# Creating a Dungeons & Dragons Character

**Select a race.** There are seven basic races, though some players may allow other races to be used. Each race has certain unique characteristics, abilities, strengths, and weaknesses. The seven basic races are:

* *Dwarves.* Short, stocky, stern, and strong. They have a connection to the earth and often live in mountains or underground lands. Stat modifiers: +2 Constitution, +2 Wisdom, –2 Charisma.
* *Elves.* Tall, long-lived, aloof, and connected to nature. Elves live in harmony with the natural world around them. Stat modifiers: +2 Dexterity, +2 Intelligence, –2 Constitution.
* *Gnomes*. Mysterious, adventure-seeking, and strange. Gnomes are the smallest common race. Stat modifiers: +2 Constitution, +2 Charisma, –2 Strength.
* *Half-elves.* Solitary, long-lived, graceful, and hearty. Half-elves are few in number, and tend to be wanderers due to their lack of homeland. Stat modifiers: +2 to one ability score.
* *Half-orcs.* Independent, strong, and distrusted. Half-orcs considered monstrosities by common folk. They are tall and powerfully built. Stat modifiers: +2 to one ability score.
* *Halflings.* Optimistic, cheerful, curious, and small in stature. On average, only 3 ft (.9 m) tall, they are agile but physically weak. Stat modifiers: +2 Dexterity, +2 Charisma, –2 Strength.
* *Humans.* Adaptive, ambitious, and well-balanced. Humans are the dominant race, and are diverse in appearance and culture. Stat modifiers: +2 to one ability score.

(Source: <https://www.wikihow.com/Create-a-Dungeons-and-Dragons-Character>)

**Choose a class.** A character’s class is like their profession. There are 11 basic classes:

* *Barbarian.* Uncivilized and strong. Considered brutal berserkers.
* *Bard.* Charismatic and crafty. Bards use skills and spells to help and harm.
* *Cleric.* A firm believer in a deity. Clerics heal, raise the dead, and direct the wrath of god.
* *Druid.* At one with nature. Druids cast spells, interact with animals, and change shapes.
* *Fighter.* Firm and brave. Fighters are capable with weapons and comfortable in armor.
* *Monk.* Martial masters. Monks train their minds and bodies for offense and defense.
* *Paladin.* Followers of what is good and just. Paladins are devoted knights.
* *Ranger.* Masters of woodcraft and wildlife. Rangers track and hunt foes.
* *Rogue.* Stealthy assassins. Rogues are cunning thieves and able scouts.
* *Sorcerer.* Natural born spellcasters. Sorcerers command strange, ancient energies.
* *Wizard.* Lifelong magic students. Years of study allow wizards to use awesome magic power.

(Source: <https://www.wikihow.com/Create-a-Dungeons-and-Dragons-Character>)

**Identify the main attributes of your character**. Every character is made up of six essential attributes. These are given scores, with high scores reflecting positive bonuses from that attribute and low scores penalties. A score of 10 in any attribute is considered to be average. The six attributes are:

* *Strength (Str).* A measure of physical power. Important for combat characters, like fighters, monks, and paladins. Strength also dictates how much weight you can carry.
* *Dexterity (Dex).* A measure of agility. This includes balance and reflexes. Important for rogues, light- to medium-armor wearing characters, and ranged attackers (like bow and sling users).
* *Constitution (Cons).* A measure of health and fortitude. Constitution increases your character’s hit points, which if depleted, can result in fainting or death.
* *Intelligence (Int).* A measure of knowledge. Important for wizards and classes requiring reason or learning, like paladins.
* *Wisdom (Wis).* A measure of correct judgement. Wisdom contributes to common sense, awareness, and willpower. Important for clerics, druids, and rangers.
* *Charisma (Cha).* A measure of magnetism. High charisma improves likability, appearance, and leadership ability. Important for bards, paladins, and sorcerers.

**Roll for your starting gold.** Gold (GP), which is the currency of D&D, is usually awarded for completing quests and defeating enemies. However, each character starts with a pre-determined amount. This amount depends on your character’s class. Calculate your starting GP by calculating:

* Barbarian, 3d6 x 10 GP
* Bard, 3d6 x 10 GP
* Cleric, 4d6 x 10 GP
* Druid, 2d6 x 10 GP
* Fighter, 5d6 x 10 GP
* Monk, 1d6 x 10 GP
* Paladin, 5d6 x 10 GP
* Ranger, 5d6 x 10 GP
* Rogue, 4d6 x 10 GP
* Sorcerer, 2d6 x 10 GP
* Wizard, 2d6 x 10 GP

(A d6 is a six-sided die. So for Barbarian, you roll 3 six-sided dice, add up the total, and multiply it by ten.)

## Part One: Character Name, race, and class selection

### A. Create the basic form in HTML5/CSS.

1. Using HTML5, create a basic form with the Name, Race, and Class fields.
   1. Name should be a basic text input.
   2. Character race should be a drop-down menu with the following choices:
      1. Dwarf
      2. Elf
      3. Gnome
      4. Half-elf
      5. Half-orc
      6. Halfling
      7. Human
   3. Character class should be a drop-down menu with the following choices:
      1. Barbarian
      2. Bard
      3. Cleric
      4. Druid
      5. Fighter
      6. Monk
      7. Paladin
      8. Ranger
      9. Rogue
      10. Sorcerer
      11. Wizard
2. Include a submit button and a reset button in the HTML.
3. Add appropriate styling using external CSS.

### B. JavaScript Validation, onclick Event Handlers, and DOM Scripting

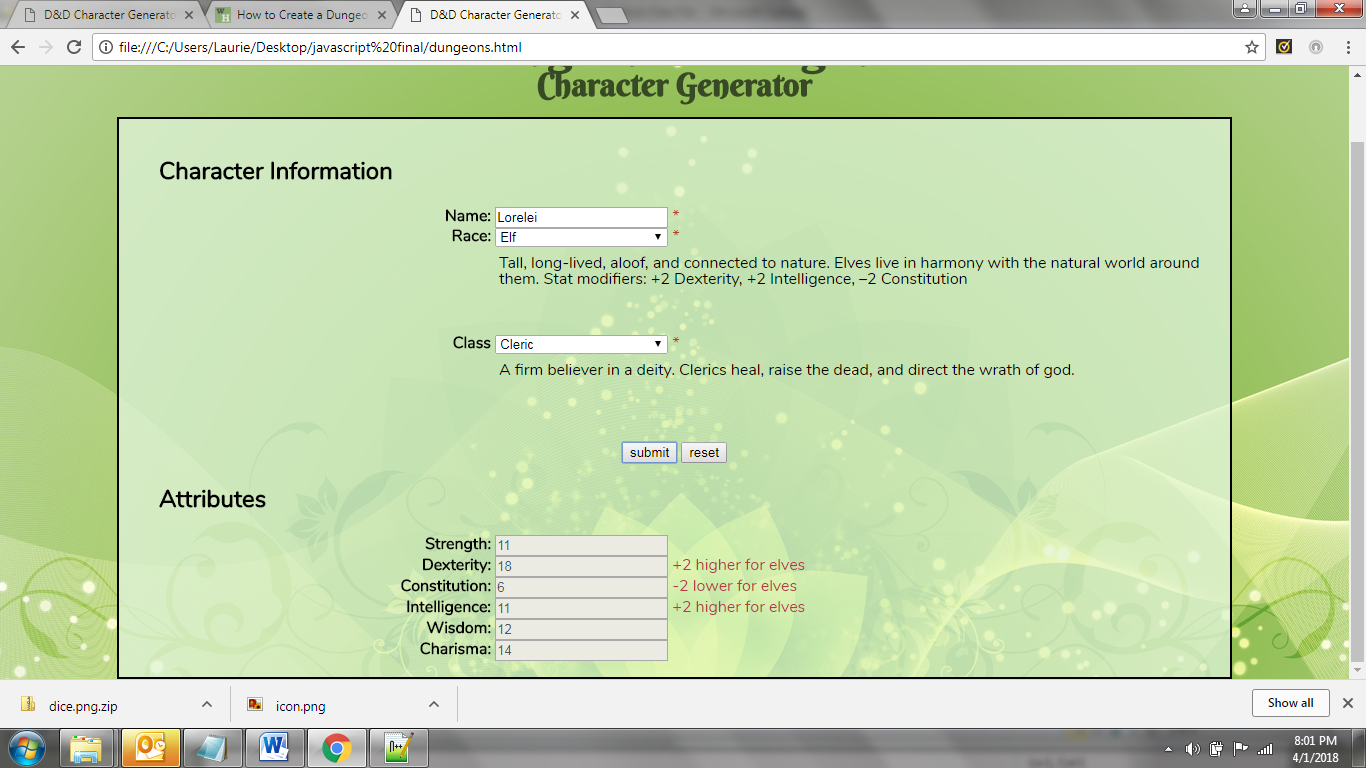
1. Use JavaScript (in an external file) to make the cursor start in the top character name field.
2. When the user clicks the Reset button, JavaScript clears all fields, removes any error messages and race/class descriptions, and returns the cursor to the top name field.
3. When the user clicks the Submit button, JavaScript validates to ensure that the user has competed all three items. If the user has omitted an item, the script will display a message to the right of the input as shown.

### C. Associative Arrays, onchange Event Handlers, and DOM Scripting

1. Store each of the race descriptions in an associative array. For example:  
   raceDescription["dwarf"] = "Short, stocky, stern, and strong. They have a connection to the earth and often live in mountains or underground lands. Stat modifiers: +2 Constitution, +2 Wisdom, –2 Charisma.";
2. Store each of the class descriptions in another associative array.
3. Create a function to display race. When the user changes a race in the race select box, the corresponding description from the array is displayed beneath the race select box in the body of the page.
4. Create a function to display class. When the user changes a class in the race select box, the corresponding description from the array is displayed beneath the class select box in the body of the page.

## Part Two: Calculating ability scores

### D. Confirm and Call rollStats()function

1. Create a JavaScript confirmation that displays the final Name, Race, and Class, and confirms that these are what the user wants. If the user clicks OK, then the script calls a rollStats() function.
2. The rollStats() function will take the place of physical dice rolls. The script will be rolling for each of the six stats: Strength (Str), Dexterity (Dex), Constitution (Cons), Intelligence (Int), Wisdom (Wis), Charisma (Cha).   
     
   In the Dungeons & Dragons game, the player rolls four regular 6-sided dice for each stat but only adds the highest three, disregarding the lowest one. For example, I rolled for my character’s strength:   
     
     
   Since 1 is the lowest, I discard the one, and my character’s strength is 3 + 5 + 5 = 13. Your function will calculate this virtually without physical dice. You will need to repeat this once for each of the six character stats. (HINT: You might want to create a separate function which rolls the stats and call it six times.)
3. Modify the stats based on the user’s character race.
   1. If the character race is Dwarf, add two points to Cons and Wis. Subtract two points from Cha.
   2. If the character race is Elf, add two points to Dex and Intel. Subtract two points from Cons.
   3. If the character race is Gnome, add two points to Cons and Cha. Subtract two points from Str.
   4. If the character race is Halfling, add two points to Dex and Cha. Subtract two points from Str.
   5. If the character race is Half-elf, Half-orc, or Human, the user gets to pick one of the six stats, and the script will add two points to that stat only. (It’s up to you how to best implement this; I used a fieldset in the HTML which was hidden until the user chose one of these three races.)
4. Include text within the span element to the right of the attribute to let the user know what has been modified and why. Example:

## E. Use a switch case to calculate GP

15. Using the formula below, calculate the user’s starting gold (GP) based on their character class and display it in the form. **You must use a switch case for this section of your code.**

* Barbarian, 3d6 x 10 GP
* Bard, 3d6 x 10 GP
* Cleric, 4d6 x 10 GP
* Druid, 2d6 x 10 GP
* Fighter, 5d6 x 10 GP
* Monk, 1d6 x 10 GP
* Paladin, 5d6 x 10 GP
* Ranger, 5d6 x 10 GP
* Rogue, 4d6 x 10 GP
* Sorcerer, 2d6 x 10 GP
* Wizard, 2d6 x 10 GP

## Part Three: Cookies

Similar to what we did in the Task Manager application in Chapter Nine, use local storage to store and retrieve the user’s complete character info (Name, Race, Class, and stats). When the page loads, it should check the local storage file and if character info exists, retrieve the existing info. Pressing the Submit button updates the content of the local storage file. Pressing the Reset button erases the content of the local storage file. (Refer to pages 286-287 in your text).

# GRADING RUBRIC

**General Web Development \_\_\_\_\_\_\_\_/30**

\_\_\_\_\_\_ All HTML documents validate as HTML5

\_\_\_\_\_\_ CSS successfully validates

\_\_\_\_\_\_ All tags are used as they were intended

\_\_\_\_\_\_ File names, page titles, and headlines are leveraged for SEO

\_\_\_\_\_\_ Alt text is supplied for necessary images

\_\_\_\_\_\_ Only external CSS is used

\_\_\_\_\_\_ Pages load quickly

\_\_\_\_\_\_ Website works on all browsers

\_\_\_\_\_\_ A reset style sheet is used

\_\_\_\_\_\_ index.html is used for the main page

**Event Handling (SLO #1) \_\_\_\_\_\_\_\_/30**

\_\_\_\_\_\_ Used external JavaScript only (no script in your HTML!) and successfully used an onload event handler to attach the script.

\_\_\_\_\_\_ Event handler is attached to a Submit button which calls the function that validates user entries and then processes the form

\_\_\_\_\_\_ Event handler is attached to a Reset button which clears all form fields and span tags and returns the cursor to the top input field.

\_\_\_\_\_\_ Event handler is attached to a drop-down select menu which displays the Race description

\_\_\_\_\_\_ Event handler is attached to a drop-down select menu which displays the Class description

\_\_\_\_\_\_ Focus method is used to begin the form with the cursor in the top input field

**Variables (SLO #2) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Script runs in strict mode

\_\_\_\_\_\_ Successfully used variables to store and retrieve information

\_\_\_\_\_\_ All variables are correctly declared in the script

**Data Arrays (SLO #3) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Successfully used an associative array to store character race descriptions

\_\_\_\_\_\_ Successfully used an associative array to store character class descriptions

\_\_\_\_\_\_ Other arrays used as appropriate

**Error Checking & Debugging (SLO #4) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Script is free of syntax errors

\_\_\_\_\_\_ Script is free of runtime errors

\_\_\_\_\_\_ Script is free of logic errors

**Document Object Model (SLO #5) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Displayed messages next to the form inputs as called for in the assignment description

\_\_\_\_\_\_ Removed unnecessary messages next to the form inputs as called for in the assignment description

\_\_\_\_\_\_ Character race descriptions are displayed near the character race select menu and change along with the menu selections

\_\_\_\_\_\_ Character class descriptions are displayed near the character class select menu and change along with the menu selections

\_\_\_\_\_\_ Other uses of the Document Object Model to change the document content as appropriate

**Input Validation (SLO #6) \_\_\_\_\_\_\_\_/10**

\_\_\_\_\_\_ Script validates user input to ensure that a name is entered

\_\_\_\_\_\_ Script validates user input to ensure that a race is entered

\_\_\_\_\_\_ Script validates user input to ensure that a class is entered

\_\_\_\_\_\_ User error messages are displayed as appropriate

**Object Classes (SLO #7) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Successfully manipulated objects as required for a functioning script

**Cookies (SLO #8) and User Authentication (SLO #9) \_\_\_\_\_\_\_\_/15**

\_\_\_\_\_\_ Used cookies to store character info

\_\_\_\_\_\_ Retrieved character info from cookies

\_\_\_\_\_\_ Reset button successfully clears cookies

**Total \_\_\_\_\_\_\_\_/160**