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-6768713019d8e079715081/]

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-6768713019d93354354322/]

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-6768713019d95943037954/]
14. Zum SSH Test einloggen eingeben. Standard Port ist 22 und sollte in der Firewall freigeschaltet sein
    [crayon-6768713019d96793122190/]

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

SQL Rekursion Parent zu Child / Child zu Parent

Erster Block im CTE: In der WHERE Bedingung wird die Child-Id übergeben

Zweiter Block: Join auf die CTE, ParentId mit der TabelleId
[crayon-676871301a14d670630964/]
Erster Block im CTE: In der WHERE Bedingung wird die ParentId mit NULL angegeben

Zweiter Block: Join auf die CTE, Id mit der Tabelle ParentId
[crayon-676871301a150877310536/]

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

HTML / CSS Container erstellen, der die Zeilen formatiert wie der Inhalt des Containers

einfach <pre> nutzen

Siehe https://www.mediaevent.de/xhtml/pre.html

SQL Server Remote Zugriff

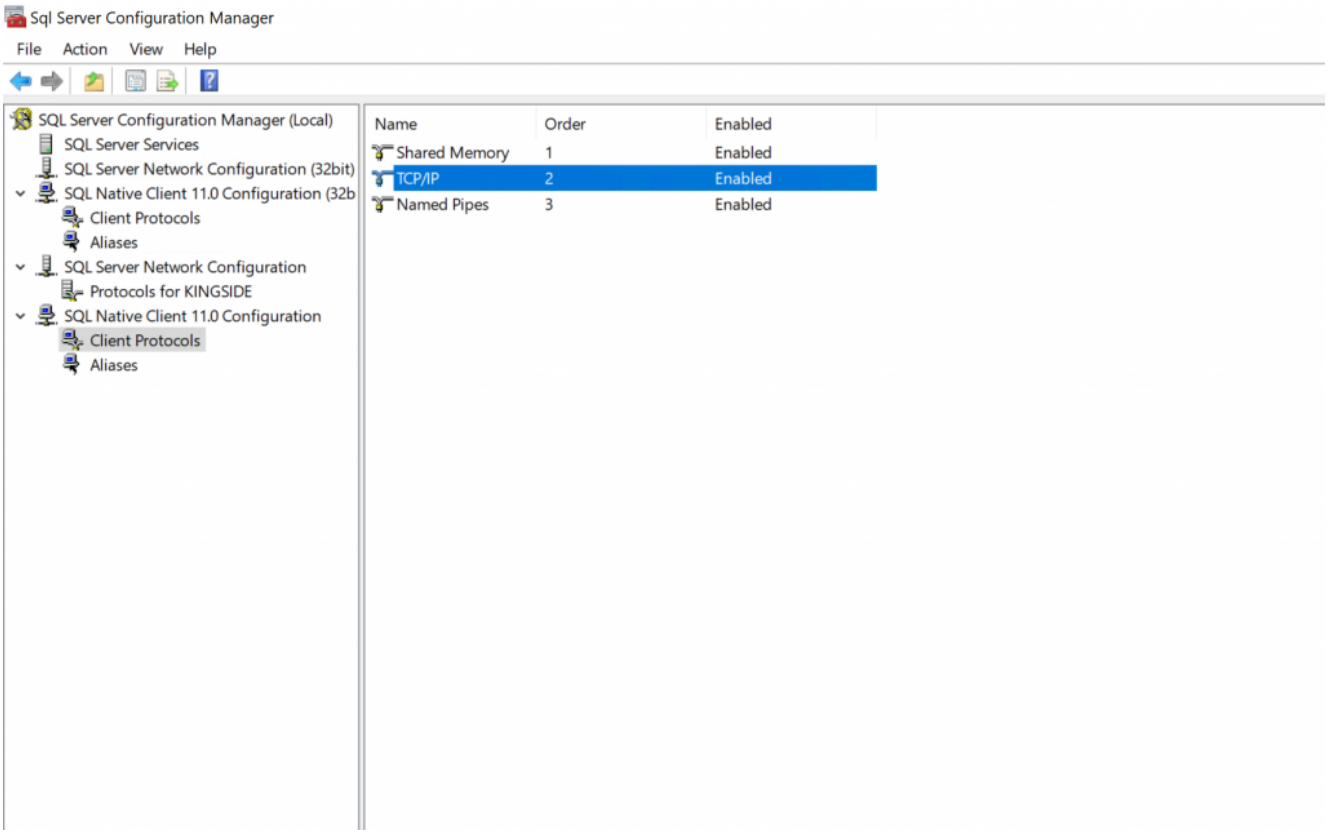
Um auf eine SQL Instanz per Remote (von außen) zugreifen zu können, muss man in der Firewall folgende Ports aufmachen:

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

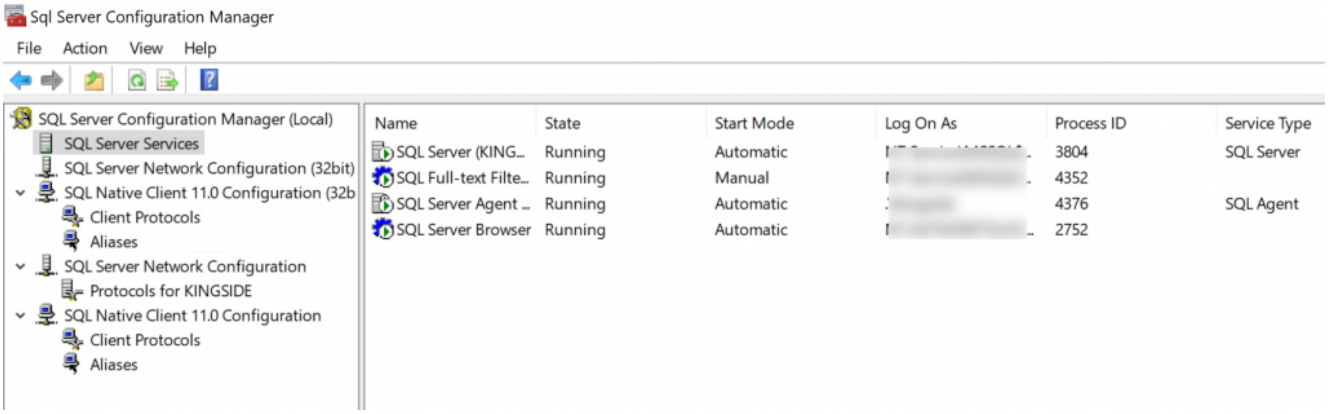
OutBound (Ausgehende): TCP 1433

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:



Async await

Wie funktionier await?

Ein Thread ist wie eine Pipeline, die einen bestimmten Code in die CPU gibt.

Mal angenommen Thread 1 (Ui Thread) will etwas downloaden und dann auf der UI darstellen. Während des Http Requests ist die Ui gesperrt. Weil der Thread 1 darauf wartet, bis der Server geantwortet hat.

Mit await wird ein freier Thread genommen und der macht den Request. Thread 1 wird nun freigegeben. Der Benutzer kann z.B. weiter die Ui wie gewohnt nutzen.

Wenn Thread 2 fertig ist, sagt er, ich bin fertig und möchte zu dem Thread zurück kehren, der ihn erzeugt hat: Thread 1.

Thread 1 übernimmt und der Codeblock wird weiter ausgeführt.

Wenn es nun egal ist, welcher Thread den Codeblock weiter ausführen soll, kann man
[crayon-676871301a2c3525141996/]
nutzen.

.Result sollte nicht genutzt werden, da im Exceptio Stacktrace Fehlermeldungen wie „MoveNext()“ erscheinen. Diese kommen aus der kompilierten Statemachine. Besser ist es
[crayon-676871301a2c6037959308/]
zu nutzen, da dann unser Typ und unsere Exception zurück gegeben werden.

Handlebars render Funktion

wie bei Mustache

[crayon-676871301a3f5483201522/]

Eigene Adblock uBlock Filterliste

Surface 2019-05-19
[crayon-676871301a960137021341/]