概述

spring-boot默认提供了数据库和数据库连接池，按照官方文档简单配置即可。若要自定义，需要修改一些配置，本文着重描述一下spring-boot如何集成mysql和阿里的druid数据库连接池。

开始

本文环境

- jdk：1.7
- tomcat：7.0.55
- spring-boot：1.2.3.RELEASE

修改application.properties文件

修改spring-boot默认配置文件 `application.properties` ，加入一下内容(根据实际情况修改)：

```
# 数据库配置
spring.mysql.datasource.driverClassName=com.mysql.jdbc.Driver
spring.mysql.datasource.url
=jdbc:mysql://127.0.0.1:3306/message?useUnicode=true&characterEncoding=utf8
spring.mysql.datasource.username=root
spring.mysql.datasource.password=
# 连接池配置
spring.mysql.datasource.filters=stat
spring.mysql.datasource.maxActive=5
spring.mysql.datasource.initialSize=1
spring.mysql.datasource.maxWait=60000
spring.mysql.datasource.minIdle=1
spring.mysql.datasource.maxIdle=3
spring.mysql.datasource.timeBetweenEvictionRunsMillis=60000
spring.mysql.datasource.minEvictableIdleTimeMillis=300000
spring.mysql.datasource.validationQuery=SELECT 'x'
spring.mysql.datasource.testWhileIdle=true
spring.mysql.datasource.testOnBorrow=false
spring.mysql.datasource.testOnReturn=false
spring.mysql.datasource.maxOpenPreparedStatements=10
spring.mysql.datasource.removeAbandoned=true
spring.mysql.datasource.removeAbandonedTimeout=1800
spring.mysql.datasource.logAbandoned=true
```

druid数据库连接池的具体配置请参考官方文档并根据项目的实际情况修改。

新建 `DataSourceConfig.java` 配置类

为数据库配置单独新建一个类，内容如下：

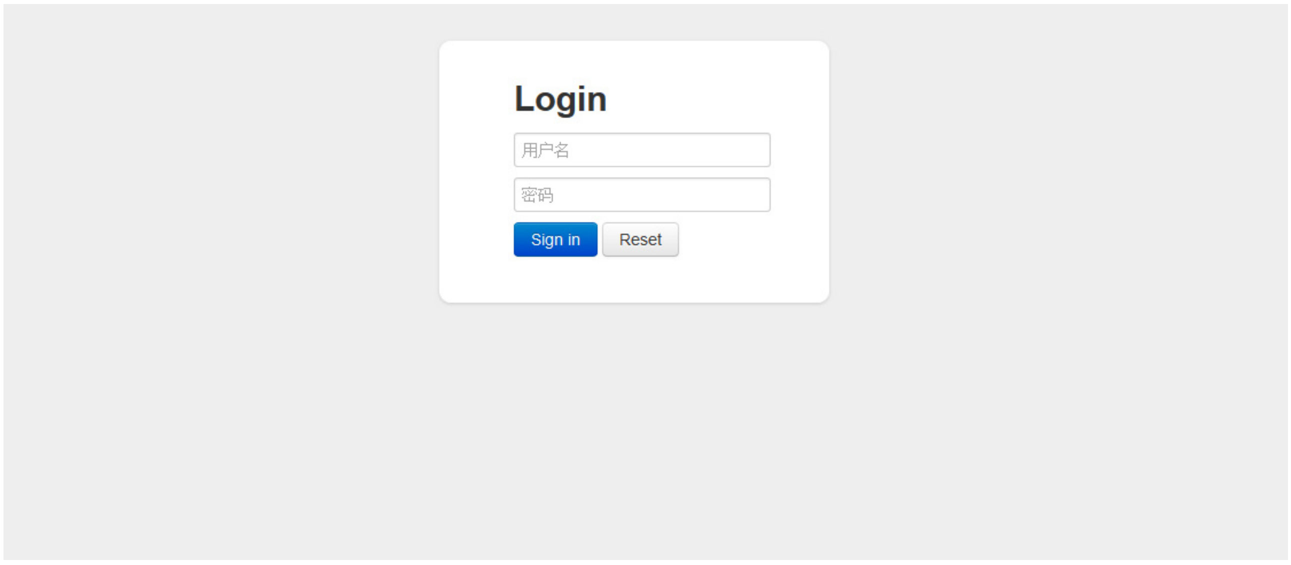
```java
@Configuration
public class DataSourceConfig {

    @Value("${spring.mysql.datasource.driverClassName}")
    private String driverClassName;
    @Value("${spring.mysql.datasource.url}")
    private String url;
    @Value("${spring.mysql.datasource.username}")
    private String username;
    @Value("${spring.mysql.datasource.password}")
    private String password;
    @Value("${spring.mysql.datasource.filters}")
    private String filters;
    @Value("${spring.mysql.datasource.maxActive}")
    private int maxActive;
    @Value("${spring.mysql.datasource.initialSize}")
    private int initialSize;
    @Value("${spring.mysql.datasource.maxWait}")
    private long maxWait;
    @Value("${spring.mysql.datasource.minIdle}")
    private int minIdle;
    @Value("${spring.mysql.datasource.timeBetweenEvictionRunsMillis}")
    private long timeBetweenEvictionRunsMillis;
    @Value("${spring.mysql.datasource.minEvictableIdleTimeMillis}")
    private long minEvictableIdleTimeMillis;
    @Value("${spring.mysql.datasource.validationQuery}")
    private String validationQuery;
    @Value("${spring.mysql.datasource.testWhileIdle}")
    private boolean testWhileIdle;
    @Value("${spring.mysql.datasource.testOnBorrow}")
    private boolean testOnBorrow;
    @Value("${spring.mysql.datasource.testOnReturn}")
    private boolean testOnReturn;
    @Value("${spring.mysql.datasource.removeAbandoned}")
    private boolean removeAbandoned;
    @Value("${spring.mysql.datasource.logAbandoned}")
    private boolean logAbandoned;
    @Value("${spring.mysql.datasource.maxOpenPreparedStatements}")
    private int maxOpenPreparedStatements;
    @Value("${spring.mysql.datasource.removeAbandonedTimeout}")
    private int removeAbandonedTimeout;

    /**
     * druid 数据库连接池
     * @return
     */
    @Bean(name = "mysqlDS")
    @Qualifier("mysqlDS")
    @Primary
    public DataSource dataSource() {
        DruidDataSource dataSource = new DruidDataSource();
        dataSource.setUrl(url);
        dataSource.setUsername(username);
        dataSource.setPassword(password);
        dataSource.setDriverClassName(driverClassName);
        dataSource.setMaxActive(maxActive);
        dataSource.setInitialSize(initialSize);
        dataSource.setMaxWait(maxWait);
        dataSource.setMinIdle(minIdle);
        dataSource
         .setTimeBetweenEvictionRunsMillis(timeBetweenEvictionRunsMillis);
        dataSource.setMinEvictableIdleTimeMillis(minEvictableIdleTimeMillis);
        dataSource.setValidationQuery(validationQuery);
        dataSource.setTestWhileIdle(testWhileIdle);
        dataSource.setTestOnBorrow(testOnBorrow);
        dataSource.setTestOnReturn(testOnReturn);
        dataSource.setMaxOpenPreparedStatements(maxOpenPreparedStatements);
        dataSource.setRemoveAbandoned(removeAbandoned);
        dataSource.setRemoveAbandonedTimeout(removeAbandonedTimeout);
        dataSource.setLogAbandoned(logAbandoned);
        try {
            dataSource.setFilters(filters);
        } catch (SQLException e) {
            return dataSource;
        }
        return dataSource;
    }

    /**
     * druid 监控页面
     * @return
     */
    @Bean
    public ServletRegistrationBean druidServletBean() {
        ServletRegistrationBean registrationBean
               = new ServletRegistrationBean();
        StatViewServlet statViewServlet = new StatViewServlet();
        registrationBean.addInitParameter("loginUsername", "admin");
        registrationBean.addInitParameter("loginPassword", "admin");
        registrationBean.addInitParameter("resetEnable", "true");
        registrationBean.addUrlMappings("/druid/*");
        registrationBean.setServlet(statViewServlet);
        return registrationBean;
    }

    /**
     * druid 资源监控过滤
     * @return
     */
    @Bean
    public FilterRegistrationBean druidWebStatFilter() {
        FilterRegistrationBean registrationBean
            = new FilterRegistrationBean();
        WebStatFilter webStatFilter = new WebStatFilter();
        registrationBean.addInitParameter("sessionStatMaxCount", "2000");
        registrationBean.addInitParameter("sessionStatEnable", "true");
        registrationBean
```

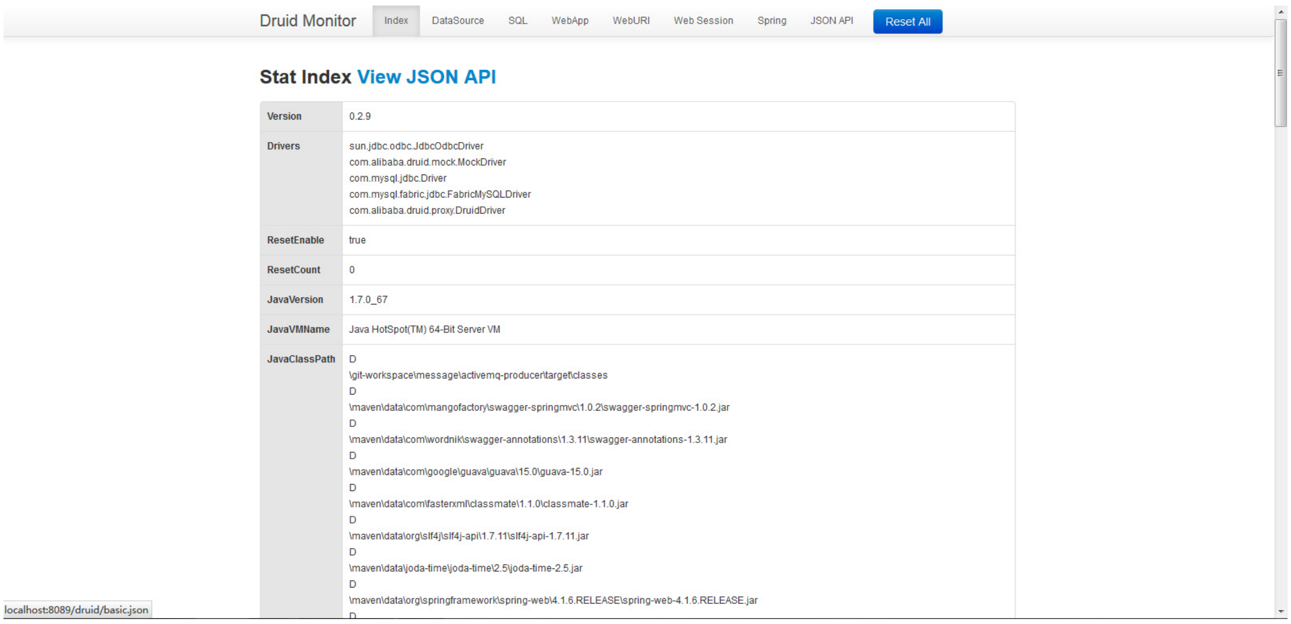
```
            .addInitParameter("principalSessionName", "session_user_key");
            registrationBean.addInitParameter("profileEnable", "true");
            registrationBean
            .addInitParameter
            ("exclusions", "*.js,*.gif,*.jpg,*.png,*.css,*.ico,*.jsp,/druid/*");
            registrationBean.setFilter(webStatFilter);
            registrationBean.addUrlPatterns("/*");
            return registrationBean;
        }

    }
```

配置了druid监控页面的登录页，用户名密码为admin。

访问druid监控页面

启动应用，本文系统tomcat的端口是8089，没有设置项目名称，因此访问的是 `http://localhost:8089/druid/login.html` 。一般而言，druid监控登录页面的入口为： `http://{IP地址}:{端口}/{项目名}/druid/login.html`





Stat Index View JSON API

Version	0.2.9
Drivers	sun.jdbc.odbc.JdbcOdbcDriver com.alibaba.druid.mock.MockDriver com.mysql.jdbc.Driver com.mysql.fabric.jdbc.FabricMySQLDriver com.alibaba.druid.proxy.DruidDriver
ResetEnable	true
ResetCount	0
JavaVersion	1.7.0_67
JavaVMName	Java HotSpot(TM) 64-Bit Server VM
JavaClassPath	D \git-workspace\message\activemq-producer\target\classes D \maven\data\com\mangofactory\swagger-springmvc\1.0.2\swagger-springmvc-1.0.2.jar D \maven\data\com\wordnik\swagger-annotations\1.3.11\swagger-annotations-1.3.11.jar D \maven\data\com\google\guava\guava\15.0\guava-15.0.jar D \maven\data\com\fasterxml\classmate\1.1.0\classmate-1.1.0.jar D \maven\data\org\slf4j\slf4j-api\1.7.11\slf4j-api-1.7.11.jar D \maven\data\joda-time\joda-time\2.5\joda-time-2.5.jar D \maven\data\org\springframework\spring-web\4.1.6.RELEASE\spring-web-4.1.6.RELEASE.jar D