11、SpringBoot与数据访问

2020年6月18日　　16:13

- 1、JDBC

```
<dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
<dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <scope>runtime</scope>
</dependency>
spring:
        datasource:
        username:root
        password:root
        url:jdbc:mysql://192.168.183.101/jdbc?characterEncoding=utf-8
        &serverTimezone=UTC
        driver-class-name:com.mysql.cj.jdbc.Driver
#直接加载sql脚本
initialization-mode:always
schema:
        -classpath:department.sql
```

  schema-*.sql、data-*.sql
  默认规则：schema.sql，schema-all.sql；
  可以使用

```
        schema:
    - classpath:department.sql
```

  指定位置

- 2、整合Druid数据源

```
@Configuration
public class DruidConfig {
   @ConfigurationProperties(prefix = "spring.datasource")
   @Bean
   public DataSource druid(){
      return new DruidDataSource();
   }
   //配置Druid监控
   //1、配置一个管理后台的Servlet
   @Bean
   public ServletRegistrationBean statViewServlet(){
      ServletRegistrationBean<StatViewServlet> bean = new
ServletRegistrationBean<>(new StatViewServlet(), "/druid/*");
      Map<String,String> initParams = new HashMap<>();
      initParams.put("loginUsername","admin");
      initParams.put("loginPassword","123456");
      initParams.put("allow","");//默认允许所有
      initParams.put("deny","192.168.31.63");
      bean.setInitParameters(initParams);
      return bean;
```

```
    }
    //2、配置一个web监控的filter
    @Bean
    public FilterRegistrationBean webStatFilter(){
        FilterRegistrationBean<Filter> bean = new FilterRegistrationBean<>();
        bean.setFilter(new WebStatFilter());
        Map<String,String> initParams = new HashMap<>();
        initParams.put("exclusions","*.js,*.css,/druid/*");
        bean.setInitParameters(initParams);
        bean.setUrlPatterns(Arrays.asList("/*"));
        return bean;
    }
}
```

- 3、整合MyBatis

  ```
  <dependency>
          <groupId>org.mybatis.spring.boot</groupId>
          <artifactId>mybatis-spring-boot-starter</artifactId>
          <version>2.1.2</version>
  </dependency>
  ```

  - 步骤：
    - 1）、配置数据源（见2、Druid）
    - 2）、给数据库建表
    - 3）、创建JavaBean
    - 4）、注解版

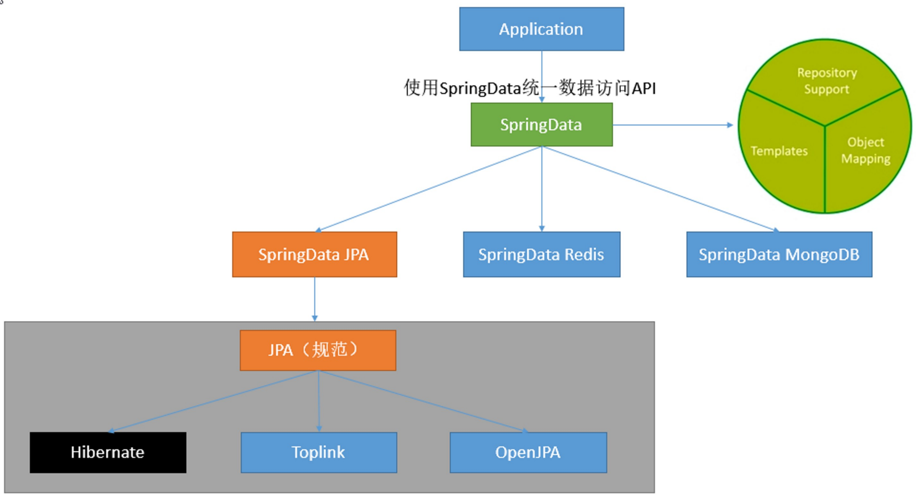
      ```
      //指定这是一个操作数据库的mapper
      //@Mapper
      public interface DepartmentMapper {
          @Select("select * from department where id=#{id}")
          public Department getDeptById(Integer id);
          @Delete("delete from department where id=#{id}")
          public int deleteDeptId(Integer id);
          @Options(useGeneratedKeys = true,keyProperty = "id")
          @Insert("insert into department(departmentName)
      values(#{departmentName})")
          public int insertDept(Department department);
          @Update("update department set
      departmentName=#{departmentName} where id=#{id}")
          public int updDept(Department department);
          @Select("select * from department")
          public List<Department> getAllDepartments();
      }
      ```

    - 5）、配置文件版

      ```
      mybatis:
              configuration:
              map-underscore-to-camel-case:true
              指定全局配置文件的位置
              #config-location:classpath:mybatis/mybatis-config.xml
              指定sql映射文件的位置
              mapper-locations:classpath:mybatis/mapper/*.xml
      ```

- 4、整合SpringData JPA

- 1）、SpringData简介



- 2）、整合SpringData JPA

  JPA:ORM（Object Relational Mapping）；

  - 1、编写一个实体类（bean）和数据表进行映射，并且配置好映射关系；

```
//使用jpa注解配置映射关系
@Entity//告诉jpa这是一个实体类（和数据表映射的类）
@Table(name = "tbl_user")//@Table来指定哪个数据表对应；如果省
略默认表名就是user；
public class User {
   @Id//这是一个主键
   @GeneratedValue(strategy = GenerationType.IDENTITY)//自增主
键
   private Integer id;
   @Column(name = "last_name",length = 50)//这是和数据表对应的
一个列
   private String lastName;
   @Column//省略默认列名就是属性名
   private String email;
  }
```

  - 2、编写一个Dao接口来操作实体类对应的数据表（Repository）

```
//继承jpaRepository来完成对数据库操作
public interface UserRepository extends
JpaRepository<User,Integer> {
}
```

  - 3、基本配置JpaProperties

```
jpa:
      hibernate:
            #更新或者创建数据库表结构
            ddl-auto:update
      #控制台显示SQL
      show-sql:true
```