Module 1 Day 15

File I/O and Exception Handling

Directory(Info), File(Info) and Path

- System.IO namespace
- Classes that allow you to navigate the file system
- Directory == folder
- Directory and File: static methods for navigating, creating and deleting folders and files
- DirectoryInfo and FileInfo: instance methods for detailed information on a single folder or file
- Path provides help parsing and combining paths together
- https://docs.microsoft.com/en-us/dotnet/api/system.io.directory?view=netcore-2.2
- https://docs.microsoft.com/en-us/dotnet/standard/io/common-i-o-tasks

Reading from a File

```
using (StreamReader stream = new StreamReader(path))
{
    // Read a line at a time.
    while (!stream.EndOfStream)
    {
        string line = stream.ReadLine();
        // Process the line however you want to here...
    }
}
```

- Use a *StreamReader*
  - Allows you to read and process chunks of a file sequentially
  - Think streaming a movie vs. downloading
- *EndOfStream* property
  - Tells when we have reached the end of the file
- *using* construct
  - Creates, uses and disposes a resource within the block
  - Important for cleaning up un-managed resources (*not* garbage-collected)
  - IDisposable

Exceptions

- Exceptions are how the .Net Framework reports runtime errors
- Exceptions are thrown when an error occurs
- Your code can catch an error and handle it
  - You can re-throw it using throw;
- Examples of runtime errors:
  - Attempting to int.Parse a non-numeric value
  - Attempting to read a File that does not exist
  - Divide by zero
  - NULL reference exception
- You can define and throw your own Exceptions

Exceptions

```
try
{
    // Do some work here...
}
catch (ArgumentNullException e)
{
    // catch most specific Exceptions first
}
catch (Exception e)
{
    // (optional) catch more general exceptions later
    // (optional) re-throw the same exception so it can be caught further up the stack
    throw;
}
finally
{
    // (optional) Do work that shouldexecute whether the above succeeded or failed
}
```