
  + List of all Weapon and Armor proficiencies the character has
  + Added to by CheckandaddWepPro
  + Removed from by removeweppro
  + Arraylist of String values
* Spells
  + Old list of spells
  + Outdated but has not been removed since used console testing code.
  + Arraylist of String values
* Savingthrow
  + These are the skills the character is proficient in to try and save
  + Holds the saving throw skills for a character
  + Determined by class
  + String array
* Weaponsheld
  + Up to 3 weapons the character is currently holding
  + Filled in by user
  + Array of Weapons class
    - Each Weapon has 4 variables
      * Weaponname String
      * Atkbns Int
      * Damage String
      * Type String
* Subclassability
  + List of all subclass features the character has
  + Added to by CheckandaddSubclassabilities
  + Removed from by removeSubclassabilities
  + Arraylist of String values
* SavingthrowLocation
  + Used for filling in the savingthrow array
  + Keeps track of location in array that has been filled in
  + Prevents out of bound errors if higher than 1.
  + Integer
* spellsknown
  + Number of spells a warlock knows
  + Set by the Warlock class at different levels
  + Integer
* Spellslots
  + Max number of spells a Warlock can have ready
  + Set by the Warlock class at different levels
  + Integer
* spellslotsLeft
  + Number of spells left a Warlock can have ready
  + Calculated by refreshing by setting equal to spellslots
  + Integer
* invocationsknown
  + Number of invocations left a Warlock can have ready
  + Set by the Warlock class at different levels
  + Integer
* numberspellsLevel
  + Number of spells a character can have at a given time
  + Set by class at different levels
  + Integer array
    - Each index in the array stands for a different spell level
      * numberspellsLevel [0] = Level 1 spells
      * numberspellsLevel [1] = Level 2 spells
      * numberspellsLevel [2] = Level 3 spells
      * numberspellsLevel [3] = Level 4 spells
      * numberspellsLevel [4] = Level 5 spells
      * numberspellsLevel [5] = Level 6 spells
      * numberspellsLevel [6] = Level 7 spells
      * numberspellsLevel [7] = Level 8 spells
      * numberspellsLevel [8] = Level 9 spells
* numberspellsLevelleft
  + Number of spells left a character can have at a given time
  + Set by refreshing so that it is equal with numberspellslevel at each index
  + Integer array
    - Each index in the array stands for a different spell level
      * numberspellsLevelleft [0] = Level 1 spells
      * numberspellsLevelleft [1] = Level 2 spells
      * numberspellsLevelleft [2] = Level 3 spells
      * numberspellsLevelleft [3] = Level 4 spells
      * numberspellsLevelleft [4] = Level 5 spells
      * numberspellsLevelleft [5] = Level 6 spells
      * numberspellsLevelleft [6] = Level 7 spells
      * numberspellsLevelleft [7] = Level 8 spells
      * numberspellsLevelleft [8] = Level 9 spells
* Newskills
  + List of Skills that can be learned by character that can learn skills based on their race
  + Compared against skills values in possibleskills method before being sent to UI
  + ArrayList<String>
* racepossibleCantrips
  + List of cantrip sills that can be learned by character that can learn cantrips based on their race
  + ArrayList<String>
* CantripsKnown
  + List of cantrips known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 0 as parameters
  + Removed from by RemoveSpell() with spellname and 0 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl1Known
  + List of Level 1 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 1 as parameters
  + Removed from by RemoveSpell() with spellname and 1 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl2Known
  + List of Level 2 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 2 as parameters
  + Removed from by RemoveSpell() with spellname and 2 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl3Known
  + List of Level 3 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 3 as parameters
  + Removed from by RemoveSpell() with spellname and 3 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl4Known
  + List of Level 4 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 4 as parameters
  + Removed from by RemoveSpell() with spellname and 4 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl5Known
  + List of Level 5 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 5 as parameters
  + Removed from by RemoveSpell() with spellname and 5 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl6Known
  + List of Level 6 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 6 as parameters
  + Removed from by RemoveSpell() with spellname and 6 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl7Known
  + List of Level 7 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 7 as parameters
  + Removed from by RemoveSpell() with spellname and 7 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl8Known
  + List of Level 8 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 8 as parameters
  + Removed from by RemoveSpell() with spellname and 8 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* Lvl9Known
  + List of Level 9 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 9 as parameters
  + Removed from by RemoveSpell() with spellname and 9 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* RaceSpellsKnown
  + List of Level 9 Spells known to character
  + Can be added to by pop up boxes or by User
  + Added to by checkandAddSpell() with the spell name and 10 as parameters
  + Removed from by RemoveSpell() with spellname and 10 as parameters
    - To remove, spellname must match exactly to existing spell
  + ArrayList<String>
* characterinfo
  + These are notes made by the user
  + To add or remove to this variable,
    - Make the changes to the Info box
    - Hit Save info button
      * Sets the text in the info box as a string to characterinfo
  + One big string variable

**5. Index**