Erstellen einer Continuous Integration Build Pipeline mit Jenkins, Docker und SQL Server

Kathegorie
SQL Server

Aaron
Priesterroth

In einer Welt mit kontinuierlicher Datenintegration und sofortiger Veröffentlichung kann es vor kommen, dass wir vielleicht mehrere unterschiedliche Builds an nur einem Tag erzeugen müssen. Dafür eignet sich besonders die Verwendung von Docker, da mit dessen Hilfe Umgebungen mit nur wenig Arbeit erzeugt und genau so leicht wieder entfernt werden können.

In dem folgenden Artikel wollen wir veranschaulichen wie man mit der Hilfe von GIT, Jenkins, Docker und SQL Server für Linux eine simple Pipeline bauen kann.

Die Pipeline

Die von uns im Folgenden erstellte Pipeline wird aus insgesamt 5 unterschiedlichen Schritten bestehen. Diese sind:

1. Auschecken eines SSDT Projekts aus einem lokalen Git Repository.
2. Konvertieren des Projekts in ein sog. **dacpac** (deploy-able artifact) mit Hilfe von msbuild.
3. Ein Container mit SQL Server für Linux wird erzeugt.
4. Das zuvor erzeugte **dacpac** wird auf den Container übertragen.
5. Der erzeugte Container wird gestoppt und entfernt.

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

Docker installieren

In diesem Artikel wird vorausgesetzt, dass Docker für die Verwendung mit Hyper-V installiert ist. Dafür muss sichergestellt werden, dass die Windows Features **Hyper-V** und **Containers** ausgewählt sind.

Außerdem muss Docker für die Verwendung von Linux Containern konfiguriert sein.

Jenkins installieren

Die Datei zum Installieren von Jenkins kann direkt über den Download-Link hier heruntergeladen werden. Es sollte darauf geachtet werden, die Option "**Mit empfohlenen Plugins**" bei der Installation zu wählen. Nach der Installation muss zusätzlich die Option ausgewählt werden, einen eigenen Service für Jenkins unter Windows zu erstellen, da Jenkins sonst vor jeder Verwendung selbstständig gestartet werden.

SQL Server 2017 Data Tier Applications Framework installieren

Das SQL Server 2017 Data Tier Applications Framework kann über den Download-Link hier heruntergeladen werden. Wir benötigen diese Applikation, um Zugriff auf die Datei sqlpackage.exe zu bekommen.

MSBuild Plugin installieren

Das MSBuild Plugin muss direkt über Jenkins installiert werden. Dafür muss mit Hilfe eines Webbrowsers die Adresse http://localhost:8080/pluginsManager/available aufgerufen oder die Jenkins Oberfläche verwendet werden, um zu **Jenkins → Manage Jenkins → Manage Plugins → Available** zu navigieren.

Nun müssen wir Jenkins noch sagen, wo das Plugin gefunden werden kann. Dafür rufen wir die folgende Seite auf: http://localhost:8080/configureTools/ (alternativ: **Jenkins → Manage Jenkins → Global Tool Configuration**). Um heraus zu finden, wo MSBuild installiert wurde, kann die "Developer Command Prompt for Visual Studio" verwendet werden. Dafür führe bitte folgenden Befehl aus:

```
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community>where msbuild
C:\Program Files (x86)\Microsoft Visual Studio\2019\Community\MSBuild\Current\
Bin\MSBuild.exe
C:\Windows\Microsoft.NET\Framework\v4.0.30319\MSBuild.exe
```

Dabei sollte drauf geachtet werden, dass das Plugin, das von Visual Studio installiert wurde, verwendet wird (hier die oberste Rückgabe).

SQL Server Data Tools Projekt herunterladen

Als Nächstes brauchen wir noch das SQL Server Data Tools Projekt, das wir verwenden wollen. Ein Demo-Projekt kann über http://localhost:8080/pluginsManager/available geklont werden:

```
aaron@DESKTOP-OBT8C77 MINGW64 ~/Projects
$ git clone https://github.com/aPriesterroth/SSDTUnitTestDemo.git
Cloning into 'SSDTUnitTestDemo'...
remote: Enumerating objects: 25, done.
remote: Counting objects: 100% (25/25), done.
remote: Compressing objects: 100% (22/22), done.
remote: Total 25 (delta 2), reused 25 (delta 2), pack-reused 0
Unpacking objects: 100% (25/25), done.
```


Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

Anschließend entfernen wir die Remote-Referenz auf Github:

```
aaron@DESKTOP-OBT8C77 MINGW64 ~/Projects
$ cd SSDTUnitTestDemo/

aaron@DESKTOP-OBT8C77 MINGW64 ~/Projects/SSDTUnitTestDemo (master)
$ git remote rm origin
```

Docker Konfigurieren

Als Erstes sollte überprüft werden, ob der Docker-Daemon läuft. Das lässt sich am einfachsten überprüfen, indem man nach einem Docker-Icon im Windows Task Tray sucht. Ist ein Symbol vorhanden, wird Docker ausgeführt. Anderenfalls sollte Docker gestartet werden.

Wenn Docker läuft, kann als nächstes das benötigte Image über den folgenden Befehl heruntergeladen werden:

```
docker pull microsoft/mssql-server-linux
```

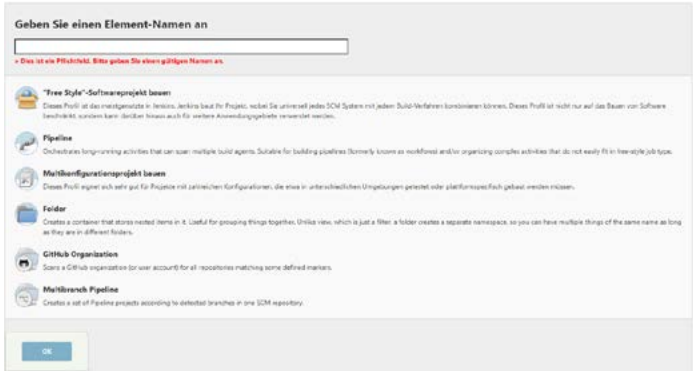
Nun müssen wir noch sicher stellen, dass der Docker-Daemon genügend Arbeitsspeicher zur Verfügung hat. Es werden mindestens 3250MB benötigt.

Um den verfügbaren Speicher zu überprüfen/erhöhen, muss mit einem Rechtsklick auf das Docker-Icon im Windows Task Tray die Einstellungen geöffnet werden. In den **Advanced**-Einstellungen kann der verfügbare Speicher mit Hilfe eines Sliders eingestellt werden. Dieser sollte auf einen Wert größer oder gleich 2350MB gesetzt werden.

Die Jenkins Build Pipeline erzeugen

Mit der URL http://localhost:8080/ kann Jenkins über einen Browser erreicht werden. Auf dem Hauptbildschirm kann jetzt mit einem Klick auf **New Item** in der oberen linken Ecke ein neues Item erzeugt werden.

Auf dem neuen Screen muss nun ein Name eigegeben werden und anschließend mit einem Klick auf **Pipeline** eine neue Pipeline erzeugt werden.



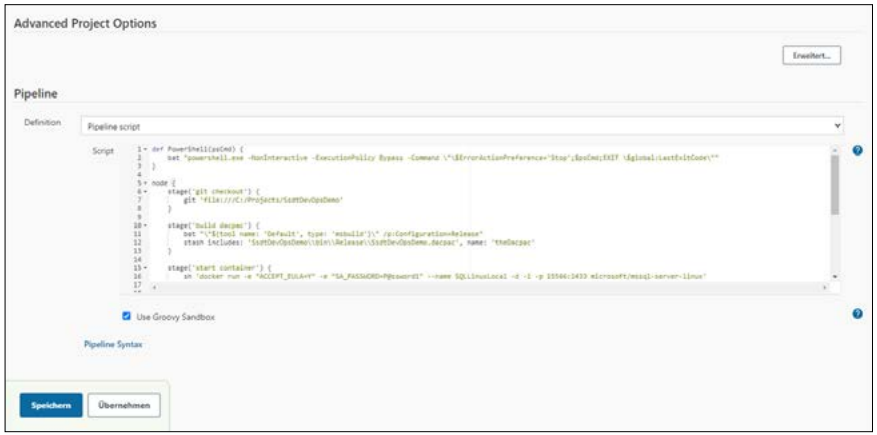
Auf dem nächsten Screen muss die Checkbox für **Github-Project** ausgewählt und anschließend für die **Project url** der Pfad zu dem zuvor heruntergeladenen Projekt angegeben werden.

Unter dem Reiter **Advanced Project Options** muss für die **Definition** der Pipeline die Option **Pipeline script** ausgewählt werden. Anschließend kann das folgende Skript an die dafür vorgesehene Stelle kopiert werden:

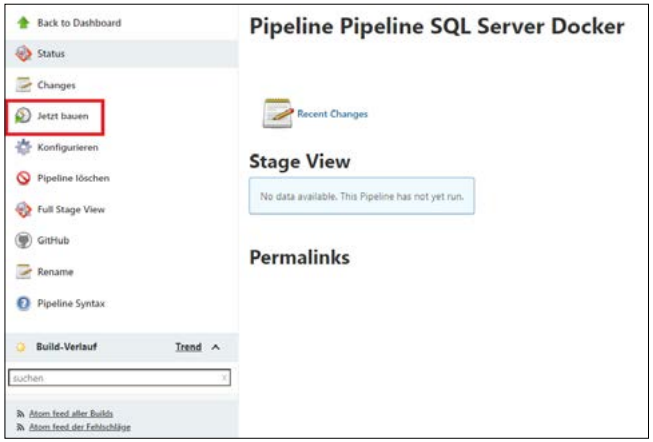
Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

```
def PowerShell(psCmd) {
    bat "powershell.exe -NonInteractive -ExecutionPolicy Bypass -Command \"\$Er-
    rorActionPreference='Stop';$psCmd;EXIT \$global:LastExitCode\""
}

node {
    stage('git checkout') {
        git 'file:///C:/Projects/SsdtDevOpsDemo'
    }

    stage('build dacpac') {
        bat "\"${tool name: 'Default', type: 'msbuild'}\" /p:Configuration=Relea-
        se"
        stash includes: 'SsdtDevOpsDemo\\bin\\Release\\SsdtDevOpsDemo.dacpac',
        name: 'theDacpac'
    }

    stage('start container') {
        sh 'docker run -e "ACCEPT_EULA=Y" -e "SA_PASSWORD=P@ssword1" --name
        SQLLinuxLocal -d -i -p 15566:1433 microsoft/mssql-server-linux'
    }

    stage('deploy dacpac') {
        unstash 'theDacpac'
        bat "\"C:\\Program Files\\Microsoft SQL Server\\140\\DAC\\bin\\sqlpa-
        ckage.exe\" /Action:Publish /SourceFile:\"SsdtDevOpsDemo\\bin\\Re-
        lease\\SsdtDevOpsDemo.dacpac\" /TargetConnectionString:\"server=lo-
        calhost,15566;database=SsdtDevOpsDemo;user id=sa;password=P@ssword1\"
        /p:ExcludeObjectType=Logins"
    }

    stage('run tests') {
        PowerShell('Start-Sleep -s 5')
    }

    stage('cleanup') {
        sh 'docker stop SQLLinuxLocal'
        sh 'docker rm SQLLinuxLocal'
    }
}
```

Zu Letzt muss noch die Checkbox mit der Aufschrift **Use Groovy Sandbox** ausgewählt werden. Mit einem Klick auf **Save** werden die Änderungen gespeichert.

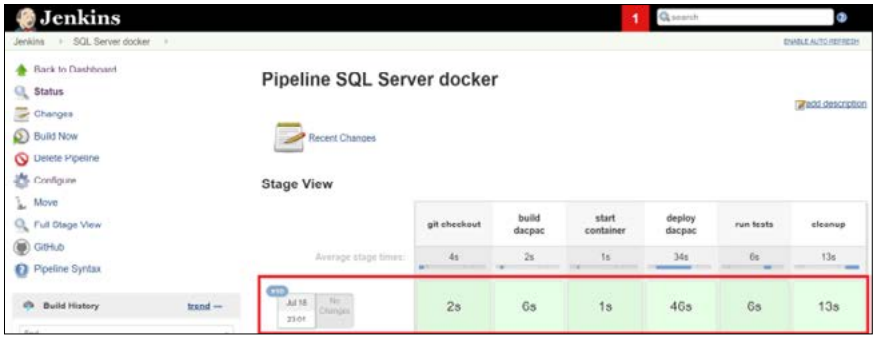


Zurück auf dem Startbildschirm müssen wir nun die eben konfigurierte Pipeline auswählen. Mit einem Klick auf **Build Now** an der linken Seite des Bildschirms wird der Build-Prozess angestoßen.



Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612

Sobald der Build-Prozess erfolgreich beendet wurde, sollte folgende Ausgabe zu sehen sein:



Dabei beschreiben die Boxen einzelne Phasen des Build-Prozesses. Jede der Boxen sollte grün hinterlegt sein.

Und schon sind wir fertig! Das hier präsentierte Ergebnis ist natürlich noch nicht das Ende. Es soll lediglich verdeutlichen wie das allgemeine Vorgehen in der Verwendung von Jenkins funktioniert. Im nächsten Beitrag werden wir unsere hier geschaffene Grundlage noch erweitern und gemeinsam eine Multi-Branch Pipeline bauen, in der die Zweige parallel verarbeitet werden.

Mainzer Datenfabrik GmbH
Rotterdamer Platz 2, 55120 Mainz
info@madafa.de
madafa.de
+49 6131 3331612