21天微服务实战营-Day4

华为云DevCloud & ServiceStage服务联合出品

Day4 微服务实例的生命周期分析

大纲

- 服务启动流程
- 服务发现
- 服务退出

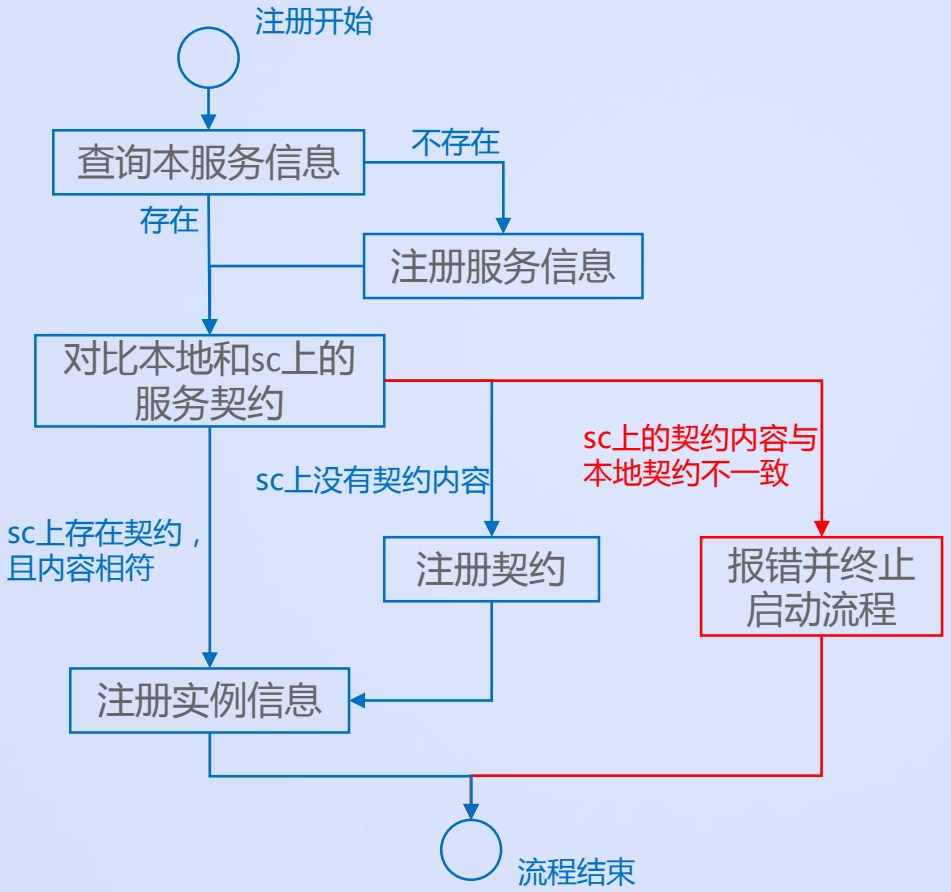
服务启动流程

一个微服务实例在启动过程中主要经历的流程有：

- 初始化日志框架
- 加载本地配置（包括System Property、环境变量、配置文件）
- 实例化Spring Bean
- 初始化SCBEngine
- 注册服务

服务启动流程

注册服务的流程如下：



- 当env为默认环境或production环境时，不允许出现服务实例本地与sc上的服务契约不一致的情况（一旦契约不一致会走左图中的红色路径）
- env=development时，服务实例会将不一致的接口契约注册到sc，覆盖sc上原有的契约（如下图）

```
APPLICATION_ID: HelloWorld
service_description:
  name: provider
  version: 0.0.1
  environment: development
```

服务启动流程

当微服务实例注册到sc上去后，启动流程完成。如果有一些操作需要在服务启动完成时执行，可以定义一个org.apache.servicecomb.core.BootListener去监听事件，并在接收到AFTER_REGISTRY事件时触发操作的执行。

```java
@Component
public class CustomBootEventListener implements BootListener {

  private static final Logger LOGGER = LoggerFactory.getLogger(CustomBootEventListener.class);

  public void onBootEvent(BootEvent bootEvent) {
    if (!EventType.AFTER_REGISTRY.equals(bootEvent.getEventType())) {
      return;
    }

    LOGGER.info("============================");
    LOGGER.info("Service startup completed!");
    LOGGER.info("============================");
  }
}
```

```
receive MicroserviceInstanceRegisterTask event, check instance Id... org.apache.servicecomb.core.SCBEngine$1.afterRegistryInstance(SCBEngine.java:182)
instance registry succeeds for the first time, will send AFTER_REGISTRY event. org.apache.servicecomb.core.SCBEngine$1.afterRegistryInstance(SCBEngine.java:184)
============================ microservice.demo.training21days.provider.bootevent.CustomBootEventListener.onBootEvent(CustomBootEventListener.java:18)
Service startup completed! microservice.demo.training21days.provider.bootevent.CustomBootEventListener.onBootEvent(CustomBootEventListener.java:19)
============================ microservice.demo.training21days.provider.bootevent.CustomBootEventListener.onBootEvent(CustomBootEventListener.java:20)
```

服务发现

Consumer服务调用provider服务时，需要去服务中心查询provider服务的契约、实例列表等信息，然后才能对provider发起调用：

- 查询条件包括AppID、serviceName、environment、versionRule，如果碰到consumer端找不到provider服务的问题，除了检查provider服务的实例有没有注册到sc，还需要检查这四个配置项是否有问题
- 查询到provider端服务信息后，consumer会从sc下载该provider服务的全部契约，加载到本地
- Consumer端加载provider服务信息的过程发生在consumer第一次调用该provider的时候，如果consumer服务的实例在启动后一直没有调用provider，则它一直不会去sc查询和加载provider服务信息

服务退出

CSEJavaSDK向JVM注册了一个shutdown hook，以实现优雅停机，在JVM进程退出时进行一系列的清理操作，其中包括：

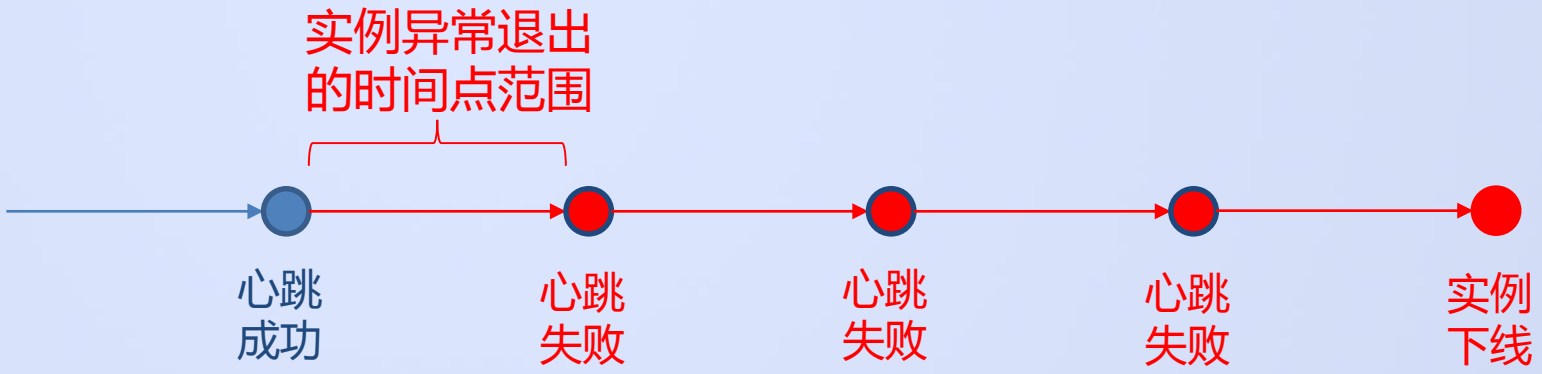
- 向服务中心注销本实例
- 停止接收请求，并等待已接受的请求处理完成

由于JVM的限制，要确保优雅停机功能正常触发，需要用户正常停止JVM进程，而不能强制杀进程。以Linux操作系统为例：

- kill ${PID} 的方式停止微服务进程可以触发优雅停机
- kill –0 ${PID} 的方式强制停止微服务进程则不会触发优雅停机

服务退出

如果微服务实例遭遇异常情况，没有调用sc接口注销自身实例就停止运行。sc会通过感知心跳超时的方式下线实例。前面的课程中提到微服务连接sc的配置中有心跳时间间隔和允许连续心跳失败次数这两个配置，假设心跳时间间隔为t，允许心跳失败次数为n，则sc检测到实例连续心跳失败n+1次的时候下线实例，从实例异常退出到sc下线实例的时延T的取值范围是 t*n < T < t*(n+1) ，按照默认值计算为90-120秒



Thank You