Title: Are spring-boot command line properties available when using spring-boot:run?

Post Body:

The short question:

How can you configure environment profiles, or even just an alternative configuration file name for spring-boot:run?

The long version:

Yes, I read the doc. http://docs.spring.io/spring-boot/docs/current/reference/html/index.html

I have my application configuration settings in src/main/resources/application-mysql.properties:

```
# Specify the DBMS spring.jpa.database = MYSQL  # Other Mysql config properties  spring.jpa.hibernate.ddl-auto=create
```

There is a corresponding application-hsql.properties, which contains the same set of configuration options for Hsql. There is no application.poperties

There is a corresponding import.sql:

```
insert into users(name, email) values ('User One',  'one@email.com') insert into users(name, email) values ('Two User', 'two@e
```

The unit tests exist only to check for the presence of these users in the repo.

I believe it to be true that whenever the test is run using the MySql configuration, those rows are added to the users table. When the test is run with hsql, the mysql db should be unaffected. I am manually dropping the users table between runs, because I want to manually see when it exists.

1) If I run mvn test, the tests use the configured db:

```
mvn -Dspring.profiles.active=mysql clean test
```

Produces mysql database rows and

```
mvn -Dspring.profiles.active=hsql clean test
```

Does not.

2) If I make a package, and then run the resulting jar file, I am able to specify a config file name:

```
java -jar -Dspring.profiles.active=mysql  ./target/app.jar
```

3) If I run with spring-boot:run, only properties in application.properties (which doesn't exist in this test scenario) are discovered.

```
mvn -Dspring.profiles.active=mysql clean spring-boot:run
```

What does spring-boot:run do differently in launching than running unit tests and kicking off the jar? The db config is one example, but in theory I'd like to be able to specify a set of dev configs when the application is being run locally vs. a production configuration.

Accepted Answer:

The Maven `spring-boot` plugin forks a new process so you'll need to send any extra parameters to it via jvmArguments, e.g.:

```
mvn spring-boot:run -Drun.jvmArguments='-Dspring.profiles.active=mysql'
```

Highest Rated Answer:

The plugin allows you to specify the active profile(s) directly (perhaps it wasn't the case 3 year ago?) :

```
mvn spring-boot:run -Drun.profiles=mysql
```