|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1. 개발 환경  1. **개발툴**    * Eclipse JAVA EE IDE Version (2019-03)을 기본 개발툴로 사용한다.    * Oracle Database11gExpress Edition을 Database 관리툴로 사용한다. 2. **버전관리계획**    * Git 및 GitHub를 이용하여 단위 개발 및Version 관리를 한다.    * 개발 진행중인 작업 내용의 경우 개인 단위의 백업관리가 이루어지며, 개발 완료 작업 내용과 일(daily)단위의 백업은 최종 개발 책임자가 맡는다. 3. **데이터베이스 접근 계획**    * 모든 개발자는 하나의 데이터베이스 서버를 공유하여 사용한다.  2. 명명 규칙  1. **디렉토리 명명 규칙**    * 디렉토리 명칭은 영문으로 사용한다.    * <context root>하위에 메뉴의 깊이를 기준으로 디렉토리를 생성하며, 약어를 사용하지 않는다. 2. **JSP 파일 명명 규칙**    * 해당 페이지의 역할에 따른 말머리로 갖고 파일의 주요 기능을 표기한다.  |  |  |  | | --- | --- | --- | | XXX\_YYY.jsp  ⓐ ⓑ ⓒ | | | | ⓐ | 디렉토리 명칭 | 대분류로 home과 project 폴더로 나누며 하위 폴더는 기능에 따라 명명한다. home 폴더는 프로젝트의 기능을 제외한 나머지(게시판, 로그인, 회원가입 등)가 들어가고 project폴더에는 프로젝트 내용이 들어간다. | | ⓑ | 구분자 | ‘\_’ | | ⓒ | 기능에 따른 파일 명칭 | 1) 메인 : WebContent/home.jsp(메인 홈페이지), WebContent/project\_list.jsp(프로젝트 화면), …  2) 게시판 : home/board/board\_crew\_list.jsp(인원 모집 게시판), …  home/board/board\_free\_list.jsp(자유 게시판),  home/board/board\_insert.jsp(글작성, 인원모집), …  3) 프로젝트 :  project/dashboard/dashboard\_list.jsp(대시보드 목록), … |  * 구분자를 제외한 모든 섹션은 영문자로 시작하며, 대문자를 사용하지 않으며 가급적 의미를 알 수 없는 약어를 사용하지 않는다. * 사용되는 확장자는 아래와 같다.  |  |  | | --- | --- | | **파일 형식** | **파일 확장자** | | Java Servlet Page(JSP) | .jsp | | Cascading Style Sheet(CSS) | .css | | JavaScript | .js | | Hyper Text Markup Language | .html | | Web Resource | .gif, .jpg | | Extensible markup language | .xml | | JAVA | .java |  1. **패키지 명명 규칙 및 자바 파일 명명 규칙**    * 패키지명은 3자리 이상으로 정의한다.      + com.toy.home.controller : Controller 파일 정의. home\_controller.java, login\_controller.java, …      + com.toy.db : 각 패키지의 XXX-mapper.xml(home, project)을 사용할 toy-config.xml, sql\_map\_config.java, db.properties 를 정의한다.      + com.toy.db.home : com.toy.db 의 파일을 활용하기 위한 home-mapper.xml을 정의하고 db 테이블 명으로 sql파일을 정의한다. user\_info.sql, …      + com.toy.db.project : project-mapper.xml, prj\_board.sql, prj\_in\_user.sql, ….      + com.toy.home.dto : 해당 db 테이블명으로 짓는다. home\_user\_info\_dto.java, home\_prj\_info\_dto.java, prj\_history\_dto.java, ….      + com.toy.home.dao : 해당 패키지 명으로 인터페이스와 이를 상속받는 파일을 생성한다. home\_dao.java, home\_daoimpl.java, login\_dao.java, login\_daoimpl.java, …      + com.toy.home.biz : home\_biz.java, home\_bizimpl, …    * 자바 파일은 3자리 이상의 package명 하위 단에 작성한다. 2. **자바스크립트 파일 명명 규칙**    * assets 폴더를 생성하여 기타 자원을 관리한다.    * 하위 폴더명은 파일이 사용되는 기능명으로 명명한다      + WebContent/assets/css/images : 이미지(png, jpg)파일 관리.      + WebContent/assets/fonts : 폰트(ttf, svg, woff) 파일 관리.      + WebContent/assets/js : js 파일 관리. main.js(메인 홈페이지 사이드 메뉴 관련 스크립트), prj.js(프로젝트 메뉴 관련 스크립트), … 3. **클래스 명명 규칙**  * DAO, DTO의 경우 해당 데이터베이스 테이블명칭을 그대로 갖는 클래스 이름을 사용하며, com.test.my.dto, com.test.my.model 을 상위 패키지로 갖는다.   예1) com.toy.home.dto*.user\_info\_dto.class*  예2) com.toy.home.dao.home\_dao.class   * Jar 파일의 경우 WEB-INF/lib 폴더에 위치 시킨다.  1. **객체 및 변수 명명 규칙**  * 객체의 경우 해당 객체가 페이지 내에서 유일할 경우 첫 글자를 소문자로 바꾼 클래스명을 그대로 사용한다. 만약 해당 객체가 페이지 내에서 유일하지 않은 경우에는 해당 객체의 역할을 상세히 나타낼 수 있는 명칭을 말머리로 하여 선언한다.   - 예1) 클래스 user\_info\_dto의 객체 user\_id\_dto  변수의 경우 단순 증감이며 for문 등의 제어문 내에서 선언되는 경우 <i,j,k> 등의 간단한 소문자를 사용한다.  - 예) *for(int i=0;i<10;i++)*   * 특정 정보를 담고 있는 변수인 경우, 해당 정보의 성격을 나타낼 수 있는 명칭을 사용한다   - 예) String *regist\_id*= “회원가입 아이디”;  - HTML내의 <input> 태그로 사용되는 필드 값을 전달받을 경우 동일한 이름의 변수에 할당한다.  - 예) <% String email = request.getParameter(“email”); %>   1. **데이터베이스 명명 규칙**  * 테이블 명칭 : 테이블의 용도를 알 수 있도록 명명한다.   예) 회원정보 테이블: USER\_INFO   * 필드 명칭 : 외부 참조키 여부에 상관없이 테이블 용도에 맞추어 명명한다.   예1) USER\_INFO 테이블의 필드들 : USER\_ID, USER\_PW, USER\_NAME, USER\_EMAIL, …   * 시퀀스 명칭 : 해당 테이블 용도 말머리로 <필드명>+’\_NUM’ 을 붙이는 형태로 사용한다.   예 1) USER\_INFO 테이블의 시퀀스 : USER\_NUM / COMMENT\_BOARD 테이블의 시퀀스 : COMMENT\_NUM 3. 개발 구조  1. **디렉토리 구조**  * 경로는 클래스 및 인터페이스 파일의 경우 java resource 폴더를, 그 외 파일들은   <context root>를 상위로 갖는다.   * Directory 이름은 의미 있는 하나 이상의 영문 단어로 구성하고 전부 소문자로 표기하도록 함. 또한 혼동을 피하기 위해 가급적 약어를 사용하지 않도록 함. * 다음은 디렉토리 구성 규칙의 예임.   + - JSP 파일 : <context root>+ ‘/XXX/XXX\_YYYjsp’     - JAVA Class 파일 : <java resource> + <package>     - JAVA Source 파일 : <java resource> + <package>     - XML 파일 : <context root> + ‘/WEB-INF/XXX.xml’     - 이미지 파일 디렉토리 : <context root> + ’/assets/css/images’     - JavaScript 디렉토리 : <context root> + ‘/assets/js’     - CSS 디렉터리 : <context root> + ‘/assets/css’  4. 코드 구조  1. **JAVA 코드 구조**    * DAO 구조 예제   스크린샷이(가) 표시된 사진  자동 생성된 설명   1. **JSP 및 HTML 코드 구조**  * 예제   스크린샷이(가) 표시된 사진  자동 생성된 설명 |