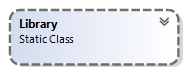
You will be learning to read from a file and building the three interconnected types illustrated below:



All the above type must decorated with the public keyword

# The SongGenre enum

This enum comprise of seven types of songs

To code this add a new item to your project and select the class option. Make the necessary changes to the header to reflect the enum type. You should decorate this enum with the Flags attribute, because values may be combined e.g. Country+Blues.

In order to make the flags work as you would expect, you must set each value to an ascending powers of 2 e.g. 0, 1, 2, 4, 8, 16 …

|  |
| --- |
| **SongGenre**  Enum |
| **Members**  Unclassified = 0  Pop = 1  Rock = 2  Blues = 4  Country = 8  Metal = 16  Soul = 32 |

# The Song class

This acts like a record for the song. The setter is missing for all the properties

|  |
| --- |
| **Song**  Class |
| **Properties**  + «property setter absent» Artist : **string**  + «property setter absent» Title : **string**  + «property setter absent» Length : **double**  + «property setter absent» Genre : **SongGenre** |
| **Methods**  + «constructor» Song(title : **string**, artist : **string**, length : **double**, genre : **SongGenre**)  + ToString() : **string** |

#### Description of the class members

##### Fields

There are no fields.

##### Properties:

This class comprise of four auto-implemented properties with public getters and setters absent. See the UML class diagram above for more details.

##### Constructor:

**public Song(string title, string artist, double length, SongGenre genre)** – This constructor that takes four arguments and assigns them to the appropriate properties.

##### Method:

**public override string ToString()** – This public method overrides the **ToString()** method of the object class. It does not take any argument and returns a string representation of the object. See the output for hints on the return value of this method.

# The Library class

This is the front end of the application. This is a static class therefore all the members also have to be static. Remember class members are accessed using the type instead of object reference.

|  |
| --- |
| **Library**  Static Class |
| **Fields**  $- songs : **List**<**Song**> |
| **Methods**  $+ LoadSongs(filename : **string**,) : **void**  $+ DisplaySongs() : **void**  $+ DisplaySongs(longerThan : **double**) : **void**  $+ DisplaySongs(genre : **SongGenre**) : **void**  $+ DisplaySongs(artist : **string**) : **void** |

#### Description of class members

##### Fields:

**songs** – this private field is a list of song object is a class variable.

##### Properties:

There are no properties.

##### Constructor:

There is no constructor for this class.

##### Methods

There are four over-loaded methods. Remember that overloading is a technique used to reduce the complexity of the API exposed by the Song class.

A foreach loop is recommended here

**public static void DisplaySongs()** – This is a public class method that does not take any argument and displays all the songs in the collection.

**public static void DisplaySongs(double longerThan)** – This is a public class method that takes a double argument and displays only songs that are longer than the argument.

**public static void DisplaySongs(SongGenre genre)** – This is a public class method that takes a SongGenre argument and displays only songs that are of this genre.

**public static void DisplaySongs(string artist)** – This is a public class method that takes a string argument and displays only songs by this artist.

**public static void LoadSongs(string fileName)** – This a class method that is public. It takes a single string argument that represents a text file containing a collection of songs. You will read all the data and create songs and add it to the songs collection. Examine the contents of the files for a better understanding of how this should be done. You will have to read four lines to create one Song. Your loop body should have four ReadLine(). The recipe below might clarify things:

The contents of text file

Baby

Justin Bebier

3.35

Pop

* Initialize the songs field to a new List of Song
* Declare four string variable (title, artist, length, genre) to store the results of four reader.ReadLine().
* The first read is the title of the song. This can and should used as a check for termination condition. If this is empty then there are no more songs to read i.e. it is the end of the file
* Do three more read and assign them to the appropriate variables.

This is a complex method and rightfully it should be a part of the **Song** Class

* Use the four variables to create a Song object (you will have to convert the third to a double and the fourth to a SongGenre).
* Add the newly created object to the collection.
* And finally do one more read for the title to re-enter the loop.

P.S. Before running the Library.LoadSongs() you will need to copy the files Week\_03\_lab09\_song.txt, Week\_03\_lab09\_song2.txt, Week\_03\_lab09\_song3.txt and Week\_03\_lab09\_song4.txt to the bin\debug folder.

### Test Harness

Insert the following code statements in the **Main()** method of your Program.cs file:

//To test the constructor and the ToString method

Console.WriteLine(new Song("Baby", "Justin Bebier", 3.35, SongGenre.Pop));

Console.WriteLine(new Song("The Promise", "Chris Cornell", 4.26, SongGenre.Country | SongGenre.Rock));

Library.LoadSongs("Week\_03\_lab\_09\_songs4.txt"); //Class methods are invoke with the class name

Console.WriteLine("\n\nAll songs");

Library.DisplaySongs();

SongGenre genre = SongGenre.Rock;

Console.WriteLine($"\n\n{genre} songs");

Library.DisplaySongs(genre);

string artist = "Bob Dylan";

Console.WriteLine($"\n\nSongs by {artist}");

Library.DisplaySongs(artist);

double length = 5.0;

Console.WriteLine($"\n\nSongs more than {length}mins");

Library.DisplaySongs(length);

### Program output

Baby by Justin Bebier (Pop) 3.35min

The Promise by Chris Cornell (Rock, Country) 4.26min

All songs

Baby by Justin Bebier (Pop) 3.35min

Fearless by Taylor Swift (Pop) 4.03min

Runaway Love by Ludacris (Pop) 4.41min

My Heart Will Go On by Celine Dion (Pop) 4.41min

Jesus Take The Wheel by Carrie Underwood (Country) 3.31min

If Tomorrow Never Comes by Garth Brooks (Country) 3.40min

Set Fire To Rain by Adele (Soul) 4.01min

Don't You Remember by Adele (Soul) 3.03min

Signed Sealed Deliverd I'm Yours by Stevie Wonder (Soul) 2.39min

Just Another Night by Mick Jagger (Rock) 5.15min

Brown Sugar by Mick Jagger (Rock) 3.50min

All I Want Is You by Bono (Metal) 6.30min

Beautiful Day by Bono (Metal) 4.08min

Like A Rolling Stone by Bob Dylan (Rock) 6.08min

Just Like a Woman by Bob Dylan (Rock) 4.51min

Hurricane by Bob Dylan (Rock) 8.33min

Subterranean Homesick Blues by Bob Dylan (Rock) 2.24min

Tangled Up In Blue by Bob Dylan (Rock) 5.40min

Love Me by Elvis Presley (Rock) 2.42min

In The Getto by Elvis Presley (Rock) 2.31min

All Shook Up by Elvis Presley (Rock) 1.54min

Rock songs

Just Another Night by Mick Jagger (Rock) 5.15min

Brown Sugar by Mick Jagger (Rock) 3.50min

Like A Rolling Stone by Bob Dylan (Rock) 6.08min

Just Like a Woman by Bob Dylan (Rock) 4.51min

Hurricane by Bob Dylan (Rock) 8.33min

Subterranean Homesick Blues by Bob Dylan (Rock) 2.24min

Tangled Up In Blue by Bob Dylan (Rock) 5.40min

Love Me by Elvis Presley (Rock) 2.42min

In The Getto by Elvis Presley (Rock) 2.31min

All Shook Up by Elvis Presley (Rock) 1.54min

Songs by Bob Dylan

Like A Rolling Stone by Bob Dylan (Rock) 6.08min

Just Like a Woman by Bob Dylan (Rock) 4.51min

Hurricane by Bob Dylan (Rock) 8.33min

Subterranean Homesick Blues by Bob Dylan (Rock) 2.24min

Tangled Up In Blue by Bob Dylan (Rock) 5.40min

Songs more than 5mins

Just Another Night by Mick Jagger (Rock) 5.15min

All I Want Is You by Bono (Metal) 6.30min

Like A Rolling Stone by Bob Dylan (Rock) 6.08min

Hurricane by Bob Dylan (Rock) 8.33min

Tangled Up In Blue by Bob Dylan (Rock) 5.40min

Press any key to continue . . .