Name: Shant Ahuja Date: 04/22/20

Lab 2 – WPF and Data Binding

Objectives

Part 1 – Functional Requirements

Part 2 – Example User Interface

Reflection Questions

Background/Scenario

Data binding is one of the most useful features in WPF. Data binding allows you to easilly display data from a data source and display their real-time values on a user interface. Data binding handles the task of querying the values and presenting them on the UI. It also has the ability to work the other direction. If the user modifies any data bound value, data binding has the ability to send the updated data back to the source.

This lab, you will design a simple program which allows a user to see what fonts are installed in the system and get some information about them. The user interface will be written in WPF and all data being displayed and requested from the user will be handed via data binding.

Compared to prior labs, this will be an exploratory lab. Instead of having step-by-step instructions, you will be presented with some functional requirements and then you will be able to tackle this lab as you see fit.

Required Resources

- Visual Studio 2017

Part 1: Functional Requirements

Summary

The goal of the application is to show which fonts are installed on the system and display an example of what the font looks like using a sample string. The application will also allow the user to mark selected fonts as favorites and leave comments on any fonts.

Technical Requirements

- The user interface must be programmed in WPF.
- All non-static content displayed in the user interface must be initialized with data binding.
- Data bound properties must implement INotifyPropertyChanged and raise the PropertyChanged event.

User Interface Requirements

- The user interface must provide:
  - A listing of all fonts currently installed in the system
  - A way for the user to select one font from the list
  - A sample string that utilizes the selected font
  - A way for the user to mark that the selected font is a favorite font
    - There should be a way to unmark a font as favorite.

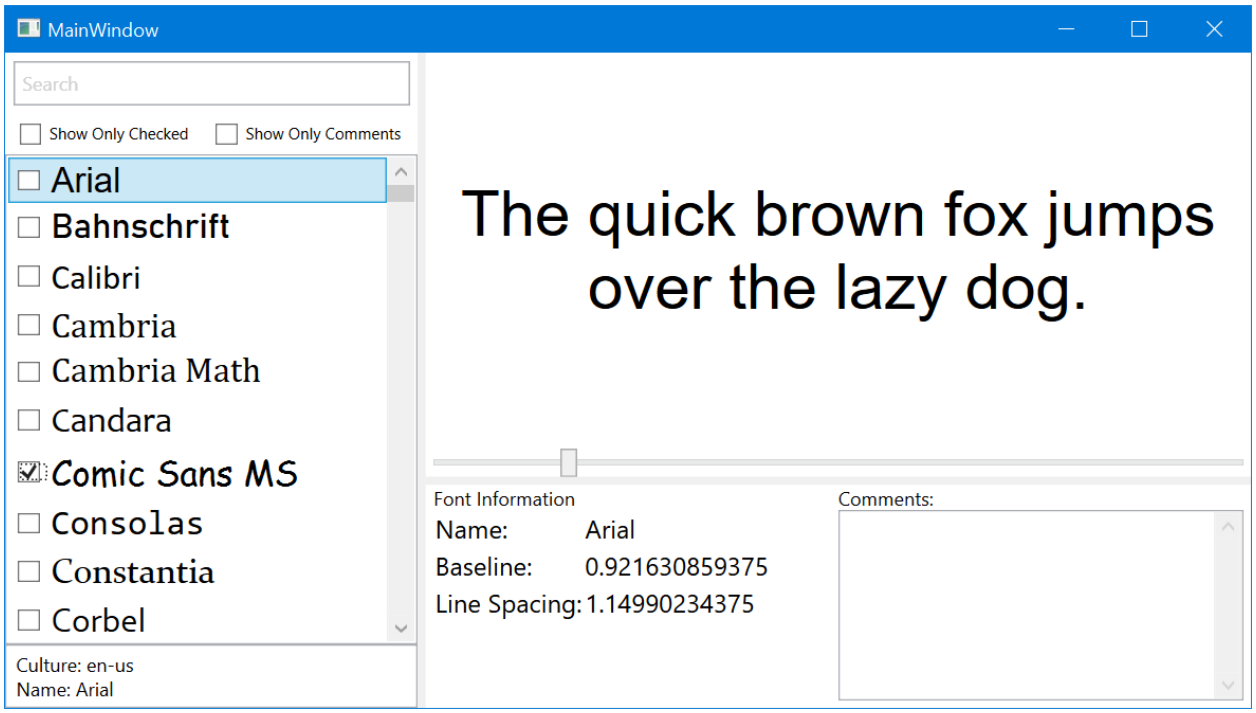
        - There should be some way to indicate that the font is or is not a favorite.
    - A way for the user to leave a comment on the selected font.
        - Allow line breaks in comments.
    - A means of filtering the list of fonts by these requirements:
        - Display only favorite fonts.
        - Display only fonts with comments.
        - Display fonts that match a search string provided by the user
        - Ensure that any or all filtering can be active simultaneously.
        - Treat multiple active filters as a logical AND.
- If a font is selected, display the following information about the selected font:
    - Font Name
    - Baseline
    - Line Spacing
    - Culture Info
- If a font has multiple localizations, display all the localizations for the font. Include the culture code and the locale-specific name.
- The sample string will be “The quick brown fox jumps over the lazy dog.”
- Any interface element that shares similar design properties must use a shared style.
    - For example, instead of declaring the FontSize parameter to 12 for every label, set a style that applies to all the labels that sets them to size 12.

Optional Requirements (Extra Credit)

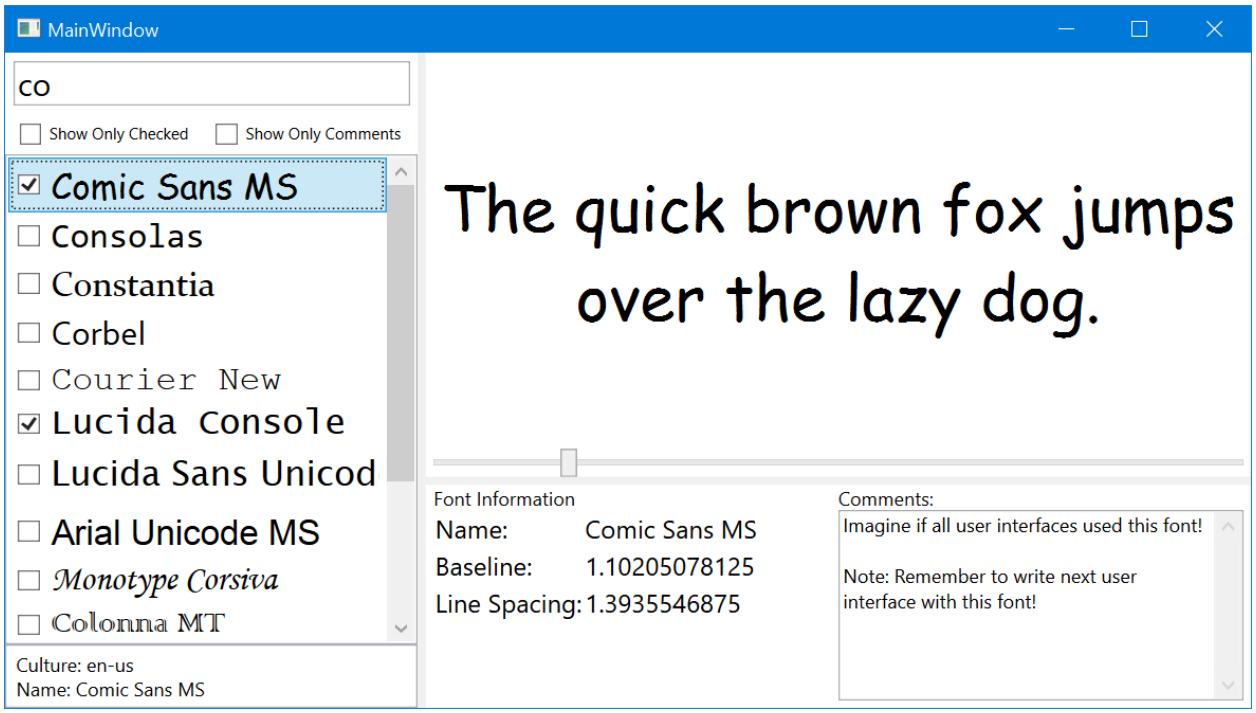
- All static string elements are to be stored in a Resource Dictionary and data bound.
- The search box searches in real-time. (No need to hit a button to enable the search)
- Define an application level template or style to style all elements a certain way.
- Utilize the MVVM design pattern instead of code behind in the UI.

Part 2: Example User Interface

Here is an example of a user interface that meets the criteria described in the previous section. Remember, you are free to design the user interface however you want as long as it meets the requirements. I highly encourage you to experiment with styles and templates to make the most beautiful and amazing user interface you can imagine.



Pictured above is the interface with no filters applied.



Above is an example of a search filter showing all the fonts with “co” in the name somewhere.

Reflection

1. What were some things you had to research to find out how to do?

   XAML class labels, how to reference a class in the backend

2. What are the different modes of data binding and what do they do? Give an example of when to use each one.

   Mode is a type of binding of the Binding property. One time means that we would only update backend once. one way would mean that we could only update the backend from the backend. Onewaytosource means the UI can update backend but backend can't update front end. Twoway means we can update backend, and backend updates front end. A time we would use this to have a checkbox that we would only want the user to see updated, but not be able to modify it. for this case, we would use oneway mode.

3. Why would you use a style over a template when customizing the look of a user interface element?

   style is beneficial because we only put the style definition once at the top of the file and it is set for all class types in the xaml. instead of repeating the styling in the element repeatedly

4. When would you define a style inside a panel compared to the window or application?

   Say if I want to give the user more perceived importance to one side of a application as opposed to another

5. What are some topics in .NET you wish to learn more about?

   full stack development