**1.- Introducción**

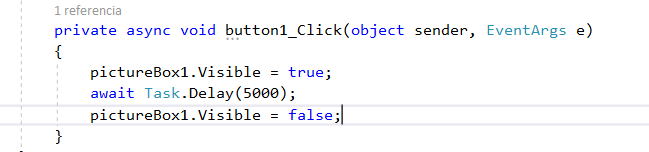
La programación en paralelo utiliza multihilos para ejecutar varias tareas en forma simultanea. Beneficio: ahorrar tiempo. Desventaja: No usar en web. Se deben ocupar en operaciones de CPU. Se ocupan varios procesadores

La programacion asincrona es aquella que utiliza los hilos de manera eficiente. Beneficio: Escalabilidad vertical(mejora en la capacidad de procesamiento), UI que no se congela, Se usan async-await, Task<T>. Se debe ocupar en operaciones con sistemas externos (I/O)

Determinismo: es la caracteristica de una funcion en la cual es posible predecir su resultado en base a sus valores de entrada. Ejemplo. Determinista Suma(), No determinista Random()

**2.- Programación asincrona**

Con async y await no se bloquea el hilo actual, sino que se le permite atender otras solicitudes hasta que se termine la tarea asincrona. Ejemplo



Await detiene la ejecucion del codigo actual y libera el hilo actual (hilo UI)

Task es una promesa de que la operacion a realizar terminara mas adelante

Una funcion asincrona puede regresar Task o Task<T>

Task.WhenAll permite esperar a que todas las tareas asincronas hayan acabado para poder continuar con el codigo

Con await Task.Run( () => {codigo}) se abre un nuevo hilo donde se ejecuta el codigo indicado

La clase SempahoreSlim permite configrar un trottling(limite de solicitudes) en un proceso asincrono mediante su metodo WaitAsync (por ejemplo dentro de un Select de LINQ)

Para obtener de Task<T> el valor de T, se ocupa la propiedad Result. Sin embargo bloquea hasta que se termine el codigo relacionado

Para implementar un ProgressBar en WindowsForms se debe agregar un control ProgressBar, crear un objeto Progress<int> (metodo que actualiza el control) e invocar el metodo Report ante cada actualizacion

**3.- Antipatrones programacion asincrona**

Sincrono dentro de asincrono: No agregar funciones sincronas(Result o una llamada sincrona a un servicio) dentro de un metodo asincrono, ya que se generarian bloqueos en la aplicacion

Asincrono dentro de sincrono: No se deben envolver funciones asincronas dentro de un metodo sincrono, ya que se genera beneficios en rendimiento

Evitar uso de async void (Excepto en eventhandlers)

Evitar Task.Factory.StartNew