# C# Basics

Loops

# Topics

- Four Types of Loops
  - while
  - do…while
  - for
  - foreach
- Jump Statements in Loops
  - break
  - continue

# Four Types of Loops

- while
  - The most basic loop
  - Checks a condition before each loop; loops if it's true
- do…while
  - Checks a condition after each loop; loops if it's true
- for
  - Most common loop structure
  - A loop structure that contains three separate statements
- foreach
  - Automatic for loop for enumerable collections

# Four Types of Loops

- while
  - The most basic loop
  - Checks a condition before each loop; loops if it's true

```
while (true) {
    print( "Loop" );
}
```

  - This will cause an *infinite loop*!!!
  - "Loop" will never appear in the Console pane because the entire Unity process will be frozen
  - This would necessitate *force quitting* Unity
  - On old, single-threaded computers, this would require turning the computer off!

# Four Types of Loops

- while – A better while loop
  - while loops need an exit condition
    - A condition that will cause the condition to evaluate to false
  - Checks a condition before each loop; loops if it's true

```
int i=0;
while ( i<3 ) {
    print( "Loop: "+i );
    i++;      // Increment operator
}
```

  - i++ will increment i on every pass through the loop
  - When i reaches 3, the conditional clause will evaluate to false, and the loop will exit

# Four Types of Loops

- do…while
  - Like a while loop, but checks *after* the loop has run
    - This allows a guarantee that the loop will run at least once
  - Checks a condition after each loop; loops if it's true

```
int i=5;
do {
    print( "Loop: "+i );
    i++;       // Increment operator
} while (i<3);
```

  - When execute the loop once before checking the conditional clause and then exiting
  - Note the semicolon after the while clause

# Four Types of Loops

- for
  - A for loop contains three separate clauses

```
for (int i=0; i<3; i++) {
    print( "Loop: "+i );
}
```

  - Initialization clause: `int i=0;`
  - Condition clause: `i<3;`
  - Iteration clause: `i++`
  - The i variable only exists within the for loop
    - It is *scoped* to the for loop
  - The iteration clause doesn't have to be ++
    - i-- is another common option for counting down instead of up

# Jump Statements Within Loops

- Jump statements change the execution of a loop
  - break
    - Breaks out of the loop entirely
  - continue
    - Breaks out of this iteration of the loop and moves on to the next

# Jump Statements Within Loops

- break
  - Breaks out of the loop completely

```
string str = "Hello";
foreach (char chr in str) {
    if (chr == 'l') {
        break;
    }
    print( chr );
}
```

  - This will print:

```
H
e
```

  - Once chr becomes 'l', it will break out of the loop
  - Can be used on any kind of loop

# Jump Statements Within Loops

- continue
  - Breaks out of the loop completely

```
string str = "Hello";
foreach (char chr in str) {
    if (chr == 'l') {
        continue;
    }
    print( chr );
}
```

  - This will print:

```
H
e
o
```

  - When chr is 'l', the loop continues without printing