1.pom文件配置：引入springboot依赖，有两种方式。第一种如下图：

```
<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>1.4.0.RELEASE</version>
</parent>
```

此种方式适合单mould的场景下，也是最简单的springboot的pom文件配置。

多Mould的情况就要使用第二种配置方式，如下图：

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-dependencies</artifactId>
    <version>1.4.0.RELEASE</version>
    <type>pom</type>
    <scope>import</scope>
</dependency>
```

在父pom中声明依赖，在子pom引入具体依赖使用。

2.spring boot依赖:

1. spring-boot-starter-actuator依赖会引入一组基本的spring项目。从而实现项目的快速配置和即时即用。
2. spring-boot-starter-web依赖会为你提供启动嵌入式Tomcat容器的自动化配置，并且提供对微服务有应用价值的端点信息，如服务器信息，应用指标以及环境详情。

3. 引入spring-boot-starter-security依赖，actuator会自动配置spring security，从而为应用提供基本的认证以及其他高级的安全特性。

4. 他还会为应用结构引入一个内部的审计框架，这个框架可以用来生成报告或者其他的用途，比如开发认证失败的锁定策略。

5. 当前不可避免的我们还要引入其他的一些依赖包

1).spring boot mybatis依赖包

```
<dependency>
    <groupId>org.mybatis.spring.boot</groupId>
    <artifactId>mybatis-spring-boot-starter</artifactId>
    <version>${mybatis-spring-boot}</version>
</dependency>
```

2).spring boot jdbc依赖包

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
    <version>${spring.boot.version}</version>
</dependency>
```

3).spring boot mysql依赖包

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>${mysql-connector}</version>
</dependency>
```

3.Spring Boot 启动类

Spring Boot 内嵌了tomcat，所以启动springboot内嵌的tomcat需要配置启动类启动tomcat。创建一个java类来作为springboot的启动类。此类需要使用注解

@SpringBootApplication，此注解包含了@Configuration，@EnableAutoConfiguration，@ComponentScan.

1).@Configuration：说到@Configuration就要提到@Bean，使用这两个注解就可以创建一个简单的spring配置类，可以用来代替相应的xml配置文件。

```html
<beans>
    <bean id = "car" class="com.test.Car">
        <property name="wheel" ref = "wheel"></property>
    </bean>
    <bean id = "wheel" class="com.test.Wheel"></bean>
</beans>
```

```java
@Configuration
public class Conf {
    @Bean
    public Car car() {
        Car car = new Car();
        car.setWheel(wheel());
        return car;
    }
    @Bean
    public Wheel wheel() {
        return new Wheel();
    }
}
```

@Configuration的注解类标识这个类可以使用Spring Ioc容器作为bean定义的来源，@Bean注解告诉Spring，一个带有@Bean的注解方法将返回一个对象，该对象应该被注册为在Spring应用程序的上下文中的Bean。

2).@EnableAutoConfiguration：能够自动配置Spring的上下文，试图猜测和配置你想要的Bean类，通常会自动根据你的类路径和你的Bean定义自动配置。

3).@ComponentScan：会自动扫描指定包下的全部标有@Component的类，并注册成Bean，当然包括@Component的子注解@Service，@Controller，@Repository。

本类中存在一个main方法，main方法调用SpringApplication.run来启动容器，还没了解run做了哪些操作，后续查看。

4.更改内嵌tomcat的端口：

启动类实现EmbeddedServletContainerCustomizer接口，有customize方法，setPost(post)，即可更改端口。

搭建问题总结：

1.Springboot默认加载sqlSessionFactory，所以在开始搭建不适用mybatis是加注解：

@EnableAutoConfiguration(exclude={DataSourceAutoConfiguration.class})