.NET Core'da Quartz Kullanımı (Yeni Versiyon)

Cron Konfigürasyon İşlemi

```
public static class CronConfigureService
{
     public static void AddJobAndTrigger<T>(this IServiceCollectionQuartzConfigurator quartz,
     IConfiguration config) where T : IJob {
          // Use the name of the IJob as the appsettings.json key
          var jobName = typeof(T).Name;

          #if DEBUG
          jobName = typeof(T).Name+"Debug"
          if (string.IsNullOrEmpty(config[$"Quartz:{jobName}"])) jobName = typeof(T).Name;
          #endif

          var configKey = $"Quartz:{jobName}";
          var cronSchedule = config[configKey];

          if (string.IsNullOrEmpty(cronSchedule))
          {
              throw new Exception($"No Quartz.NET Cron schedule found for job in configuration at
{configKey}");
          }

          // register the job as before
          var jobKey = new JobKey(jobName);
          quartz.AddJob<T>(opts => opts.WithIdentity(jobKey));

          quartz.AddTrigger(opts => opts
          .ForJob(jobKey)
          .WithIdentity(jobName + "-trigger")
          .WithCronSchedule(cronSchedule)); // use the schedule from configuration
     }
}
```

Job Tanımlama

```
public class StopHesSearchJob : IJob
{
     private readonly IHesService _hesService;
     private static readonly ILog Logger =
LogManager.GetLogger(MethodBase.GetCurrentMethod()?.DeclaringType);

     public StopHesSearchJob(IHesService hesService)
     {
          _hesService = hesService;
     }

     public Task Execute(IJobExecutionContext context)
     {
          Logger.Info("Stop hes search job çalışıyor ");
          _hesService.UpdateHesStatusPassive();
```

```
            return Task.CompletedTask;
        }
}
```

Burada 3. adımda yer alan IHesService otomatik olarak inject edilebilmesi için, **UseMicrosoftDependencyInjectionScopedJobFactory()** metodu kullanılarak aşağıdaki gibi .NET Core'da Program.cs dosyasında ayar yapılması gerekmektedir:

```
public class Program
{
     public static void Main(string[] args)
     {
         CreateHostBuilder(args).Build().Run();
     }

     public static IHostBuilder CreateHostBuilder(string[] args) =>
     Host.CreateDefaultBuilder(args)
     .ConfigureServices((hostContext, services) => { ConfigureQuartz(services, hostContext); })
     .ConfigureWebHostDefaults(webBuilder =>
     {
         webBuilder.UseStartup<Startup>();
     });

     private static void ConfigureQuartz(IServiceCollection services, HostBuilderContext
hostContext)
     {
         services.AddQuartz(q =>
         {
             q.UseMicrosoftDependencyInjectionScopedJobFactory();
             q.UseSimpleTypeLoader();
             q.UseInMemoryStore();
         q.UseDefaultThreadPool(tp => { tp.MaxConcurrency= 10; });
         q.AddJobAndTrigger<StopHesSearchJob>(hostContext.Configuration); //Yukarıda tanımlanmıştır.
     });
     // ASP.NET Core hosting
     services.AddQuartzServer(options =>
     {
         // Çalışan job'ları durdurmak için bu özellik true olmalıdır.
         options.WaitForJobsToComplete= true;
     });
}
```

Nuget Gerekli Kütüphaneler

```
<PackageReference Include="Microsoft.Extensions.DependencyInjection.Abstractions" Version="5.0.0" />
<PackageReference Include="Quartz.AspNetCore" Version="3.2.4" />
<PackageReference Include="Quartz" Version="3.2.4" />
<PackageReference Include="Quartz.Extensions.DependencyInjection" Version="3.2.4" />
```

appsettings.json dosyasının güncellenmesi

```
    "Quartz": {
     "StopHesSearchJob": "0 0 1 ? * * *",
     "StopHesSearchJobDebug": "0 * * ? * *"
   }
```

Burada dikkat edilecek husus şudur: **StopHesSearchJob** isminde bir **Job** sınıfı olmak zorundadır!