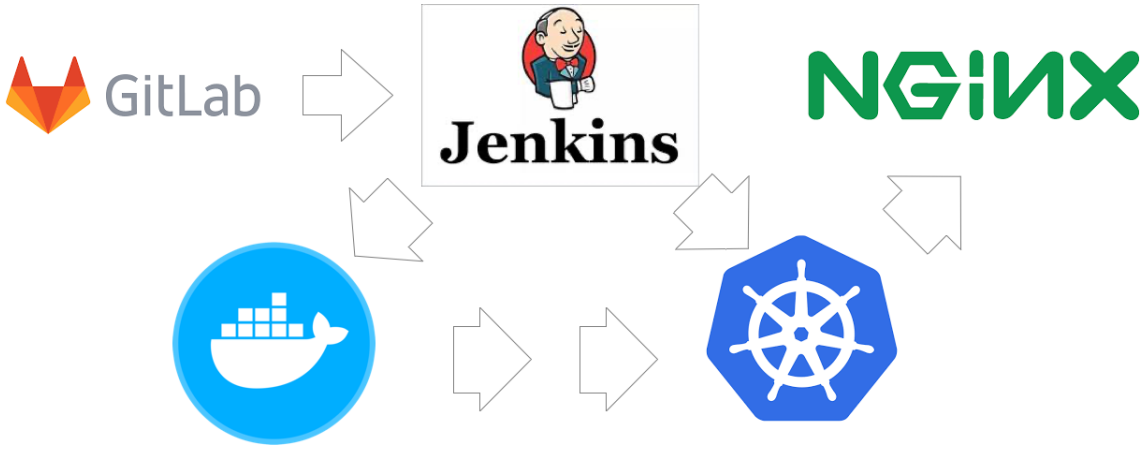
讲师(老司机)

容器虚拟化技术和自动化部署



初识skywalking

简介

随着微服务的兴起与流行，一些微服务架构下的问题也会越来越突出，服务之间的依赖关系愈发复杂。如果某个服务出现问题，寻找问题往往需要很长的时间，可能需要一个一个服务的查找问题，由此我们就有了一个新的工具去方便定位这个问题。这个工具就是APM(应用性能管理)，他可以帮助我们在出现问题时，快速定位问题原因。

Skywalking就是这样的一款APM工具，他提供了完善的链路追踪能力。随着微服务的兴起与流行，一些微服务架构下的问题也会越来越突出，服务之间的依赖关系愈发复杂。如果某个服务出现问题，寻找问题往往需要很长的时间，可能需要一个一个服务的查找问题，由此我们就有了一个新的工具去方便定位这个问题。这个工具就是APM(应用性能管理)，他可以帮助我们在出现问题时，快速定位问题原因。

Skywalking就是这样的一款APM工具，他提供了完善的链路追踪能力。

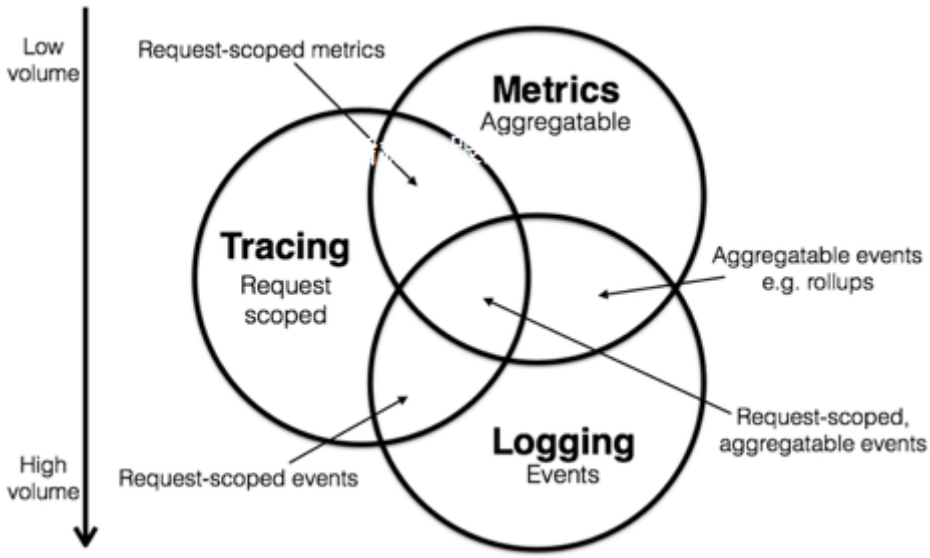
SkyWalking项目是由华为大牛吴晟开源的个人项目，目前已经加入Apache孵化器。SkyWalking项目的核心目标是针对微服务、Cloud Native、容器化架构提供应用性能监控和分布式调用链追踪功能。

APM背景

APM (Application Performance Management) 即应用性能管理系统，是对企业系统即时监控以实现对应用程序性能管理和故障管理的系统化的解决方案。应用性能管理，主要指对企业的关键业务应用进行监测、优化，提高企业应用的可靠性和质量，保证用户得到良好的服务，降低IT总拥有成本。

随着微服务、云计算的发展，提升了应用的可用性、扩展性，同时也带来了运维、监控的复杂性，当我们需要从众多服务依赖中，查询出问题根源也越来越难。

系统的监控分为三个维度：



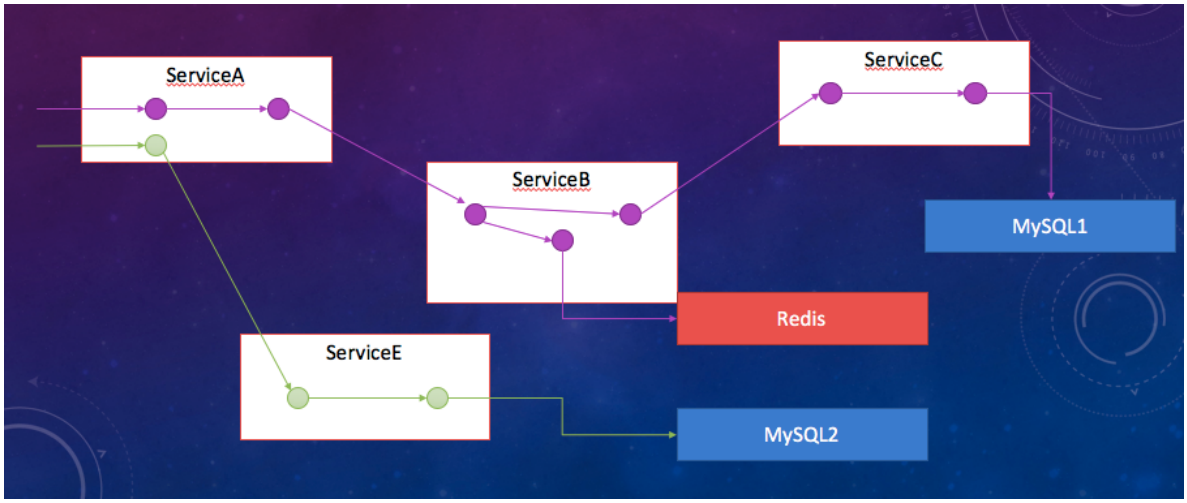
1、Metrics 指标性统计
一个服务的正确率、成功率、流量、TPS、响应时间等

2、Tracing 分布式追踪
从请求开始到结束，服务节点之间的调用链路

3、 Logging 日志记录
程序在执行的过程中间发生了一些日志，会一帧一帧地跳出来给大家去记录这个东西，这是日志记录

分布式追踪

分布式链路追踪就是将一次分布式请求还原成调用链路，将一次分布式请求的调用情况集中展示，比如各个服务节点上的耗时、请求具体到达哪台机器上、每个服务节点的请求状态等等。



skywalking简介

- 2015年开始开源，由原华为技术专家吴晟主导实现的专门为微服务架构和云原生架构系统而设计并且支持分布式链路追踪的APM系统
- 专为微服务、云原生架构和基于容器（Docker、K8s、Mesos）架构而设计
- apache顶级项目

特性

1. 多种监控手段，语言探针和service mesh

2. 多语言探针：Java、.NET Core、PHP、Node.JS、Go
3. 支持多种数据存储方式: ElasticSearch、MySql、TiDB、H2、Sharding Sphere
4. 强大的UI可视化界面
5. 高性能、高可用：agent端探针性能消耗低，服务端支持多种集群部署方式
6. 支持多种指标的监控告警
7. 微核+插件式扩展的高扩展框架

主要部分及其功能

SkyWalking主要由三大部分组成：Agent，Collector，Web：

- Agent：作用是使用JavaAgent字节码增强技术将Agent的代码织入程序中，完成Agent无代码侵入地获取程序内方法的上下文并进行增强和收集信息并发送给Collector；与Collector进行心跳，表明Agent客户端的存活。
- Collector：作用是维护存活的Agent实例，并收集从Agent发送至Collector的数据，进行处理及持久化。
- Web：作用是将Collector收集的参数进行不同维度的展示

同类型对比

同类型的开源APM产品有：大众点评的cat、韩国的pinpoint、老牌APM zipkin，综合比较skywalking是之中最值得推荐使用的产品。

skywalking优势

1. 国内开源项目，社区成熟，且可与作者直接沟通
2. 支持语言更多：Java、.NET Core、PHP、Node.JS、Golang
3. 支持java自动探针，代码无侵入，只要简单配置，就可以自动实现各种指标、调用链的埋点
4. 众多存储选择：ElasticSearch、MySql、TiDB、H2、Sharding Sphere
5. 扩展性：支持模块、插件的可拔插
6. 支持java语言的几十种插件，例如:Tomcat、RabbitMq、Kafka，且易于二次开发
7. 性能：比其他开源软件性能都强

官网地址

apache官网

```
官网地址：
http://skywalking.apache.org/

中文官网
http://skywalking.apache.org/zh/

下载地址：
https://skywalking.apache.org/zh/downloads/
```

github官网

```
https://github.com/apache/skywalking
```

准备工作

服务器规划

生成至少三台虚拟机

主机名	主机IP
k8s-master01	192.168.198.156
k8s-node01	192.168.198.157
k8s-node02	192.168.198.158
k8s-node03	192.168.198.159
harbor-155	192.168.198.155
jenkinsagent-154	192.168.198.154
skywalking-151	192.168.198.141
windows-mysql-199	192.168.198.199

下载源码包

```
https://archive.apache.org/dist/skywalking/6.6.0/apache-skywalking-apm-6.6.0.tar.gz

清华大学镜像网站：
https://mirrors.tuna.tsinghua.edu.cn/apache/skywalking/8.1.0/apache-skywalking-apm-8.1.0.tar.gz
```

skywalking安装

1. centos7+H2内存数据库方式
2. docker+ES7数据库方式
3. docker-compose+ES7方式
4. K8S+helm+ES7方式
5. window+mysql5方式

centos7安装

安装JDK8

oap

```
解压压缩包
tar zxf apache-skywalking-apm-6.6.0.tar.gz

mv apache-skywalking-apm-bin/ skywalking

cd bin/

执行启动命令
./startup.sh

耐心等待skywalking启动：
http://192.168.198.154:8080/
```

目录介绍

整体目录

```
agent:javaagent探针模块
bin:Collector和Web模块的启动脚本
config:Collector模块的配置
webapp:Web模块的jar包和配置
oap-libs:skywalking依赖的jar包配置
```

agent目录介绍

```
cd agent

1.config目录：配置文件
2.plugins目录：组件的所有插件
3.optional-plugins目录：可选插件，部分插件在使用上会影响整体的性能或者由于版权问题放置于可选插件包中，不会直接加载，如果需要使用，将可选插件中的jar包拷贝到plugins包下。
```

docker安装

前置条件

文件创建数

修改Linux系统的限制配置，将文件创建数修改为65536个：

1. 修改系统中允许应用最多创建多少文件等的限制权限。Linux默认来说，一般限制应用最多创建的文件是65535个。但是ES至少需要65536的文件创建数的权限。
2. 修改系统中允许用户启动的进程开启多少个线程。默认的Linux限制root用户开启的进程可以开启任意数量的线程，其他用户开启的进程可以开启1024个线程。必须修改限制数为4096+。因为ES至少需要4096的线程池预备。

```
vi /etc/security/limits.conf
#新增如下内容在limits.conf文件中
es soft nofile 65536
es hard nofile 65536
es soft nproc 4096
es hard nproc 4096
```

系统控制权限

修改系统控制权限，ElasticSearch需要开辟一个65536字节以上空间的虚拟内存。Linux默认不允许任何用户和应用程序直接开辟这么大的虚拟内存。

```
vi /etc/sysctl.conf

添加参数:新增如下内容在sysctl.conf文件中，当前用户拥有的内存权限大小
vm.max_map_count=262144

重启生效:让系统控制权限配置生效
sysctl -p
```

6.x版本

安装elasticsearch版本7.5.1，oap版本为6.6.0，UI版本为6.6.0。

安装elasticsearch

docker官网

```
https://hub.docker.com/_/elasticsearch
```

基础镜像

```
docker pull elasticsearch:7.5.1
```

安装elasticsearch

```
docker run -itd --name=es7 \
-p 9200:9200 -p 9300:9300 \
-e "discovery.type=single-node" elasticsearch:7.5.1
```

检查启动情况

```
等待30秒左右，查看docker日志，如果有出现
"publish_address {172.17.0.2:9300}, bound_addresses {0.0.0.0:9300}"
说明启动成功了。
```

创建持久化目录，并重启启动elasticsearch

```
mkdir -p /data/elasticsearch

docker cp es7:/usr/share/elasticsearch/data /data/elasticsearch/
docker cp es7:/usr/share/elasticsearch/logs /data/elasticsearch/

docker stop es7
docker rm  es7
```

持久化安装elasticsearch

```
docker run -d --name=es7 \
  --restart=always \
  -p 9200:9200 -p 9300:9300 \
  -e "discovery.type=single-node" \
  -v /data/elasticsearch/data:/usr/share/elasticsearch/data \
  -v /data/elasticsearch/logs:/usr/share/elasticsearch/logs \
elasticsearch:7.5.1
```

测试elasticsearch

```
google浏览器访问elasticsearch
http://192.168.198.141:9200/
```

安装oap

官网地址

```
https://hub.docker.com/r/apache/skywalking-oap-server
```

基础镜像

```
docker pull apache/skywalking-oap-server:6.6.0-es7
```

安装oap

注意：

1. **等待elasticsearch完全启动之后，再启动oap**

2. **SW_STORAGE参数没有区分elasticsearch的版本。**

```
说明：这里指定elasticsearch 来存储数据

docker run --name oap --restart always -d \
-e TZ=Asia/Shanghai \
-p 12800:12800 \
-p 11800:11800 \
--link es7:es7 \
-e SW_STORAGE=elasticsearch \
-e SW_STORAGE_ES_CLUSTER_NODES=es7:9200 \
apache/skywalking-oap-server:6.6.0-es7
```

安装UI

docker官网

```
https://hub.docker.com/r/apache/skywalking-ui
```

基础镜像

```
docker pull apache/skywalking-ui:6.6.0
```

安装UI

注意：这里映射的端口为8088，防止端口冲突。

```
docker run -d --name skywalking-ui \
--restart=always \
-e TZ=Asia/Shanghai \
-p 8088:8080 \
--link oap:oap \
-e SW_OAP_ADDRESS=oap:12800 \
apache/skywalking-ui:6.6.0
```

启动UI

在window系统中使用google浏览器访问skywalking-ui界面

```
http://192.168.198.141:8088/
```

8.x版本

升级elasticsearch版本7.9.0，oap版本为8.1.0，UI版本为8.1.0

安装elasticsearch

docker官网

```
https://hub.docker.com/_/elasticsearch
```

基础镜像

```
docker pull elasticsearch:7.9.0
```

安装elasticsearch

```
docker run -itd --name=es7 \
-p 9200:9200 -p 9300:9300 \
-e "discovery.type=single-node" elasticsearch:7.9.0
```

检查启动情况

```
等待30秒左右，查看docker日志，如果有出现
"publish_address {172.17.0.2:9300}, bound_addresses {0.0.0.0:9300}"
说明启动成功了。
```

创建持久化目录，并重启启动elasticsearch

```
mkdir -p /data/elasticsearch

docker cp es7:/usr/share/elasticsearch/data /data/elasticsearch/
docker cp es7:/usr/share/elasticsearch/logs /data/elasticsearch/

docker stop es7
docker rm  es7
```

安装elasticsearch

```
docker run -d --name=es7 \
  --restart=always \
  -p 9200:9200 -p 9300:9300 \
  -e "discovery.type=single-node" \
  -v /data/elasticsearch/data:/usr/share/elasticsearch/data \
  -v /data/elasticsearch/logs:/usr/share/elasticsearch/logs \
elasticsearch:7.9.0
```

测试elasticsearch

```
google浏览器访问elasticsearch
http://192.168.198.141:9200/
```

具体信息如下

```
{
  "name" : "24a6002d98e5",
  "cluster_name" : "docker-cluster",
  "cluster_uuid" : "X1RoxJ5cSJyo5uARFui8TA",
  "version" : {
    "number" : "7.9.0",
    "build_flavor" : "default",
    "build_type" : "docker",
    "build_hash" : "a479a2a7fce0389512d6a9361301708b92dff667",
    "build_date" : "2020-08-11T21:36:48.204330Z",
    "build_snapshot" : false,
    "lucene_version" : "8.6.0",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

安装oap

官网地址

```
https://hub.docker.com/r/apache/skywalking-oap-server
```

基础镜像

```
docker pull apache/skywalking-oap-server:8.1.0-es7
```

安装oap

注意事项:

1. **SW_STORAGE参数严格区分elasticsearch的版本**

```
docker run --name oap --restart always -d \
-e TZ=Asia/Shanghai \
-p 12800:12800 \
-p 11800:11800 \
--link es7:es7 \
-e SW_STORAGE=elasticsearch7 \
-e SW_STORAGE_ES_CLUSTER_NODES=es7:9200 \
apache/skywalking-oap-server:8.1.0-es7
```

安装UI

docker官网

```
https://hub.docker.com/r/apache/skywalking-ui
```

基础镜像

```
docker pull apache/skywalking-ui:8.1.0
```

安装UI

注意：这里映射的端口为8088，防止端口冲突。

```
docker run -d --name skywalking-ui \
--restart=always \
-e TZ=Asia/Shanghai \
-p 8088:8080 \
--link oap:oap \
-e SW_OAP_ADDRESS=oap:12800 \
apache/skywalking-ui:8.1.0
```

启动ui

在window系统中使用google浏览器访问skywalking-ui界面

```
http://192.168.198.141:8088/
```

docker-compose安装

官网地址

```
根据官网地址进入docker目录中，修改docker-compose.yml文件内容
https://github.com/apache/skywalking
```

基础镜像

```
docker pull elasticsearch:7.9.0
docker pull apache/skywalking-oap-server:8.1.0-es7
docker pull apache/skywalking-ui:8.1.0
```

docker-compose文件

```yaml
version: '3.3'
services:
  elasticsearch:
    image: elasticsearch:7.9.0
    container_name: elasticsearch
    restart: always
    ports:
      - 9200:9200
    environment:
      discovery.type: single-node
      TZ: Asia/Shanghai
    ulimits:
      memlock:
        soft: -1
        hard: -1
  oap:
    image: apache/skywalking-oap-server:8.1.0-es7
    container_name: oap
    depends_on:
      - elasticsearch
    links:
      - elasticsearch
    restart: always
    ports:
      - 11800:11800
      - 12800:12800
    environment:
      SW_STORAGE: elasticsearch7  # 指定ES版本
      SW_STORAGE_ES_CLUSTER_NODES: elasticsearch:9200
      TZ: Asia/Shanghai
  ui:
    image: apache/skywalking-ui:8.1.0
    container_name: ui
    depends_on:
      - oap
    links:
      - oap
    restart: always
    ports:
      - 8080:8080
    environment:
      SW_OAP_ADDRESS: oap:12800
      TZ: Asia/Shanghai
```

查看启动

```
docker-compose ps
```

启动ui

在window系统中使用google浏览器访问skywalking-ui界面

```
elasticsearch启动时间比较长，需要耐心等待几分钟
http://192.168.198.141:8080/
```

备份与还原

```
docker save elasticsearch:7.9.0 apache/skywalking-oap-server:8.1.0-es7
apache/skywalking-ui:8.1.0 -o skywalking8.1.0.tar

docker load -i skywalking8.1.0.tar
```

k8s安装

基础镜像

```
docker pull docker.elastic.co/elasticsearch/elasticsearch:7.5.1
docker pull apache/skywalking-oap-server:8.1.0-es7
docker pull apache/skywalking-ui:8.1.0
docker pull busybox:1.30
```

安装helm3

```
下载地址
https://github.com/helm/helm/releases

本教程下载helm版本为3.3.0
helm-v3.3.0-linux-amd64.tar.gz

tar zxf helm-v3.3.0-linux-amd64.tar.gz

mv linux-amd64/helm /usr/local/bin
chmod 777 /usr/local/bin/helm

helm version

增加repo信息
helm repo add elastic https://helm.elastic.co

```

```
查看安装仓库列表
helm repo list
```

下载项目

```
最新版本：
下载skywalking在k8s环境上部署的项目:git clone项目的master版本。容易安装失败。不推荐使用
git clone https://github.com/apache/skywalking-kubernetes

推荐版本：
下载skywalking在k8s环境上部署项目的最新版本
https://github.com/apache/skywalking-kubernetes/releases/tag/v3.1.0
```

修改配置文件

oap-svc.yml

由于Apache SkyWalking Kubernetes默认的安装方式是采用的ClusterIP,我们需要改为NodePort方式。在skywalking-kubernetes/chart/skywalking/templates下找到oap-svc.yaml文件，修改其中的配置，添加targetPort和nodePort。如果仅仅用于安装演示skywalking。本步骤可以跳过不执行。

```yaml
apiVersion: v1
kind: Service
metadata:
  name: {{ template "skywalking.oap.fullname" . }}
  labels:
    app: {{ template "skywalking.name" . }}
    chart: {{ .Chart.Name }}-{{ .Chart.Version }}
    component: "{{ .Values.oap.name }}"
    heritage: {{ .Release.Service }}
    release: {{ .Release.Name }}
spec:
  type: {{ .Values.oap.service.type }}
  ports:
  - port: {{ .Values.oap.ports.rest }}
    targetPort: {{ .Values.oap.ports.rest }}  #新添加
    nodePort: 31800  #固定端口，也可以通过helm3模板方式配置
    name: rest
  - port: {{ .Values.oap.ports.grpc }}
    targetPort: {{ .Values.oap.ports.grpc }} #新添加
    name: grpc
    nodePort: 31801  #固定端口，也可以通过helm3模板方式配置
  selector:
    app: {{ template "skywalking.name" . }}
    component: "{{ .Values.oap.name }}"
    release: {{ .Release.Name }}
```

values.yaml

在skywalking-kubernetes/chart/skywalking下找到values.yaml文件

- 将oap.service.type改为NodePort
- 将ui.service.type改为NodePort
- 新增ui.service.nodePort端口固定为31080
- 将所有replicas副本数为1
- 将elasticsearch.image改为：elasticsearch

```
UI关键代码：
ui:
  name: ui
  replicas: 1
...
  service:
    type: NodePort
    ...
    nodePort: 31080
```

启动skywalking

```
helm install -f values.yaml skywalking .
```

window系统安装

`skywalking`是一个开源的观测平台, 用于从服务和云原生基础设施收集, 分析, 聚合以及可视化数据.。生产环境下该平台是安装在`linux`系统上，或者是docker容器化运行。但如果要在本地开发的话可能免不了要在windows系统下安装。

安装须知

1. 安装之前请确保被监控的服务器上的系统时间和`OAP`服务器上的系统时间是相同的。
2. `JDK 8`
3. 本教程只适合运行`Skywalking`的`Backend`和`UI`来进行预览或演示，可能并不适合长期部署使用。如果需要在生产环境使用，请参考下边三个官方文档，进行进一步设置：

```
1.Backend setup document:
https://github.com/apache/skywalking/blob/v7.0.0/docs/en/setup/backend/backend-setup.md

2.UI setup document:
https://github.com/apache/skywalking/blob/v7.0.0/docs/en/setup/backend/ui-setup.md

3.CLI set up document:
https://github.com/apache/skywalking-cli
```

安装oap

windows系统安装skywalking特别简单。解压apache-skywalking-apm-8.1.0.tar.gz，进入bin目录，在cmd窗口执行命令

```
快速运行
win+r

打开DOS窗口
cmd

运行 startup.bat，启动skywalking
startup.bat
```

访问ui

测试的windows系统在虚拟机中。windows系统默认防火墙不支持远程访问。在虚拟机的浏览器端进行访问测试：

```
http://localhost:8080
```

spring-boot实例部署

springboot项目

application.yml

springboot项目tomcat服务器默认8080端口，与skywalking-UI端口冲突，默认修改application.yml的项目端口号

```
server:
  port: 8088
```

SkywalkingDemoController

controller/SkywalkingDemoController

```
@RestController
public class SkywalkingDemoController {
    @GetMapping("/")
    public String hello() {
        return "hello skywalking!!!";
    }

    @GetMapping("skywalkinghello")
    public String hello1() {
        return "test1";
    }
}
```

运行skywalking-agent.jar包

修改项目的 VM 运行参数，点击菜单栏中的 `Run` -> `EditConfigurations...`，此处我们以 `skywalkingdemo1` 项目为例，修改参数如下：

```
-javaagent:D:\skywalking\agent\skywalking-agent.jar -
Dskywalking.agent.service_name=skywalkingdemo1 -
Dskywalking.collector.backend_service=localhost:11800
```

启动skywalkingdemo1项目

```
http://192.168.198.199:8088/
http://192.168.198.199:8088/skywalkinghello


http://192.168.198.199:8088/
```

优化skywalking

修改启动端口

修改D:\skywalking\webapp\webapp.yml 。 主要修改port: 18080

```
server:
  port: 18080

collector:
  path: /graphql
  ribbon:
    ReadTimeout: 10000
    # Point to all backend's restHost:restPort, split by ,
    listOfServers: 127.0.0.1:12800
```

更改数据库为mysql

1. 下载mysql-connector-java-5.1.47.jar到oap-libs下。只要是mysql-connector-java-5的版本就可以。
2. 打开D:\skywalking\config\application.yml。 注释H2，开启mysql 。
3. 到Mysql中创建一个 swtest数据库。

```
1.更改storage.selector: ${SW_STORAGE:mysql}为mysql数据库。
2.dataSource.user: ${SW_DATA_SOURCE_USER:root}，更新为自己的mysql数据库用户名
3.dataSource.password: ${SW_DATA_SOURCE_PASSWORD:root}，更新为自己的mysql数据库
用户的密码


storage:
  selector: ${SW_STORAGE:mysql}
#  h2:
#    driver: ${SW_STORAGE_H2_DRIVER:org.h2.jdbcx.JdbcDataSource}
#    url: ${SW_STORAGE_H2_URL:jdbc:h2:mem:skywalking-oap-db}
#    user: ${SW_STORAGE_H2_USER:sa}
#    metadataQueryMaxSize: ${SW_STORAGE_H2_QUERY_MAX_SIZE:5000}
  mysql:
```

```
    properties:
      jdbcUrl: ${SW_JDBC_URL:"jdbc:mysql://localhost:3306/swtest"}
      dataSource.user: ${SW_DATA_SOURCE_USER:root}
      dataSource.password: ${SW_DATA_SOURCE_PASSWORD:admin}
      dataSource.cachePrepStmts: ${SW_DATA_SOURCE_CACHE_PREP_STMTS:true}
      dataSource.prepStmtCacheSize:
${SW_DATA_SOURCE_PREP_STMT_CACHE_SQL_SIZE:250}
      dataSource.prepStmtCacheSqlLimit:
${SW_DATA_SOURCE_PREP_STMT_CACHE_SQL_LIMIT:2048}
      dataSource.useServerPrepStmts:
${SW_DATA_SOURCE_USE_SERVER_PREP_STMTS:true}
      dataSource.useSSL: false
    metadataQueryMaxSize: ${SW_STORAGE_MYSQL_QUERY_MAX_SIZE:5000}
```