Асинхронный код в C#

Как надо

1. Обращения к БД должны быть асинхронными, например, вместо `Count` нужно использовать `CountAsync` . Метод, содержащий асинхронный вызов нужно пометить как `async`, он должен возвращать `Task<T>`, а перед вызовом `CountAsync` нужно указать `await`.
2. Ко всем асинхронным вызовам на уровнях DAL и BLL нужно добавлять (`ConfigureAwait(false)`).

   Пример:

   ```
   // Асинхронный метод в сервисе (BLL)
   public async Task<RequestVM> GetRequestForDate(DateTime date)
   {
       Request request = await _requestRepo.GetRequestForDate(date).ConfigureAwait(false);
       return _mapper.Map<RequestVM>(request);
   }
   ```

3. Если асинхронный метод *A* просто "пробрасывает" результат асинхронного метода *B*, то можно не помечать метод *A* как `async` и не использовать `await` перед вызовом *B*.

   Использовать с осторожностью, т.к. могут возникнуть проблемы с обработкой исключений: если в *A* до вызова *B* есть код, который может выбросить исключение, то это исключение на уровне выше может приобрести невнятный вид (см. https://blog.stephencleary.com/2016/12/eliding-async-await.html).

   Пример:

   ```
   // Асинхронный метод в сервисе (BLL), вызывающий метод репозитория
   public Task<int> GetRequestsCount()
   {
       return _requestRepo.CountAsync();
   }
   ```

Как НЕ надо

1. Использовать `Task.FromResult`. Это костыль для имитации асинхронного использования синхронного кода.
2. Получать результат работы асинхронного метода через `.Result`. Это костыль для синхронного вызова асинхронного метода. Вся цепочка вызовов должна быть асинхронной.

Ссылки

- Асинхронное программирование с использованием async и await https://docs.microsoft.com/ru-ru/dotnet/csharp/programming-guide/concepts/async/
- Антипаттерны в работе с async/await в .NET https://habr.com/ru/company/dodopizzadev/blog/435666/