CPSC 1181 – Assignment 5 Part 2 [70 marks]

Submission:

- **This is the second part of a multi-part assignment.**
  - It builds upon what was asked for in Part 1.
  - The Part 2 submission will be graded as normal.
  - Penalties based on the first submission may be applied
  - Penalties for submitting late, not submitting in a ZIP file/etc. will be applied to your final submission
- Submit a **zip file** containing **only** the .java files to Brightspace prior to the **due date set in Brightspace**. Do not include class or other files
- Submissions that are less than 24 hours late receive a 1% per hour late penalty. Submission that are more than 24 hours late will not be accepted.
- You can use one of your 2 day extensions if you request it before the due date/time.
- Submissions that are unzipped or that contain .class or other unneeded files will be penalized.
- **IMPORTANT: If you are unsure of your submission for any reason, submit it AND email it to me.**

Working With a Partner

- You must tell me **in-person** with your partner that you are working as partners to have permission.
  - This must be done **the week that the assignment is posted**
- If either partner asks for an extension, both partners use up one of their extension.
- If one partner no longer has an extension, neither partner can use an extension.
- Only one group submission is required
- **You must inform me if you are not working with your partner from the previous assignment.**

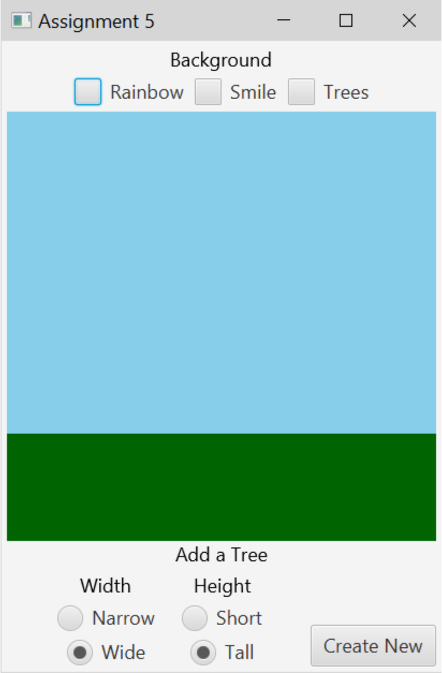
JavaFX Notes

It is important, even more so than usual, to follow the methods taught in the slides for JavaFX. If you stray from them, you will likely receive significant penalties. If there is something you'd like to try and are unsure if it is allowed, contact your instructor.

JavaFX Layout Panes You Can Use: HBox, VBox, BorderPane

You are going to build a JavaFX Application built around the elements created in Part 1. Everything drawn from Part 1 will be placed on a Pane that will be placed in the center area of a BorderPane. Checkboxes, radio buttons, and button will change what is displayed in the center pane. The application must include

- Three or more checkboxes. When clicked they hide/display elements. Two will hide/display elements of the background **(don't worry whether an element is truly background or not. As long as an element appears and disappears, it is ok)** One will hide/display all of the foreground objects added using the controls at the bottom.
- Two sets of two radio buttons each that control the size of the next new object.
- A button that adds a new object based on the radio buttons
- All graphical elements affected by the controls should involve at least two shapes. The foreground objects must meet the requirements specified in Part 1.
- Elements can be hidden/visible when your Application starts, as long as the graphics match the state of the checkboxes. For example, if the checkbox is checked, the element must be shown.



CPSC 1181 – Assignment 5 Part 2 [70 marks]

Example Application Video: https://youtu.be/GAkT7sVp7H4

Layout

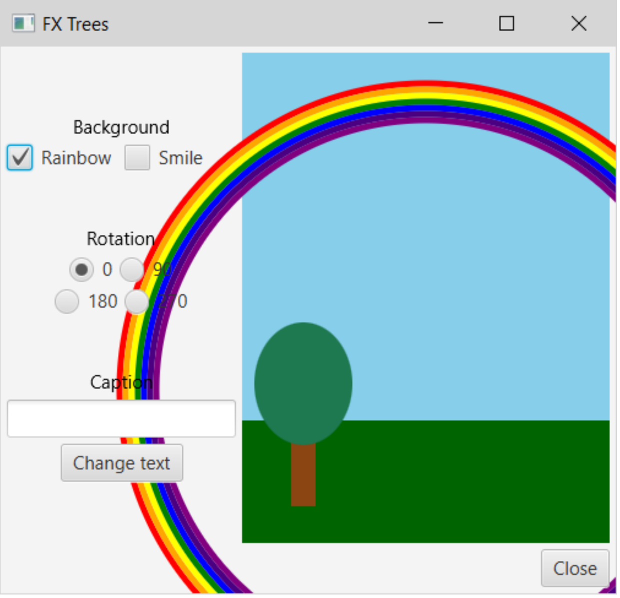
- Take your Part 1 work. You can just rename the root Pane that you are drawing everything on. Then make a BorderPane the new root pane. Add the original Pane to the center area of the BorderPane. Give that Pane the correct size for your graphics.
- Set up the rest of the elements in the layout. Add only HBoxex and VBoxes, and use alignment/margins/padding to position everything. You don't need to match my example exactly, but you should produce something similar. It should
  - Use the same basic structure with similar clusters on the top/bottom.
  - The basic alignment (left/center/right) of elements should be the same
  - There should be padding/margin used to create separation from the edge of the window and between elements in the window

Solving Clipping (only if necessary)

If your graphics from Part 1 went beyond the edge of the Pane, they would have been cut off naturally. But with the BorderPane layout, they will bleed into the other areas of the BorderPane like the rainbow in the image to the right. But, this is easy to solve.

You need to create a Rectangle that is positioned at 0,0 and the same dimensions as the pane for your graphics. Then use the clip method. Then everything will be cut (clipped) off at the edge of the pane.

```
treePane = new Pane();
treePane.setPrefWidth(300);
treePane.setPrefHeight(400);
Rectangle clip = new Rectangle(0,0,300,400);
treePane.setClip(clip);
```



☑ Rainbow □ Smile

● 0 ○ 90 ○ 180 ○ 270

Make Everything Work By Adding Event Handlers

Now, create event handler inner classes for the needed operations. **You must create full inner classes.**

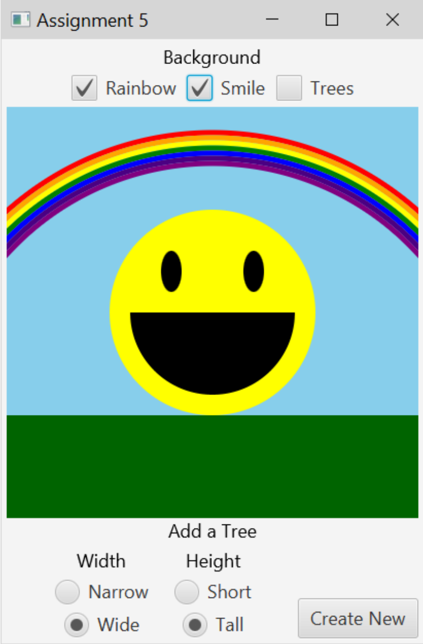
- Don't create a separate event handler for every checkbox, radio button, and other element, but don't just create a single event handler called by everything.
- The CheckBoxes should immediately hide/display the related element. When the application starts, there should be none of the foreground objects to hide (trees in my case).
- When the button is clicked to create a new object, create it sized based on the radio buttons and position it randomly within a majority of your Pane. (Some areas may be excluded)
  - Wide must be at least twice the width of narrow and similar for tall/short
  - While there are only 4 different sizes created with this setup, the constructor for your object must accept and work with any values for width/heigh
  - If the button is clicked and the related checkbox is not checked, you still create the object, but set it to hidden
  - Add your objects to the center Pane, and to an ArrayList. Then you can use that ArrayList to hide/display all elements with a loop when the checkbox is clicked

CPSC 1181 - Assignment 5 Part 2 [70 marks]

Hiding/Showing Groups (and your object class is a group) have the method setVisible(boolean) that you can use to show or hide the group. You can add your background elements to groups to make hiding/displaying them easier, but it is not required

Example Application Video: https://youtu.be/GAkT7sVp7H4

More Example Images



Assignment 5

Background

☑ Rainbow ☑ Smile □ Trees

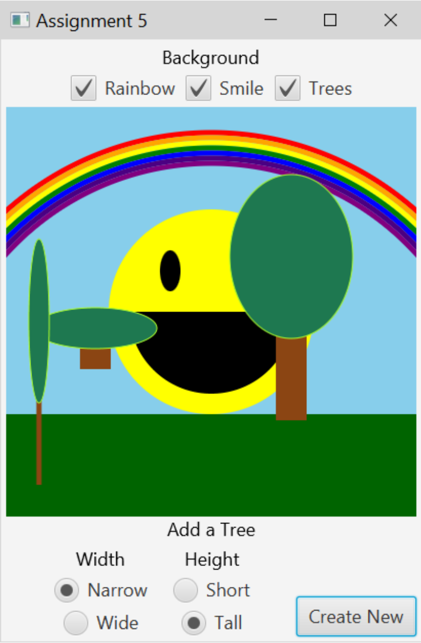
Add a Tree

Width Height

○ Narrow ○ Short

● Wide ● Tall

Create New



Assignment 5

Background

☑ Rainbow ☑ Smile ☑ Trees

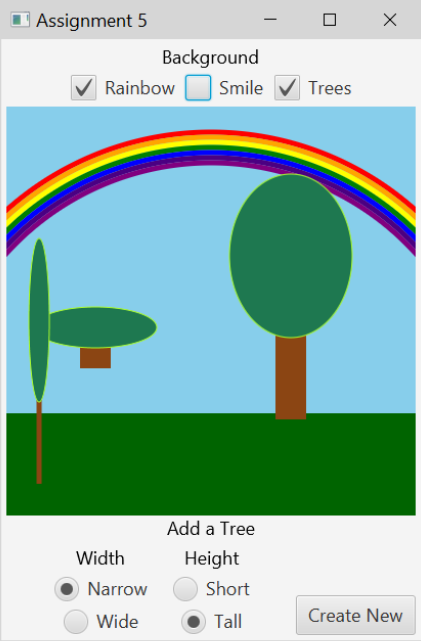
Add a Tree

Width Height

● Narrow ○ Short

○ Wide ● Tall

Create New



Assignment 5

Background

☑ Rainbow □ Smile ☑ Trees

Add a Tree

Width Height

● Narrow ○ Short

○ Wide ● Tall

Create New

Marking Rubric:

Style, Convention, Documentation[5 marks]

You only need to document the outer class at the top of the file. No documentation needed at the method level.

Background Image [10 Marks]

Foreground Item Inner Class [15 marks]

JavaFX Elements/Layout [20 marks]

Checkbox Behavior [10 marks]

Foreground Element Creation [10 marks]