Lab 2: Appendix A Beginning Your Project

- READ EACH STEP CAREFULLY -

Create your project in QtCreator:

Follow the steps below to create a **Qt Quick Project**

(from) http://doc.qt.io/qtcreator/quick-projects.html

1. Select **File** > **New File or Project** > **Application** > **Qt Quick Application** or **Qt Quick Controls Application** > **Choose**.
2. In the **Minimal required Qt version** field, select the Qt version to develop with. The Qt version determines the Qt Quick imports that are used in the QML files. Select the highest version available (5.6).
3. Uncheck "**With .ui.qml file".** Using the Qt Designer will not be covered in this course. Students are welcome to use it but we encourage you write all code yourself. Inserting code with "drag-and-drop" tools only furthers you from what is actually happening.
4. Select kits for running and building your project, and then click **Next**.
5. Review the project settings, and click **Finish** (on Windows and Linux) or **Done** (on OS X) to create the project.

You now will have an empty project with a mouse and text area. ***Delete*** the mouse and text areas so your main.qml only has a **Window** object.

Now at the top of main.qml add a file header comment with the following items.

Name: (your name here)

Lab: (number and lab name)

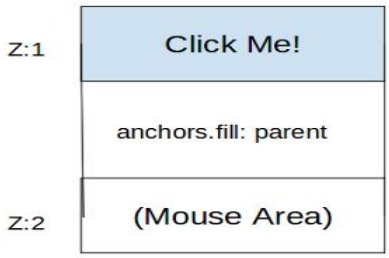
Project Name: (project name here)

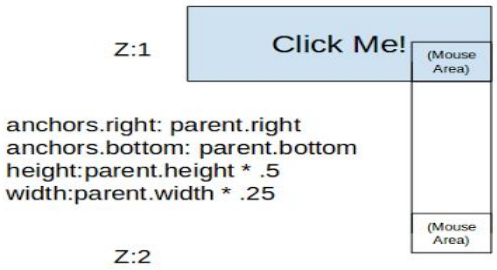
Lab 2: Appendix B Creating Base Components

Build 3 Base Components:

Create a **Button**, **TextRect**, and **Dialog** component.

Button: The button is a basic widget that combines a MouseArea and a Rectangle to form a single object to handle drawing and user input. Location of the MouseArea is important because visually (and through training) the user expects the entire button region to accept the mouse click. (See Below)





The button should have the following signals exposed at minimum.

- onClicked
- onHoverStarted
- onHoverFinished

This component will be used by the rest of the team so think about consumer vs producer responsibilities. Developers should be able to use your button like:

```
Button{
    id:iDisappear
    height: 180
    width: parent.width * .35
    anchors.bottom: parent.bottom
    mouseArea.onClicked:{
        iDisappear.visible = false
    }
    onHoverStarted: {
        color = "blue";
    }
    onHoverFinished: {
        color = "red";
    }
}
```

Lab 2: Appendix B Creating Base Components

Build 3 Base Components (continued):

TextRect: The TextRect will pair a Text object and and Rectangle to create a widget to display text information to the user. The message should not be hardcoded into the TextRect. The idea of building components is to be reusable. Engineer a reusable component that can display text like:

The TextRect should have external properties to set what *text* is displayed, what *color* the background is, and the *font* properties (use an **alias**).

Lab 2: Appendix B Creating Base Components

Build 3 Base Components (continued):

Dialog: The Dialog object pairs the Button and the TextRect object together to create a dialog widget. Similar to the button think about how the consumer of Dialog object will expect them to work. How will the Developer know when internal buttons are clicked? The Dialog should be draggable when clicking on the inside of the rectangle. You should be able to drag the dialog around the application.

All three of these components should be as modular as possible (large chunks should be broken into smaller components). Use your best judgment. In general follow the rules below.

Rules for Building Components:

- Reference **ONLY id**s declared *internally*. (NO Spaghetti code)
- When building new components often change the base '**Item**' to a **Rectangle** instead (adds visual properties).
- When changing scope (working inside of a component) think about that component only. Not the other things in the scene. But instead think about all items relevant to the parent item (of the component).
- Have a big picture in mind of the relationship between Components.

Lab 2: Appendix C Creating Splash Screen

Create your splash screen/home screen :

The code for your splash screen can be written in main.qml. But following the component rules above it might be good to build a SplashScreen/HomeScreen object.

Your screen should have 2 dialogs. One will be hidden behind the other (use Z stacking order).

The dialog should contain:

- A Paragraph explaining your project (can be from your ReadMe).
- A Button that says 'Ok' - when clicked will close (hide) the dialog
- Behind the dialog will be a smaller dialog that says "Hello from, (Your Name)".
- On the smaller dialog, the button labeled 'Goodbye' will close the application when clicked.