Title: Applications not registering to eureka when using docker-compose

Post Body:

I took this example https://github.com/paulc4/microservices-demo and I created 3 docker images from it, with the following Dockerfiles:

springdocker-registration:

```
FROM openjdk:8-jdk-alpine VOLUME /tmp ADD target/microservice-demo-1.1.0.RELEASE.jar app.jar EXPOSE 1111 ENTRYPOINT exec java
```

springdocker-accounts:

```
FROM openjdk:8-jdk-alpine VOLUME /tmp ADD target/microservice-demo-1.1.0.RELEASE.jar app.jar EXPOSE 2222 ENTRYPOINT exec java
```

springdocker-web:

```
FROM openjdk:8-jdk-alpine VOLUME /tmp ADD target/microservice-demo-1.1.0.RELEASE.jar app.jar EXPOSE 3333 ENTRYPOINT exec java
```

If I run the three images separately everything works ok, the `web` and `accounts` services register to the `registration` service (which is an implementation of the eureka registry) and I can use my application. However when using `docker-compose` with the following `docker-compose.yml` file

```
version: '3.4' services:  registration:   image: springdocker-registration   ports:    - '1111:1111'   accounts:   image: spri
```

the services `web` and `accounts` are not able to register to the `registration` service. Here are the configuration files for the applications:

registration-server.yml:

```
eureka:   instance:     hostname: localhost   client:       registerWithEureka: false     fetchRegistry: false     serviceUrl:
```

accounts-server.yml:

```
spring:   application:      name: accounts-service     freemarker:     enabled: false             thymeleaf:     cache: false
```

web-server.yml

```
spring:   application:     name: web-service    freemarker:     enabled: false        thymeleaf:     cache: false            p
```

I can post the full console log of `docker-compose up` but I think this is the interesting point:

```
1: ERROR RedirectingEurekaHttpClient - Request execution error com.sun.jersey.api.client.ClientHandlerException: java.net.Conn
```

Accepted Answer:

Since its running in docker, don't use `localhost`. Docker compose lets you refer to container names.

```
eureka:   client:     serviceUrl:       defaultZone: http://registration:1111/eureka
```

Side note: `defaultZone` must be exact, I spent 2 days wondering why it wouldn't work since intellij auto completes to do `default-zone` which wont work.

Highest Rated Answer:

For those who are working with docker-compose and if in your services yml file you have the following:

```
eureka:   client:     serviceUrl:       # Will get overridden in docker-compose       defaultZone: http://localhost:1111/eurek
```

You can override this in your docker-compose file like below while running `docker-compose up`

```
version: '3.3' services:   # Other services registered above   web:   image: springdocker-web   environment:     # Important f
```

Tested on Docker version 19.03.8