Title: Environment variables and @Value can't work together on Spring Boot

Post Body:

I have a Spring boot app that connects to a Redis instance that works as a cache. When I'm in dev environment, I have the following:

```
---  spring:   profiles: default redis:    host: localhost   port: 6379
```

And my cache configuration class is like this:

```
@Configuration @EnableCaching public class CacheConfiguration {     @Value('${redis.host}')     String redisHost;     @Value('
```

In production, this app is Dockerized, and I have the following `docker-compose.yml` file:

```
redis:    image: tutum/redis   ports:     - '6379:6379'   volumes:     - /data app:    build: .   ports:     - '8080:8080'   l
```

And the `application.yml` is:

```
---  spring:   profiles: docker redis:    host: redis   port: 6379
```

To start the app on Docker, I run with `-Dspring.profiles.active=docker`, but when the app is starting up, the following error happens:

```
Caused by: org.springframework.beans.factory.BeanCreationException: Could not autowire field: private int com.inkdrop.config.c
```

For some reason, Spring Boot is reading the `redis.port` as `tcp://172.17.0.3:6379`. So for tests proposes, I removed the `@Value` annotations from `CacheConfiguration` class, and set it manually to `redis` as host and `6379` as port and it worked. Seems like when using environment variables and `@Value`, Spring get lost. Anyone have an idea?

Accepted Answer:

Based on the Docker documentation:

> Compose uses Docker links to expose services containers to one another. Each linked container injects a set of environment variables, **each of which begins with the uppercase name of the container**.

Docker Compose would create an Environment Variable representing the *Full URL* of the container using ***name_PORT*** format, e.g. `REDIS_PORT=tcp://172.17.0.5:6379`.

And based on your `docker-compose.yml` file:

```
redis:    image: tutum/redis   ports:     - '6379:6379'   volumes:     - /data
```

You would have an Environment Variable named `REDIS_PORT` with a value equals to `tcp://172.17.0.3:6379`.Since OS environment variables have more precedence with respect to Profile-specific application properties, Spring Boot would pick up the `REDIS_PORT` value over `redis.port`, hence the error:

> Caused by: org.springframework.beans.factory.BeanCreationException: Could not autowire field: private int com.inkdrop.config.cache.CacheConfiguration.redisPort; nested exception is org.springframework.beans.TypeMismatchException: Failed to convert value of type [java.lang.String] to required type [int]; nested exception is java.lang.NumberFormatException: For input string: 'tcp://172.17.0.3:6379'

As a workaround for this problem, you either should override the `REDIS_PORT` environment variable with your port value or rename your config name from `redis.name` to anything less controversial.

Kinda off topic but just quoting from `tutum-docker-redis` Github repository:

> This image will be deprecated soon. Please use the docker official image: https://hub.docker.com/_/redis/

Highest Rated Answer: None