

INFT3960 – Game Production

Week 02

Module 2.2

Introducing C#

Course Overview - 2019

Lec	Date	Modules	Assignments
1	Tuesday 30 Jul	Mod 1.1, Mod 1.2	
2	Tuesday 5 Aug	Mod 2.1, Mod 2.2, Mod 2.3, Mod 2.4	
3	Tuesday 12 Aug	Mod 3.1, Mod 3.2, Mod 3.3	★ Assign 1 12 Aug, 11:00 pm
4	Tuesday 19 Aug	Mod 4.1, Mod 4.2	
5	Tuesday 26 Aug	Mod 5.1, Mod 5.2	
6	Tuesday 3 Sep	Mod 6.1, Mod 6.2, Mod 6.3	
7	Tuesday 10 Sep	Mod 7.1, Mod 7.2	★ Assign 2 12 Sep, 11:00 pm
8	Tuesday 17 Sep	Mod 8.1	
9	Tuesday 24 Sep	Mod 9.1	
10	Tuesday 15 Oct	Mod 10.1, Mod 10.2	
11	Tuesday 22 Oct	Mod 11.1, Mod 11.2	★ Assign 3 24 Oct, 11:00pm
12	Tuesday 29 Oct	Mod 12.1, Mod 12.2	

Course Details

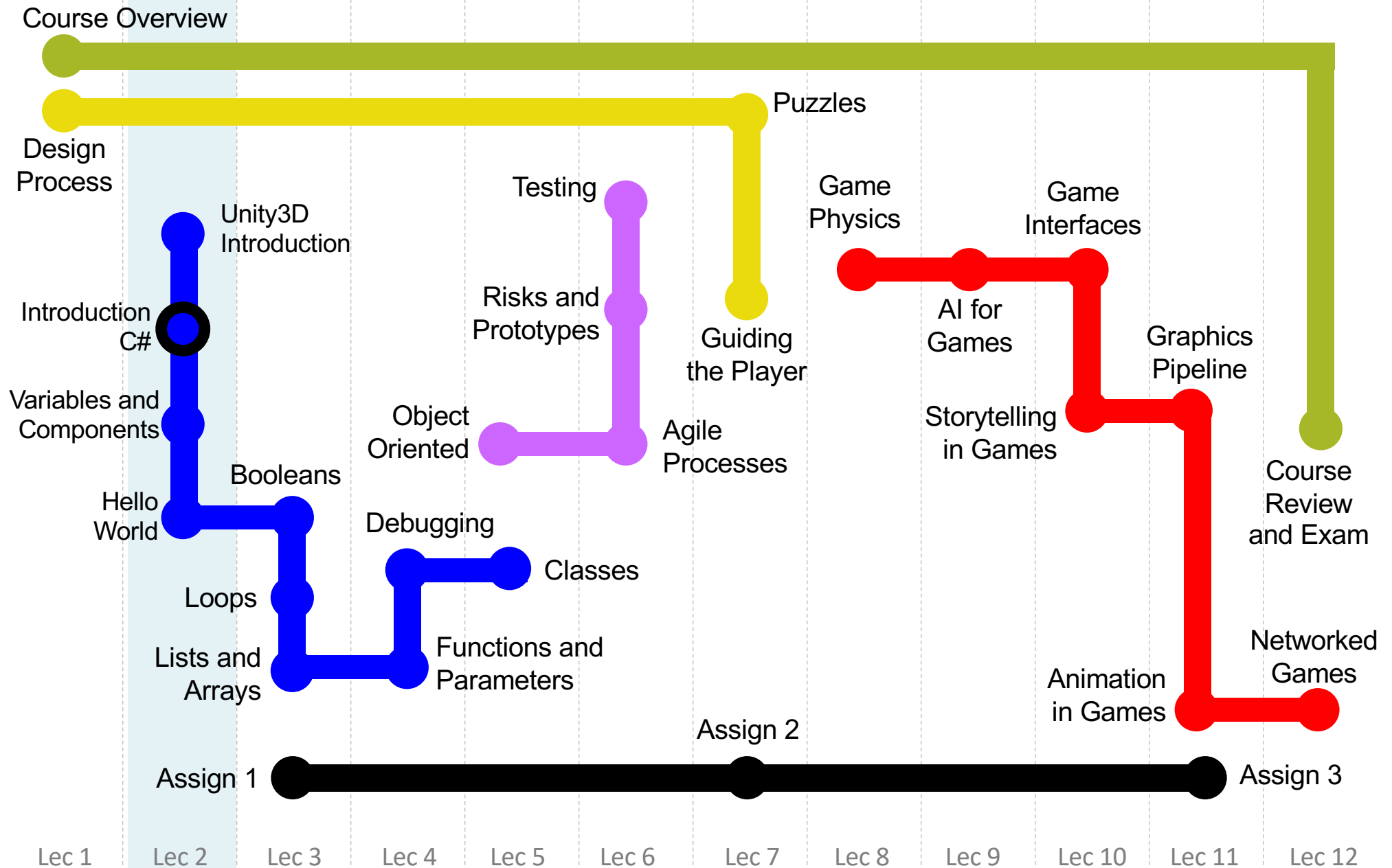
Game Design

Unity 3D and C#

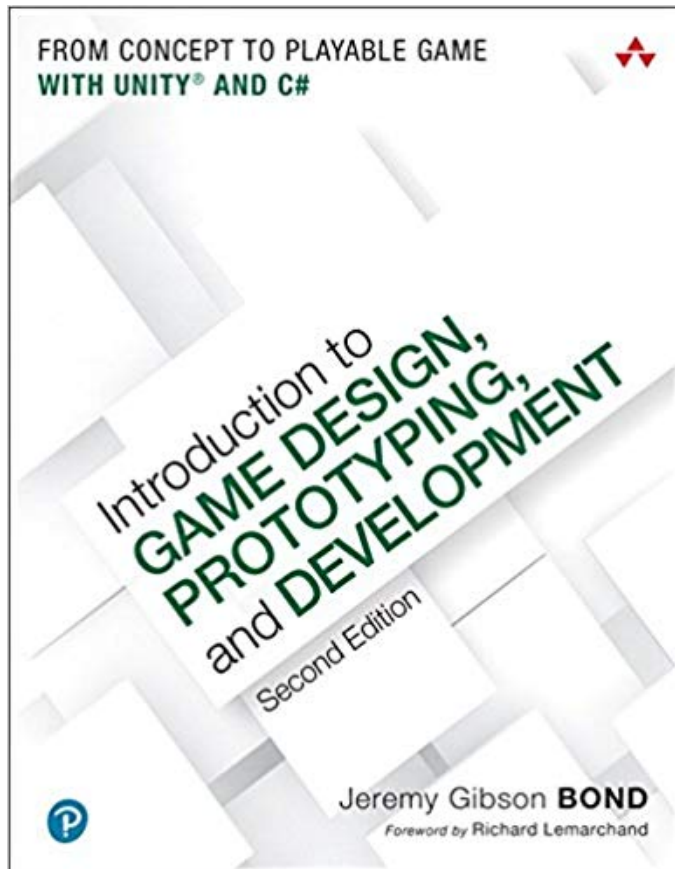
Development Process

Core Game Concepts

Assignments



Textbook – (Chapter 18)



INTRODUCING OUR
LANGUAGE: C#

Introduction C# – Topics

- The Features of C#
- C# is a Compiled Language
- C# is Managed Code
- C# is Strongly Typed
- C# is Function Based
- C# is Object-Oriented
- Reading and Understanding C# Syntax

Machine Language

C# is a Compiled Language

Computer chips only understand machine language

000000 00001 00010 00110 00000 100000

(This is what punch cards were)

Authoring languages were created to be an intermediary language between humans and computers


Two kinds of authoring languages:

- Interpreted
- Compiled

Interpreted Languages

e.g., JavaScript, PHP, and Python

Two-Step Process

- 
1. Programmer writes the code
 2. An interpreter converts the code into machine language in real-time

Benefits

- Portability: Can run on any kind of computer as long as there's an interpreter

Detriments

- Lack of Speed: The processing power used to interpret the code is not spent on the game itself
- Lack of Efficiency: Because the code can run on any computer, it's not optimized for any specific computer

Compiled Languages

e.g., C#, Basic, Java, C++

Three-Step Process

1. Programmer writes the code
2. Programmer uses a compiler to convert the code into machine language
3. Computer executes the code

Benefits

- Speed: Computer spends more processor power on game itself
- Efficiency: Code is optimized for a specific processor architecture

Detriments

- Lack of Portability: Compiled for only one kind of machine
- Extra Compilation Step

Computer Languages

Computer Programming Languages			
Machine-Readable	Human-Readable Authoring Languages		
<div>Machine Language</div>	Compiled		Interpreted
	Unmanaged	Managed	<div>JavaScript</div> <div>PHP</div> <div>Python</div>
	<div>BASIC</div> <div>C++</div>	<div>C#</div> <div>Java</div>	

Memory Management

C# is managed code

Computers have a limited amount of Random Access Memory (RAM)

Older compiled languages like BASIC and C++ require the programmer to manually allocate and deallocate RAM

In managed code, allocation and deallocation are handled automatically

This makes it less likely that you will accidentally claim all of the memory (Doing so is known as a "memory leak")

VARIABLES IN COMPUTER LANGUAGES

A variable is a name that can hold a value
This concept is borrowed from algebra

$$x = 5$$

$$x + 2 = ?$$

Variables in computer languages can hold more than just simple numbers

Numbers

Words, sentences, paragraphs, novels...

Images, sounds, 3D models, animations...

Functions and methods

Classes and GameObjects

Strongly Typed

C# is strongly typed

In a non-strongly typed language, a variable can hold any kind of data

The same variable could hold a number one moment and an animation the next (this can be helpful or spell disaster)

A strongly typed language restricts the type of data that can be held by any variable

<code>int x = 5;</code>	– An int x can only hold an integer number
<code>float y = 3.4f;</code>	– A float y can only hold a floating point number

Strongly Typed

C# is strongly typed

Strong typing allows accurate syntax checking - The compiler can check your code for correctness

Strong typing also allows robust code-completion -The code editor can guess what you want to type and auto-complete

C# is function based

Computer programs used to be just a series of commands

This was like giving driving directions to someone

- From school, head north on Vermont
- Head west on I-10 for about 7.5 miles (about 12Km)
- At the intersection with I-405, take the 405 south for 2mi (3.2Km)
- Take the exit for Venice Blvd.
- Turn right onto Sawtelle Blvd.
- My place is just north of Venice on Sawtelle.

C# is function based

Computer programs used to be just a series of commands

This was like giving driving directions to someone

- From school, head north on Vermont
- Head west on I-10 for about 7.5 miles (about 12Km)
- At the intersection with I-405, take the 405 south for 2mi (3.2Km)
- Take the exit for Venice Blvd.
- Turn right onto Sawtelle Blvd.
- My place is just north of Venice on Sawtelle.

Functional languages allow the encapsulation of commands

"If you see a store on the way, please BuySomeMilk()."

The BuySomeMilk() function encapsulates the many actions involved in finding and purchasing milk.

C# is Object Oriented

Functions and data used to be separate

Object-oriented programming introduced classes

Classes combine functions and data into a single object

- Variables in classes are called fields
- Functions in classes are called methods

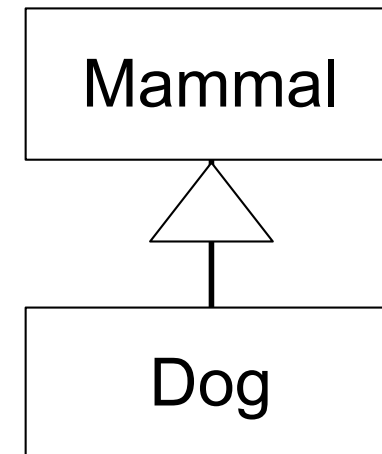
This enables things like a flock of birds where each bird thinks for itself...

...rather than being controlled by a single, monolithic program

C# is Object Oriented

Object-orientation also allows class inheritance

- A subclass can inherit the fields and methods of its superclass
- e.g., a Dog would inherit all the fields and methods of its superclass Mammal, which would in turn inherit from Animal



Languages and Syntax

All languages have syntax

The dog barked at the squirrel.

[Subject] [verb] [object].

At the squirrel the dog barked.

[Object] [subject] [verb].

The dog at the squirrel. barked

[Subject] [object]. [verb]

barked The dog at the squirrel.

[verb] [Subject] [object].

Only one of these sentences is correct

- Only one follows the rules of syntax of the English language

The other sentences have syntax errors

C# Syntax Rules

Example

```
int x = 5;
```

Declares a variable named `x` of the type `int`

If a statement starts with a type, the second word of the statement becomes a new variable of that type

C# Syntax Rules

Example

```
int x = 5;
```

Defines the value of `x` to be `5`

The `=` is used to assign values to variables

All C# statements end with a semicolon (`;`)

A semicolon is used because the period (`.`) is already used in decimal numbers (and dot notation)

C# Syntax Rules

Example

```
int x = 5;  
int y = x * ( 3 + x );
```

Declares a variable named y of the type int

Adds 3 + x for a value of 8 (because x = 5)

Just as in algebra, order of operations follows parentheses first

Multiplies x * 8 for a value of 40 (5 * 8 = 40)

Defines the value of y to be 40

Ends with a semicolon (;)

C# Syntax Rules

Example

```
string greeting = "Hello World!";
```

Declares a variable named greeting of the type string
strings can hold a series of characters like a word or novel

Defines the value of greeting to be "Hello World!"

Anything between double quotes is a string literal, a value to be
assigned to a string variable

Ends with a semicolon (;)

C# Syntax Rules

Example

```
string greeting = "Hello World!";  
print( greeting );
```

Calls the function print()

When a function is called, it executes its actions

Passes the argument greeting into the function print()

Some functions take arguments: data that changes how the function acts

The print() function will print the string greeting to the Console pane in Unity - The Console pane will display "Hello World!"

Ends with a semicolon (;)

Summary

You learned important features of C#

- C# is a Compiled Language
- C# is Managed Code
- C# is Strongly Typed
- C# is Function Based
- C# is Object-Oriented

You learned to read and understand some C#