Linux部署（以ubuntu16.04为例）

本文提供ChainEval项目Linux上的两种部署方案。

# 本机部署

## 安装.net core环境

参考：https://www.microsoft.com/net/learn/get-started/linuxubuntu

curl https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > microsoft.gpg

sudo mv microsoft.gpg /etc/apt/trusted.gpg.d/microsoft.gpg

sudo sh -c 'echo "deb [arch=amd64] https://packages.microsoft.com/repos/microsoft-ubuntu-xenial-prod xenial main" > /etc/apt/sources.list.d/dotnetdev.list'

sudo apt-get update

sudo apt-get install dotnet-sdk-2.0.2

## 安装node环境（以使用编译好的文件安装为例）

(1)官方下载最新的node安装包node-v6.11.5-linux-x64.tar.xz，拷贝到Linux

(2)

xz -d node-v6.11.5-linux-x64.tar.xz

sudo cp node-v6.11.5-linux-x64.tar /usr/local

sudo tar -xvf  /usr/local/node-v6.11.5-linux-x64.tar

sudo rm /usr/local/node-v6.11.5-linux-x64.tar

sudo mv /usr/local/node-v6.11.5-linux-x64 /usr/local/nodejs

sudo rm -rf /usr/local/bin/node

sudo rm -rf /usr/local/bin/npm

sudo ln -s /usr/local/nodejs/bin/node /usr/local/bin

sudo ln -s /usr/local/nodejs/bin/npm /usr/local/bin

## 修改配置

(1)数据库配置

aspnet-core/src/Hoopox.ChainEval.Migrator/appsettings.json文件下

"ConnectionStrings": {

Default": "Server=数据库IP;Database=ChainEvalDb;charset=utf8;user=root;password=123456;",

"Redis": "localhost"

},

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

"ConnectionStrings": {

Default": "Server=数据库IP;Database=ChainEvalDb;charset=utf8;user=root;password=123456;",

"Redis": "localhost"

},

(2)跨域问题配置

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

比如前端页面是http://10.1.1.6，此处就添加”, http://10.1.1.6”。

"App": {

"ServerRootAddress": "http://localhost:21021/",

"ClientRootAddress": "http://localhost:4200/",

"CorsOrigins": "http://localhost:4200/, 前端页面url"

},

(3)Zabbix Server IP配置

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

"Zabbix": {

"ServerAddress": "http://10.1.1.170",

"ApiRequestUrl": "api\_jsonrpc.php",

"User": "Admin",

"Password": "zabbix",

"ItemValueType": 3,

"ItemHistoryLimit": 100,

//load item history \* minutes ago

"ItemHistoryTimeSpan": 30

},

(4)修改服务端url和前端url

angular/src/assets/appconfig.json

服务端url，比如“http://10.1.1.6:21021”，前端url，比如

“http://10.1.1.6”。

{

"remoteServiceBaseUrl": "服务端url",

"appBaseUrl": "前端页面url"

}

## 编译服务端项目

(1)拷贝过来项目，cd到aspnet-core目录。

(2)还原整个解决方案的依赖库

dotnet restore -s "https://nuget.cnblogs.com/v3/index.json" -s "https://api.nuget.org/v3/index.json" "aspnet-core/Hoopox.ChainEval.sln"

(3)发布服务端项目

dotnet publish -o publish -c Release "src/Hoopox.ChainEval.Migrator/Hoopox.ChainEval.Migrator.csproj"

dotnet publish -o publish -c Release "src/Hoopox.ChainEval.Web.Host/Hoopox.ChainEval.Web.Host.csproj"

数据库迁移程序会发布在src/Hoopox.ChainEval.Migrator/publish目录下。

服务端程序会发布在src/Hoopox.ChainEval.Web.Host/publish目录下。

## 编译前端项目

(1)拷贝过来项目，cd到angular目录。

(2)

npm install --registry=https://registry.npm.taobao.org

npm run build:prod-aot

前端程序会编译到dist目录下。

## 配置MySQL环境

(1)安装最新版的MySQL或者MariaDB。（以本地MySQL为例）

sudo apt-get install mysql-server

(2)修改配置文件

sudo vim /etc/mysql/mysql.conf.d/mysqld.cnf

添加

[mysql]

default-character-set = utf8

[mysqld]

character-set-server = utf8

[mysqld\_safe]

default-character-set = utf8

[mysql.server]

default-character-set = utf8

[client]

default-character-set = utf8

(3)重启MySQL

/etc/init.d/mysqld restart

## 运行程序

(1)运行数据库迁移程序

dotnet aspnet-core/src/Hoopox.ChainEval.Migrator/publish/Hoopox.ChainEval.Migrator.dll

(2)运行服务端程序

dotnet aspnet-core/src/Hoopox.ChainEval.Web.Host/publish/Hoopox.ChainEval.Web.Host.dll

(3)运行前端程序

使用nginx

# Docker部署

## 安装.net core环境（同本机部署）

参考：https://www.microsoft.com/net/learn/get-started/linuxubuntu

curl https://packages.microsoft.com/keys/microsoft.asc | gpg --dearmor > microsoft.gpg

sudo mv microsoft.gpg /etc/apt/trusted.gpg.d/microsoft.gpg

sudo sh -c 'echo "deb [arch=amd64] https://packages.microsoft.com/repos/microsoft-ubuntu-xenial-prod xenial main" > /etc/apt/sources.list.d/dotnetdev.list'

sudo apt-get update

sudo apt-get install dotnet-sdk-2.0.2

## 安装node环境（以使用编译好的文件安装为例）（同本机部署）

(1)官方下载最新的node安装包node-v6.11.5-linux-x64.tar.xz，拷贝到Linux

(2)

xz -d node-v6.11.5-linux-x64.tar.xz

sudo cp node-v6.11.5-linux-x64.tar /usr/local

sudo tar -xvf  /usr/local/node-v6.11.5-linux-x64.tar

sudo rm /usr/local/node-v6.11.5-linux-x64.tar

sudo mv /usr/local/node-v6.11.5-linux-x64 /usr/local/nodejs

sudo rm -rf /usr/local/bin/node

sudo rm -rf /usr/local/bin/npm

sudo ln -s /usr/local/nodejs/bin/node /usr/local/bin

sudo ln -s /usr/local/nodejs/bin/npm /usr/local/bin

## 修改配置

(1)数据库配置

aspnet-core/src/Hoopox.ChainEval.Migrator/appsettings.json文件下

"ConnectionStrings": {

Default": "Server=数据库IP;Database=ChainEvalDb;charset=utf8;user=root;password=123456;",

"Redis": "localhost"

},

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

"ConnectionStrings": {

Default": "Server=数据库IP;Database=ChainEvalDb;charset=utf8;user=root;password=123456;",

"Redis": "localhost"

},

(2)跨域问题配置

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

比如前端页面是http://10.1.1.6，此处就添加”, http://10.1.1.6”。

"App": {

"ServerRootAddress": "http://localhost:21021/",

"ClientRootAddress": "http://localhost:4200/",

"CorsOrigins": "http://localhost:4200/, 前端页面url"

},

(3)Zabbix Server IP配置

aspnet-core/src/Hoopox.ChainEval.Web.Host/appsettings.json文件下

"Zabbix": {

"ServerAddress": "http://10.1.1.170",

"ApiRequestUrl": "api\_jsonrpc.php",

"User": "Admin",

"Password": "zabbix",

"ItemValueType": 3,

"ItemHistoryLimit": 100,

//load item history \* minutes ago

"ItemHistoryTimeSpan": 30

},

(4)修改服务端url和前端url

angular/src/assets/appconfig.json

服务端url，比如“http://10.1.1.6:21021”，前端url，比如

“http://10.1.1.6”。

{

"remoteServiceBaseUrl": "服务端url",

"appBaseUrl": "前端页面url"

}

## 编译服务端项目（同本机部署）

(1)拷贝过来项目，cd到aspnet-core目录。

(2)还原整个解决方案的依赖库

dotnet restore -s "https://nuget.cnblogs.com/v3/index.json" -s "https://api.nuget.org/v3/index.json" "aspnet-core/Hoopox.ChainEval.sln"

(3)发布服务端项目

dotnet publish -o publish -c Release "src/Hoopox.ChainEval.Migrator/Hoopox.ChainEval.Migrator.csproj"

dotnet publish -o publish -c Release "src/Hoopox.ChainEval.Web.Host/Hoopox.ChainEval.Web.Host.csproj"

数据库迁移程序会发布在src/Hoopox.ChainEval.Migrator/publish目录下。

服务端程序会发布在src/Hoopox.ChainEval.Web.Host/publish目录下。

## 编译前端项目（同本机部署）

(1)拷贝过来项目，cd到angular目录。

(2)

npm install --registry=https://registry.npm.taobao.org

npm run build:prod-aot

前端程序会编译到dist目录下。

## 安装Docker

wget -qO- https://get.docker.com/ | sh

## 打包Docker镜像

如果之前运行过镜像，请先停止并删除旧的容器。

docker stop chainevalmigrator || true

docker stop chainevalhost || true

docker stop chainevalwebsite || true

docker rm -f chainevalmigrator || true

docker rm -f chainevalhost || true

docker rm -f chainevalwebsite || true

如果存在旧的镜像，请先删除旧的镜像。

docker rmi -f chainevalmigrator || true

docker rmi -f chainevalhost || true

docker rmi -f chainevalwebsite || true

下面打包镜像

cd aspnet-core/src/Hoopox.ChainEval.Migrator/publish/

docker build -t chainevalmigrator .

cd –

cd aspnet-core/src/Hoopox.ChainEval.Web.Host/publish/

docker build -t chainevalhost .

cd –

cd angular

docker build -t chainevalwebsite .

使用docker images可以查看现有镜像。

数据库迁移程序打包的镜像是chainevalmigrator。

服务端程序打包的镜像是chainevalhost。

前端程序会打包的镜像是chainevalwebsite。

## 配置MySQL环境

(1)建立目录/etc/mysql作为MySQL的配置文件目录。

(2)创建文件/etc/mysql/my.cnf

vim /etc/mysql/my.cnf

内容如下

[mysql]

default-character-set = utf8

[mysqld]

character-set-server = utf8

[mysqld\_safe]

default-character-set = utf8

[mysql.server]

default-character-set = utf8

[client]

default-character-set = utf8

(3)下载MySQL或者MariaDB的Docker镜像。（以MySQL为例）

docker pull mysql

(4)启动mysql

docker run -d --restart always \

-v /var/lib/mysql:/var/lib/mysql \

-v /etc/mysql:/etc/mysql \

-p 3306:3306 \

-e MYSQL\_ROOT\_PASSWORD=123456 \

--name mysql mysql:latest

此处设定root密码是123456

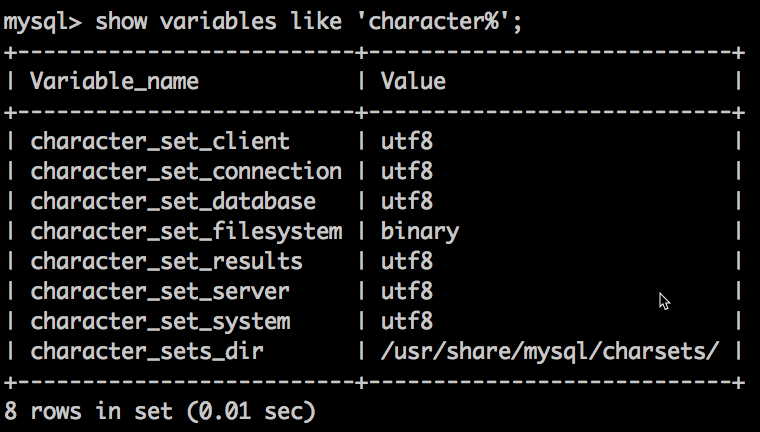
(5) 验证配置是否生效

docker exec -it mysql /bin/bash

mysql -uroot -p123456

show variables like 'character%';

可以看到字符集已经更改为 utf-8



## 运行程序

docker run -d --name chainevalmigrator chainevalmigrator

docker run -d --name chainevalhost -p 21021:21021 -e "ASPNETCORE\_URLS=http://\*:21021" chainevalhost

docker run -d --name chainevalwebsite -p 80:80 chainevalwebsite