SSMAOP日志

1.数据库与表结构

1.1.日志表信息描述sysLog

序号	字段名称	字段类型	字段描述
1	id	VARCHAR2	主键 无意义uuid
2	visitTime	timestamp	访问时间
3	username	VARCHAR2	操作者用户名
4	ip	VARCHAR2	访问ip
5	url	VARCHAR2	访问资源url
6	executionTime	int	执行时长
7	method	VARCHAR	访问方法

1.2.sql语句

```
CREATE TABLE sysLog(
    id VARCHAR2(32) default SYS_GUID() PRIMARY KEY,
    visitTime timestamp,
    username VARCHAR2(50),
    ip VARCHAR2(30),
    url VARCHAR2(50),
    executionTime int,
    method VARCHAR2(200)
)
```

1.3.实体类

```
public class SysLog {

    private String id;
    private Date visitTime;
    private String visitTimeStr;
    private String username;
    private String ip;
    private String url;
    private Long executionTime;
    private String method;
  }
```

2.基于AOP日志处理

2.1.页面syslog-list.jsp

详细内容请查看资源中页面信息

2.2.创建切面类处理日志

```
@Component
@Aspect
public class LogAop {

    @Autowired
    private HttpServletRequest request;

    @Autowired
    private ISysLogService sysLogService;

    private Date startTime; // 访问时间
    private Class executionClass;// 访问的类
    private Method executionMethod; // 访问的方法
    // 主要获取访问时间、访问的类、访问的方法

    @Before("execution(* com.itheima.ssm.controller.*.*(..))")
    public void doBefore(JoinPoint jp) throws NoSuchMethodException, SecurityException {
        startTime = new Date(); // 访问时间
        // 获取访问的类
        executionClass = jp.getTarget().getClass();
        // 获取访问的方法
        String methodName = jp.getSignature().getName();// 获取访问的方法的名称

        Object[] args = jp.getArgs();// 获取访问的方法的参数
        if (args == null || args.length == 0) {// 无参数
            executionMethod = executionClass.getMethod(methodName); // 只能获取无参数方法
        } else {
            // 有参数，就将args中所有元素遍历，获取对应的Class,装入到一个Class[]
            Class[] classArgs = new Class[args.length];
```

```
            for (int i = 0; i < args.length; i++) {
                classArgs[i] = args[i].getClass();
            }
            executionMethod = executionClass.getMethod(methodName, classArgs);// 获取有参数方法
        }
    }

    // 主要获取日志中其它信息，时长、ip、url...
    @After("execution(* com.itheima.ssm.controller.*.*(..))")
    public void doAfter(JoinPoint jp) throws Exception {

        // 获取类上的@RequestMapping对象
        if (executionClass != SysLogController.class) {
            RequestMapping classAnnotation = (RequestMapping)
executionClass.getAnnotation(RequestMapping.class);
            if (classAnnotation != null) {
                // 获取方法上的@RequestMapping对象
                RequestMapping methodAnnotation =
executionMethod.getAnnotation(RequestMapping.class);

                if (methodAnnotation != null) {

                    String url = ""; // 它的值应该是类上的@RequestMapping的value+方法上的
@RequestMapping的value

                    url = classAnnotation.value()[0] + methodAnnotation.value()[0];

                    SysLog sysLog = new SysLog();
                    // 获取访问时长
                    Long executionTime = new Date().getTime() - startTime.getTime();
                    // 将sysLog对象属性封装
                    sysLog.setExecutionTime(executionTime);
                    sysLog.setUrl(url);
                    // 获取ip
                    String ip = request.getRemoteAddr();
                    sysLog.setIp(ip);

                    // 可以通过securityContext获取，也可以从request.getSession中获取
                    SecurityContext context = SecurityContextHolder.getContext(); //
request.getSession().getAttribute("SPRING_SECURITY_CONTEXT")
                    String username = ((User)
(context.getAuthentication().getPrincipal())).getUsername();
                    sysLog.setUsername(username);

                    sysLog.setMethod("[类名]" + executionClass.getName() + "[方法名]" +
executionMethod.getName());

                    sysLog.setVisitTime(startTime);

                    // 调用Service，调用dao将sysLog insert数据库
                    sysLogService.save(sysLog);
                }

            }
```

```
        }
    }
}
```

在切面类中我们需要获取登录用户的username，还需要获取ip地址，我们怎么处理?

- username获取

  SecurityContextHolder获取

- ip地址获取

  ip地址的获取我们可以通过request.getRemoteAddr()方法获取到。

  在Spring中可以通过RequestContextListener来获取request或session对象。

2.3.SysLogController

```
@RequestMapping("/sysLog")
@Controller
public class SysLogController {

    @Autowired
    private ISysLogService sysLogService;

    @RequestMapping("/findAll.do")
    public ModelAndView findAll() throws Exception {
        ModelAndView mv = new ModelAndView();
        List<SysLog> sysLogs = sysLogService.findAll();
        mv.addObject("sysLogs", sysLogs);
        mv.setViewName("syslog-list");
        return mv;
    }
}
```

2.4.Service

```
@Service
@Transactional
public class SysLogServiceImpl implements ISysLogService {

    @Autowired
    private ISysLogDao sysLogDao;

    @Override
    public void save(SysLog log) throws Exception {
        sysLogDao.save(log);
    }


    @Override
```

```
    public List<SysLog> findAll() throws Exception {
        return sysLogDao.findAll();
    }
}
```

2.5.Dao

```
public interface ISysLogDao {

    @Select("select * from syslog")
    @Results({
        @Result(id=true,column="id",property="id"),
        @Result(column="visitTime",property="visitTime"),
        @Result(column="ip",property="ip"),
        @Result(column="url",property="url"),
        @Result(column="executionTime",property="executionTime"),
        @Result(column="method",property="method"),
        @Result(column="username",property="username")
    })
    public List<SysLog> findAll() throws Exception;

    @Insert("insert into syslog(visitTime,username,ip,url,executionTime,method) values(#
{visitTime},#{username},#{ip},#{url},#{executionTime},#{method})")
    public void save(SysLog log) throws Exception;

}
```