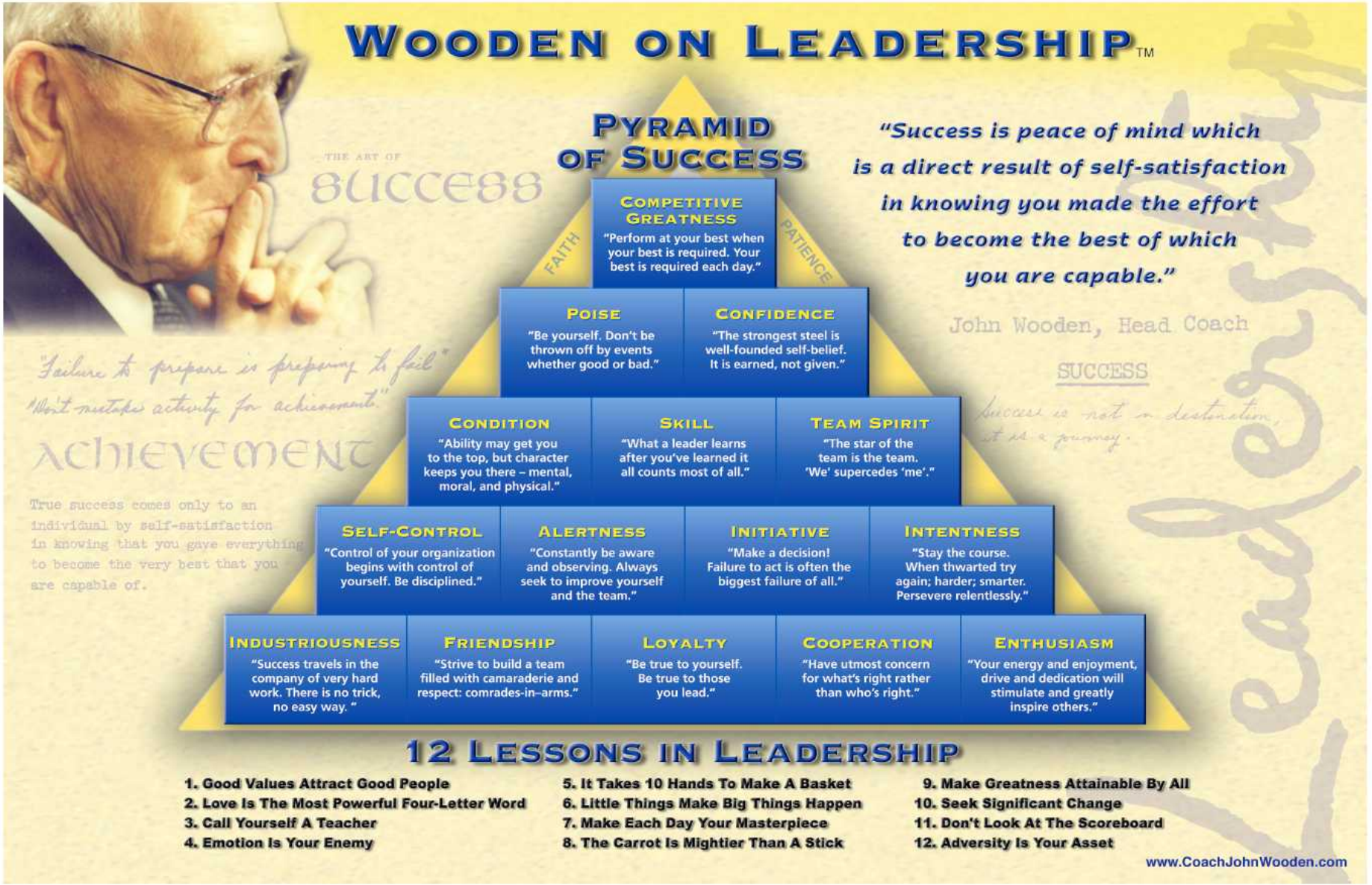
Chapman University
CPSC 340 - Spring 2011

Class #1
August 30, 2011
William Wood Harter
Adjunct Professor

Introduction

- About Me
- The Syllabus (gamedev360.com)
  - I use gamedev360 because I can no longer update my ~harter website at Chapman using the VPN
- Weekly game chalk talk. How does that game do that? What's going on under the hood. Any game any concept. Image formats, Zaxxon, etc. Come with a question and let's talk about it with the whole class.
- Let's talk about pong.

You should define success



Skills

- We will focus the first few weeks on core game programming skills.
- They may not seem like game programming skills

Availability

- Feel free to use me as a resource

Where you might be right now

- Based on what I know about the rest of the slides, if you are not a programmer and haven't taken CPSC 231. You took the red pill.

Where you might be right now

- And might be in for some stress. Ready?

C#

- Just another programming language
- C became C++ became Java (cloned as C#)
- Main difference between C++ and C# is that C++ has much closer view of memory. C# hides some of that complexity.
- All these languages are strongly typed.
- The hardest part about learning a new language is typically learning the tools and learning to navigate the libraries.

Data types

- Char
  - char – a Unicode character
- Numbers
  - int - Whole numbers
  - float/double – Real numbers
- Boolean
  - bool
- String
  - string – A string (array) of Unicode characters

What is Unicode?

- A mapping of number to characters that started as ASCII numbers. (http://jimprice.com/jim-asc.shtml)
- Unicode uses 2 bytes (simplified answer) instead of 1 to allow for Non-European languages.

ASCII value	Character	Control character	ASCII value	Character	ASCII value	Character	ASCII value	Character
000	(null)	NUL	032	(space)	064	@	096	
001	☺	SOH	033	!	065	A	097	a
002	☻	STX	034	"	066	B	098	b
003	♥	ETX	035	#	067	C	099	c
004	♦	EOT	036	$	068	D	100	d
005	♣	ENQ	037	%	069	E	101	e
006	♠	ACK	038	&	070	F	102	f
007	(beep)	BEL	039	'	071	G	103	g
008	◘	BS	040	(	072	H	104	h
009	(tab)	HT	041	)	073	I	105	i
010	(line feed)	LF	042	*	074	J	106	j
011	(home)	VT	043	+	075	K	107	k
012	(form feed)	FF	044	'	076	L	108	l
013	(carriage return)	CR	045	-	077	M	109	m
014	♫	SO	046	.	078	N	110	n
015	☼	SI	047	/	079	O	111	o
016	►	DLE	048	0	080	P	112	p
017	◄	DC1	049	1	081	Q	113	q
018	↕	DC2	050	2	082	R	114	r
019	!!	DC3	051	3	083	S	115	s
020	¶	DC4	052	4	084	T	116	t
021	§	NAK	053	5	085	U	117	u
022	▬	SYN	054	6	086	V	118	v
023	↨	ETB	055	7	087	W	119	w
024	↑	CAN	056	8	088	X	120	x
025	↓	EM	057	9	089	Y	121	y
026	→	SUB	058	:	090	Z	122	z
027	←	ESC	059	;	091	[	123	{
028	(cursor right)	FS	060	<	092	\	124	¦
029	(cursor left)	GS	061	=	093	]	125	}
030	(cursor up)	RS	062	>	094	^	126	~
031	(cursor down)	US	063	?	095	_	127	⌂

What are variables?

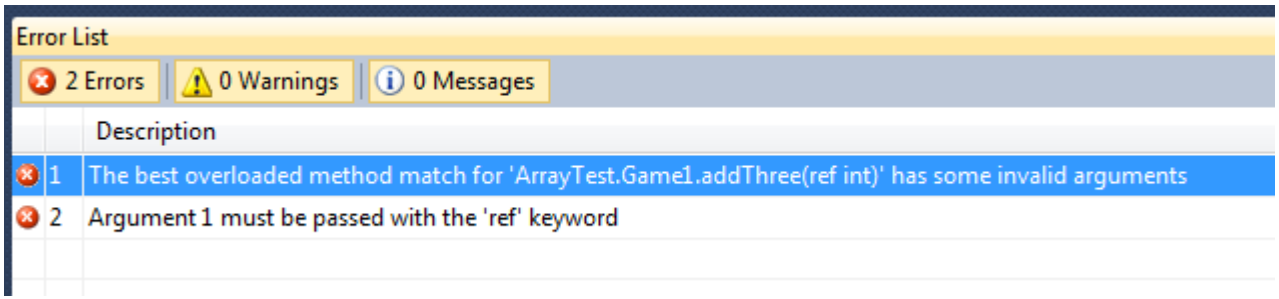
- Variables are just locations in memory.
- int x;
  - X is a location in memory where we are storing a number. We can change x and the value changes.
- String name = “Wood”;
  - The variable name is a string of four characters.
  - name = “Thomas”;
  - Now we changed the value held by name.

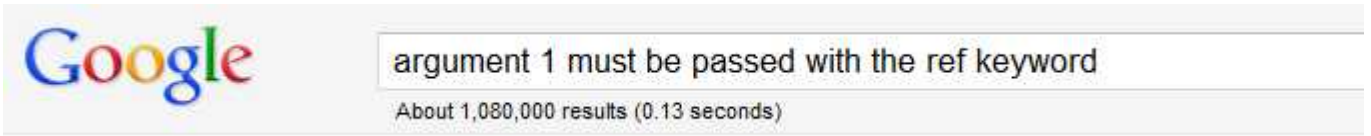
What is strongly typed?

- Many people have taken Python, a language that is not strongly typed like C#.
- X=4; will not work in C#
- You have to first declare the variable and give it a type.
- int X;
- X = 4;
- This allows the compiler to do safety checking before running the program.
- Neither is better, just different paradigms.

Some Important Skills

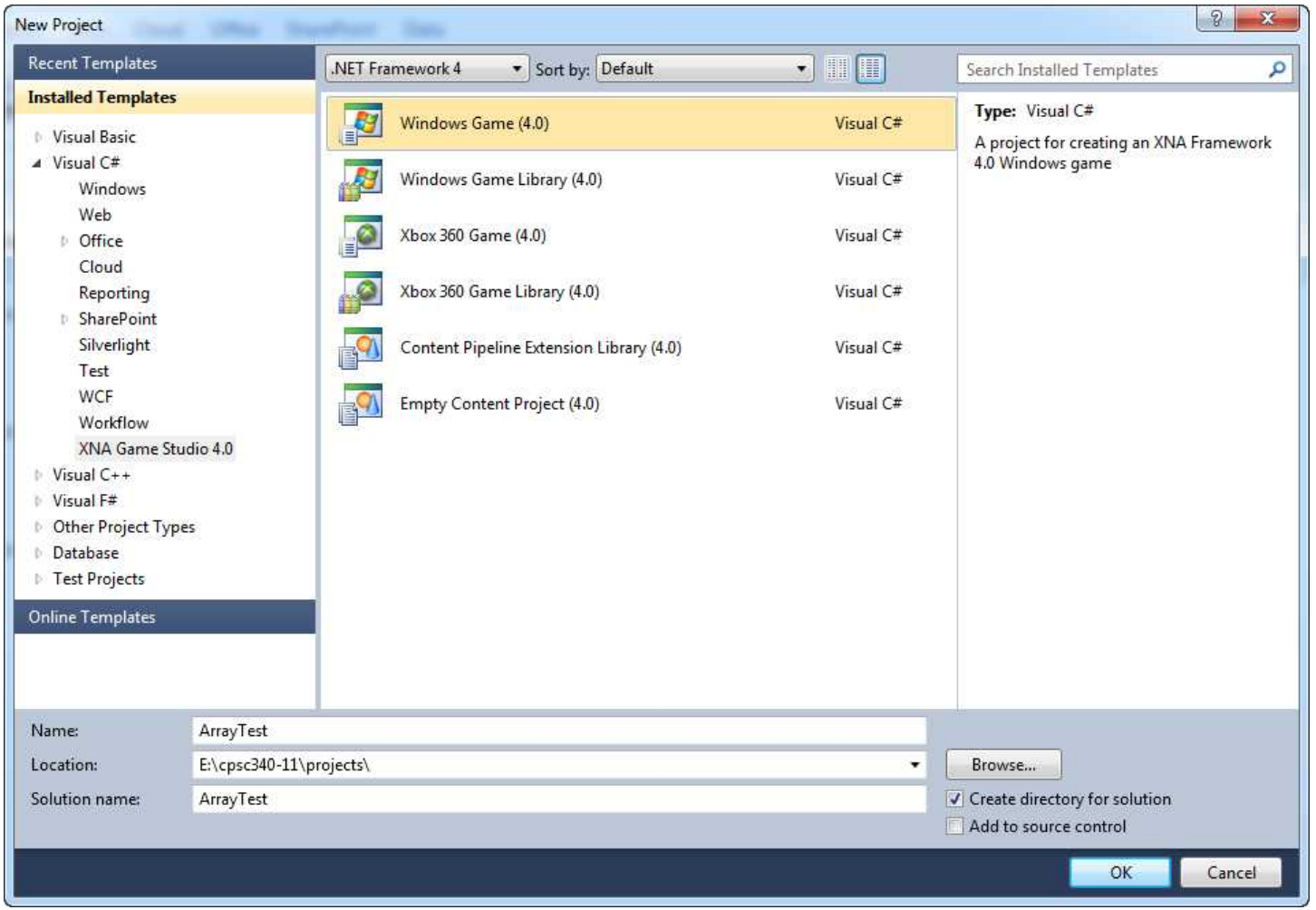
- Reading
- Math
- Communication
- Looking stuff up with google.





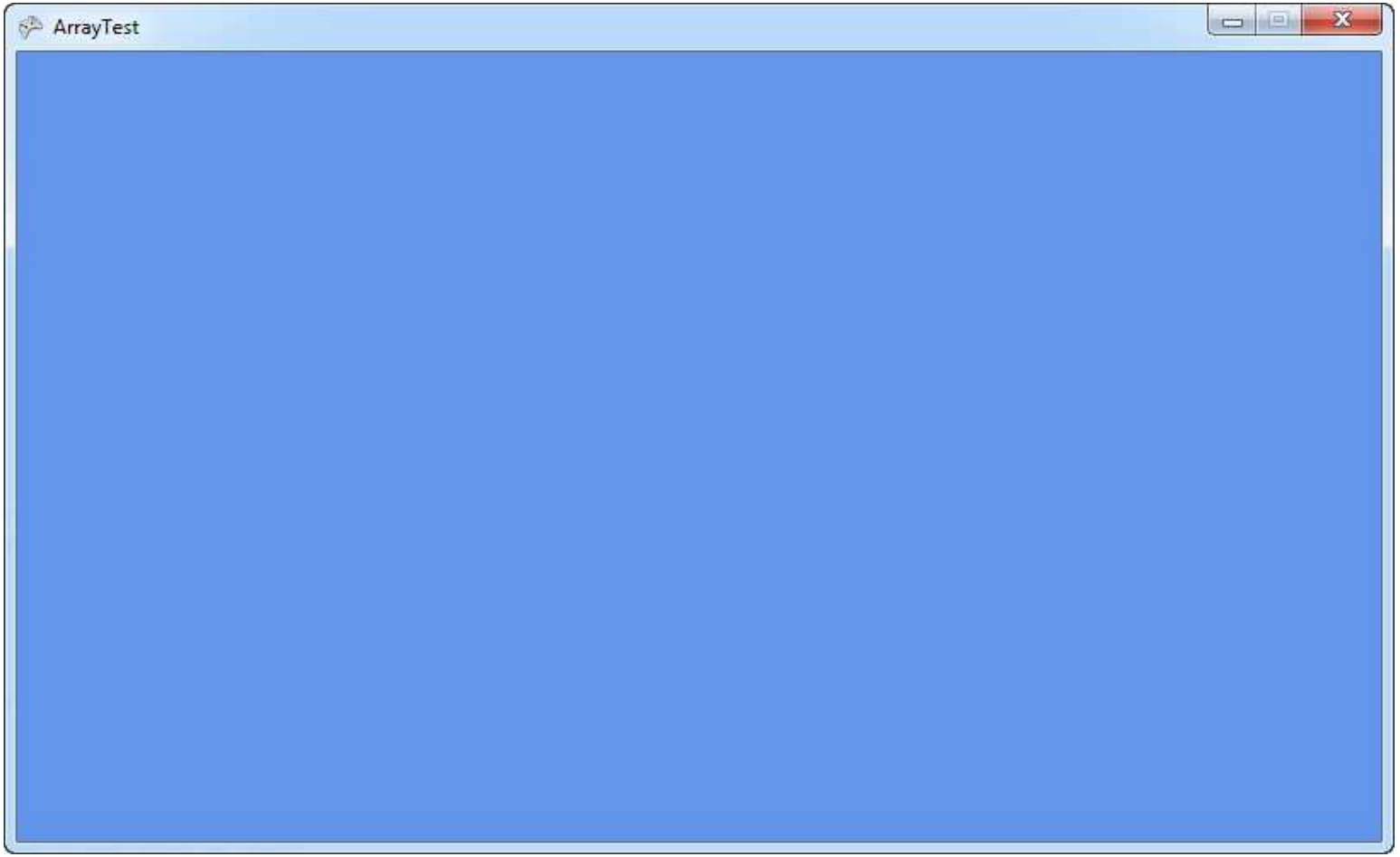
Let's Jump In

- Create an XNA project called ArrayTest



Run It

- Hey! You just built your first game!



Let's talk more about programming

- Arrays
- Functions
- XNA applications
- Output

Arrays

- The first extension of built-in data types
- Why?
- What would you use an array for?
- Declare an array with
  - int[] myarray;
- Declare and initialize an array
  - int[] myarray = {1, 2, 3, 4};
- You can find the length of an array
  - int x = myarray.Length

For loops

- For loops allow you to count in sequence.
- For loops take three variables.
  - The starting value of the counter
  - When to quit
  - What and how much to increment
- for (i=0; i<23; i=i+1)
  - Will loop 23 times and stop when i is 23

Functions

- Functions form the central means of creating small “tools” that can be reused over and over again.
- $f(x) = x^2$
- Are functions the same as methods?
  - Methods just combine functions with data.

Functions

- Functions can return values or not
  - void myfunction()
    - Does not return anything
  - int myfunction()
    - Returns an integer value

```
void myfunction()
{
        int i= 0;
        i = i + 1;

}

myfunction();
```

```
int myfunction()
{
        Int x= 0;
        x = x + 1;
        return x
}

int y = myfunction()
```

Functions

- Functions can be passed values $f(x) = x^2$

```
void myfunction(int passedin)
{
      passedin = passedin + 3;
}

int y = 4;
myfunction(y);

Passed by value (y not changed!)
```

```
int myfunction(int passedin)
{
      passedin = passedin + 3;
      return passedin;
}

int y = 4;
y = myfunction(y);
int x = myfunction(y);
```

Functions

- Values can be passed to functions by reference meaning that the function is referencing the original value.

```
void addThree(ref int myval)
{
    myval = myval + 3;
}

int myx = 4;
addThree(ref myx);
```

Methods

- Methods have access restrictions
  - public
  - private
  - protected
- Methods also have inheritance which is beyond this discussion
  - virtual
  - override

Anatomy of the XNA template

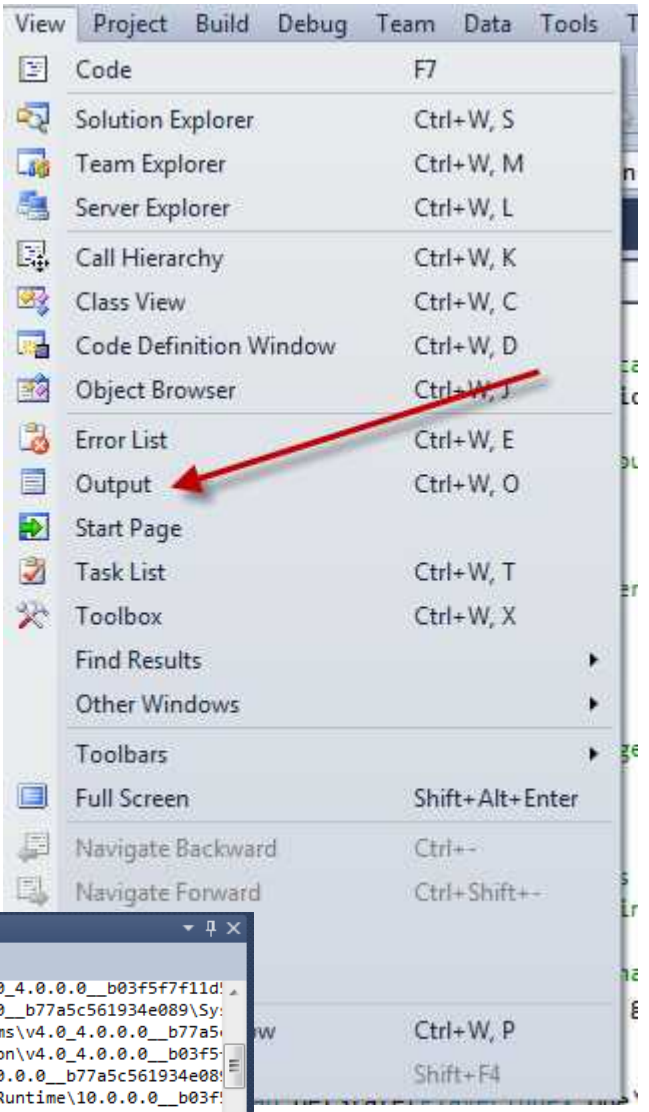
- Initialize
  - Called when the game starts
- Load Content
  - Called when the game starts
- Unload Content
  - Called at the end (you can ignore it for now)

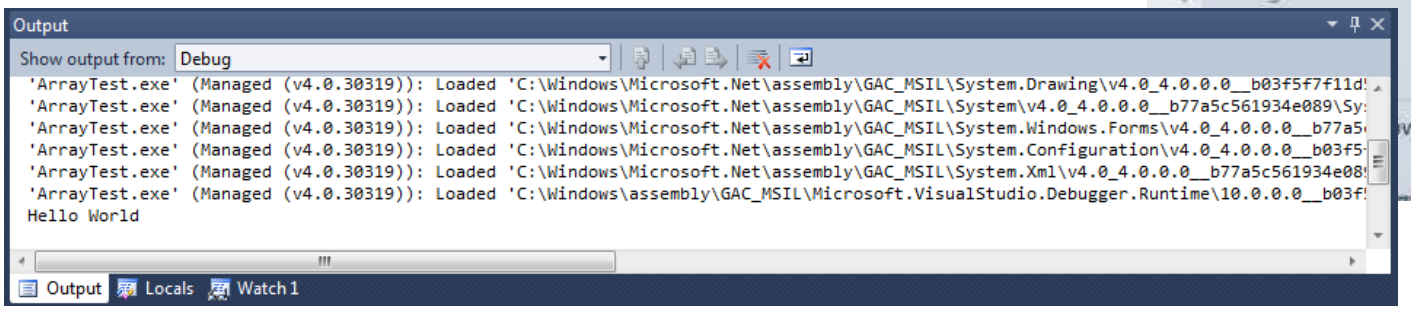
Anatomy of the XNA template

- Update
  - Called over and over to tell you time has passed
  - Use this to check input and move things on the screen.
- Draw
  - Draw things in/on the window/screen

Early CRT output still common

- Printing information to the console
- Counsole.WriteLine("hello world")
- May need to view output to make this work
- What is the difference between
  - Console.Write
  - Console.WriteLine





First exercise

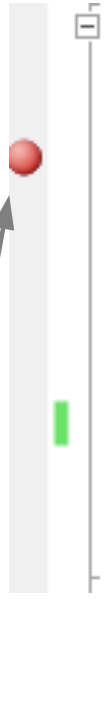
- Create an xna 4.0 project.
- Add this array code to your first project
- Call it from your Initialize method

```
protected void printArray(int[] ar)
{
    int i;
    for (i = 0; i < ar.Length; i++)
    {
        Console.Out.WriteLine(ar[i]);
    }
}
```

```
protected override void Initialize()
{
    // TODO: Add your initialization logic here
    Console.Out.WriteLine("Hello World");

    int[] myarray = { 1, 2, 3, 4, 5, 6 };
    printArray(myarray);

    base.Initialize();
}
```

Debugging

- Set a break point on the for loop line.

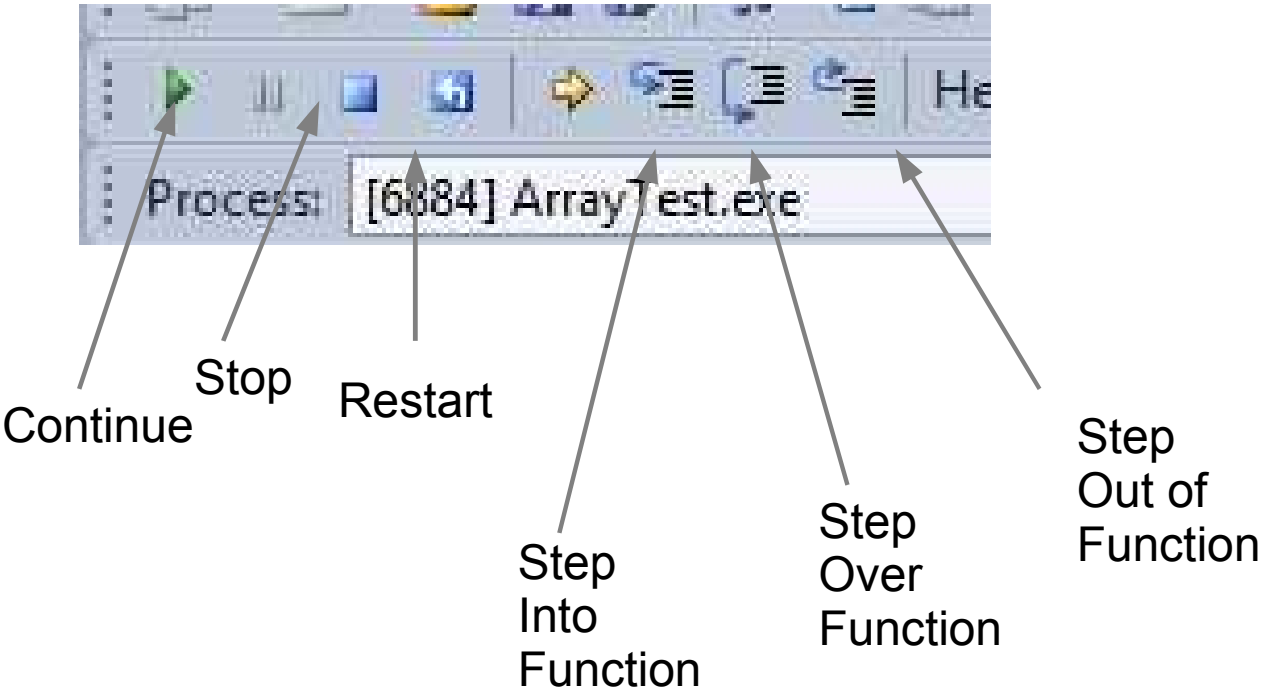


```
protected override void Initialize()
{
    // TODO: Add your initialization logic here
    Console.Out.WriteLine("Hello World");

    int[] myarray = { 1, 2, 3, 4, 5, 6 };
    printArray(myarray);

    int x = 4;
    addThree(ref x);

    base.Initialize();
}
```

Click here to set or unset a breakpoint

Debugging

- Run the program and step through the code.



Second Exercise

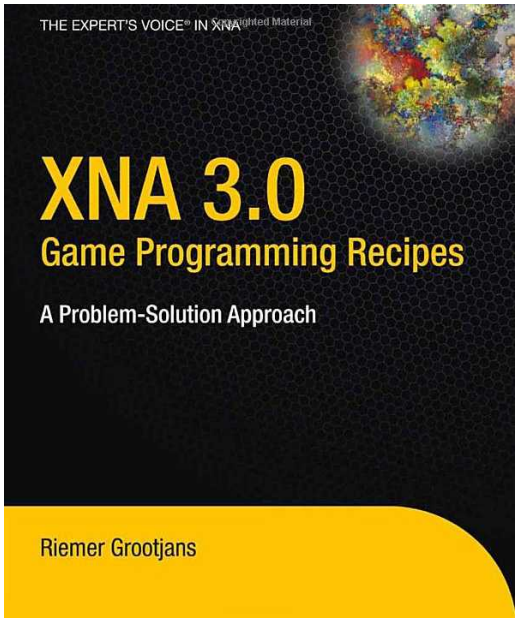
- We are going to create our own button
- We will use images to draw the button in the various states.
- We will capture mouse location and understand when the button was pressed.

Properties of buttons

- Location
- A set of images
- Different states
  - Normal
  - Up
  - Down
  - Down but not over – can just use normal

Your book

- Yes it's 3.0, but will work perfectly well for XNA 4.0
- A reference, not a read from cover to cover.
- Find a topic you are trying to solve and read about a solution



XNA 3.1 and XNA 4.0 differences

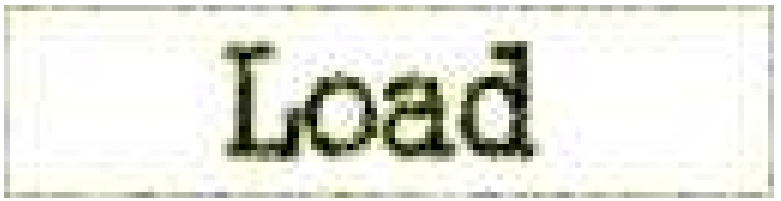
- The world is currently transitioning from xna 3.1 to XNA 4.0
- There are some big differences between the XNA versions.
- Use google to find solutions
- This is a great cheat sheet on the differences.
  - http://www.nelxon.com/blog/xna-3-1-to-xna-4-0-cheatsheet/
- Dealing with this sort of versioning is a common problem

Vector

- In XNA, a vector is 2 or three floating point numbers
- Vector2 is X,Y
- Vector3 is X,Y,Z
- Vector2 myvect = new Vector2(10.0f, 20.0f);
- It is simply a location on the screen.
- Vector3 is simply a location in space.

Drawing Images

- XNA is a 2D and 3D platform
- Loading an image is an important skill
- Book Chapter 3
  - Recipe 3-1 Display 2D Images
- What is a SpriteBatch?
- Create a button image with Photoshop
  - Save it as a png image



Batching

- SpriteBatch brings up a common programming term Batching.
- What is Batching?
- What are types of batches?
- Can you think of things in nature that are batched?

Iterative development

- Always take small iterative steps
  - Make small changes
  - Test the small changes
  - Move onto the next small change
- Why?

ButtonTest project

- Create a new project called ButtonTest
- Put your image in the content directory.
- Add the button image images to the content project.

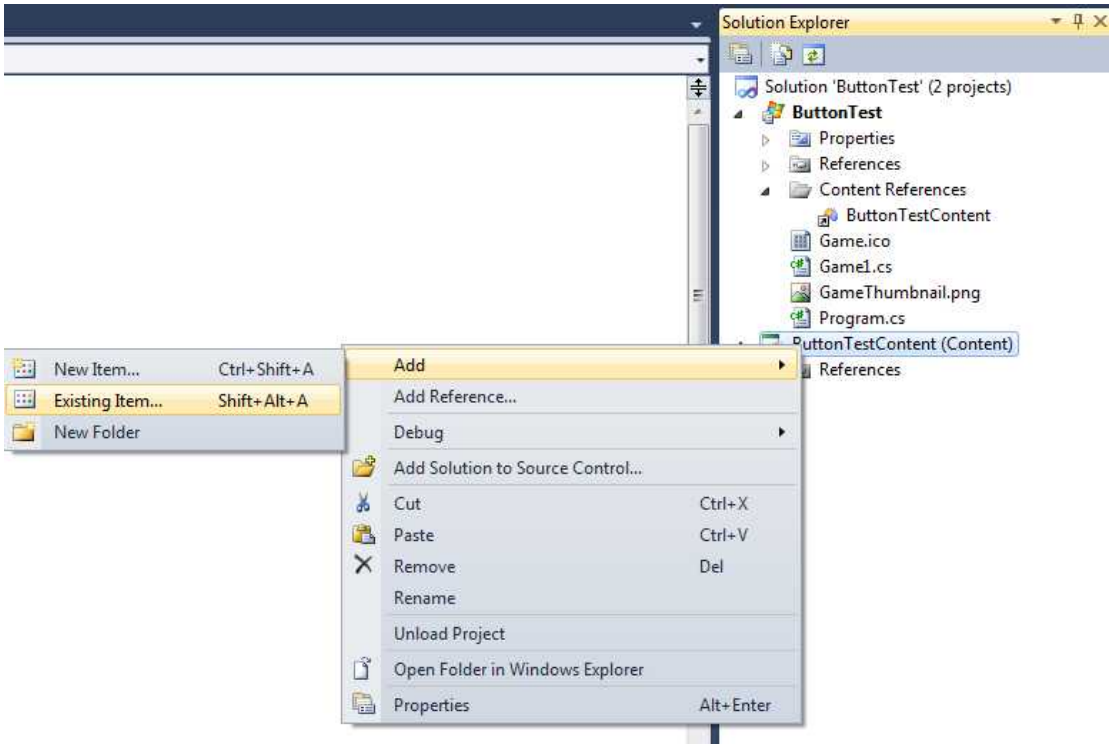
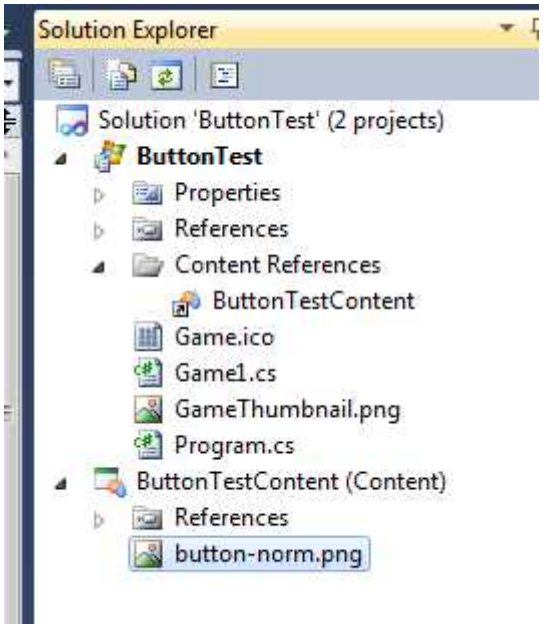


Image Content

- The image is now part of the content.
- Should probably be in sub-directories for real projects.



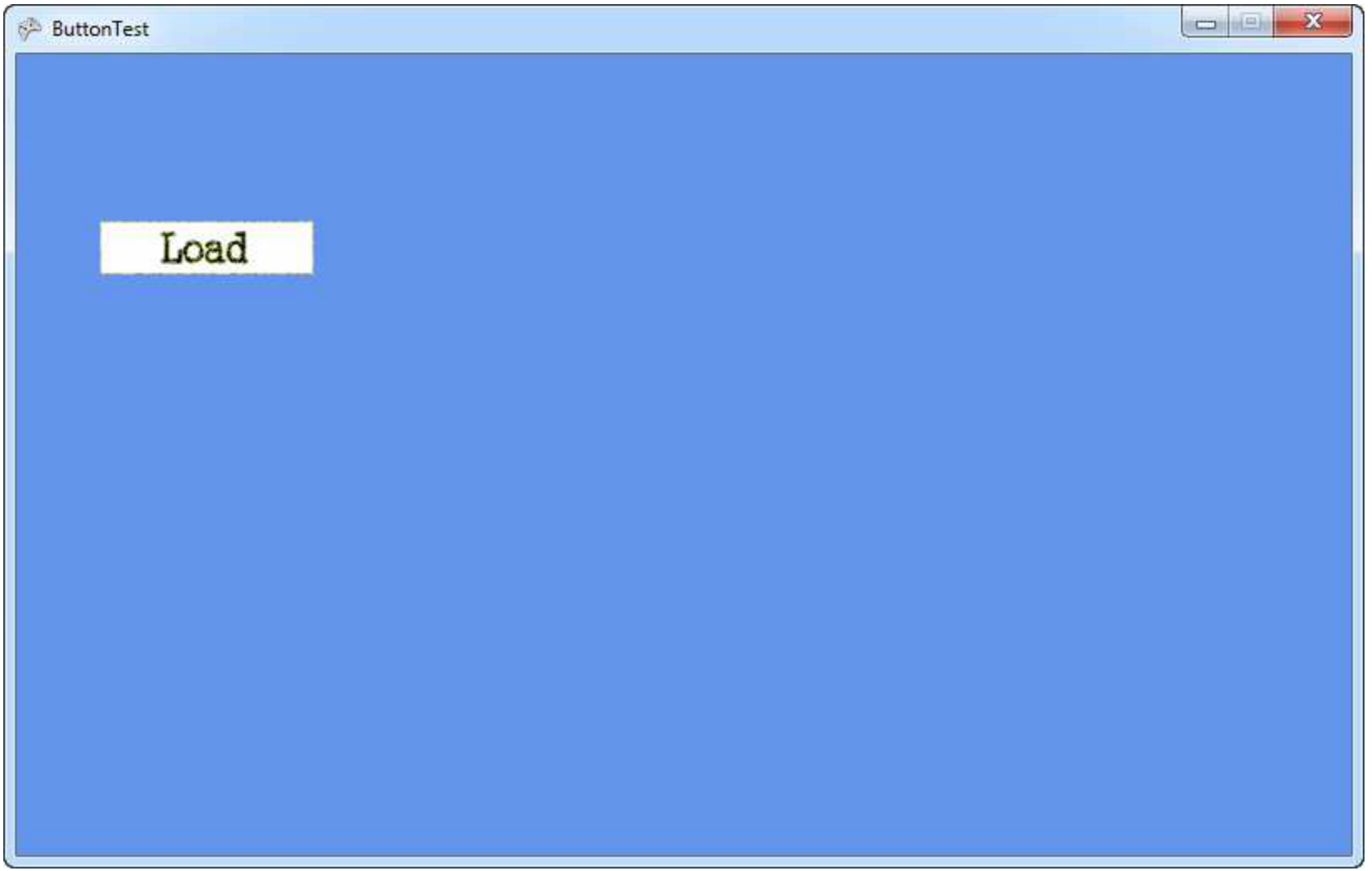
Texture Variable and LoadContent

- Declare the texture variable
- Make changes to the LoadContent method to load the image.

```
/// </summary>
public class Game1 : Microsoft.Xna.Framework.Game
{
    GraphicsDeviceManager graphics;
    SpriteBatch spriteBatch;

    Texture2D ButtonLoadNorm;
```

```
/// all of your content.
/// </summary>
protected override void LoadContent()
{
    // Create a new SpriteBatch, which can be used to draw textures.
    spriteBatch = new SpriteBatch(GraphicsDevice);

    ButtonLoadNorm = Content.Load<Texture2D>("button-norm");
}
```

Your second first game



Improvements

- What is wrong with the new Vector(50,100) in the context of the Draw method?

```
protected override void Draw(GameTime gameTime)
{
    GraphicsDevice.Clear(Color.CornflowerBlue);

    spriteBatch.Begin();

    // draw the button
    spriteBatch.Draw(ButtonLoadNorm, new Vector2(50, 100), Color.White);

    spriteBatch.End();

    base.Draw(gameTime);
}
```

Fix the overactive Vector

- It's time to move the location of the image to a variable.
- Add a variable for the location

```
Texture2D texButtonLoadNorm;
Vector2 vButtonLoadLoc = new Vector2(50, 100);
```

- Then use that variable when drawing the button.

```
spriteBatch.Draw(texButtonLoadNorm, vButtonLoadLoc, Color.White);
```

Create your own class

- Classes are easy to create. They simply encapsulate both data and functions. The functions associated with a class are called methods.

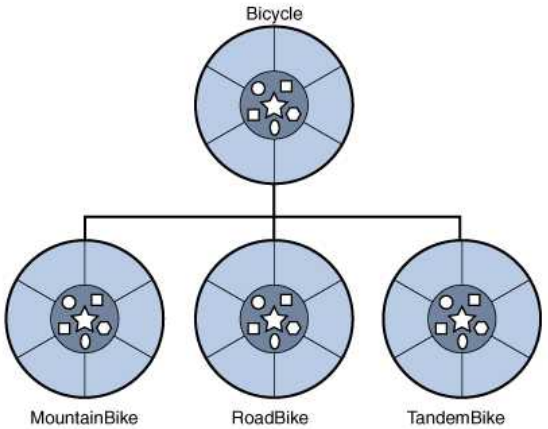
```
public class MyClass
{
        public int xyz;

        Void printXYZ()
        {
                Console.WriteLine(xyz);
        }
}
…

MyClass mc = new MyClass()
mc.printXYZ();
```

When to create new classes

- Create new classes for functionality that you will use over and over again.
  - A Car in a game
  - A Button
- Create new classes for functionality that you might use in your next game as well as the game you are working on.
  - An animation system
  - A game saving system
  - A button system

Class inheritance

- Class inheritance is something that can't be explained. You have to use it and figure it out. We'll continue to talk more about it as we continue, but I wanted to tell you it exists and it is very powerful and important.
  - http://download.oracle.com/javase/tutorial/java/concepts/inheritance.html
- This type of object would allow you to have an array of Bicycles not caring what sub-types actually exist. This is handy once you start to feel the pain of working with objects of different types. Remember that thing about “Strongly Typed”?



Variable scope

- All variables have scope
- Scope tells where a variable is active and available
- The variables you have been adding have had object scope. Meaning they are part of an object. One for each object of that type.
- Some variables have a method scope and disappear when the method/function returns, ie. passed parameters.

More variable scoping

- Variables that are scoped to an object can only be accessed when an instance of that object has been created.

```
class MyClass
{
	public int myvar = 4;
}

function xyz()
{
	MyClass mc = new MyClass();
	Console.WriteLine(mc.myvar);
	mc.myvar = 8;
}
```

More variable scoping

- Some variables can be scoped to the class. Meaning they are accessible even when no objects are created. They are like global variables.
- These are called static variables

```
public class MyClass
{
	public static int currentState = 0;
}
…

MyClass.currentState = 4;
```

Some Coding Conventions

- I like to use a Hungarian notation where the variable type is identified in the name.
  - Vector vDirection;
  - int iScore;
  - Texture2D texButtonImage;
- Lots of comments.
  - Leave comments not only for others, but for yourself, you wont remember what it did a month from now.
- Descriptive file and method headers.
  - User /** to create tags for auto documentation at the beginning of functions.

Some Coding Conventions

- Align those brackets.
  - Brackets on the end of lines are only done to save space in books. Always put your brackets on their own line and make sure they line up.
  - Current editors do a lot of this work for you. Lucky you.

Yes

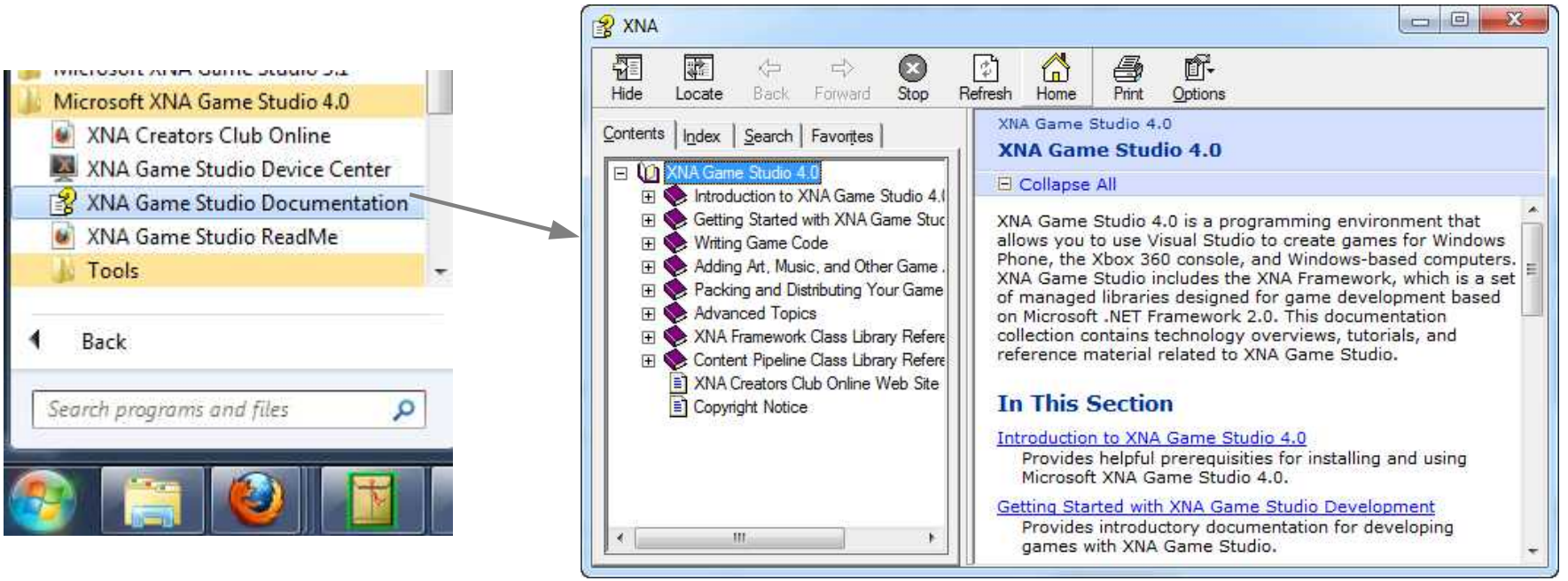
```
void MyFunction()
{
   Xyz = 40;
}
```

No

```
Void MyFunction() {
   Xyz = 40;
}
```

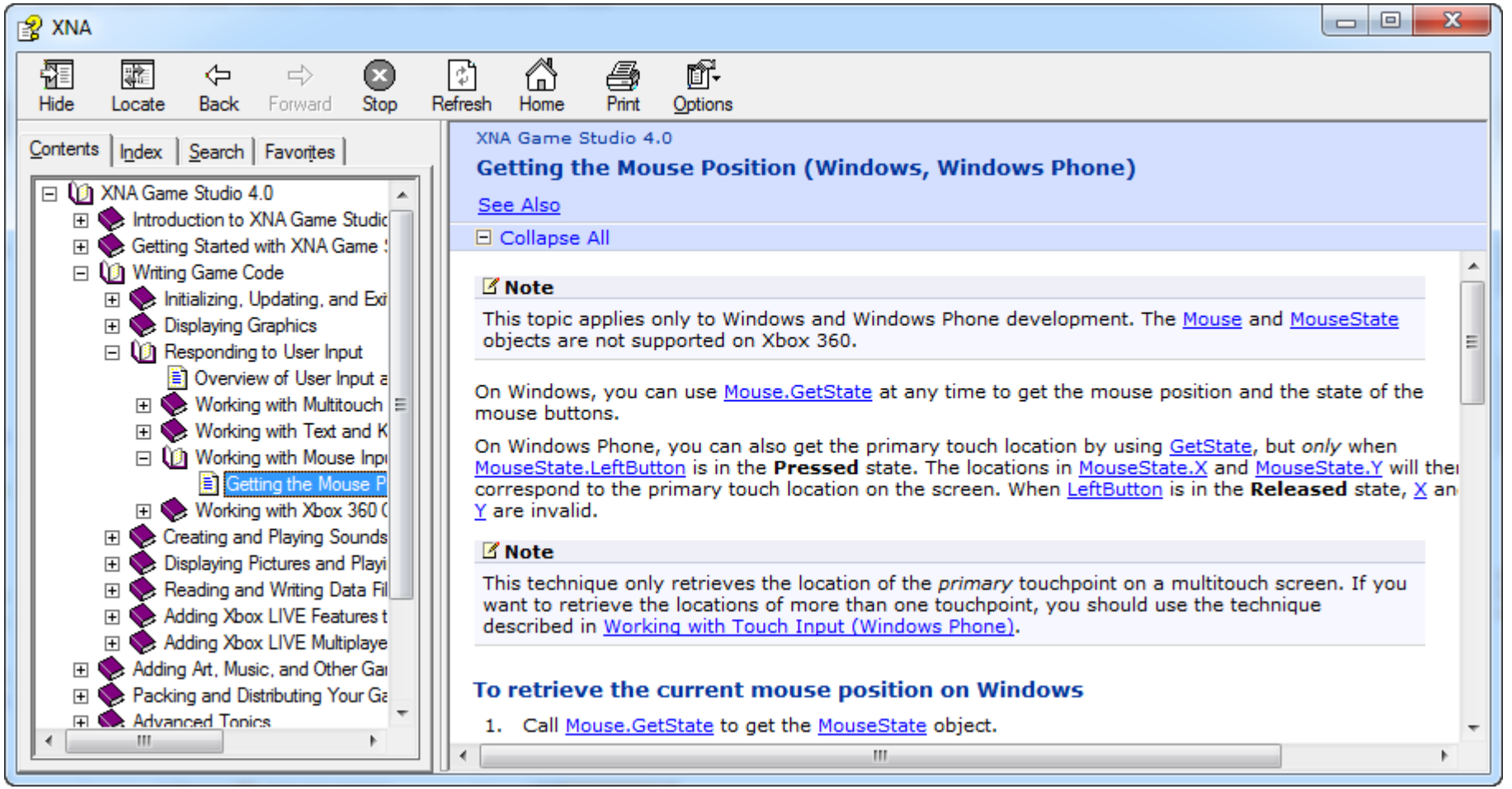
Detecting a mouse over

- How do you detect a mouse over?
- Update method?
- How about using the XNA help system. What? No one really uses 'help' do they?



Detecting a mouse over

- It's all there.



Detecting a mouse over

- In the update method, you can get mouse state which includes the X,Y location and compare it to the location of your button.

```
            MouseState ms = Mouse.GetState();

            // on Windows, the current state of the mouse cursor can be obtained at any time.
#if WINDOWS
            curMousePos.X = ms.X;
            curMousePos.Y = ms.Y;
#endif

            // if the mouse button is held down (or the touchscreen is pressed for phone), drop
            // a piece of food at the location given.
            if (canDrop && ms.LeftButton == ButtonState.Pressed)
            {
                foodLocations.Add(new Vector2(ms.X, ms.Y));
                canDrop = false;
            }
            else if (ms.LeftButton == ButtonState.Released)
            {
                // wait until the button is released before allowing the player to drop another
                // piece of food.
                canDrop = true;
            }
```

Comparison?

- How do I compare?
- All languages include a handy thing called an if statement. Try something like this.

```
if ((ms.X > vButtonLoadLoc.X) &&
   (ms.X < vButtonLoadLoc.X+132) &&
   (ms.Y > vButtonLoadLoc.Y) &&
   (ms.Y < vButtonLoadLoc.Y+32))
{
      Console.WriteLine("I'm over!!!");
}
else
{
      Console.WriteLine("I'm not over!!");
}
```

Performance and Console.Write

- If you have performance issues, first make sure your code isn't using Console.Write.
- Even a few calls to Console.Write can bring an application to a crawl.

Comparison

- If statement operations.

Symbol	Use
&&	AND
||	OR
>	Greater Than
<	Less Than
!=	Not Equal
>=	Greater Than and Equal
<=	Less Than and Equal

Changing the button image

- You need a way to change which image is showing when the mouse is over. A boolean value is a great way to switch between two images.
  - bool isOverLoad = false
  - bool isLoadPressed = false;

```
if ((ms.X > vButtonLoadLoc.X) &&(ms.X < vButtonLoadLoc.X+132) &&
   (ms.Y > vButtonLoadLoc.Y) && (ms.Y < vButtonLoadLoc.Y+32))
{
      isOverNorm = true;
}
else
{
      isOverNorm = false;
}
```

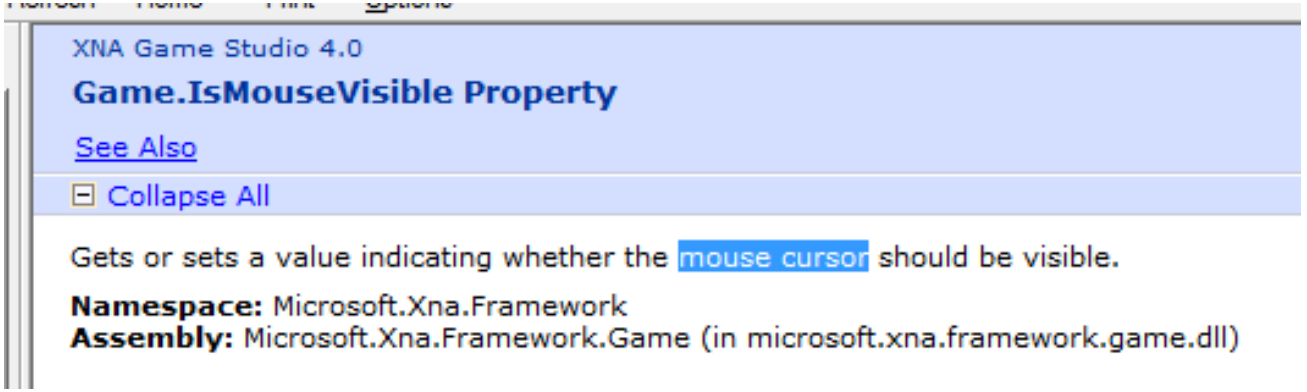
Changing the button image

- In the draw method you have to draw one image or the other

```
// draw the button
if (isOverLoad)
{
    spriteBatch.Draw(texButtonLoadOver, vButtonLoadLoc, Color.White);
}
else
{
    spriteBatch.Draw(texButtonLoadNorm, vButtonLoadLoc, Color.White);
}
```

Cursor

- Did you notice the cursor doesn't show. Me too and I had to look it up in the XNA help



- In Initialize I now call
- this.IsMouseVisible = true.
- Why this? Did you notice that the Game1 class for this game you are creating inherits type Game?

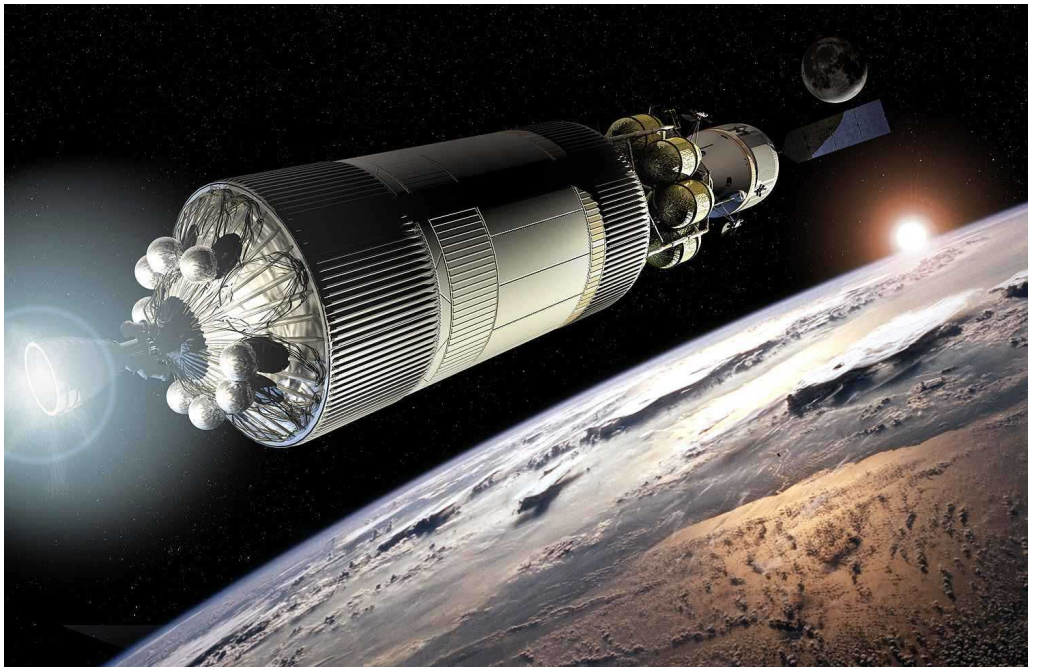
Homework #1

- I'm need a method that reverses an array of integers of any size. I will call it like this.
- int myarray = {1, 2, 3, 4, 5};
- ReverseArray(ref myarray)
- You need to write the ReverseArray function. Send it to me via e-mail before 7pm on Sept 6.
- It will need to handle any size of array, including a zero length array.

Homework #2

- Create a screen with two buttons. The buttons only have to deal with mouse over since we didn't cover mouse clicks yet.
- New Game
- Exit
- Each button should have a normal and an over image and show the correct image when the mouse is over the image or not.
- Zip the project up, give it the name Assign1JohnDoe.zip. Use your name for JohnDoe. 5 points for the correct name. Upload it before 7pm on 9/8 for full credit. 50% credit until 9/15, zero after that.
- Consider using a class to contain your button. This will save you time later. Your choice.

X Prize Homework



- If you feel compelled to work on a some really hard C# problems to help advance the PantherEngine, let me know. I have some bugs that need looking into. Lua interpreter debugging anyone?