22.09.2021

TASK

KEVIN ORTEGA

Índex

Architecture

The application is based on a WPF with the version of Framework .NET 4.7.2. The design is implemented under the concept of the “MVVM” pattern.

The people class was made by transforming the given xml file to a class.c with the xsd.exe tool. A Microsoft Net Framework 4.8 tool.

The development was carried out with Visual Studio 2019.

Used Libraries

For the development of this application the following libraries have been used:

Race simulation:

For the simulation of the race, the Swizz simulator has been used.

For MVVM pattern:

For the implementation of this design, the use of **System.Collections** has been required for the interaction with data, in addition to use of the **ICommand** Interface was needed the “**System.Windows.Input**” to command elements of our UI, for example the buttons that have been implemented.

Data manipulation

To manage the data has been used the **System.Linq**. Thus to be able to access the data within the collections of the system.

Input and output data

For the input and output data has been used **System.IO** and **System.Xml.Serialization**. With these tools we can import and export an **“.xml**” file through the file dialog window.

Configuration

System.Configuration has been used to make use of app.config. With this, it has been sought to be flexible in the selection of the career mode. This allows us to select the career mode in app.config

The application

The application is designed to load an “.xml” file that will list the participants who wish to compete in a race. This data will be uploaded in the table "Uploaded people", indicated below in Image 1.

Then we can select a person and move them to the Competitors section. This function is carried out when the following button is pressed ->, who will delete it from the list of people loaded to move it to the list of competitors.

If the competitor regrets or something has happened and he will not compete, then he can withdraw from the list of competitors by selecting this person and pressing the <- button. On this way, this person is eliminated from the competitors list and added to the list of loaded people again.

when everything is ready to start the competition then the "Start" button must be pressed to start the race.

Results are displayed at the end when the race is over. Showing the ranking of each competitor along with their race time. The competitors who are sanctioned with an IRM will be shown in their respective order below the table according to the rules of the game.

At the end of the degree, the data can be exported to an "xml" file assisted by a window of file dialogs, indicated below on button Export Race to XML in Image 2.

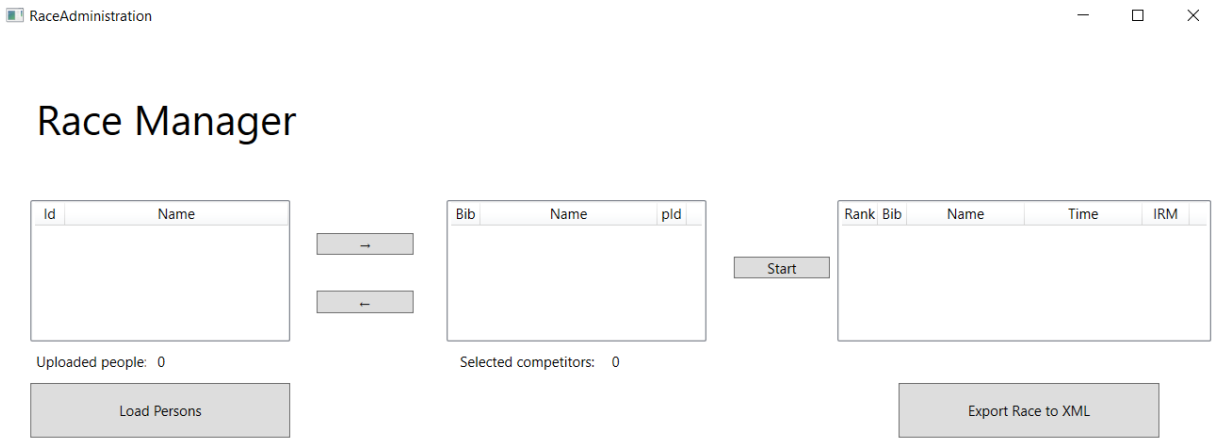

Image 1 App without loaded people

Image 2 App with loaded people and ran

Pending and improvements

The application has the following pending points:

- Loading the list of competitors who have finished the race in real time. It is estimated that the implementation should be done with "Backgroundworker" or some other method for thread management and to have a parallelization effect in the application.
- Change of language in the application. Fundamental so that our application can be used anywhere in the world.
- Enable the edition of IRMs in each Competitor who has finished the race.

The application has the following improvements points:

- The action of the buttons once the race has started must be deactivated.
- The race mode must be able to be set from the race manager.

Attachments

- Project Solution.
- Implementation Document.

Conclusions

The time to implement my solution tacked around 10 Hours. I started on Monday 20, 4 Hours. On Tuesday 21, 3 Hours and today 3 Hours more. This include too the elaboration of the document.

The Project I found very nice and very interesting. We are with this “small project” allow to improve more for example to add a connection to a database and too to implement a better design or create forms to edit more about one competitor.