ASP Core Configuration vererben

public IConfiguration Configuration { get; }

in ServiceCollection

services.Configure<BaseConfiguration>(this.Configuration);
services.Configure<ChildConfiguration>(this.Configuration);

Die ChildConfiguration erbt von BaseConfiguration

Hat die ChildConfiguration Unterknoten, erben auch die Unterknoten. So wird die Konfiguration „erweitert“

Integrations Test mit Dependency Injection MsTest v2 .Net Core 3.1

Möchte man einen Integrationstest schreiben und dabei dependency injection nutzen, muss man folgendermaßen vorgehen:

1. Im Besten Fall teilt ma die Ausführende Applikation z.B. Web Applikation, Console etc und eine Bibliothek.
2. In die Bibliothek kommt die Auslagerung der StartUp: [crayon-6779c7c7c381c073699739/]
3. Im Testprojekt wird eine Basis Klasse definiert, die dieses Modul einliest. Nun kann in CofigureServices services.UseMyModule() aufgerufen werden oder der Host

selbst gebaut werden:
[crayon-6779c7c7c3823566190184/]

4. Die eigentliche Test Klasse erbt nun von diesen BaseUnitTest. Nun kann ein Propertiy erzeugt werden, welche einen Service aus der Dependency Injection ausliest:
[crayon-6779c7c7c3826972365145/]

.gitignore SSIS dtsx Dateien hinzufügen

in der VisualStudio .gitignore: https://www.gitignore.io/api/visualstudio

wird das komplette Verzeichnis „obj“ exkludiert. Die packages (.dtsx) aus SSIS befindet sich aber gerade dort.

Daher muss man folgende Zeile auskommentieren:
[crayon-6779c7c7c3d45984683380/]
und unter diesen Block einen neuen Block hinzufügen:
[crayon-6779c7c7c3d48924865794/]
meine komplette .gitignore sieht demnach so aus:
[crayon-6779c7c7c3d4a388312132/]

Step 1: Gitea installieren

1. Chocolaty ist eine Windows Umgebung um Anwendungen über Kommandozeile zu installieren:
    1. https://chocolatey.org/install
2. GO installieren
    1. https://golang.org/dl/
3. cmd öffnen und echo %GOPATH% eingeben. Wenn als Ergebnis wieder %GOPATH% steht, Windows neustarten und nochmal prüfen. Andernfalls unter Systemsteurung -> System -> Erweitert -> Umgebungsvariablen die GOPATH Variable setzen
4. cmd als Admin starten und folgende Befehle eingeben:
5. [crayon-6779c7c7c3f10812444476/]

6. Nun ist der Service unter http://localhost:3000 erreichbar, zeigt aber Fehler an. Daher muss man das Package herunterladen:
    1. https://dl.gitea.io/gitea/1.11.0/gitea-1.11.0-windows-4.0-amd64.exe
    2. Diese direkt in das Verzeichnis c:\gitea kopieren
7. [crayon-6779c7c7c3f13324417640/]

MS SQL Server vorbereiten

8. Im SQL Management Studio nun eine Neue Datenbank mit dem Namen „Gitea“ erstellen
9. Unter Security -> Logins einen Neuen SQL User „gitea-user“ erstellen.
10. R. Maustaste auf den User -> Properties -> Server Roles und nur die neue Gitea Datenbank zuweisen
11. Mindestens einmal als dieser User einlogen und das neue Passwort setzen
12. Wieder mit Properties -> Global die Haken bei Enforce Password expiration raus nehmen

SSH Server einrichten

13. PowerShell als Admin öffnen
    [crayon-6779c7c7c3f15597153165/]
14. Zum SSH Test einloggen eingeben. Standard Port ist 22 und sollte in der Firewall freigeschaltet sein
    [crayon-6779c7c7c3f16986308421/]

Microservices mit .Net Framework und Core ohne Docker

Momentan liest man überall von Microservices und welche Vorteile diese Architektur mit sich bringt. Zwar wird immer wieder aufgeführt, dass Microservices Polyglot (Eine Architektur mit unterschiedlichen Programmiersprachen) unterstützen können. Tatsächlich findet man aber nur Anleitungen (In der Microsoft Welt) Zu .Net Core, in Verbindung mit Docker und Azure.

Mein Ziel ist es eine Architektur ohne Docker (Da auf VMs nicht unterstützt wird) und eine Verbindung aus Core und .Net Framework herzustellen. Aus einer Reihe von nachhaltigen Frameworks soll die Entwicklung in CD (Continues Delivery) gestaltet werden. TFS und GitLab scheiden wegen ihren Lizenzmodell aus.

- CodeRepository: Git -> https://git-scm.com/download/win
- Build Server: Jenkins -> https://jenkins.io/download/
    - https://www.guru99.com/jenkin-continuous-integrati

on.html
- Api Gateway (Kommunikation ClientApi zu Microservices) Ocelot : https://github.com/ThreeMammals/Ocelot
- Kommunikationsprotokoll zwischen Microservices: gRPC (Da schneller als Http)
- Sicherheit unter Microservices: JWT Token
- Authentifizierung Microservice mit OAuth 2.0 und OWIN Middleware: https://oauth.net/code/dotnet/
- Repository: Implementierung mit Dapper: https://github.com/StackExchange/Dapper
- Repository Cache: https://github.com/MichaCo/CacheManager
- Datenbank Code Migration SSDT: SQL Server Data Tools
    - https://www.youtube.com/watch?v=6ass_PYECmM&t
    - https://arapaima.uk/post/2017-04-04-jenkins-windows-git-ssdt-profit/
- (Optional) Search Engine: https://www.elastic.co/guide/en/elasticsearch/client/net-api/current/introduction.html
    - https://www.red-gate.com/simple-talk/dotnet/net-development/how-to-build-a-search-page-with-elasticsearch-and-net/

Windows Bug Tooltip bleibt hängen

Wahrscheinlich jeder kennt es, wenn plötzlich irgendwo im Bildschirm ein Tooltip hängt und nicht weg geht.

Dann kann man einfach Windows+D 2x drücken und dann ist der weg

---

Visual Studio Build Events

Wenn man nach einem Build eine Applikation starten möchte, muss man einen Umweg um eine bat Datei gehen.

R. Maustaste auf das Projekt in Visual Studio, dann auf „Build“ und unter Pre oder Post folgendes eingeben

call meinPfadZurBatDatei\startPublisher.bat

und unter diesen Pfad müssen wir eine bat Datei erstellen, die wiederum eine exe ausführt

einfach start meinPfadZurExe.exe in die bat Datei schreiben

SQL Server Remote Zugriff

Um auf eine SQL Instanz per Remote (von außen) zugreifen zu können, muss man in der Firewall folgende Ports aufmachen:

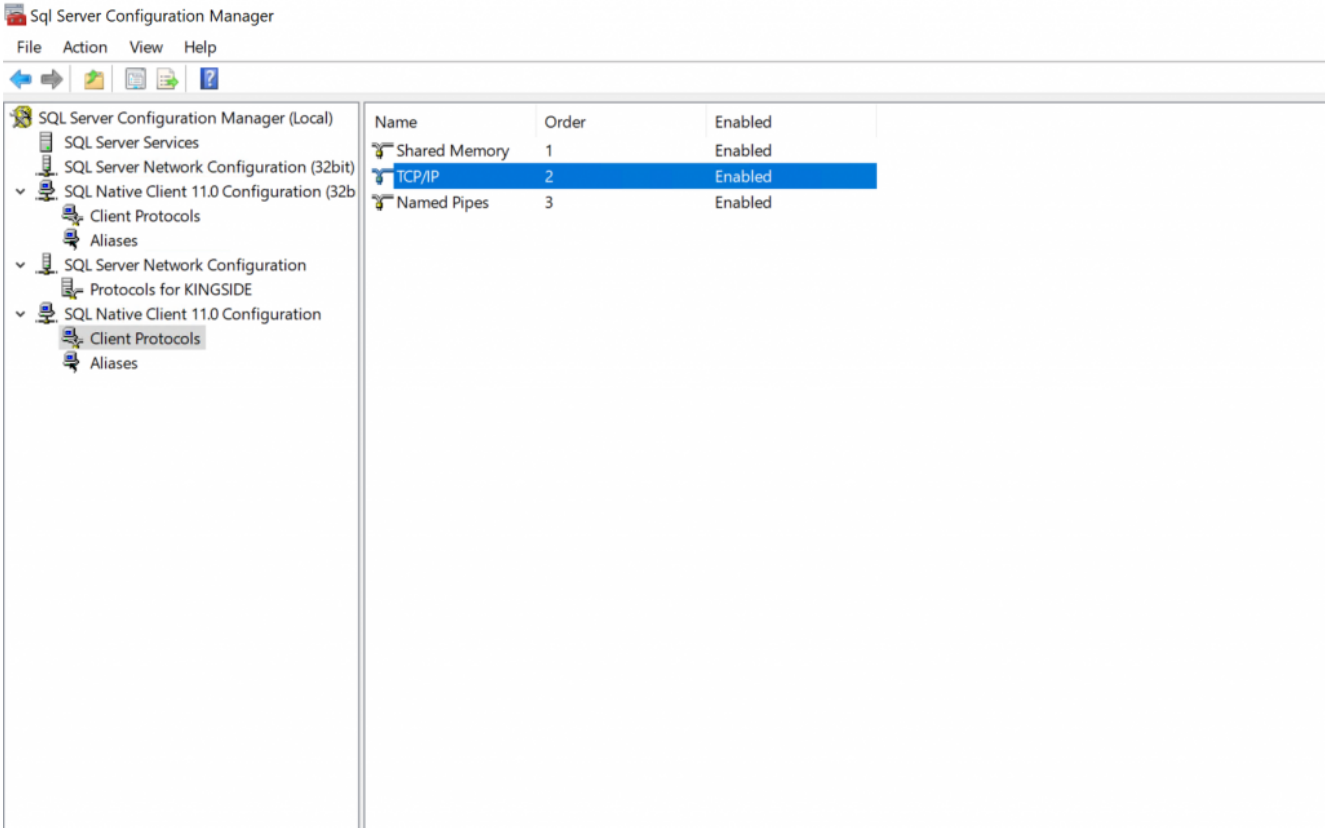
InBound (Eingehende): TCP 1433 (für z.B. SSMS), UDP 1434 (für ODBC Verbindungen)

OutBound (Ausgehende): TCP 1433

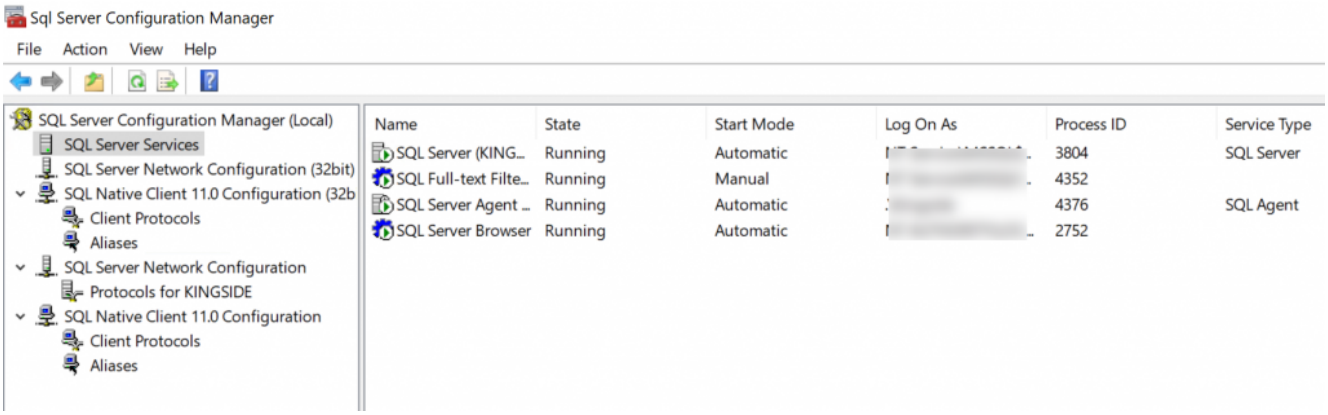
Zusätzlich die Dynamic aus „SQL Server Network Configuration“ -> „Protocols for [INSTANZ]“ -> „TCP/IP“ -> Reiter „IPAdresses“ -> Im Feld „IPAll“ -> TCP Dynamic Ports

Auserdem muss der TCP/IP Client Protokoll in SQL Configuration

eingeschaltet werden:



außerdem muss der SQL Browser Service laufen:



Eigene Adblock uBlock

Filterliste

Surface 2019-05-19
[crayon-6779c7c7c40b7919258152/]

Pokemon Tower Defense Tricks

// Letzte Trades

http://www.ptdtrading.com/latestTrades.php?whichProfile=1

JQuery hinzufügen dynamisch
[crayon-6779c7c7c424c481664530/]
Blendet alle Regulären Pokemon aus
[crayon-6779c7c7c424f026850966/]
Blendet Pidgeot aus
[crayon-6779c7c7c4251726159070/]
[crayon-6779c7c7c4252134845530/]