Chaining Methods

namespace ChainingMethods

{

/// <summary>

/// 1. Chain Method pattern is you can keep calling property or methods of a class in a chain

/// 2. Key implementation is to return this, the instnace of the current class

/// </summary>

internal class User

{

internal String? UserName

{

get; set;

}

internal User Login(string username)

{

this.UserName = username;

Console.WriteLine($"User {username} has login successfully!");

return this;

}

internal User SayHello()

{

if(String.IsNullOrEmpty(UserName))

{

throw new Exception("User hasn't login in yet. Please call Login(username)");

}

Console.WriteLine($"Hello from {UserName}.");

return this;

}

}

}

namespace ChainingMethods

{

class Program

{

/// <summary>

/// 1. Class, Struct and Interface default Access Modifier is Internal

/// 2. The access level for class members and struct members, including nested classes and structs, is private by default

/// 3. Interface members are public by default because the purpose of an interface is to enable other types to access a class or struct.

/// </summary>

static void Main()

{

User user = new User();

try

{

user.SayHello();

}

catch(Exception ex)

{

Console.WriteLine($"Exception caught: {ex.Message}");

}

user.Login("John Li").SayHello();

Console.Write("Press any key to exit ...");

Console.ReadKey();

}

}

}