Title: Spring Boot And Multi-Module Maven Projects

Post Body:

I am struggling to get a multi-module Spring Boot Maven project setup correctly. Are there any example projects that I can reference? Currently I have my project structured like this.

Data Module: Basically my data layer. Is contains my POJOs and my database repo interfaces (PagingAndSortingRepository). This module will be a dependency of the other modules in the project. Currently I have placed a Config class in the module that looks like this

```
public class Config {    @Configuration   @Profile('cloud')   static class CloudConfiguration extends AbstractCloudConfig {
```

I figured this configuration would be common between the other two modules so it belonged in the data module.

Text Module: The is a very simple module, it contains a single REST API controller that will be called when a text message is sent to a certain phone number. It stores the text message in the DB and uses one of the repo interfaces from the cdata module to do so. This module is being built as an executable jar file and contains a class which implements EmbeddedServletContainerCustomizer.

```
@Configuration @ComponentScan @EnableAutoConfiguration @EnableJpaRepositories public class App implements EmbeddedServletConta
```

When I run the jar I get errors saying that rest controller class cannot autowire beans from my data module. I saw some posts saying that you should add the package name to the @ComponentScan annotation. For example

@ComponentScan('com.example.data')

Doing this will let the embedded tomcat server start, but I think adding that package alone causes Spring to not find my REST controller in my text module because I am getting 404s when hitting the API. So I added my text package as well

@ComponentScan({'com.example.data','com.example.text'})

However this brings me back to the same error, Spring cannot find my beans from the data module to autowire to my REST controller.

```
java.lang.reflect.InvocationTargetException     at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)     at sun.refl
```

Does anyone have any pointers on how to do this properly?

Accepted Answer:

The repositories are scanned automatically from the package of your application class (the one annotated with `@EnableAutoConfiguration`). If that default does not suit you, you can easily fallback on `@EnableJpaRepositories` with the relevant packages.

I can see `com.example.data` and `com.example.text`. I guess you might have a project specific package as `com.example` is probably too broad. So one way to fix that would be to put your application in `com.example` (or whatever the root package of your app is).

Check also the doc

Highest Rated Answer:

Verify beans annotation which demands to have a parent annotation in Application class, like @Transactional - parent: @EnableTransactionManagement.

There are some of them that provide AOP and proxy objects and provokes beans injection lifecycle failures if they are wrong or missing. Happened to me with JPA & REST multimodule project (e.g thought was a multi-module component scan failure).

Beans Resolution Hierarchy

@Controller <- @Service <- @Component (Repository/Aggregation Facade) <- @Repository