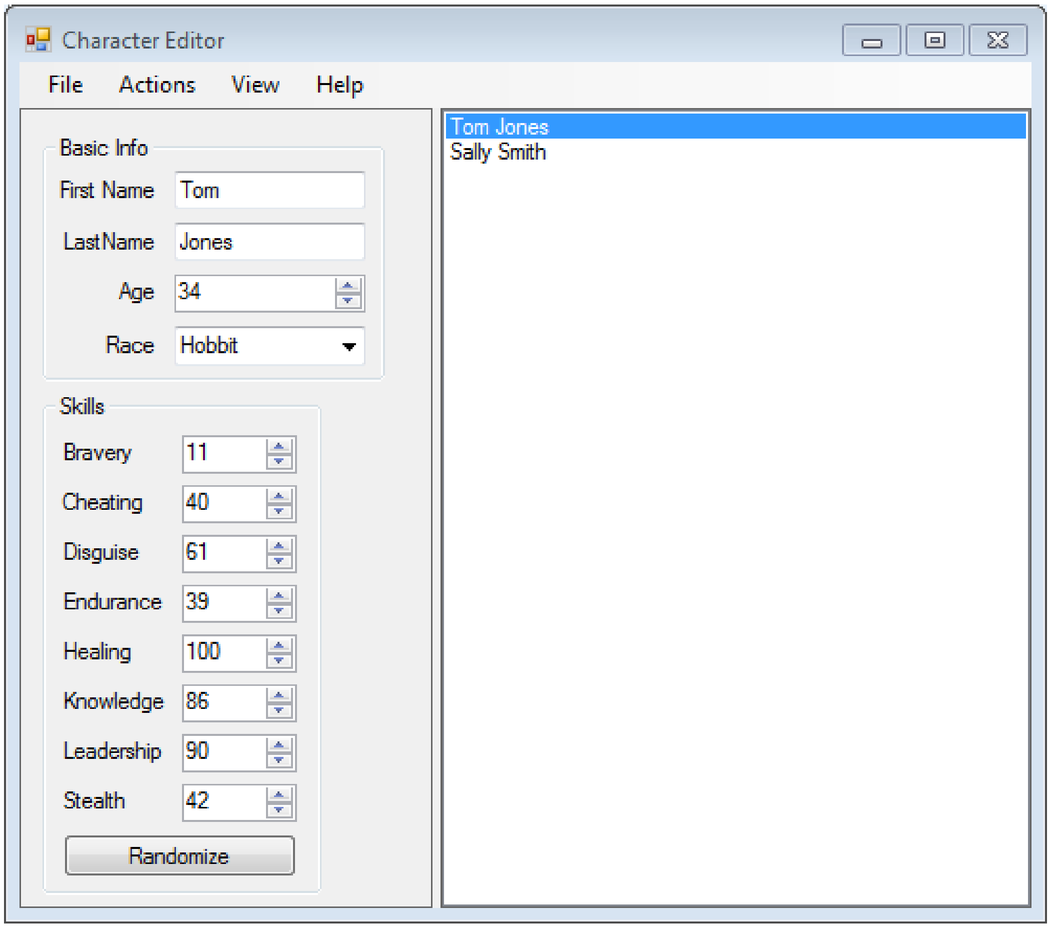
# Character Editor

## Objectives

The purpose of the Character Editor is to start becoming familiar with the .NET framework and C#. Many aspects of this assignment will reappear on the midterm examination. The most important ones are:

1. Being able add new instances of the Character class to the Listbox and control what the ListBox displays by overriding the Character's ToString method.
2. When items are selected in the ListBox you need to be able to retrieve the selected Character and place its values back into the detail controls.
3. When the details of a Character are changed and the update menu item is clicked you need to be able to make the changes permanent. That also includes updating the string displayed in the ListBox.
4. You need to be able to deselect items in the ListBox.
5. You need to be able to remove items from the ListBox.

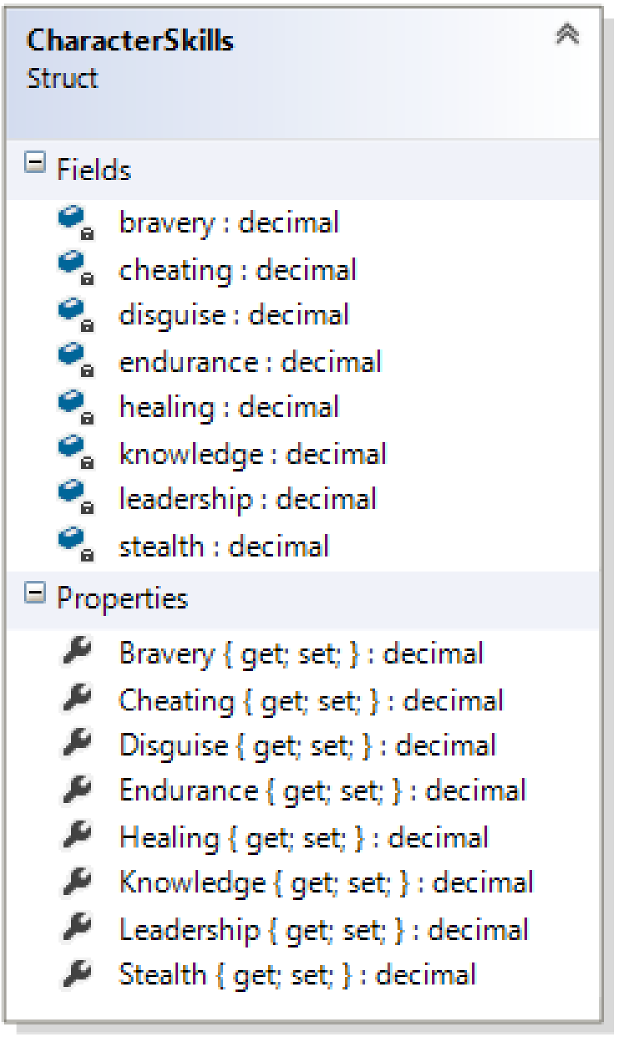


| | |
|---|---|
| First Name | Tom |
| LastName | Jones |
| Age | 34 |
| Race | Hobbit |
| Bravery | 11 |
| Cheating | 40 |
| Disguise | 61 |
| Endurance | 39 |
| Healing | 100 |
| Knowledge | 86 |
| Leadership | 90 |
| Stealth | 42 |

Tom Jones
Sally Smith

## Supporting Objects

In order to construct the Character class according to specifications a structure and enumeration will need to be defined:



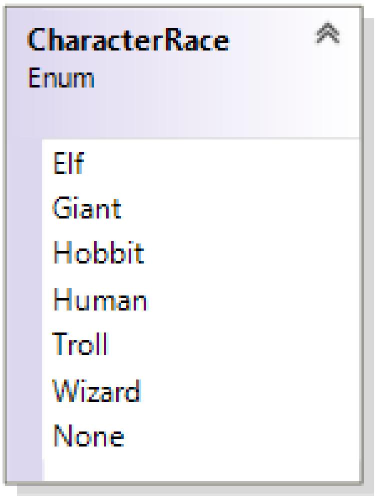
**CharacterSkills**
Struct

Fields

- bravery : decimal
- cheating : decimal
- disguise : decimal
- endurance : decimal
- healing : decimal
- knowledge : decimal
- leadership : decimal
- stealth : decimal

Properties

- Bravery { get; set; } : decimal
- Cheating { get; set; } : decimal
- Disguise { get; set; } : decimal
- Endurance { get; set; } : decimal
- Healing { get; set; } : decimal
- Knowledge { get; set; } : decimal
- Leadership { get; set; } : decimal
- Stealth { get; set; } : decimal

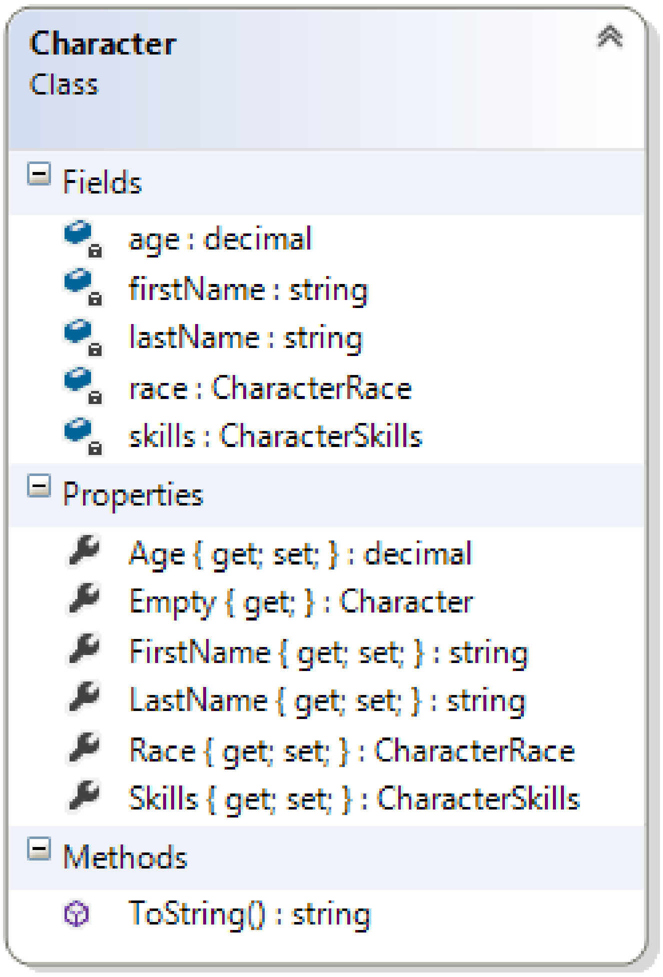


**CharacterRace**
Enum

- Elf
- Giant
- Hobbit
- Human
- Troll
- Wizard
- None

## The Character Class

The following class will incorporate the supporting structure and enumeration into a single object that defines a character in a game.



Character
Class

Fields
- age : decimal
- firstName : string
- lastName : string
- race : CharacterRace
- skills : CharacterSkills

Properties
- Age { get; set; } : decimal
- Empty { get; } : Character
- FirstName { get; set; } : string
- LastName { get; set; } : string
- Race { get; set; } : CharacterRace
- Skills { get; set; } : CharacterSkills

Methods
- ToString() : string

## Strategy

The essence of this program will be to create, edit, save and retrieve a list of characters and to start to become familiar with C# and .NET programming. Because the Character class is complex your might want to start simply and work up from there. The following might present a good strategy for tackling the assignment.

1. Begin with a simple Character class that just contains a few basic pieces of information like the characters name.
2. Now, get the basic functionality working with this simple class. Be able to add characters to the ListBox; retrieve them when the character is selected in the ListBox; update, remove and deselect this simple character type.

3. Now add the CharacterRace enumeration to the character class and get that working.
4. Next create the CharacterSkills structure described above and get that working.
5. The saving and opening of the list is the last thing to tackle.

## Hints

### ComboBox and Enums

The CharacterRace enumeration will be selectable through a ComboBox. The ComboBox works almost just like the ListBox. To fill the ComboBox with the enumeration values check out its DataSource property and the Enum class's GetValues method.

### Saving the Characters to a File

To get a filename from the user use the SaveFileDialog class. To write the file use System.IO.StreamWriter. If each Character is written to a separate line in the file with its values separated by commas this is known as the Comma Separated Value format.

### Retrieving the Characters from a File

To get a filename from the user use the OpenFileDialog class. To read the file use StreamReader. If you retrieve each whole line as a string use the String class's Split method to separate the individual values. To convert a CharacterRace enumeration value from a string to the enumeration value itself use Enum.Parse.

### Hiding Windows

To make a control or container invisible use the Visible property.

### Disabling or Checking Menu Items

To enable and disable menu items use the ToolStripMenuItem.Enabled property. To check or uncheck them use the Checked property.

### Generate Random Numbers

To randomize the character's skills use the Random class.