로그관리

개요

로그관리는 시스템 사용시 발생하는 각종 로그를 검색, 조회하는 기능을 제공한다.

설명

로그조회는 시스템로그의 **등록, 조회, 목록, 삭제, 요약**의 기능을 수반한다.

```
① 로그등록 : 로그정보를 등록한다. - AOP 기능을 이용
② 로그조회 : 로그정보의 상세내용을 조회한다.
③ 로그목록 : 로그정보의 목록을 검색, 조회한다.
④ 로그삭제 : 로그정보를 삭제한다. - 실행환경의 Scheduling 기능을 이용
⑤ 로그요약 : 로그정보를 요약하여 Summary를 생성한다. - 실행환경의 Scheduling 기능을 이용
```

패키지 참조 관계

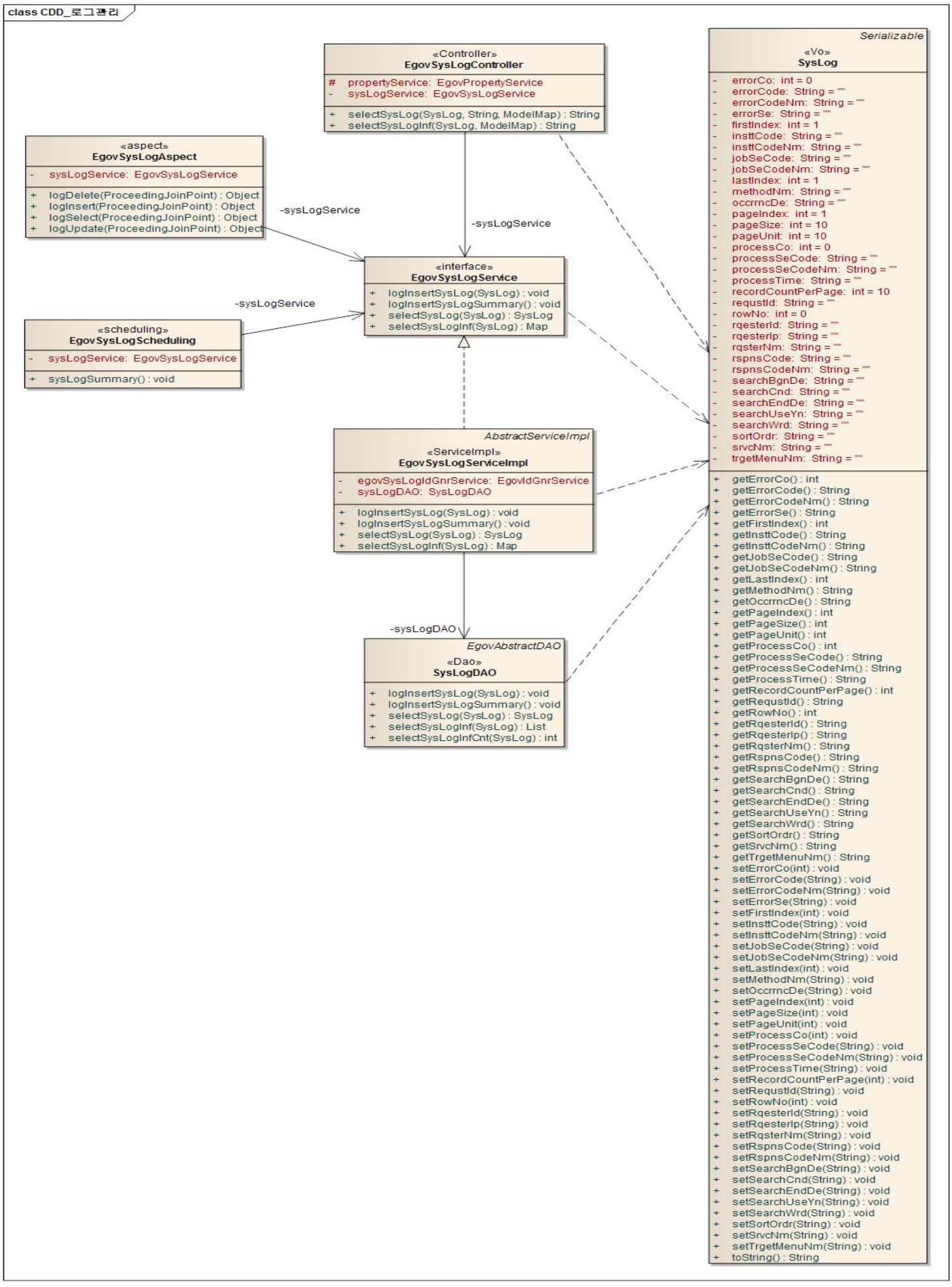
로그관리 패키지는 요소기술의 공통(cmm) 패키지에 대해서만 직접적인 함수적 참조 관계를 가진다. 하지만, 컴포넌트 배포 시 오류 없이 실행되기 위하여 패키지 간의 참조관계에 따라 달력 패키지와 함께 배포 파일을 구성한다.

- 패키지 간 참조 관계 : 시스템관리 Package Dependency

관련소스

유형	대상소스명	비고
Controller	egovframework.com.sym.log.lgm.web.EgovSysLogController.java	로그관리를 위한 컨트롤러 클래스
Service	egovframework.com.sym.log.lgm.service.EgovSysLogService.java	로그관리를 위한 서비스 인터페이스
ServiceImpl	egovframework.com.sym.log.lgm.service.impl.EgovSysLogServiceImpl.java	로그 관리를 위한 서비스 구현 클래스
Model	egovframework.com.sym.log.lgm.service.SysLog.java	로그관리를 위한 Model 클래스
DAO	egovframework.com.sym.log.lgm.service.impl.LogManageDAO.java	로그관리를 위한 데이터처리 클래스
Aspect	egovframework.com.sym.log.lgm.service.EgovLogManageAspect.java	로그 등록을 위한 Aspect 클래스
Scheduler	egovframework.com.sym.log.lgm.service.EgovLogManageScheduling.java	로그 삭제, 요약을 위한 Scheduling 클래스
JSP	/WEB-INF/jsp/egovframework/com/sym/log/lgm/EgovSysLogList.jsp	로그 목록을 위한 jsp페이지
JSP	/WEB-INF/jsp/egovframework/com/sym/log/lgm/EgovSysLogInqire.jsp	로그 조회를 위한 jsp페이지
QUERY XML	resources/egovframework/sqlmap/com/sym/log/lgm/EgovSysLog_SQL_Mysql.xml	로그관리 MySQL용 QUERY XML
QUERY XML	resources/egovframework/sqlmap/com/sym/log/lgm/EgovSysLog_SQL_Oracle.xml	로그관리 Oracle용 QUERY XML
QUERY XML	resources/egovframework/sqlmap/com/sym/log/lgm/EgovSysLog_SQL_Tibero.xml	로그관리 Tibero용 QUERY XML
QUERY XML	resources/egovframework/sqlmap/com/sym/log/lgm/EgovSysLog_SQL_Altibase.xml	로그관리 Altibase용 QUERY XML
Idgen XML	resources/egovframework/spring/context-idgen.xml	로그관리 Id생성 Idgen XML

클래스 다이어그램


class CDD_로그관리
«Controller»
EgovSysLogController
propertyService: EgovPropertyService
- sysLogService: EgovSysLogService
+ selectSysLog(SysLog, String, ModelMap) : String
+ selectSysLogInf(SysLog, ModelMap) : String
«aspect»
EgovSysLogAspect
- sysLogService: EgovSysLogService
+ logDelete(ProceedingJoinPoint) : Object
+ logInsert(ProceedingJoinPoint) : Object
+ logSelect(ProceedingJoinPoint) : Object
+ logUpdate(ProceedingJoinPoint) : Object
-sysLogService
-sysLogService
«interface»
EgovSysLogService
+ logInsertSysLog(SysLog) : void
+ logInsertSysLogSummary() : void
+ selectSysLog(SysLog) : SysLog
+ selectSysLogInf(SysLog) : Map
-sysLogService
«scheduling»
EgovSysLogScheduling
- sysLogService: EgovSysLogService
+ sysLogSummary() : void
AbstractServiceImpl
«ServiceImpl»
EgovSysLogServiceImpl
- egovSysLogIdGnrService: EgovIdGnrService
- sysLogDAO: SysLogDAO
+ logInsertSysLog(SysLog) : void
+ logInsertSysLogSummary() : void
+ selectSysLog(SysLog) : SysLog
+ selectSysLogInf(SysLog) : Map
-sysLogDAO
EgovAbstractDAO
«Dao»
SysLogDAO
+ logInsertSysLog(SysLog) : void
+ logInsertSysLogSummary() : void
+ selectSysLog(SysLog) : SysLog
+ selectSysLogInf(SysLog) : List
+ selectSysLogInfCnt(SysLog) : int
Serializable
«Vo»
SysLog
- errorCo: int = 0
- errorCode: String = ""
- errorCodeNm: String = ""
- errorSe: String = ""
- firstIndex: int = 1
- insttCode: String = ""
- insttCodeNm: String = ""
- jobSeCode: String = ""
- jobSeCodeNm: String = ""
- lastIndex: int = 1
- methodNm: String = ""
- occrrncDe: String = ""
- pageIndex: int = 1
- pageSize: int = 10
- pageUnit: int = 10
- processCo: int = 0
- processSeCode: String = ""
- processSeCodeNm: String = ""
- processTime: String = ""
- recordCountPerPage: int = 10
- requstId: String = ""
- rowNo: int = 0
- rqesterId: String = ""
- rqesterIp: String = ""
- rqsterNm: String = ""
- rspnsCode: String = ""
- rspnsCodeNm: String = ""
- searchBgnDe: String = ""
- searchCnd: String = ""
- searchEndDe: String = ""
- searchUseYn: String = ""
- searchWrd: String = ""
- sortOrdr: String = ""
- srvcNm: String = ""
- trgetMenuNm: String = ""
+ getErrorCo() : int
+ getErrorCode() : String
+ getErrorCodeNm() : String
+ getErrorSe() : String
+ getFirstIndex() : int
+ getInsttCode() : String
+ getInsttCodeNm() : String
+ getJobSeCode() : String
+ getJobSeCodeNm() : String
+ getLastIndex() : int
+ getMethodNm() : String
+ getOccrrncDe() : String
+ getPageIndex() : int
+ getPageSize() : int
+ getPageUnit() : int
+ getProcessCo() : int
+ getProcessSeCode() : String
+ getProcessSeCodeNm() : String
+ getProcessTime() : String
+ getRecordCountPerPage() : int
+ getRequstId() : String
+ getRowNo() : int
+ getRqesterId() : String
+ getRqesterIp() : String
+ getRqsterNm() : String
+ getRspnsCode() : String
+ getRspnsCodeNm() : String
+ getSearchBgnDe() : String
+ getSearchCnd() : String
+ getSearchEndDe() : String
+ getSearchUseYn() : String
+ getSearchWrd() : String
+ getSortOrdr() : String
+ getSrvcNm() : String
+ getTrgetMenuNm() : String
+ setErrorCo(int) : void
+ setErrorCode(String) : void
+ setErrorCodeNm(String) : void
+ setErrorSe(String) : void
+ setFirstIndex(int) : void
+ setInsttCode(String) : void
+ setInsttCodeNm(String) : void
+ setJobSeCode(String) : void
+ setJobSeCodeNm(String) : void
+ setLastIndex(int) : void
+ setMethodNm(String) : void
+ setOccrrncDe(String) : void
+ setPageIndex(int) : void
+ setPageSize(int) : void
+ setPageUnit(int) : void
+ setProcessCo(int) : void
+ setProcessSeCode(String) : void
+ setProcessSeCodeNm(String) : void
+ setProcessTime(String) : void
+ setRecordCountPerPage(int) : void
+ setRequstId(String) : void
+ setRowNo(int) : void
+ setRqesterId(String) : void
+ setRqesterIp(String) : void
+ setRqsterNm(String) : void
+ setRspnsCode(String) : void
+ setRspnsCodeNm(String) : void
+ setSearchBgnDe(String) : void
+ setSearchCnd(String) : void
+ setSearchEndDe(String) : void
+ setSearchUseYn(String) : void
+ setSearchWrd(String) : void
+ setSortOrdr(String) : void
+ setSrvcNm(String) : void
+ setTrgetMenuNm(String) : void
+ toString() : String

ID Generation

ID Generation 관련 DDL 및 DML

- ID Generation Service를 활용하기 위해서 Sequence 저장테이블인 COMTECOPSEQ에 **SYSLOG_ID** 항목을 추가한다.

```
                table_name varchar(16) NOT NULL,
                next_id DECIMAL(30) NOT NULL,
                PRIMARY KEY (table_name));

INSERT INTO COMTECOPSEQ VALUES('SYSLOG_ID','0');
```

ID Generation 환경설정(context-idgen.xml)

```
<bean name="egovSysLogIdGnrService"
  class="egovframework.rte.fdl.idgnr.impl.EgovTableIdGnrService"
  destroy-method="destroy">
  <property name="dataSource" ref="dataSource" />
  <property name="strategy"   ref="sysLogStrategy" />
  <property name="blockSize"  value="1"/>
  <property name="table"      value="COMTECOPSEQ"/>
  <property name="tableName"  value="SYSLOG_ID"/>
</bean>

<bean name="sysLogStrategy"
  class="egovframework.rte.fdl.idgnr.impl.strategy.EgovIdGnrStrategyImpl">
  <property name="prefix" value="SYSLOG_" />
  <property name="cipers" value="13" />
  <property name="fillChar" value="0" />
</bean>
```

테이블

테이블명	테이블명(영문)	비고
시스템로그	COMTNSYSLOG	시스템로그 정보를 관리한다.
시스템로그요약	COMTSSYSLOGSUMMARY	시스템로그 요약정보를 관리한다.

AOP

context-syslogaop.xml

```
        <bean id="syslog" class="egovframework.com.sym.log.lgm.service.EgovSysLogAspect" />

        <aop:config>
                <aop:aspect id="sysLogAspect" ref="syslog">
                        <!--  insert로 시작되는 service Method -->
                        <aop:around pointcut="execution(public * egovframework.com..impl.*Impl.insert*(..))"
method="logInsert" />
                        <!--  update로 시작되는 service Method -->
                        <aop:around pointcut="execution(public * egovframework.com..impl.*Impl.update*(..))"
method="logUpdate" />
                        <!--  delete로 시작되는 service Method -->
                        <aop:around pointcut="execution(public * egovframework.com..impl.*Impl.delete*(..))"
method="logDelete" />
                        <!--  select로 시작되는 service Method -->
                        <aop:around pointcut="execution(public * egovframework.com..impl.*Impl.select*(..))"
method="logSelect" />
                </aop:aspect>
        </aop:config>
```

시스템로그 등록 기능구현을 위하여 AOP를 설정한다.

- 시스템로그 등록 기능구현을 위하여 EgovLogManageAspect 클래스를 생성한다.

```
package egovframework.com.sym.log.lgm.service;

import javax.annotation.Resource;

import org.aspectj.lang.ProceedingJoinPoint;
import org.springframework.util.StopWatch;

import egovframework.com.cmm.LoginVO;
import egovframework.com.cmm.util.EgovUserDetailsHelper;
```

```java
public class EgovSysLogAspect {

	@Resource(name="EgovSysLogService")
	private EgovSysLogService sysLogService;

	/**
	 * 시스템 로그정보를 생성한다.
	 * sevice Class의 insert로 시작되는 Method
	 *
	 * @param ProceedingJoinPoint
	 * @return Object
	 * @throws Exception
	 */
	public Object logInsert(ProceedingJoinPoint joinPoint) throws Throwable {

		StopWatch stopWatch = new StopWatch();

		try {
			stopWatch.start();

			Object retValue = joinPoint.proceed();
			return retValue;
		} catch (Throwable e) {
			throw e;
		} finally {
			stopWatch.stop();

			SysLog sysLog = new SysLog();
			String className = joinPoint.getTarget().getClass().getName();
			String methodName = joinPoint.getSignature().getName();
			String processSeCode = "C";
			String processTime = Long.toString(stopWatch.getTotalTimeMillis());
			String uniqId = "";
			String ip = "";

			/* Authenticated  */
			Boolean isAuthenticated = EgovUserDetailsHelper.isAuthenticated();
			if(isAuthenticated.booleanValue()) {
				LoginVO user = (LoginVO)EgovUserDetailsHelper.getAuthenticatedUser();
				uniqId = user.getUniqId();
				ip = user.getIp();
			}

			sysLog.setSrvcNm(className);
			sysLog.setMethodNm(methodName);
			sysLog.setProcessSeCode(processSeCode);
			sysLog.setProcessTime(processTime);
			sysLog.setRqesterId(uniqId);
			sysLog.setRqesterIp(ip);

			sysLogService.logInsertSysLog(sysLog);

		}

	}

	/**
	 * 시스템 로그정보를 생성한다.
	 * sevice Class의 update로 시작되는 Method
	 *
	 * @param ProceedingJoinPoint
	 * @return Object
	 * @throws Exception
	 */
	public Object logUpdate(ProceedingJoinPoint joinPoint) throws Throwable {

		StopWatch stopWatch = new StopWatch();

		try {
			stopWatch.start();

			Object retValue = joinPoint.proceed();
			return retValue;
		} catch (Throwable e) {
			throw e;
		} finally {
			stopWatch.stop();

			SysLog sysLog = new SysLog();
			String className = joinPoint.getTarget().getClass().getName();
			String methodName = joinPoint.getSignature().getName();
			String processSeCode = "U";
			String processTime = Long.toString(stopWatch.getTotalTimeMillis());
			String uniqId = "";
			String ip = "";

			/* Authenticated  */
			Boolean isAuthenticated = EgovUserDetailsHelper.isAuthenticated();
			if(isAuthenticated.booleanValue()) {
				LoginVO user = (LoginVO)EgovUserDetailsHelper.getAuthenticatedUser();
				uniqId = user.getUniqId();
				ip = user.getIp();
			}

			sysLog.setSrvcNm(className);
			sysLog.setMethodNm(methodName);
			sysLog.setProcessSeCode(processSeCode);
			sysLog.setProcessTime(processTime);
			sysLog.setRqesterId(uniqId);
			sysLog.setRqesterIp(ip);

			sysLogService.logInsertSysLog(sysLog);

		}

	}

	/**
	 * 시스템 로그정보를 생성한다.
	 * sevice Class의 delete로 시작되는 Method
	 *
	 * @param ProceedingJoinPoint
	 * @return Object
	 * @throws Exception
	 */
	public Object logDelete(ProceedingJoinPoint joinPoint) throws Throwable {

		StopWatch stopWatch = new StopWatch();
```

```
                try {
                        stopWatch.start();

                        Object retValue = joinPoint.proceed();
                        return retValue;
                } catch (Throwable e) {
                        throw e;
                } finally {
                        stopWatch.stop();

                        SysLog sysLog = new SysLog();
                        String className = joinPoint.getTarget().getClass().getName();
                        String methodName = joinPoint.getSignature().getName();
                        String processSeCode = "D";
                        String processTime = Long.toString(stopWatch.getTotalTimeMillis());
                        String uniqId = "";
                        String ip = "";

                /* Authenticated  */
                Boolean isAuthenticated = EgovUserDetailsHelper.isAuthenticated();
                if(isAuthenticated.booleanValue()) {
                                LoginVO user = (LoginVO)EgovUserDetailsHelper.getAuthenticatedUser();
                                uniqId = user.getUniqId();
                                ip = user.getIp();
                }

                        sysLog.setSrvcNm(className);
                        sysLog.setMethodNm(methodName);
                        sysLog.setProcessSeCode(processSeCode);
                        sysLog.setProcessTime(processTime);
                        sysLog.setRqesterId(uniqId);
                        sysLog.setRqesterIp(ip);

                        sysLogService.logInsertSysLog(sysLog);

                }

        }

        /**
         * 시스템 로그정보를 생성한다.
         * sevice Class의 select로 시작되는 Method
         *
         * @param ProceedingJoinPoint
         * @return Object
         * @throws Exception
         */
        public Object logSelect(ProceedingJoinPoint joinPoint) throws Throwable {

                StopWatch stopWatch = new StopWatch();

                try {
                        stopWatch.start();

                        Object retValue = joinPoint.proceed();
                        return retValue;
                } catch (Throwable e) {
                        throw e;
                } finally {
                        stopWatch.stop();

                        SysLog sysLog = new SysLog();
                        String className = joinPoint.getTarget().getClass().getName();
                        String methodName = joinPoint.getSignature().getName();
                        String processSeCode = "R";
                        String processTime = Long.toString(stopWatch.getTotalTimeMillis());
                        String uniqId = "";
                        String ip = "";

                /* Authenticated  */
                Boolean isAuthenticated = EgovUserDetailsHelper.isAuthenticated();
                if(isAuthenticated.booleanValue()) {
                                LoginVO user = (LoginVO)EgovUserDetailsHelper.getAuthenticatedUser();
                                uniqId = user.getUniqId();
                                ip = user.getIp();
                }

                        sysLog.setSrvcNm(className);
                        sysLog.setMethodNm(methodName);
                        sysLog.setProcessSeCode(processSeCode);
                        sysLog.setProcessTime(processTime);
                        sysLog.setRqesterId(uniqId);
                        sysLog.setRqesterIp(ip);

                        sysLogService.logInsertSysLog(sysLog);

                }

        }
}
```

Scheduling

context-scheduling-sym-log-lgm.xml (src/main/resources/egovframework/spring/com/context-scheduling-sym-log-lgm.xml)

```
  <!-- 시스템 로그 요약  -->
  <bean id="sysLogging"
    class="org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean">
    <property name="targetObject" ref="egovLogManageScheduling" />
    <property name="targetMethod" value="sysLogSummary" />
    <property name="concurrent" value="false" />
  </bean>

  <!-- 시스템 로그 요약  트리거-->
  <bean id="sysLogTrigger"
    class="org.springframework.scheduling.quartz.SimpleTriggerBean">
    <property name="jobDetail" ref="sysLogging" />
```

```
    <!-- 시작하고 1분후에 실행한다. (milisecond) -->
    <property name="startDelay" value="60000" />
    <!-- 매 1시간마다 실행한다. (milisecond) -->
    <property name="repeatInterval" value="3600000" />
  </bean>

  <!-- 스케쥴러 등록-->
  <bean id="logSummaryScheduler" class="org.springframework.scheduling.quartz.SchedulerFactoryBean">
    <property name="triggers">
      <list>
        <ref bean="sysLogTrigger" />
        <ref bean="webLogTrigger" />
        <ref bean="trsmrcvLogTrigger" />
        <ref bean="userLogTrigger" />
      </list>
    </property>
  </bean>
```

시스템로그 삭제, 요약 기능구현을 위하여 Scheduling을 설정한다.

- 시스템로그 삭제, 요약 기능구현을 위하여 EgovLogManageScheduling 클래스를 생성한다.

```
@Service("egovSysLogScheduling")
public class EgovSysLogScheduling {

        @Resource(name="EgovSysLogService")
        private EgovSysLogService sysLogService;

        /**
         * 시스템 로그정보를 요약한다.
         * 전날의 로그를 요약하여 입력하고, 일주일전의 로그를 삭제한다.
         *
         * @param
         * @return
         * @throws Exception
         */
        public void sysLogSummary() throws Exception {
                sysLogService.logInsertSysLogSummary();
        }

}
```

관련기능

로그관리는 **로그 목록조회**, **로그 상세조회** 기능으로 구분된다.

로그 목록조회

비즈니스 규칙

시스템로그 목록은 페이지 당 10건씩 조회되며 페이징은 10페이지씩 이루어진다. 검색조건은 발생일자와 처리구분에 대해서 수행된다.
시스템로그 상세조회 기능을 수행하기 위해서는 **상세보기** 버튼을 클릭한다.

관련코드

N/A

관련화면 및 수행매뉴얼

Action	URL	Controller method	QueryID
목록조회	/sym/log/lgm/SelectSysLogList.do	selectSysLogInf	"SysLogDAO.selectSysLogInf",
			"SysLogDAO.selectSysLogInfCnt"

조회: 조회하기 위해서는 발생일자/처리구분 조건을 입력한 후 조회 버튼을 클릭한다.
목록클릭: **로그관리 상세조회** 화면으로 이동한다.

시스템로그 상세조회

비즈니스 규칙

시스템로그 상세조회는 팝업창으로 구성되며, **닫기** 버튼을 클릭하면 창을 닫는다.

관련코드

N/A

관련화면 및 수행매뉴얼

Action	URL	Controller method	QueryID
상세조회	/sym/log/lgm/InqireSysLog.do	selectSysLog	"SysLogDAO.selectSysLog"

참고자료

- 실행환경 참조 : AOP
- 실행환경 참조 : Scheduling
- 실행환경 참조 : ID Generation