Scientific Computing and Object Oriented Programming - Homework 2

On demand (video) content in C#

Deadline 10-04-2018 23:59

Objective Design and implementation of hierarchy of classes to model diverse types of video content that can be played using a streaming service.

As lead software architect of the mass media and entertainment company, you need to re-design the software for the media streaming and on-demand video services in order to make the software more modular and increase code reuse.

As in Homework 1, there are three types of videos: documentary, series episode, and movie. Besides the two common attributes (title and duration in seconds) introduced in Homework 1, the three types of video share an additional attribute: the artwork (image) to represent the video in the homepage or in the search result list. An artwork is modelled by the interface `IImage`, which is defined as follows:

```
namespace scoopflix
{
   public interface IImage
   {
      string GetPath();

      int GetWidth();

      int GetHeight();

   }
}
```

Each video is made available through a streaming service. There are different interactions with a video that must be supported: play the video, pause the video, stop the video, and get the current state of the video. A video can be in one of the following states: "PLAYING", "STOPPED" or "ON_PAUSE".

Write the following classes and interfaces:

- the class `Artwork` that implements the interface `IImage`
- the interface `IVideo` with the following methods: `GetTitle()`, `GetDurationInMinutes()` and `GetArtwork()`; the last method returns an `IImage`
- the interface `IPlayable` with the following methods: `Play()`, `Stop()`, `Pause()` and `GetState()`. The methods `Play()`,

`Stop()` and `Pause()` change the state of a video respectively in "PLAYING", "STOPPED" and "ON_PAUSE". The method `GetState()` returns a string which represents the current state of the video.

- the class `Documentary` which is a *video* with the additional attribute `topic` — see Homework 1; the class should implement both the `IVideo` and the `IPlayable` interfaces
- the class `Movie` which is a *video* with the additional attribute `genre` — see Homework 1; the class should implement both the `IVideo` and the `IPlayable` interfaces
- the class `SeriesEpisode` which is a *video* with the additional attributes `series`, `season` and `episode` — see Homework 1; the class should implement both the `IVideo` and the `IPlayable` interfaces
- the class `MainApp` that consists only in the `static Main()` method, where one instance for each type of video is created; for only one of the instances, the following actions should be performed:
  1. play the video;
  2. print the state of the video
  3. pause the video
  4. print the state of the video
  5. play the video
  6. print the state of the video
  7. stop the video
  8. print the state of the video
  9. play the video
  10. print the state of the video

Note: The classes `Documentary`, `Movie` and `SeriesEpisode` share some common fields and methods: use inheritance to avoid rewriting the same code three times. Differently from Homework 1, set methods are not necessary.

Submission Within the deadline indicated above, each student must submit

- A zipped archive (`Homework2_StudentId.zip`) with source code (`Homework2_StudentId.cs`) and compiled file (`Homework2_StudentId.exe`) should be uploaded on moodle and
- All the classes which *could* be on a single file `Homework2_StudentId.cs`, where `StudentId` is your identifier (numero di matricola).

Submission steps

1. access the web page of the course on moodle: `https://elearning.unipd.it/dicea/course/view.php?id=792`
2. click on *Homework2* in the *Homeworks* section

3. click on *add submission*
4. upload the zipped archive through the available form
5. click on *Edit submission* to modify, if needed, your submission
6. click on *Submit assignment* to submit the homework; once the assignment is submitted you will not be able to make any more changes

If an error occurs during the submission, send the zipped archive via email to `emanuele.dibuccio@unipd.it` and `michele.schimd@unipd.it`