

GUI design in C++

Using MFC (Microsoft Foundation Classes)

By: Rishinder Paul

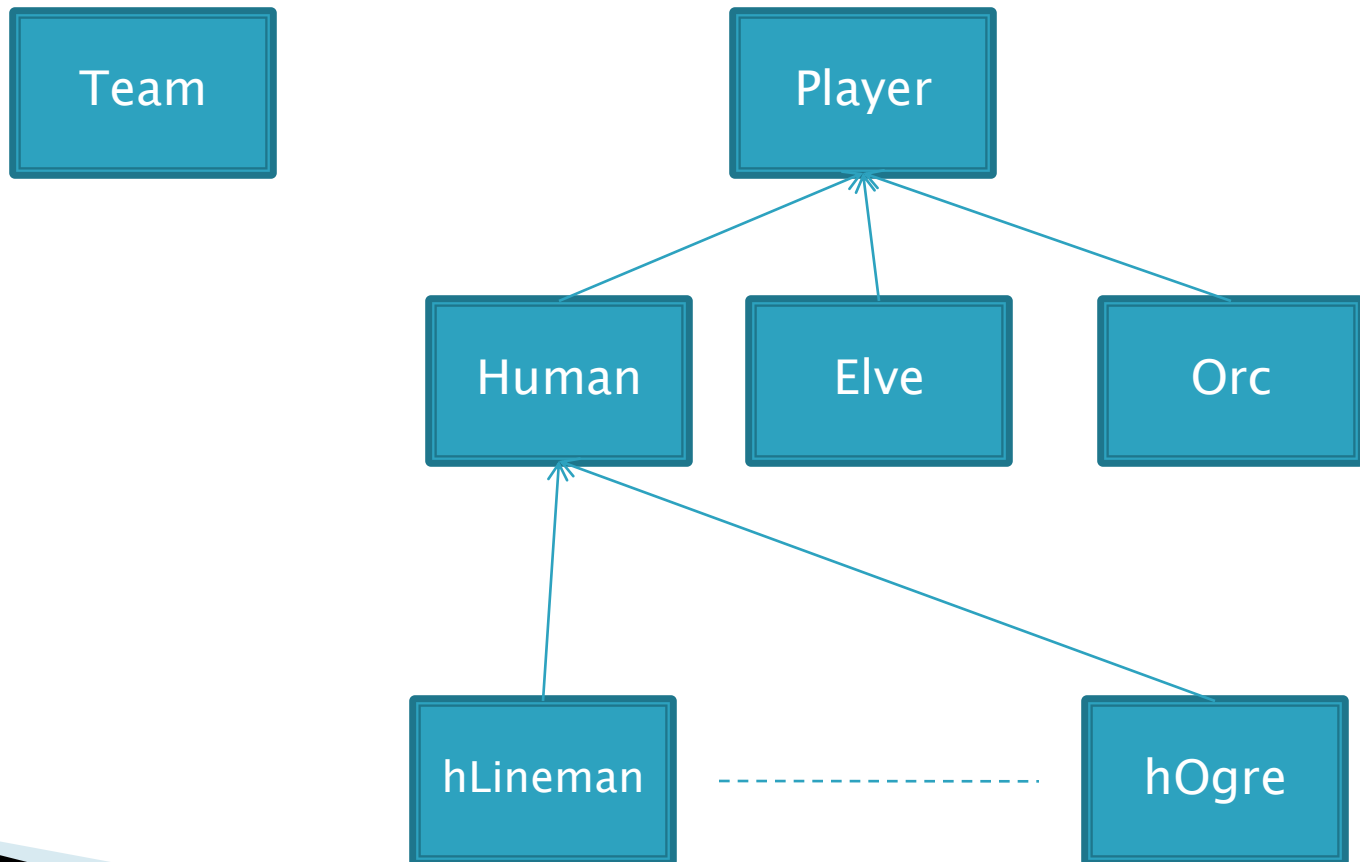
Programming with Dialogs

Designing the Team Editor

- ▶ Steps:
 - Design program Structure and Logic
 - Design Dialogs and GUI
 - Link the logic with GUI

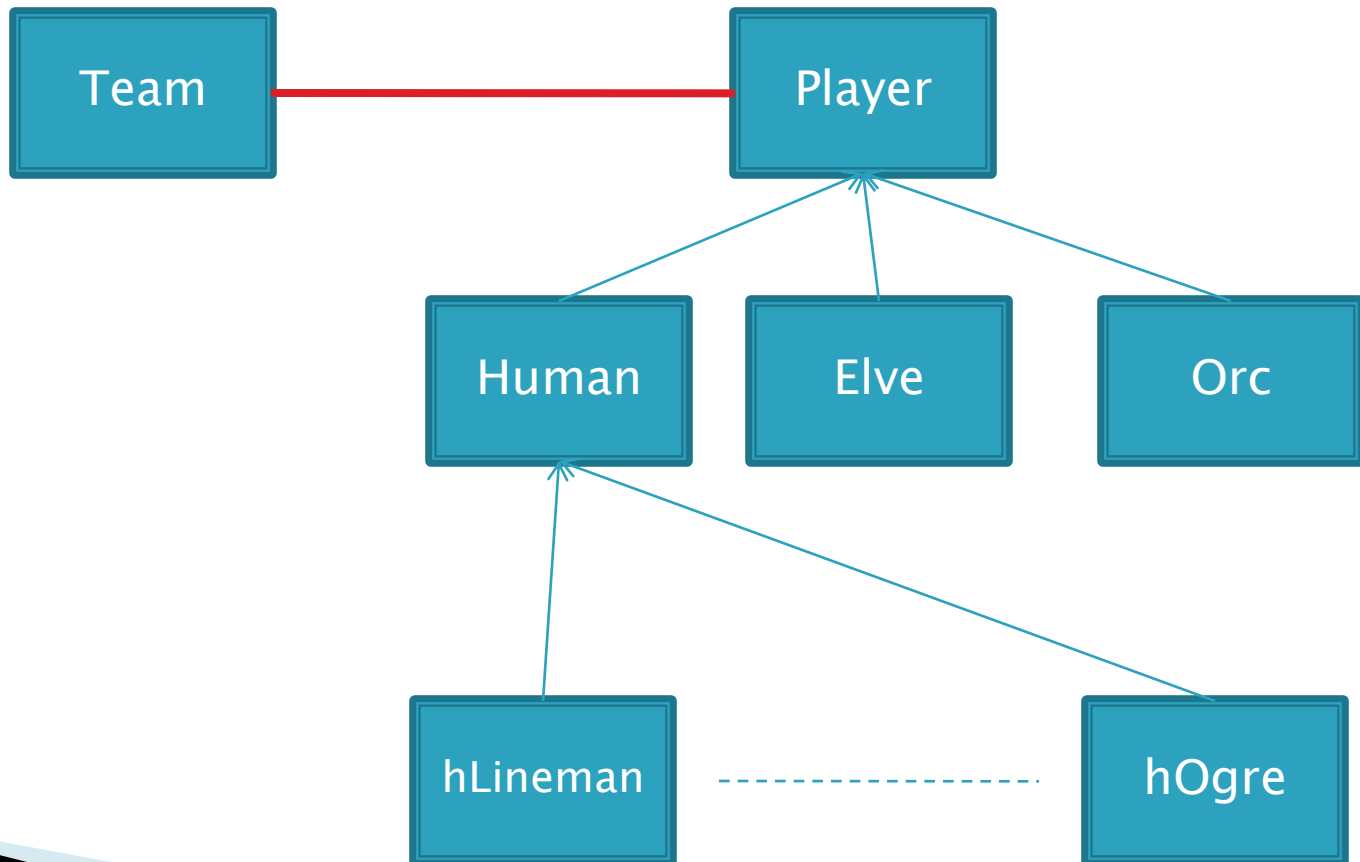
Program Structure

- ▶ Class diagrams



Some Coupling ☹️

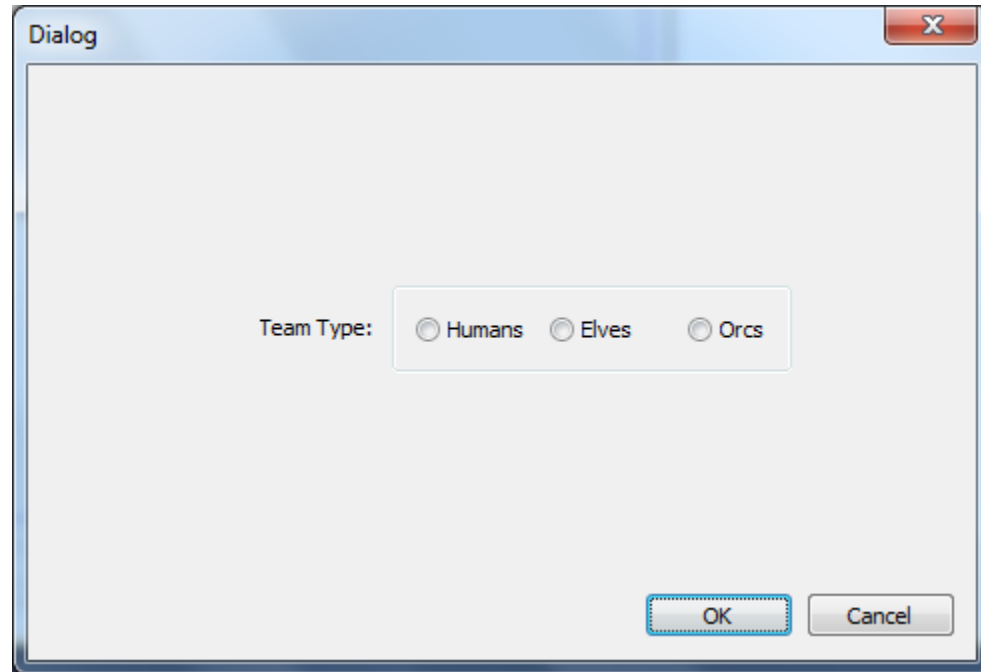
- ▶ Class Relationships



Dialog Based GUI

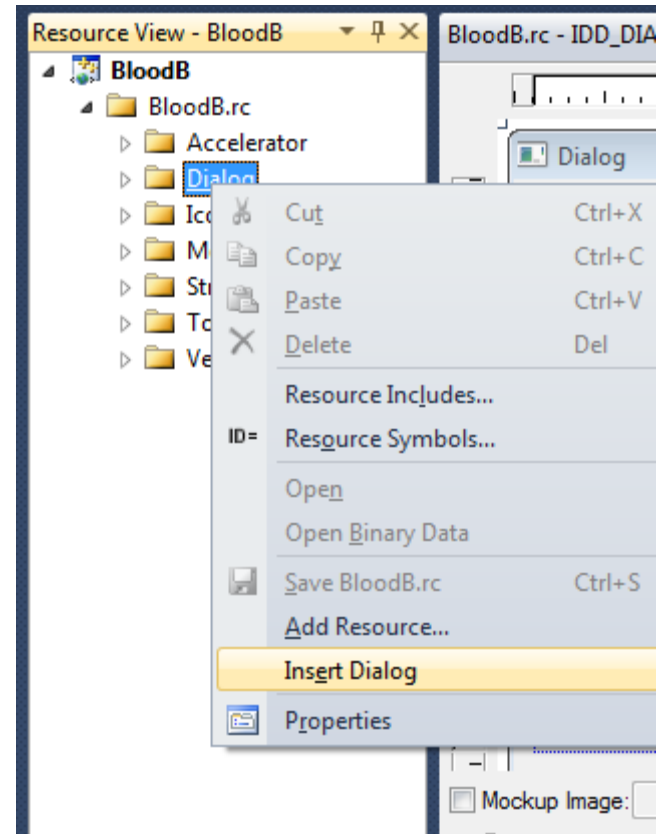
- ▶ Dialog based User Interface makes a lot of stuff easier for us:
 - It makes use of Drag-and-Drop just like VB.
 - You can pick whichever component you need from the toolbox and place it on your window.
 - It provides you with a huge selection of components.
 - It saves you from coding. 😊

An example Dialog



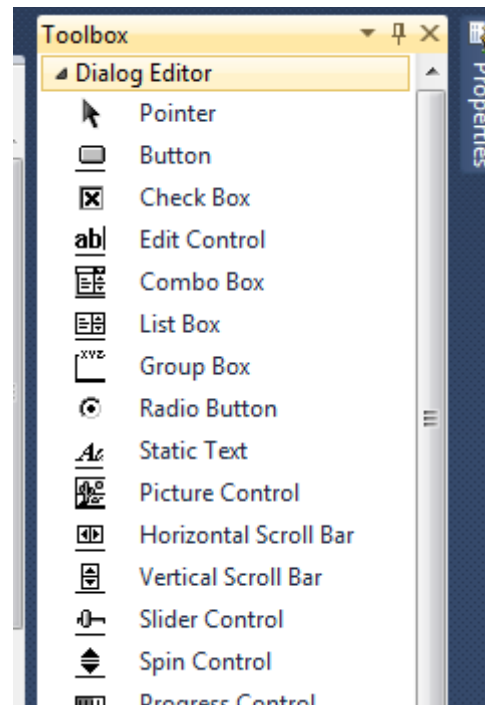
Creating a Dialog

- ▶ Go to menu bar: View > Resource View or Hit CTRL+SHIFT+E
- ▶ Expand the tree In the left hand panel.
- ▶ Right Click: Dialog
- ▶ Choose: Insert Dialog

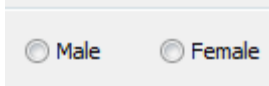


The Toolbox

- ▶ Go to menu bar: View > Toolbox or Hit CTRL+ALT+X



Some Basics ☹️

- ▶ Whenever we create a User Interface, we have to understand what sort of components we need.
 - For example, if we want that the user should choose one from the two available options, like Gender (M/F). We should use the Radio Button control.
 - Giving a Textbox for this choice may not work out because the user may enter “Man” or “Boy” which the computer cannot understand.

Some more Basics ☹️

- ▶ Using complex components for simple tasks is not a good idea. E.g. Using a List-Box control for the previous example.
 - Since we know that we have just two options to choose from, a list box will prove unnecessary as it may take more space and more time to code than Radio Buttons.

Enough of Basics 😐

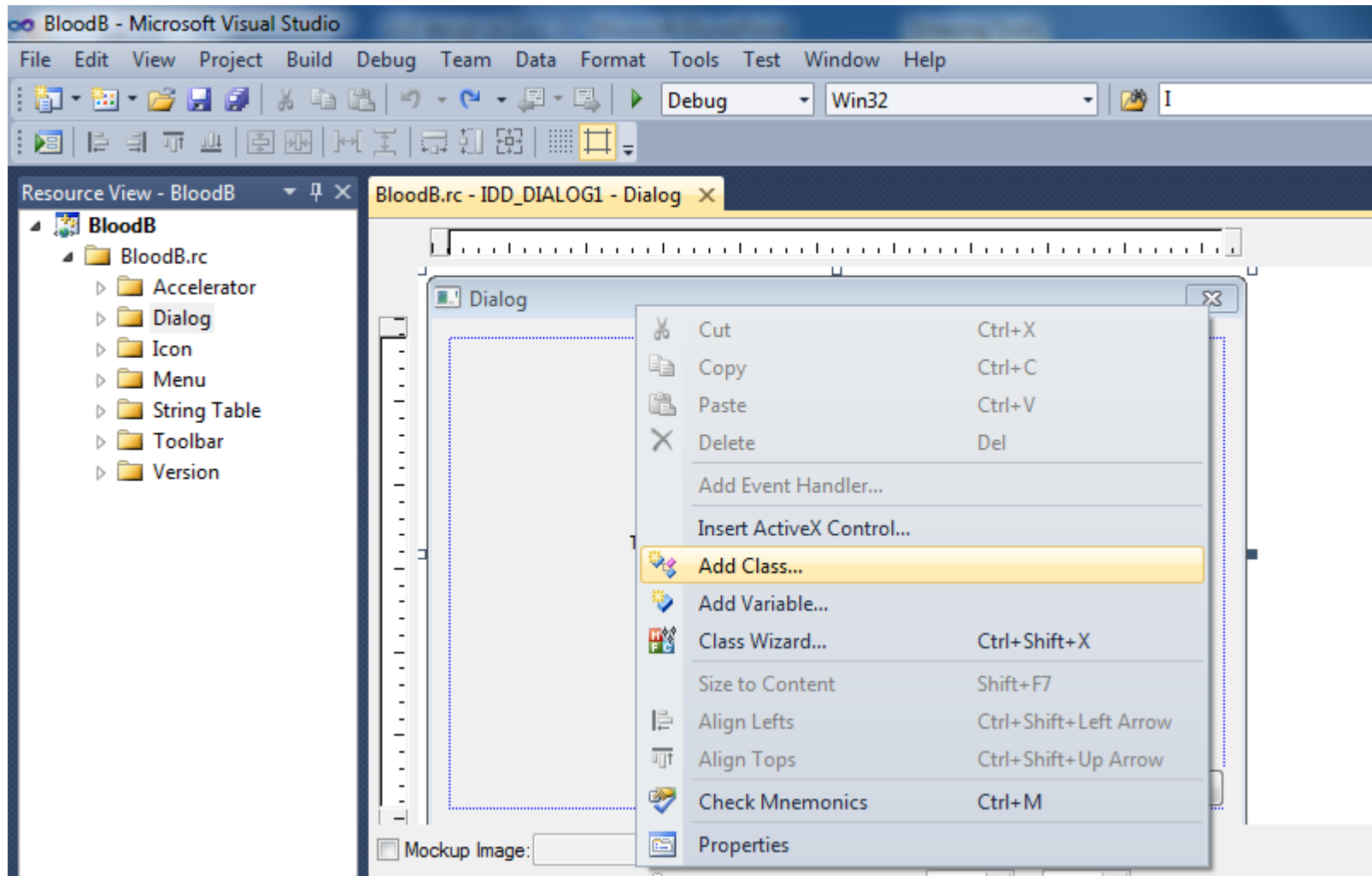
- ▶ In the following slides we will discuss about some basic UI controls.
- ▶ We will understand how to link our program logic with these controls.
- ▶ Finally, some end notes. 😊

Dialog Properties

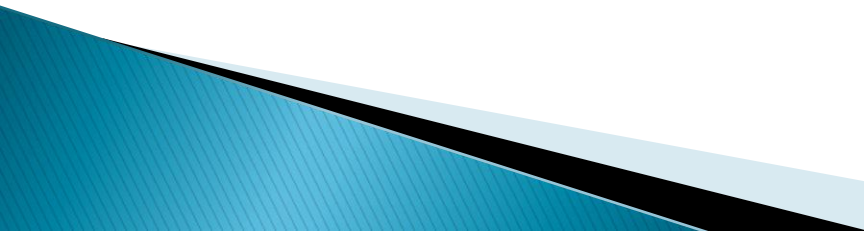
- ▶ Each dialog has a set of properties associated with it.
- ▶ To access these properties, Right-Click the dialog from the editor and choose “Properties”.
- ▶ You will see a huge list of options. For now the options which are important for us are:
 - Caption: It lets us change the title of our dialog
 - ID: It lets us choose an ID for this dialog. We don't have to bother about it for now 😊

Using Dialogs

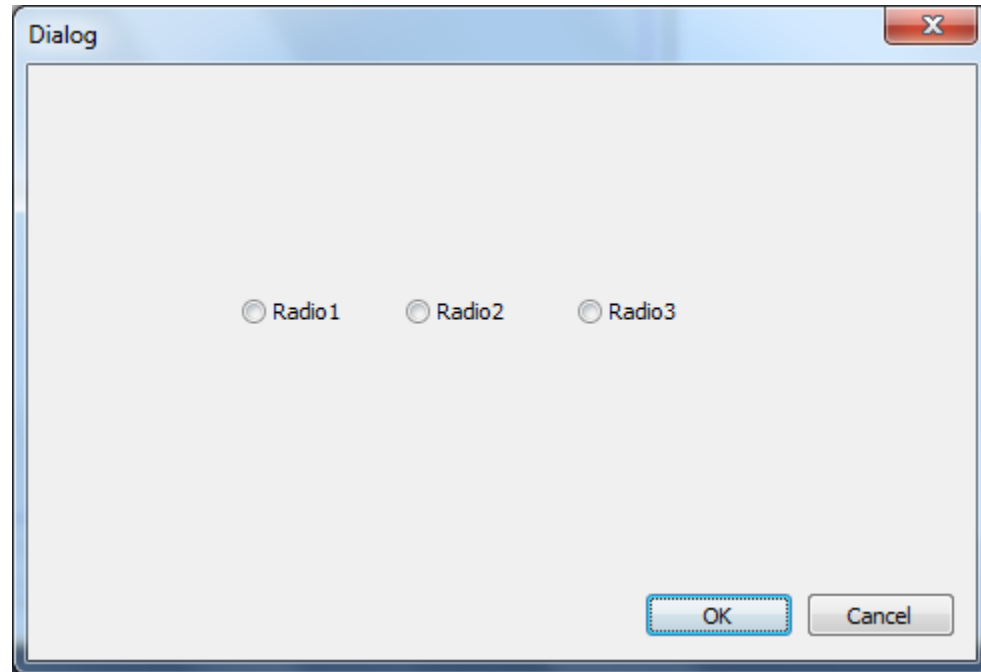
- ▶ Whenever we create a dialog in MFC, we have to associate a class with it.
- ▶ This class helps us to carry out tasks like Invoking the dialog, handle its events and lots more.
- ▶ The steps for doing this:
 - Right-Click on the dialog and choose Add Class.
 - From the “Add Class Wizard”, choose a class name e.g. “TeamChoiceDg”. And click Finish



Using Dialogs Cont...

- ▶ After adding the class, we will see the code for the class in it's own header file.
 - ▶ For now we don't have to worry about this and get back to our dialog editor.
 - ▶ Now from the toolbox, we have to choose some UI controls that we need for our application.
 - ▶ Drag some radio button controls. We need 3 of them for choosing Humans, Elves and Orgs.
- 

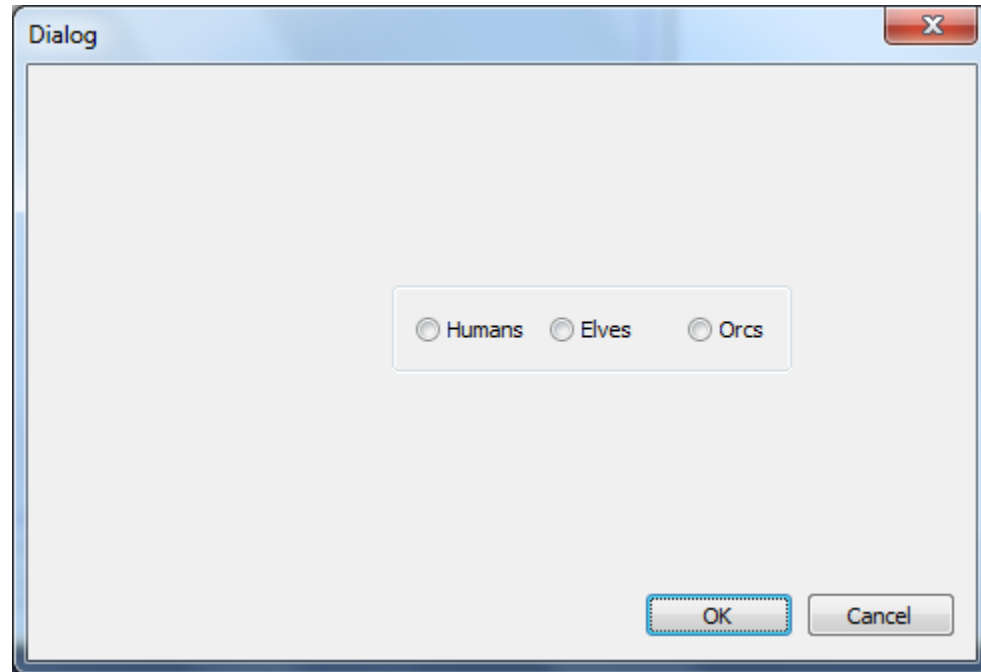
Radio Buttons



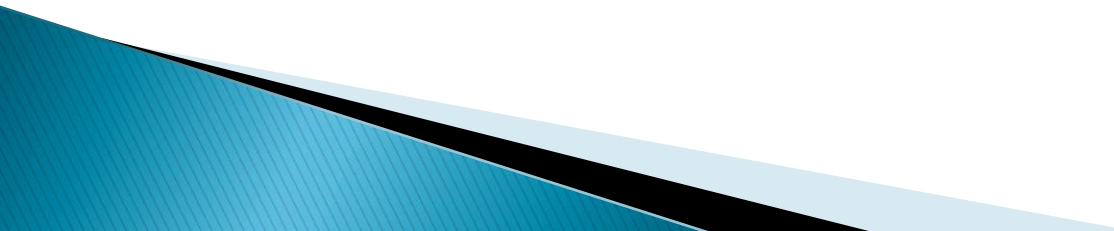
Radio Buttons

- ▶ We can change the text from “Radio1” to “Humans” from the **captions** column in Properties box.
- ▶ To use radio buttons, we have to take care that just one of them is selected. For this we need to group them using a group box.
- ▶ Select the Group Box from the toolbar and then draw it just like you draw a regular rectangle and cover all three radio buttons.

Radio Buttons



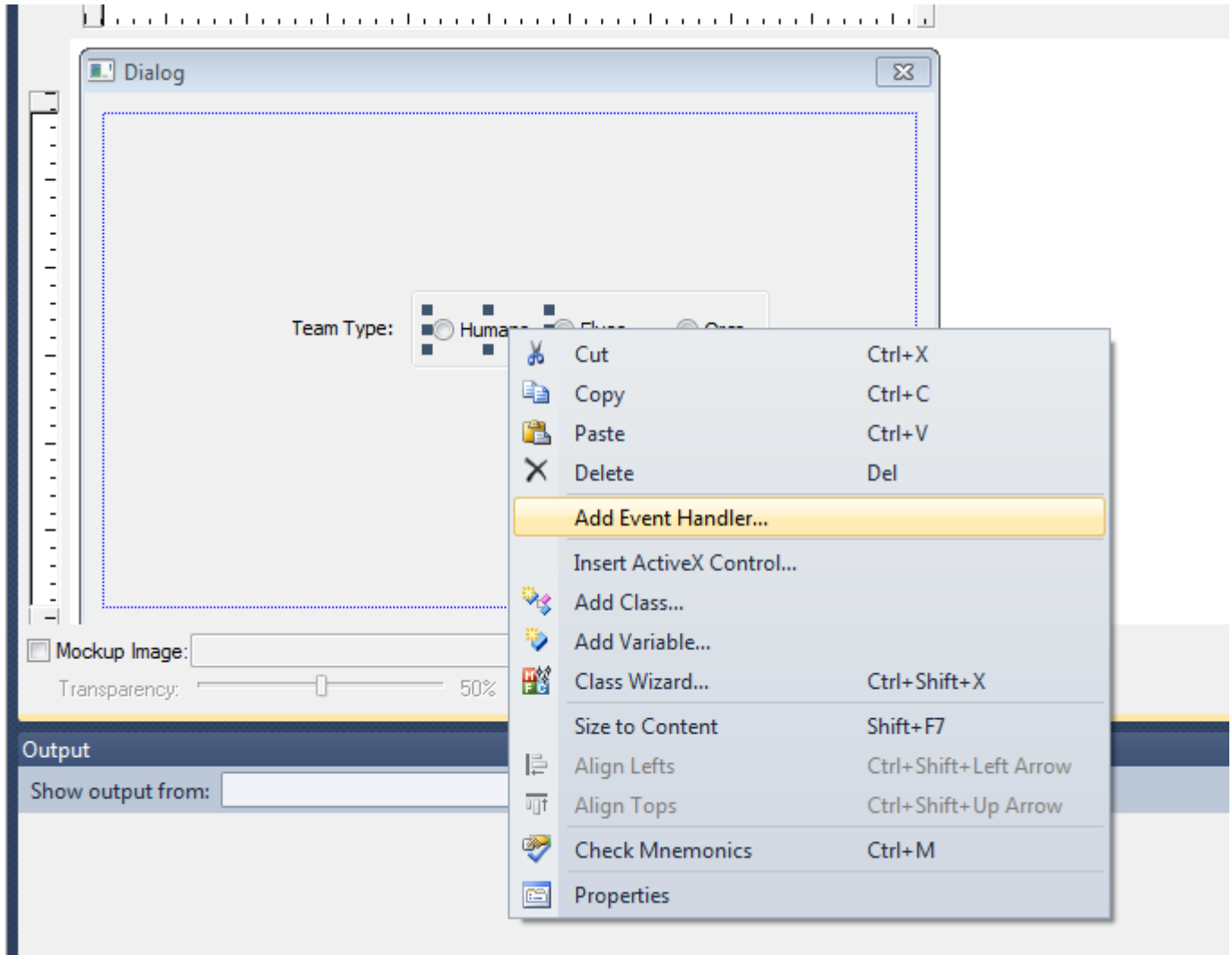
Handling Events

- ▶ To use any control, we need to create an even handler.
 - ▶ This handler informs the system about the event that has just occurred, so that the system can respond to this even by doing something.
- 


Handling Events

- ▶ To use Radio buttons to do something, we need to create an event handler for them.
- ▶ For this:
 - Right-Click on the radio button
 - Choose Add Event Handler
 - Choose a name for our event in the **Function Handle Name** option.
 - E.g. if you have clicked on the “human” button, you can choose a name like “OnSelectHumans”.
 - Click **Add and Edit**

```
B.rc*
:accelerator
alog
| IDD_ABOUTBOX
| IDD_DIALOG1
| IDD_DIALOG2
on
enu
ring Table
olbar
rsion
```



Event Handler Wizard - BloodB



Welcome to the Event Handler Wizard

Command name:

Message type:

BCN_DROPDOWN
BCN_HOTITEMCHANGE

Class list:
hLineman
hOgre
hThrower
humans
pitch
player

Function handler name:

Handler description:

Some coding.

- ▶ After clicking on **Add and Edit**, there will be the function `OnSelectHumans()`, where we can place our code for the case, when the user selects “Human”.
- ▶ Now, think what can happen when a user clicks on “Human”, from the programming perspective.
- ▶ For example, you have a variable that changes its value once the user selects the *race*.

Some more coding

```
void ClassName::OnSelectHuman()  
{  
    teamType=1;  
}
```


Click OK 😊

- ▶ We have to repeat the same process for the other two radio buttons and set the value for our variable (e.g. `teamType`) accordingly.
- ▶ To create an event for the OK button, just double click it and then you can write your code in the function body.

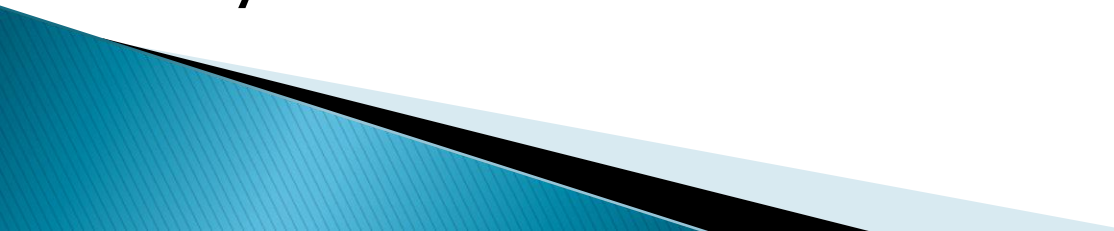
Invoking Dialogs

- ▶ Now to use this dialog in our application, we have to include the header file for the dialog class i.e. TeamChoiceDg.h in our main application file.
- ▶ If we are using the drawing class, we can include it in the “ClassNameView.cpp” file.

Invoking Dialogs

- ▶ Once we have included the header files for our dialog in our application, we can use it to invoke the dialog.
- ▶ Firstly we create a new object for the dialog class. E.g.
 - `TeamChoiceDg* tcDg=new TeamChoiceDg();`
- ▶ Then we can invoke this dialog by:
 - `tcDg->DoModal()`
- ▶ The `DoModal()` function invokes the dialogs.

Going further

- ▶ Well, here we have seen how we can use the Radio Buttons in a dialog based application and how to invoke the dialog we have designed.
 - ▶ The concept is the same for using other controls like ListBox, CheckBox, etc.
 - ▶ The only thing that differs is their use and syntax.
- 

End Notes

- ▶ To know more about MFC, go through the MSDN tutorials at:
<http://msdn.microsoft.com/en-ca/visualc/bb496952.aspx>
- ▶ Try to use the right programming approach. If this is right, you will find it helpful in designing your User Interface.
- ▶ You can always ask your questions by mail at: rishinder2007@yahoo.com