12、自定义starter

2020年6月18日 16:39

starter：

- 1、这个场景需要使用到的依赖是什么？
- 2、如何编写自动配置

  ```
  @Configuration  //指定这个类是一个配置类
  @ConditionalOnXXX  //在指定条件成立的情况下自动配置类生效
  @AutoConfigureAfter  //指定自动配置类的顺序
  @Bean  //给容器中添加组件

  @ConfigurationPropertie结合相关xxxProperties类来绑定相关的配置
  @EnableConfigurationProperties //让xxxProperties生效加入到容器中
  ```

  自动配置类要能加载

  将需要启动就加载的自动配置类，配置在META-INF/spring.factories

  ```
  org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
  org.springframework.boot.autoconfigure.admin.SpringApplicationAdminJmxAutoConfiguration,\
  org.springframework.boot.autoconfigure.aop.AopAutoConfiguration,\
  ```

- 3、模式：

  启动器只用来做依赖导入；

  专门来写一个自动配置模块；

  启动器依赖自动配置；别人只需要引入启动器（starter）

  mybatis-spring-boot-starter；自定义启动器名-spring-boot-starter

  步骤：

  - 1）启动器模块

    ```
    <?xmlversion="1.0"encoding="UTF-8"?>
    <projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.lhq</groupId>
    <artifactId>lhq-spring-boot-starter</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <!--启动器-->
    <dependencies>
    <!--引入自动配置模块-->
        <dependency>
            <groupId>com.lhq</groupId>
            <artifactId>lhq-spring-boot-starter-autoconfigurer</artifactId>
            <version>0.0.1-SNAPSHOT</version>
        </dependency>
    </dependencies>
    </project>
    ```

  - 2）自动配置模块

    ```
    <?xmlversion="1.0"encoding="UTF-8"?>
    ```

```
<projectxmlns="http://maven.apache.org/POM/4.0.0"xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0https://maven.apache.org/xsd/maven-4.0.0.xsd">
<modelVersion>4.0.0</modelVersion>
<parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.3.0.RELEASE</version>
        <relativePath/><!--lookupparentfromrepository-->
</parent>
<groupId>com.lhq</groupId>
<artifactId>lhq-spring-boot-starter-autoconfigurer</artifactId>
<version>0.0.1-SNAPSHOT</version>
<name>lhq-spring-boot-starter-autoconfigurer</name>
<description>DemoprojectforSpringBoot</description>
<properties>
        <java.version>1.8</java.version>
</properties>
<!--引入spring-boot-starter-->
<dependencies>
        <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter</artifactId>
        </dependency>
</dependencies>
</project>
```

```
package com.lhq.lhqspringboot;
import org.springframework.boot.context.properties.ConfigurationProperties;
@ConfigurationProperties(prefix = "lhq.hello")
public class HelloProperties {
    private String prefix;
    private String suffix;

    public String getPrefix() {
        return prefix;
    }
    public void setPrefix(String prefix) {
        this.prefix = prefix;
    }
    public String getSuffix() {
        return suffix;
    }
    public void setSuffix(String suffix) {
        this.suffix = suffix;
    }
}
```

```
package com.lhq.lhqspringboot;
public class HelloService {
    HelloProperties helloProperties;
    public HelloProperties getHelloProperties() {
        return helloProperties;
    }
    public void setHelloProperties(HelloProperties helloProperties) {
        this.helloProperties = helloProperties;
    }
    public String sayHelloLhq(String name){
       return helloProperties.getPrefix()+"-"+name +
helloProperties.getSuffix();
    }
}

package com.lhq.lhqspringboot;
import org.springframework.beans.factory.annotation.Autowired;
import
org.springframework.boot.autoconfigure.condition.ConditionalOnWeb
Application;
import
org.springframework.boot.context.properties.EnableConfigurationPrope
rties;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
@Configuration
@ConditionalOnWebApplication
@EnableConfigurationProperties(HelloProperties.class)
public class HelloServiceAutoConfiguration {
    @Autowired
    HelloProperties helloProperties;
    @Bean
    public HelloService helloService(){
        HelloService service = new HelloService();
        service.setHelloProperties(helloProperties);
        return service;
    }
}
```