목차

[1. 표준화 요건 3](#_Toc359242617)

[1.1. 패키지 및 폴더 구조 3](#_Toc359242618)

[1.1.1. 설정파일 3](#_Toc359242619)

[1.1.2. JAVA 3](#_Toc359242620)

[1.1.3. JSP 4](#_Toc359242621)

[1.1.4. 정적 콘텐트 (JavaScript, CSS, Images) 4](#_Toc359242622)

[1.2. 포멧팅 5](#_Toc359242623)

[1.3. JAVA 5](#_Toc359242624)

[1.3.1. Controller Class : 처리 프로세스 전 검증 및 UI 처리 5](#_Toc359242625)

[1.3.2. Service Class : 업무 로직 처리 6](#_Toc359242626)

[1.3.3. DAO 6](#_Toc359242627)

[1.3.4. VO 6](#_Toc359242628)

[2. 기능별 적용 가이드 6](#_Toc359242629)

[2.1. 사용 권한 적용 6](#_Toc359242630)

[2.1.1. 권한 유형 6](#_Toc359242631)

[2.1.2. 적용방법 7](#_Toc359242632)

[2.1.3. 적용 예 7](#_Toc359242633)

[2.2. JavaScript Include 7](#_Toc359242634)

[2.2.1. 설정 7](#_Toc359242635)

[2.2.2. 적용 방법 7](#_Toc359242636)

[2.2.3. 적용 결과 7](#_Toc359242637)

[2.3. Parameter 정책 및 적용 8](#_Toc359242638)

[2.3.1. resources/config/commons/global-config.xml 8](#_Toc359242639)

[2.4. Validate 8](#_Toc359242640)

[2.4.1. VO 작성 8](#_Toc359242641)

[2.4.2. Java 9](#_Toc359242642)

[2.4.3. JSP 10](#_Toc359242643)

[2.5. 개인정보 필터링 11](#_Toc359242644)

[2.5.1. JAVA 소스적용 11](#_Toc359242645)

[2.6. 암.복호화 (용도에 맞도록 사용. 비번/주민번호 등 비교만을 위한 경우 단방향 사용) 12](#_Toc359242646)

[2.7. XSS 보안취약점 보안 적용 12](#_Toc359242647)

[2.8. 환결설정 동기화 13](#_Toc359242648)

[2.8.1. JAVA 사용 13](#_Toc359242649)

[2.8.2. JSP 사용 13](#_Toc359242650)

[2.8.3. JavaScript 사용(위 JSP에서 해당 키를 호출해야 함) 14](#_Toc359242651)

[2.9. Schedule (지정된 시점에 대상 프로세스 구동 Batch Process) 14](#_Toc359242652)

[2.9.1. 설정 14](#_Toc359242653)

[2.10. Logging 변경 14](#_Toc359242654)

[2.10.1. Apache commons-logging을 더 이상 사용하지 않음 14](#_Toc359242655)

[2.10.2. slf4j 라이브러리로 변경 됨 14](#_Toc359242656)

[~~2.10.3. TODO~~ 15](#_Toc359242657)

[2.11. Decotator (레이아웃 관리) 15](#_Toc359242658)

[2.11.1. 네이밍 15](#_Toc359242659)

[3. JavaScript 가이드 15](#_Toc359242660)

[3.1. 프레임워크 표준 스크립트 15](#_Toc359242661)

[3.1.1. resources/openworks/spi/openworks/openworks. global.js 15](#_Toc359242662)

[3.1.2. resources/openworks/spi/openworks/openworks.function.js 15](#_Toc359242663)

[3.1.3. resources/openworks/spi/openworks/openworks.form.js 15](#_Toc359242664)

[3.1.4. resources/openworks/spi/openworks/openworks.ui.js 16](#_Toc359242665)

[3.2. 네이밍에 따른 자동화 16](#_Toc359242666)

[3.2.1. 목록 테이블 ROW 별 컬러 적용 16](#_Toc359242667)

[3.2.2. 키워드 하이라이트 16](#_Toc359242668)

[3.3. 트리 패널 17](#_Toc359242669)

[3.3.1. 태그 적용 17](#_Toc359242670)

[3.3.2. 설정 (기본값이 없는 항목은 필수 입력) 17](#_Toc359242671)

[4. 기타 17](#_Toc359242672)

[4.1. 참조 소스 17](#_Toc359242673)

# 표준화 요건

## 패키지 및 폴더 구조

### 설정파일

|  |
| --- |
|  |

#### commons : xml 기반 설정 파일 위치(기본 설정. 전역변수, 메시지, 리소스관리 등)

#### ibatis : iBatis Data Mapper 설정

#### log4j : Log4J 로그 설정(로그 파일경로, 로그 파일명, 로그 레벨, 로그 유형 등)

#### sitemesh : SiteMesh2 기본 설정 및 Decoration 설정

#### spring : SpringFramework 설정(Spring 기본 설정, 권한적용, IBatis 연동, DataBase 연결, 트랜잭션, 스케줄, View Class, 세션 체크, 캐시 Class 설정 등)

### JAVA

|  |
| --- |
|  |

#### base : Web Application 관련 Class(Support, Helper, TagLib 등)

#### core : OpenWorks 관련 핵심 Class

#### openworks : Web Application Class

#### common : Cache, Schedule 등 개별 동작 프로세스 Class

#### commponent : 공통 프로세스 Class (첨부파일 관리, Java-JS 환경설정 공유 등)

#### intra : 내부 업무 프로세스 Web Application

#### web : 외부 공개 프로세스 Web Application

#### samples : 개발 참조 샘플 Web Application

### JSP

|  |
| --- |
|  |

#### common : 화면이동, Page Decorator, Error Page, TagLib UI 관련

#### component : 전반적 공통 프로세스의 UI (첨부파일관리)

#### intra : 내부 업무 프로세스 UI

#### web : 외부 공개 프로세스 UI

#### login : 시스템 접속 인증 UI

#### samples : 개발 참조 샘플 UI

### 정적 콘텐트 (JavaScript, CSS, Images)

|  |
| --- |
|  |

#### component :전반적 공통 프로세스 UI 콘텐트

#### openworks : 코어에 해당되는 UI 콘텐트

#### intra : 내부 업무 프로세스 UI 콘텐트

#### web : 외부 공개 프로세스 UI 콘텐트

#### samples : 개발 참조 샘플 UI 콘텐트

## 포멧팅

|  |
| --- |
|  |

#### clean-up.xml : 포멧팅 및 미사용 변수, Annotation 등 자동 보정 정보

#### codetemplates.xml : 파일생성 등 주석 표준 정보 - 메뉴 선택(또는 단축키)을 통한 자동 생성시 적용

#### formatter.xml : 들여쓰기, 띄어쓰기 등 기본 포멧팅 정보 - Ctrl + Shift + F 단축키 사용

#### Clean Up 적용방법 - 프로젝트 Properties를 열고 아래 그림과 같은 메뉴를 선택

#### - 각 메뉴에서 "Import" 버튼을 클릭하여 해당 파일을 선택하여 등록한다. - clean-up 적용 : 해당 java 소스를 선택 후 아래 그림과 같이 적용

|  |
| --- |
|  |
|  |

#### - Finish 버튼 클릭 완료. (변경 내역을 확인하려면 Next 버튼을 클릭 후 확인)

## JAVA

### Controller Class : 처리 프로세스 전 검증 및 UI 처리

#### 세션 정보 처리

#### 파라 미터 처리

#### 입력 파라 미터 유효성 검증

#### 요청 결과 통보 및 요청 결과에 따른 페이지 이동

#### 화면 VIEW UI 설정

#### 부모 Controller의 자동화된 응답 메소드

|  |  |  |
| --- | --- | --- |
| **메소드** | **유형** | **설명** |
| alertAndBack | Script | JavaScript 기반  alert("메시지"); history.back(); |
| alertAndClose | Script | JavaScript 기반  alert("메시지"); self.close(); |
| alertAndRedirect | Script | JavaScript 기반  alert("메시지"); location.href=url; |
| confirmAndRedirect | Script | JavaScript 기반  confirm("메시지");  true : location.href=url;  false : history.back(); |
| responseText | AJAX | 문자열을 인자로 사용하며, 인자를 그대로 출력 |
| responseJson | AJAX | 인자로 VO 또는 Collection 객체 등을 사용하며, JSON 타입으로 변환 후 출력  여러 응답 유형을 제공하므로 사용시 확인 |
| responseDownload | File | 인자로 File 또는 FileVO 객체를 사용하며, 파일 저장기능 제공 |
| responseExcel | File | 인자로 IExcelVO 구현 객체를 사용하며, 파일 저장기능 제공 |
| responsePDF | File | 인자로 IPdfVO 구현 객체를 사용하며, 파일 저장기능 제공 |
| responseRSS | HTTP | 인자로 FeedVO 객체를 사용하여 서비스 제공. |
| responseAtom | HTTP | 인자로 FeedVO 객체를 사용하여 서비스 제공. |

### Service Class : 업무 로직 처리

#### DAO Class에 Database관련 처리 로직 위임

#### 트랜잭션 처리**(반드시 결과는 Service 단에서 검증)** 예를 들어 수정, 삭제 시 몇 건이 처리되었는지 등

#### 요청에 따른 관련 업무 프로세스 그룹 처리

### DAO

#### Database 질의 및 결과 반환

#### 페이징 처리

#### 부수적 쿼리 동시 수행

### VO

#### Database 질의 및 결과를 담는 Class

#### 요청 파라 미터 설정

# 기능별 적용 가이드

## 사용 권한 적용

### 권한 유형

\* zes.base.auth Pakage 참조

#### AuthType.READ : 조회권한

#### AuthType.BASIC : 기본권한(일기, 쓰기, 수정, 삭제. 이외 개인데이터 관리여부는 각 기능에서 추가로 확인한다.(예: 게시판의 내 글에 대해서만 수정, 삭제 가능)

#### AuthType.GENERALIZE: 총괄권한(운영자권한)

### 적용방법

|  |
| --- |
| 대상 Controller의 Method에 Annotation 추가  조회 : @AuthIntra(authType = AuthType.READ)  기본 : @AuthIntra(authType = AuthType.BAGIC)  총괄 : @AuthIntra(authType = AuthType.GENERALIZE) |

### 적용 예

|  |
| --- |
| @AuthIntra(authType = AuthType.READ)  @RequestMapping(value = "BD\_crud.list.do", method = RequestMethod.GET)  **public** **void** crudList(HttpServletRequest request, ModelMap model, CrudVO crudVo) {  ....  } |

## JavaScript Include

### 설정

|  |
| --- |
|  |

#### 파일을 열어 기존 내용을 참조하여 js, css 태그 쌍으로 설정

#### 공통 스크립트의 버전, 기능 관리 등을 위함으로 개별 모듈의 파일은 등록하지 않는다.

### 적용 방법

#### 데코레이터에 정의 되어 있는 태그이므로 개별 화면에 적용 불필요

|  |
| --- |
| <op:jsTag type="spi" items="jquery,jnotify,notice,blackbird" />  <op:jsTag type="openworks" /> |

#### 세부 추출(js 또는 css 만 추출시 . 으로 구분해서 사용)

|  |
| --- |
| <op:jsTag type="spi" items="jquery.css, extjs.js " /> |

### 적용 결과

|  |
| --- |
| <script type="text/javascript" src="/resources/openworks/spi/jquery/jquery-1.7.1.min.js"></script>  <link rel="stylesheet" type="text/css" href="/resources/openworks/spi/jquery/jnotify/jquery.jnotify.min.css" />  <script type="text/javascript" src="/resources/openworks/spi/jquery/jnotify/jquery.jnotify.min.js"></script>  <link rel="stylesheet" type="text/css" href="/resources/openworks/spi/jquery/notice/jquery.notice.css" />  <script type="text/javascript" src="/resources/openworks/spi/jquery/notice/jquery.notice.js"></script>  <link rel="stylesheet" type="text/css" href="/resources/openworks/spi/jquery/blackbird/blackbird.css" />  <script type="text/javascript" src="/resources/openworks/spi/jquery/blackbird/blackbird.js"></script>  <link rel="stylesheet" type="text/css" href="/resources/openworks/theme/default/css/openworks\_all.css" />  <script type="text/javascript" src="/component/javascript/MD\_globalConfig.do"></script>  <script type="text/javascript" src="/resources/openworks/spi/openworks/openworks.global.js"></script>  <script type="text/javascript" src="/resources/openworks/spi/openworks/openworks.functin.js"></script> |

## Parameter 정책 및 적용

### /resources/config/commons/global-config.xml

#### 모든 파라미터 정책은 아래 설정을 기준으로 작동

|  |
| --- |
| <global>  <!-- 기본 검색 파라미터 Prefix -->  <prefixSearchParam>q\_</prefixSearchParam>  </global> |

#### Controller 적용. OpHelper.bindSearchMap(VO, HttpServletRequest)

|  |
| --- |
| **public** **void** crudList(HttpServletRequest request, ModelMap model, CrudVO crudVo) {  **OpHelper.bindSearchMap(crudVo, request);**  model.addAttribute(GlobalConfig.KEY\_PAGER, crudService.crudList(crudVo));  } |

#### JSP 적용.

#### - 목록 : <op:pagerParam title="셈플게시물 목록" />

|  |
| --- |
|  |

#### - 상세 : <op:pagerParam view="view" />

|  |
| --- |
| <input type="hidden" name="q\_seq" id="q\_seq" value="1" />  <input type="hidden" name="q\_currPage" id="q\_currPage" value="1" />  <input type="hidden" name="q\_rowPerPage" id="q\_rowPerPage" value="10" />  <input type="hidden" name="q\_searchKey" id="q\_searchKey" value="" />  <input type="hidden" name="q\_searchVal" id="q\_searchVal" value="" />  <input type="hidden" name="q\_sortName" id="q\_sortName" value="" />  <input type="hidden" name="q\_sortOrder" id="q\_sortOrder" value="" /> |

## Validate

### VO 작성

#### 검증이 필요한 속성에 셈플과 같이 Annotation 추가

|  |
| --- |
| /\*\* 제목 \*/  @Required(message = "제목은 필수 입력입니다.")  @RangeLength(min = 3, max = 200)  **private** String title;  /\*\* 분류코드 \*/  @Required(message = "분류를 선택하세요.")  **private** Integer ctgCd;  /\*\* 공지여부 \*/  **private** String noticeYn;  /\*\* 내용 \*/  @Required  @Length(min = 10)  **private** String contents; |

#### 지원 목록 (필요시마다 추가 됨)

* (message = "제목은 필수 입력입니다.") message 항목은 사용자 정의 메시지 사용시만 사용

|  |  |  |
| --- | --- | --- |
| **항목** | **내역** | **속성** |
| Alphabetic | 영문 여부 |  |
| AlphaLower | 영문 소문자 여부 |  |
| AlphaUpper | 영문 대문자 여부 |  |
| AlphaNumeric | 영문 숫자 구성 여부 |  |
| ContainsValue | 입력 값이 속성 값에 존재 여부 | compareValues : {"1,"2","3"} |
| Date | 속성 패턴과 일치 여부 | pattern : "YYYY-MM-DD" |
| Email | 이메일 유효성 여부 |  |
| JuminNo | 주민번호 유효성 여부 |  |
| Digits | 숫자로만 구성 여부 |  |
| Max | 숫자 값 최대 크기 초과 여부 | max :최대수치 |
| Min | 숫자 값 최소 크기 미만 여부 | min: 최소수치 |
| Range | 숫자 값 최대/최소 범위 여부 | min: 최소수치, max :최대수치 |
| MaxLength | 문자열의 최대 길이 초과 여부 | max: 최대길이 |
| MinLength | 문자열의 최소 길이 미만 여부 | min: 최소길이 |
| RangeLength | 문자열의 최대/최고 길이 범위 여부 | min: 최소길이, max: 최대길이 |
| NotNull | null 여부 (입력값 != null) |  |
| Numeric | 숫자(실수 포함) 여부 |  |
| NoWhiteSpace | 공배 포함 여부(없어야 함) |  |
| Pattern | 정규 표현식 유효성 여부 | regexp : 정규표현식 |
| Required | 값의 존재 여부 |  |
| RequireFrom | 대상 필드 값에 따른 입력 값 존재 여부 | fieldName: 대상 필드명  fieldValue: 대상 비교 값  fieldDesc: 대상 한글명 |
| Size | 입력 값의 길이(Array, List 등 배열 형) 예 : checkbox 선택 개수 | min : 최소, max : 최대 |
| URL | URL 유효성 여부 |  |

### Java

#### 입력 화면 : JavaScript를 통한 자동 검증 정보 전달

|  |
| --- |
| ValidateResultHolder holder = ValidateUtil.validate(crudVo);  model.addAttribute(GlobalConfig.KEY\_VALIDATE, holder); |

#### 등록/수정 : 입력된 값을 검증

|  |
| --- |
| ValidateResultHolder holder = ValidateUtil.validate(crudVo);  **if**(holder.isValid()) {  // 실행코드  } **else** {  **return** alertAndBack(model, Messages.COMMON\_VALIDATE\_FAIL);  } |

### JSP

#### 일반 적용(**jquery 제외**)

* 코드

|  |
| --- |
| <valid:script type="msgbox" /> **🡨 검증 결과를 우상단 메시지 박스에 표시**  <valid:script type="alert" /> **🡨 검증 결과를 alert() 스크립트로 표시** |
| $("#dataForm").submit(**function**() {  **if**(**validate()**) { **🡨 검증 실행**  **return** **true**;  }  **return** **false**;  }); |

* 결과 소스

|  |
| --- |
| <script type="text/javascript">  //<![CDATA[  var validate = function() {  var $validateTarget;  $validateTarget = $("[name='title']");    if(!$validateTarget.requiredValidator()) {  jsErrorBox("제목은 필수 입력입니다.");  if($validateTarget.length <= 1) {  $validateTarget.focus();  } else {  $validateTarget[0].focus();  }  return false;  }  …..  **// 자동화 이외 사용자 정의 검증 실행(함수 존재 시 호출)**  if(customValidate()) {  return customValidate();  }  }  //]]>  </script>  <script type="text/javascript">  //<![CDATA[  **// 필요시 사용자 정의 검증 작성**  var customValidate = function() {  return true;  };  //]]>  </script> |

* 메시지 표시 설정

|  |
| --- |
| <tr>  <th><label for="title">제목<span class="tx\_red tx\_b">\*</span></label></th>  <td>  <input type="text" name="title" id="title" class="w85\_p"  title="제목을 입력해주세요" value="${dataVo.title}" />  **<valid:msg name="title" /> 🡨 검증 메시지 설정**  </td>  </tr> |

* 결과 화면

|  |
| --- |
|  |

#### jquery 적용

* 코드

|  |
| --- |
| $("#dataForm").validate({  /\* 유효성 검증 BEAN Annotation 기반 자동 설정 \*/  <valid:script type="jquery" />, **🡨 검증 결과를 jquery.validate 스크립트로 표시**  submitHandler : **function**(form) {  /\* submit() 코드 구현 \*/  }  }); |

* 결과 소스

|  |
| --- |
| $("#dataForm").validate({  /\* 유효성 검증 BEAN Annotation 기반 자동 설정 \*/  rules: {  title : {  required : **true**,  minlength : [3],  maxlength : [300]  }  },  messages:{  title : {  required : "제목은 필수 입력입니다.",  minlength : "최소 길이에 맞게 입력하세요. (3자)",  maxlength : "최대 길이에 맞게 입력하세요. (300자)"  }  },  submitHandler : **function**(form) {  /\* submit() 코드 구현 \*/  }  }); |

* 결과 화면

|  |
| --- |
|  |

## 개인정보 필터링

* 주민등록번호, 여권번호, 운전면허번호, 휴대전화번호, 신용카드번호, 건강보험번호, 계좌번호가 개인정보에 해당된다.

### JAVA 소스적용

* 지원사항 : 단순문자열(String 객체) 지원, 파일 (MS office /PDF/text) 지원
* 파일 등 확장이 필요한 경우 zes.base.privacy 패키지를 참조
* 테스트 참조 파일 : /src/test/java/zes/base/privacy/PrivacyFilterTest.java

|  |
| --- |
| // 문자열 예시  String content = "신용카드번호 테스트 4333 0333 8333 1333 ";  boolean tf = PrivacyFilterFactory.getInstance(content).doFilter();  // 파일명 예시  String content = "sample.xlsx";  boolean tf = PrivacyFilterFactory.getInstance(content).doFilter();  // 파일객체 예시  File file = new File("sample.xlsx");  boolean tf = PrivacyFilterFactory.getInstance(file).doFilter();  나눠서 사용하는 경우  PrivacyFilter pf = PrivacyFilterFactory.getInstance(content);  if(pf.doFilter()) { … } |

## 암.복호화 (용도에 맞도록 사용. 비번/주민번호 등 비교만을 위한 경우 단방향 사용)

* 양방향 : DES(보안 취약), SEED, ARIA(권장은 ARIA이며, SEED를 사용해도 상관없다.)
* 단방향 : SHA256
* 테스트 참조 파일 : /src/test/java/zes/base/privacy/PrivacyFilterTest.java

|  |
| --- |
| 양방향 예 (DES/ SEED/ARIA 중 선택)  Crypto cry = CryptoFactory.getInstance(EncryptType.SEED);  // 아래처럼 문자열로도 가능  // Crypto cry = CryptoFactory.getInstance("SEED");  String enc = cry.encrypt("암호화 대상 문자열");  String dec = cry. decrypt(enc);  단방향 예  Crypto cry = CryptoFactory.getInstance(EncryptType.SHA256);  // 아래처럼 문자열로도 가능  // Crypto cry = CryptoFactory.getInstance("SHA256");  String str1 = cry.encrypt("암호화 대상 문자열 "); |

## XSS 보안취약점 보안 적용

* 신규로 추가된 XSS 보안취약점 보안은 Filter로 구현되어 있으며 web.xml 파일을 참조한다

|  |
| --- |
| **web.xml 파일 내용**  XSS 취약점 보안 적용  XssInclude : , (쉼표)로 복수 URI를 지정하며 길이는 제한이 없다.   * 취약점 보안 대상 URI   XssType : 적용 유형을 설정(기본값은 script)   * all (모든 HTML TAG 무효화) * script (javascript 태그와 특정이벤트 무효화)   <context-param>  <param-name>XssInclude</param-name>  <param-value>/user/, /www/, /web/,/samples/</param-value>  </context-param>  <context-param>  <param-name>XssType</param-name>  <param-value>script</param-value>  </context-param>  …….  아래 예제는 \*.do 인 Spring 프레임워크에만 적용되어 있다.  jsp 도 필요시에는 filter-mapping 항목을 추가하여 url-pattern 을 \*.jsp 형식으로 추가한다.  <filter>  <filter-name>XssFilter</filter-name>  <filter-class>zes.openworks.common.filter.XSSFilter</filter-class>  </filter>  <filter-mapping>  <filter-name>XssFilter</filter-name>  <url-pattern>\*.do</url-pattern>  </filter-mapping> |

## 환결설정 동기화

* /config/commons/global-config.xml 파일 내용중 global 키와 url 키는 데코레이터에서 자동으로 지원 됨.

|  |
| --- |
| - 설정파일    - 동기화 컨트롤러 |

### JAVA 사용

|  |
| --- |
| String encoding = Config.getString("global.encoding", "utf-8"); |

### JSP 사용

|  |
| --- |
| - 메시지 설정 동기화 (/config/commons/message-config.xml 파일내용)  <script type="text/javascript" src="/component/javascript/ MD\_massageConfig.do ">  </script>  - 기타 설정 동기화  <script type="text/javascript" src="/component/javascript/ MD\_customConfig.do?rootKey=system">  </script> |

### JavaScript 사용(위 JSP에서 해당 키를 호출해야 함)

|  |
| --- |
| Config는 변수명 충돌을 피하기 위한 prefix에 해당하며 최상위 ROOT Key에 해당.  rootKey에 해당하는 하위 자식을 모두 가지므로 아래 모든 설정이 사용가능  참조 : zes.openworks.component.javascript.JavaScriptController  // 기본 항목  var encoding = Config.global.encoding;  var serviceMode = Config.global.serviceMode;  …  var prefixSearchParam = Config.global.serviceMode;  // 메시지 사용 예  var processFail = Message.msg.processFail;  // 사용자 정의 사용 예  var appNm = Custom.system.appNm; |

## Schedule (지정된 시점에 대상 프로세스 구동 Batch Process)

### 설정

|  |
| --- |
|  |

#### context-scheduled.xml : 작동 설정

#### context-scheduled-readme.txt : 상세 작동 시간 설정 설명서

## Logging 변경

### Apache commons-logging을 더 이상 사용하지 않음

#### 기존 소스를 변경할 필요는 없으나 이하 변경 점에 대한 적용은 권장사항임.

#### 신규 작성 코드에서는 권장 사항을 준수하도록 해야 함

### slf4j 라이브러리로 변경 됨

#### 적용 방법

|  |
| --- |
| private static Logger logger = LoggerFactory.getLogger(XXXX.class); |

#### 권장 코딩 패턴 ( **{}** 하나당 하나의 파라 미터가 바인딩 됨)

|  |
| --- |
| logger.debug("요청 정보 {} ", "폼 설정 ");  logger.debug("요청 정보 {}을 {}했습니다. ", "폼 설정", "성공"); |

#### 로그 결과

|  |
| --- |
| [04-17 23:51:26] DEBUG RequestInfoLogInterceptor.java:81 : 요청 정보 폼설정  [04-17 23:51:26] DEBUG RequestInfoLogInterceptor.java:81 : 요청 정보 폼설정을 성공했습니다. |

#### 변경요인

* 기존의 로깅보다 적게 30배 이상의 성능 개선
* if(logger.isDebug()) { ... } 과 같이 이전처럼 성능을 위한 설정 확인이 불필요하므로 개발자의 설정 미확인으로 인한 성능 저하 방지 및 코딩 양의 축소 효과

### ~~TODO~~

#### ~~Log4J를 LogBack으로 대체~~

## Decotator (레이아웃 관리)

### 네이밍

|  |  |  |
| --- | --- | --- |
| **별칭** | **전체 명칭** | **설명** |
| BD | Base Decorator | 디자인 포함 지원 |
| PD | Popup Decorator | 팝업 페이지 디자인 |
| SD | Simple Decorator | 디자인은 없으나 공통 Javascript & CSS 파일 지원 |
| ND | No Decorator | 데코레이터를 사용하지 않고 별개로 구현  - 등록/수정/삭제 등의 액션  - 프레임워크의 디자인이 아닌 완전 개별 디자인  페이지 구현등(프레임워크 첫 화면 레이아웃의  extjs를 이용한 구성형태, 로그인 페이지) |
| INC | JSP Include | File 명칭 또는 Ajax 를 이용한 화면 로딩  즉! 컨텐츠(html, text, json 등)를 가져 오지만 데코레이터는 사용하지 않는 요청을 통칭함 |

# JavaScript 가이드

## 프레임워크 표준 스크립트

**\* JavaScript Include 내용 참조**

### /resources/openworks/spi/openworks/openworks. global.js

|  |  |
| --- | --- |
| **메소드** | **설명** |
| jsShowFileHistory | 파일 다운이력 보기(colorbox 필요) |
| viewProp | 객체 속성보기 (디버깅용) 예 : viewProp(대상 Object); |

### /resources/openworks/spi/openworks/openworks.function.js

|  |  |
| --- | --- |
| **메소드** | **설명** |
| jsMessage | 일반 알림 메시지 |
| jsSuccessBox | 프로세스 성공시 알림 메시지 |
| jsWarningBox | 프로세스 경고시 알림 메시지 |
| jsErrorBox | 프로세스 실패시 알림 메시지 |
| jsCloseWin | 팝업 또는 colorbox 윈도우 닫기 |
| jsSysErrorBox | 프로그램 오류 알림 메시지  - isDebugEnabled 설정에 따라서 시스템 오류를 보여주거나,  정식 서비스에 사용될 대체 메시지로 구분하여 출력한다. |

### /resources/openworks/spi/openworks/openworks.form.js

|  |  |
| --- | --- |
| **메소드** | **설명** |
| jsMovePage | 페이지 이동 |
| jsList | 목록 페이지로 이동 |
| jsView | 상세 페이지로 이동 |
| jsInsertForm | 등록 페이지로 이동 |
| jsUpdateForm | 수정 페이지로 이동 |
| jsDelete | 삭제 처리 |
| jsDeleteList | 복수 항목 삭제 처리 |
| jsSearchQueryString | 파라미터 정책("q\_")에 따른 항목을 QueryString로 반환(?는 제외) |
| jsQueryString | form의 input 속성을 QueryString로 반환(?는 제외) |
| jsSearchReset | 파라미터 정책("q\_")에 따른 항목을 Form에서 값을 초기화 기본 dataForm 명칭을 사용 |
| jsSearchResetForm | 파라미터 정책("q\_")에 따른 항목을 Form에서 값을 초기화  지정한 form의 명칭을 사용 |
| jsSelected | select 박스의 선택 설정  \* JSTL if 문 대용 |

### /resources/openworks/spi/openworks/openworks.ui.js

#### input[name=chk-all] 속성은 자동으로 전체 토글 기능이 반영됨

#### input[name=seqs] 속성은 자동으로 토글 기능이 반영됨

#### "span.sort" 속성은 자동으로 컬럼 정렬 기능이 반영됨(단! 컬럼 정의는 목록 html에서 정의)

|  |  |
| --- | --- |
| **메소드** | **설명** |
| jsHightlight | 단어 하이라이트 |
| jsAllCheck | 대상 체크박스 전체 선택/비선택 시 해당 Row의 컬러 변경 |
| jsToggleCheck | 대상 체크박스 선택/비선택 시 해당 Row의 컬러 변경 |
| jsTabUi | 프레임워크 Tab UI 헨들러  프레임워크 디자인 대응  참조 : /docs/산출물/openworks.sample/openworks.ui.tabs.txt |

## 네이밍에 따른 자동화

### 목록 테이블 ROW 별 컬러 적용

#### 적용

|  |
| --- |
| <op:jsTag type="openworks" items="ui" />  **id="odd-color" id 적용시 안쪽 <tr /> 태그에 자동 적용됨**  <tbody id="odd-color">  <tr><td>내용</td></tr> // 일반색  <tr><td>내용</td></tr> // 자동적용 컬러  </tbody>  <table **id="odd-color"**>  <tr><td>내용</td></tr> // 일반색  <tr><td>내용</td></tr> // 자동적용 컬러  </table> |

### 키워드 하이라이트

#### 기본 검색어 자동 인식

|  |
| --- |
| // 검색어 자동인식 사용시 필요 id="q\_searchVal"  <op:jsTag type="spi" items="highlight" />  <op:jsTag type="openworks" items="ui" /> |

#### 직접 지정시

|  |
| --- |
| <op:jsTag type="spi" items="highlight" />  keyword : 하이라이팅 대상 단어  contented : 하이라이팅 적용 영역(미지정시 body 태그 내 전체 적용)  jsHightlight(keyword, contentId); |

## 트리 패널

### 태그 적용

|  |
| --- |
| // 트리 컴포넌트  <op:jsTag type="spi" items="extjs" />  // 공통 영역 구현체  <op:jsTag type="openworks" items="tree" /> |

### 설정 (기본값이 없는 항목은 필수 입력)

|  |  |  |
| --- | --- | --- |
| **속성** | **내용** | **기본값** |
| **속성 설정** | | |
| TREE\_TITLE | 트리 패널 상단의 제목 | 목록 선택 |
| TREE\_EL\_ID | 트리를 표시할 div 태그의 id 명 | "treeDiv" |
| TREE\_DETAIL\_ID | 항목 클릭시 내용을 표시할 div 태그의 id명 | "treeDetailDiv" |
| ROOT\_VISIBLE | 최상위 노드 표시 여부 | false |
| ROOT\_TREE\_ID | 최상위 노드의 고유 id 명(예: 최상위 부서코드) |  |
| ROOT\_TREE\_TEXT | 최상위 노드의 이름(예: 최상위 부서명) |  |
| PARENT\_NODE\_ID\_VAR | 부모 노드의 고유 id명(예: 상위 부서코드) |  |
| NODE\_ID\_VAR | 현재 선택된 노드의 고유 id명(예: 부서코드) |  |
| NODE\_NM\_VAR | 노드 추가(DB 저장 시)시 명칭(예: 추가부서) |  |
| **기타 설정** | | |
| ENABLE\_DND | 항목을 끌어 놓기 여부(예: 순서 변경 시) | false |
| IS\_APPEND\_LEAF | leaf 항목 하위로 항목 추가 가능 여부 설정 | false |
| IS\_USE | 요청 URL에 사용여부(useYn=Y) 설정 여부 | false |
| RESIZABLE | 트리 패널 크기 조절 가능 여부 | false |
| PANEL\_SIZE\_HEIGHT | 패널 높이 사이즈 | 400 |
| PANEL\_SIZE\_WIDTH | 패널 넓이 사이즈 | 200 |
| **URL 관련** | | |
| TREE\_JSON\_URL | 트리 목록을 불러올 URL (json 형식) |  |
| **메소드** | | |
| METHOD | 요청방식 지정(get/post) | "post" |
| VIEW\_FUNC | 항목 클릭 시 액션이 정의 된 메소드 객체 | jsView |
| MOVE\_FUNC | ENABLE\_DND 설정 후 항목 이동 시 액션이 정의된 메소드 객체 | jsMove |
|  |  |  |
|  |  |  |

# 기타

## 참조 소스

|  |
| --- |
| Web Application   * zes.openworks.samples 이하 셈플 참조 * zes.openworks.samples.crud 패키지 참조   JUnit 테스트 코드 참조 |