**Asynchronous programming** By Kevin Singh

Asynchronous programming allows us to call multiple function calls simultaneously.

In C# there is a type called Async. This allows us to convert our function to an Asynchronous type. When converting a function to an Aysnc type we must include the following:

public async Task MyFunctionAsync()

{

await Task.Run(() =>

Code you want to run asynchronously

);

}

We must specify an “async Task” return value for our function we want to make asynchronous. Also, we must use the await call on a Task.Run (()=>); lambda call.

Also – Task.Run without an Await specifier will allow us to call an asynchronous function call. Example:

Inside the main call:

Task.Run(() =>

runAsync()

);

The async function we are trying to call:

public async Task runAsync()

{

}

Coding example:

The code below uses an external JSON library to convert a get request value to a nice clean format for us to use. You will also see an LIST of type Task which holds all of our Async function calls. Which we then finally call in a await Task.WhenAll(tasks);

List<Task> tasks = new List<Task>();

List<Stock> stock\_list = new List<Stock>();

private Stock\_Loader()

{

}

// A private static instance of the same class

private static Stock\_Loader instance = null;

public static Stock\_Loader GetInstance()

{

// create the instance only if the instance is null

if (instance == null)

{

instance = new Stock\_Loader();

}

// Otherwise return the already existing instance

return instance;

}

public void Load(Task obj)

{

tasks.Add(obj);

}

public void Load\_ex(Stock obj)

{

stock\_list.Add(obj);

tasks.Add(obj.Update\_ValuesAsync());

}

public void Load\_ex\_list(List<Stock> obj\_list)

{

stock\_list = obj\_list;

foreach (Stock stck in stock\_list)

{

tasks.Add(stck.Update\_ValuesAsync());

}

}

public async Task RunAsyc()

{

await Task.WhenAll(tasks);

}

}

public class Stock

{

private String symbol;

private float current\_Value;

private WebClient wclient;

private Uri url;

// private Label mylabel;

public Stock(string symbol)

{

this.symbol = symbol;

wclient = new WebClient();

url = new Uri("https://api.iextrading.com/1.0/stock/" + this.symbol + "/batch?types=quote");

}

public float Stock\_Value

{

get { return this.current\_Value; }

set { this.current\_Value = value; }

}

//

public async Task Update\_ValuesAsync()

{

// we will need to setup web client

while (true)

{

//Thread.Sleep(500);

if (!wclient.IsBusy)

{

await Task.Run(() =>

wclient.DownloadDataAsync(this.url));

wclient.DownloadDataCompleted += Wclient\_DownloadDataCompleted;

}

}

}

private void Wclient\_DownloadDataCompleted(object sender, DownloadDataCompletedEventArgs e)

{

if (wclient != null)

{

//Console.WriteLine(Encoding.UTF8.GetString(e.Result));

JObject o = JObject.Parse(Encoding.UTF8.GetString(e.Result));

//Console.WriteLine(this.symbol + ":" + o["quote"]["latestPrice"]);

try

{

this.current\_Value = Int64.Parse(o["quote"]["latestPrice"].ToString());

}

catch

{

this.current\_Value = 0;

}

}

//Console.WriteLine("Done!");

}

}