COMP 2690 - .NET Windows App Development Part 1

Coding Standards

Please understand that there is no "correct" coding standard. Your next employer, for instance, may insist on its own coding style that is different and unique. For this course, however, you must follow the rules outlined below, most which match the conventions that Microsoft and the .NET community already use.

Note: This document may change during the course, probably for additions. I will email you any changes that occur.

Spacing

Use blank space appropriately so as to maximize the readability of your code.

Blocks

A *block* is any code surrounded by { } curly braces.

Always use blocks for `if`, `while`, `do`, and `for` constructs, *even when* they only consist of one line. Here are some examples:

Incorrect	Correct

`for (index=0; index<MAX; ++index)`
`    printf("%d\t", index);`
	`for (index=0; index<MAX; index++)`
	`{`
	`    printf("%d\t", index);`
	`}`

`if (index < 0)`
`    printf("Invalid index");`
	`if (index < 0)`
	`{`
	`    printf("Invalid index");`
	`}`
	
	OR
	
	`if (index < 0) {`
	`    printf("Invalid index");`
	`}`

Tabbing

Tabs shall always be exactly 4 characters wide. (This is the default for the Visual Studio.)

Tabs shall be used to indent the entire contents of any block. A block is <u>*any*</u> code or comments surrounded by the { } curly braces.

Here are some examples:

Incorrect	**Correct**

int main()
{
printf("Hello world!\n");
 	
etc.
}
	int main()
	{
	$\xrightarrow{\text{TAB}}$ printf("Hello world!\n");
	
	    etc.
	}

int main()
{
if(*expression*)
{
printf("One");
}
else
{
printf("Zero");
}
 	
etc.
}
	int main()
	{
	    if(*expression*)
	    {
	    $\xrightarrow{\text{TAB}}$ printf("One");
	    }
	$\xrightarrow{\text{TAB}}$ else
	    {
	    $\xrightarrow{\text{TAB}}$ printf("Zero");
	    }
	
	    etc.
	}

Naming

Variable names shall be descriptive—names such as `i` and `x` are not permitted in this course, with the exception of the following case: you may use "trivial" variable names when the code that uses them is trivial.

Variable names shall have "camel case" capitalization, which means that the first letter of each word within the variable name is capitalized, except the first one. Some examples are: `shoeSize`, `previousRecordCount`, `firstName`, etc.

All collection-type variables shall be named in the *plural* sense. Examples are: `highScores`, `temperatureReadings`, `scanCodes`, etc.

Class Names

SentenceCapitalization

A class representing a *single* item when instantiated shall be named in the singular. For example, `GamePiece`, `BankAccount`, `SystemUser`.

Public Members
SentenceCapitalization

This includes methods and properties, unlike Java.

Protected or Private Members
camelCaseCapitalization

This includes methods, properties, instance variables.

Local variables
camelCaseCapitalization

Constant Names

Constant names shall be all in uppercase with underscores to separate each word of the name. Examples are: `MAX_NAME_LENGTH`, `PROGRAM_VERSION`, `DEFAULT_PROMPT`.