Design Guide: Classroom Program Requirements and Naming Standards(VB)

Note:

Programs submitted that fail, as submitted, to compile are an Automatic Grade of Zero (0). Test your programs before you submit them.

Note:

Option Strict On, Option Infer Off, Option Explicit On are required to be present in all/each module, class and form (code) for All submitted programs. They should also be set in your Project/Compile settings (All). Failure to do so could result in a grade of Zero(0). No program will be graded without these settings.

Note:

Programs submitted with only form design elements, but no appreciable code to meet the lab requirements, are an Automatic grade of Zero (0) without further review. No partial credit will be given.

Visual Basic Lab/ Project Specifications

A. The following must be done in all Visual Basic Projects/Labs as it applies to each Project/Lab.

1. Use the **Hungarian** notation with **Pascal** case (aka: upper CamelCase) to name all internal variable names, controls, etc. and make the names of them representative of what they do or contain. **EG: intLoopCounter, decMoneyPaid, strFirstName, btnClickMe, lblLastNamePrompt, getLastName(), setFirstName(strFirstName)**
2. All concepts covered in each chapter and/or lecture must be utilized for the current lab as appropriate, plus all subsequent labs. Failure to use the required class topic for a particular module, even if the program works, will result in an Automatic Grade of Zero(0). IE: The topic for the module is **Arrays**, but your lab submission for that module does not use Arrays… your grade is automatically a Zero even if your program works.
3. The first or primary form (Object/Class) and form file (frmMain.vb) in a project must always be named "**frmMain"** (this applies to the **file name**, **class name** and **form name**).
4. **All** controls (IE: Labels, Textboxes, etc.) on a form must be prefixed and named to classroom standards. (See course **Prefixes** Guide)
5. When using **TRY/CATCH** validate each field individually, do not excessively group code together in the same Try/Catch. **Try/Catches** should be used in conjunction with **IF/ELSE** statements (once **IF/ELSE** has been covered as a topic).
6. When converting numeric data types, **.TryParse** will be used. Additionally, **.TryParse** should be used in conjunction with **IF/ELSE** statements, once **IF/ELSE** has been covered as a topic. [IE: if **Integer.TryParse**(tbxNumber.text, intMyNumber) then…].
7. Tabs Stops must be set from left to right across the screen and then top to bottom of the screen unless otherwise dictated by the lab/project.
8. All Forms **must** be centered on the screen unless specified by the lab not to be centered.
9. Maximize Box (button) will not be displayed unless required by the lab.
10. Forms **will NOT** have a sizable border unless required (or appropriate) for the lab.

11. **Ensure there are no UNHANDLED EXCEPTIONS** when doing error proofing, validating or calculating. (To include form load/startup). Unhandled exceptions have a substantial negative effect on your grade and could result in a grade of Zero.
12. All controls placed on the form must be named using the correct classroom (separate attachment) naming (prefix and Pascal Naming) convention.
13. Comments must be placed inside each EVENT, FUNCTION, or PROCEDURE briefly describing what it is doing.
14. **DO NOT COMMENT EACH LINE**. You may comment lines as you need to help you design the program, but do not comment each line in what you submit for grading.
15. There will be **no empty functions/subs** (functions/subs with no code).
16. All programs, unless otherwise specified by the instructor or the Lab/Project, must include an "**Exit**" button with the code **"Application.Exit()"**.
17. All **MessageBox**'s must be fully formed. (All four elements: Text, Title, Button, Icon). Example:
    MessageBox.Show("Text to Display", "Title to Display", MessageBoxButtons.OK, MessageBoxIcon.Information)
18. All forms and output must be professional in appearance, function and flow.
19. No Excessive "empty space" on forms or output.
20. All forms must have an appropriate text/title for the Lab.
21. All forms must a have ControlBox unless otherwise specified by the lab or instructor.
22. All forms must a have Minimize  Button unless otherwise specified by the lab or instructor.
23. The following remarks are required in the frmMain of every project you submit. Failure to include them will adversely affect your grade. The areas containing {} should be replaced with the appropriate required entries. (IE: 'Programmer: James Flippin):

    '
    '
    'Programmer: {Your name goes here}
    'Course: ITSE 1332.xxxx (VB)
    'Program purpose: {Program Purpose Goes here}
    '
    '
24. If a screen layout (Screen Snapshot) is given/present as part of the Lab Instructions/requirements, then your screen **MUST** match as closely as is reasonable the provided screen layout.

25. The following Mandatory Compiler Directives (MDC) must be present in **EVERY** form, code module, class, etc. in the appropriate location. Failure to include them will adversely affect your grade. If found to be missing from your submission, I will take off points from your grade and then add the MDC before testing your code. If your code then fails to compile as a result of adding the MDC, your grade will automatically be a grade of Zero.

    Mandatory Compiler Directives (MDC):

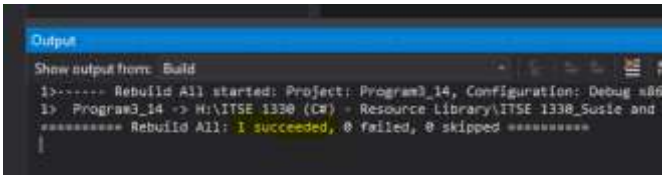
    #Region "Compiler_Directives"
    '
    Option Explicit On
    Option Strict On
    Option Infer Off
    #Disable Warning IDE1006
    '
    #End Region

B. <u>Control Name Prefixes (</u><u>Use the Hungarian notation with Pascal aka upper CamelCase</u>**<u>):</u>**
(See Attached/Separate Handout)

C. <u>Building, Debugging and running your programs:</u>

To test your code, First select "**Build**" from the Menu, then select "**Rebuild Solution**".

Check the **Output window** for errors, if errors were found. Stop and correct them. Then rebuild again. Repeat this process until there are no more errors.



If a line of code is in error, you can double click on the error message in the output window to be taken directly to that line of code.

If there are no errors, you can run the program by either selecting "**Start Debugging**" or "**Start Without Debugging**" depending upon what you are trying to do. Until you have fully "debugged" your program, you should be using "**Start Debugging**". Please note the Keyboard Shortcut (F5 – Start Debugging, Ctrl+F5 Without debugging) keys that are available for each of these.


File Edit View Project Build Incredibuild Debug Team Tools Architecture Test R Tools
Windows
Graphics
Start Debugging F5
Start Without Debugging Ctrl+F5
Attach Unity Debugger
Performance Profiler... Alt+F2
Toolbox
General
There are no usable controls in this