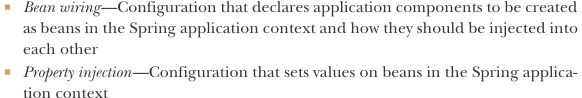
6. Working with configuration properties

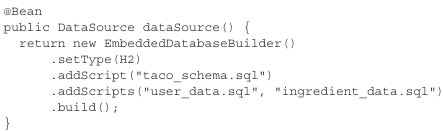
# 6.1 Fine-tunning autoconfiguration

-Kinds of configuration in Spring:



In Spring’s XML and Java configuration, these 2 types of configurations are often declared explicitly in the same place.

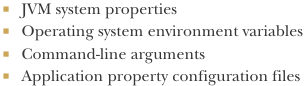
-In Java configuration, a @Bean method instantiate a bean and then set values to its properties:

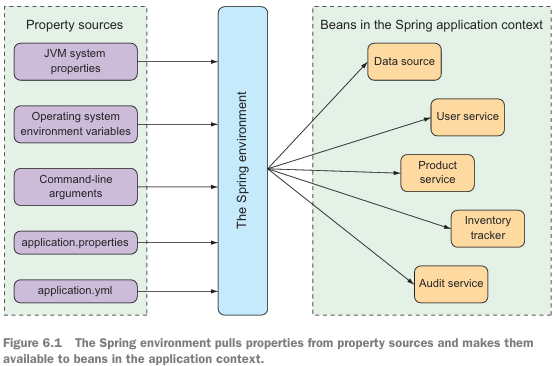


+Autoconfiguration makes this method unnecessary. If H2 dependency is available in runtime classpath, SB automatically creates in Spring application context an DataSource bean.

## 6.1.1 Understanding Spring’s environment abstraction

-Spring environment abstraction is one-stop shop for any configurable property. It abstracts the origins of properties so that beans needing those properties can consume them from Spring. Spring environment pulls from several property sources + aggregates them into a single source from which Spring beans can be injected.





-The beans that are automatically configured by SB are all configurable by properties drawn from Spring environment.

+Example: change default port

+Use command-line argument:

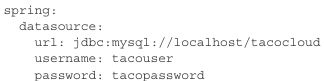


+Set it one time as an OS environment variable:

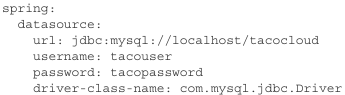


## 6.1.2 Configuring a data source

-Configure URL and credentials for database:



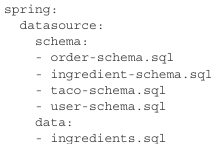
+If there’s a problem, set driver class:



+SB uses this connection data when autoconfiguring DataSource bean. This bean will be pooled using HikariCP connection pool if it’s available on classpath. If not, SB looks for and uses one of 2 other implementations:



-Specify the database initialization scripts to run:



+Instead, configure data source in Java Naming and Directory Interface JNDI (<http://mng.bz/MvEo>).



## 6.1.3 Configuring the embedded server

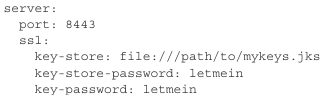
-Set the servlet container’s port by server.port. Start randomly chosen available port: 

-One of the most common things to do with underlying container is to set it up to handle HTTPS requests:  
+Create a keystore using JDK’s keytool command-line:



You’ll be asked several questions (name, orga, pass)

+Set properties to enable HTTPS on command-line or application.yml:



## 6.1.4 Configuring logging

## 6.1.5 Using special property values

# 6.2 Creating your own configuration properties

## 6.2.1 Defining configuration property holders

## 6.2.2 Declaring configuration property metadata

# 6.3 Configuring with profiles

## 6.3.1 Defining profile-specific properties

## 6.3.2 Activating profiles

## 6.3.3 Conditionally creating beans with profiles

# -Summary