텍스트이(가) 표시된 사진

자동 생성된 설명

개발표준 가이드

VER 0.5

2022년 7월

|  |
| --- |
|  |

개 정 이 력

| 버전 | 변경일 | 변경내용 | 작성자 | 승인자 |
| --- | --- | --- | --- | --- |
| 0.5 | 2022.07.04 | 최초 작성 | 전경일 |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

목 차

[1. 개발환경 구성 1](#_Toc127780633)

[1.1 크롬 브라우저 1](#_Toc127780634)

[1.2 Visual Studio Code 1](#_Toc127780635)

[1.3 Visual Studio Code 플러그인 1](#_Toc127780636)

[1.3.1 Korean Language Pack for Visual Studio Code 1](#_Toc127780637)

[1.3.2 indent-rainbow 1](#_Toc127780638)

[1.3.3 Auto Rename Tag 2](#_Toc127780639)

[1.3.4 CSS Peek 2](#_Toc127780640)

[1.3.5 HTML to CSS autocompletion 2](#_Toc127780641)

[1.3.6 HTML CSS Support 3](#_Toc127780642)

[1.3.7 Live Server 3](#_Toc127780643)

[1.3.8 ESLint 3](#_Toc127780644)

[1.3.9 Vue Language Features (Volar) 4](#_Toc127780645)

[1.3.10 Vue VSCode Snippets 4](#_Toc127780646)

[1.3.11 Prettier - Code formatter 4](#_Toc127780647)

[1.4 크롬 플러그인 5](#_Toc127780648)

[1.4.1 Vue.js devtools 5](#_Toc127780649)

[2. 프로젝트 시작 6](#_Toc127780650)

[2.1 Quasar CLI 설치 6](#_Toc127780651)

[2.2 Quasar 프로젝트 6](#_Toc127780652)

[2.2.1 프로젝트 생성 6](#_Toc127780653)

[2.2.1 프로젝트 설정 8](#_Toc127780654)

[2.2.2 프로젝트 실행 9](#_Toc127780655)

[2.2.1 프로젝트 구조 11](#_Toc127780656)

[2.3 한글 용어(논리명) 표준정의 원칙 14](#_Toc127780657)

[2.3.1 한글용어(논리명) 기본원칙 14](#_Toc127780658)

[2.4 영문 용어(물리명) 표준정의 원칙 15](#_Toc127780659)

[2.4.1 영문용어(물리명) 기본원칙 15](#_Toc127780660)

[2.4.2 테이블명 명명규칙 15](#_Toc127780661)

[2.4.3 컬럼명 명명규칙 15](#_Toc127780662)

[2.4.4 기타 물리 데이터 오브젝트 명명규칙 15](#_Toc127780663)

[2.5 요약 21](#_Toc127780664)

[2.5.1 테이블명 등 예시 요약 21](#_Toc127780665)

[3. 프로그래밍 표준 23](#_Toc127780666)

[2.1. 포맷 규칙(Formatting rule) 23](#_Toc127780667)

[3.1.1 들여쓰기 23](#_Toc127780668)

[3.1.2 라인 작성 기준 23](#_Toc127780669)

[3.2 클래스 명명 규칙 25](#_Toc127780670)

[3.2.1 접미사 25](#_Toc127780671)

[3.3 메소드 명명 규칙 26](#_Toc127780672)

[3.3.1 메소드 명명 규칙 26](#_Toc127780673)

[3.4 변수 명명 규칙 26](#_Toc127780674)

[3.4.1 상수 26](#_Toc127780675)

[4. 소스 파일 작성 규칙 27](#_Toc127780676)

[4.1 HTML, JS, JAVA 27](#_Toc127780677)

[4.2 주석 27](#_Toc127780678)

[4.2.1 한줄 주석 27](#_Toc127780679)

[4.2.2 여러줄 주석 27](#_Toc127780680)

[4.2.3 코드명 부여원칙 예외 28](#_Toc127780681)

[4.3 선언부 28](#_Toc127780682)

[4.3.1 import 28](#_Toc127780683)

[4.3.2 대소문자 규칙 29](#_Toc127780684)

[5. 산출물 30](#_Toc127780685)

[5.1 디렉토리 정의 30](#_Toc127780686)

[5.2 명명 규칙 31](#_Toc127780687)

[5.2.1 명명 규칙 31](#_Toc127780688)

# 개발환경 구성

## 크롬 브라우저

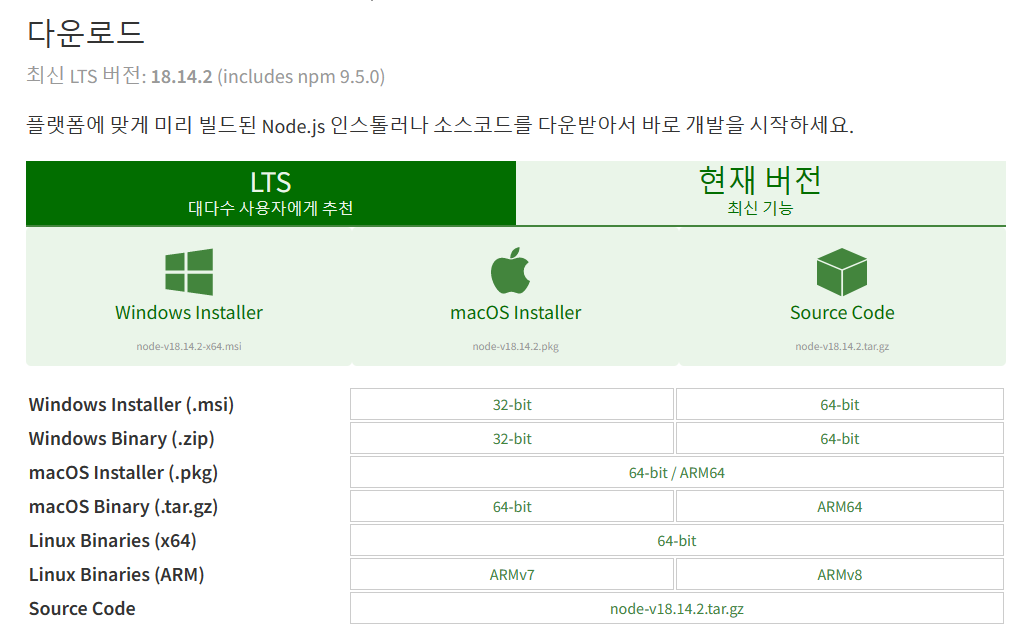
프론트엔드 화면의 테스트 및 실행을 위한 표준 브라우저로 필요 시 다른 브라우저를 설치한다.

## Node.js

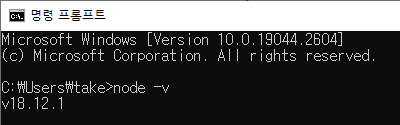
개발을 위한 Node 설치

- 설치 주소 : <https://nodejs.org/ko/download/>

- 아래 화면에서 개발 OS에 맞는 프로그램 다운 후 설치 진행한다.



- 설치 확인



## Visual Studio Code

개발 IDE로 Visual Studio Code를 설치한다.

- 설치 주소 : <https://code.visualstudio.com/>

## Visual Studio Code 플러그인

### Korean Language Pack for Visual Studio Code

- 비주얼 스튜디오 코드 툴을 한국어로 사용할 수 있다.

- 설치 후 VSCode 재시작 필요함

텍스트이(가) 표시된 사진

자동 생성된 설명

### indent-rainbow

- Tab 영역을 컬러별로 다르게 표시하여 코드라인이 길어졌을때 읽기 편하게 한다

텍스트이(가) 표시된 사진

자동 생성된 설명

### Auto Rename Tag

- HTML Tag에서 태그명 변경 시 여는태그 또는 닫는태그 명도 함께 변경함

텍스트이(가) 표시된 사진

자동 생성된 설명

### CSS Peek

- HTML 문서에서 정의된 CSS를 쉽게 찾을 수 있도록 함. ctrl + 클릭 하면 선언된 곳으로 이동함.

텍스트, 모니터, 스크린샷, 화면이(가) 표시된 사진

자동 생성된 설명

### HTML to CSS autocompletion

- HTML 문서에 선언된 class 명을 .css파일에서 입력할 때 자동완성 기능을 제공

텍스트, 스크린샷, 화면, 닫기이(가) 표시된 사진

자동 생성된 설명

### HTML CSS Support

- HTML 문서에서 CSS의 자동완성을 이용할 수 있다.

텍스트이(가) 표시된 사진

자동 생성된 설명

### Live Server

- HTML 파일 수정 시 새로고침 없이 바로바로 즉각 적용되도록 도와준다.

텍스트이(가) 표시된 사진

자동 생성된 설명

### ESLint

- 코드 검사기로써 에러가 있는지 검사해주는 도구.

텍스트이(가) 표시된 사진

자동 생성된 설명

### Vue Language Features (Volar)

- Vue 3언어의 코딩을 지원하는 플러그인입니다.

텍스트이(가) 표시된 사진

자동 생성된 설명

### Vue VSCode Snippets

- 상용구 코드를 빠르게 타이핑 할 수 있도록 도와주는 플러그인

텍스트이(가) 표시된 사진

자동 생성된 설명

### Prettier - Code formatter

- 코드 포멧터로써 코드를 일관성있고 예쁘게 작성할 수 있도록 도와주는 도구

텍스트이(가) 표시된 사진

자동 생성된 설명

## 크롬 플러그인

### Vue.js devtools

- 크롬 앱스토어 > vue dev tools 검색 > vue 3용 devtools 설치

텍스트이(가) 표시된 사진

자동 생성된 설명

# 프로젝트 시작

## Quasar CLI 설치

Quasar 에서 권장하고 다양한 기능을 사용할 수 있는 Quasar CLI를 이용하여 프로젝트를 생성한다.

- 참고 : <https://quasar.dev/start/quasar-cli>

- nodejs 최신버전을 먼저 설치한다.

|  |
| --- |
| PS C:\dev\vue\_test\quve> node -v  v18.12.1 |

* Quasar CLI설치

$ npm i -g @quasar/cli

## Quasar 프로젝트

### 프로젝트 생성

* VSCode에서 파일열기로 프로젝트 Root 로 이동

Ex) C:\dev

* 새로운 터미널창 오픈
* Quasar 프로젝트 생성

npm init quasar

|  |
| --- |
| ? What would you like to build? » - Use arrow-keys. Return to submit.  > App with Quasar CLI, let's go! - spa/pwa/ssr/bex/electron/capacitor/cordova  AppExtension (AE) for Quasar CLI  Quasar UI kit  [Enter]  ? Project folder: » quve  ? Pick Quasar version: » - Use arrow-keys. Return to submit.  > Quasar v2 (Vue 3 | latest and greatest) - recommended  Quasar v1 (Vue 2)  ? Pick script type: » - Use arrow-keys. Return to submit.  > Javascript  Typescript  ? Pick Quasar App CLI variant: » - Use arrow-keys. Return to submit.  > Quasar App CLI with Webpack  Quasar App CLI with Vite  ? Package name: » quve  ? Project product name: (must start with letter if building mobile apps) » Lubentis Frontend Framework  ? Project description: » Lubentis Frontend Framework  'git':(��) ���� �Ǵ� �ܺ� ���, ������ �� �ִ� �wα׷�, �Ǵ�  ��ġ ������ �ƴմϴ�.  √ Author: ...  [Enter]  ? Pick your CSS preprocessor: » - Use arrow-keys. Return to submit.  > Sass with SCSS syntax  Sass with indented syntax  None (the others will still be available)  ? Check the features needed for your project: »  Instructions:  ↑/↓: Highlight option  ←/→/[space]: Toggle selection  a: Toggle all  enter/return: Complete answer  (\*) ESLint  ( ) State Management (Pinia) - https://pinia.vuejs.org  ( ) State Management (Vuex) [DEPRECATED by Vue Team]  ( ) Axios  ( ) Vue-i18n  ? Pick an ESLint preset: » - Use arrow-keys. Return to submit.  > Prettier - https://github.com/prettier/prettier  Standard  Airbnb  ? Install project dependencies? (recommended) » - Use arrow-keys. Return to submit.  > Yes, use npm  No, I will handle that myself  [Enter] |

* VSCode에서 파일열기로 생성한 프로젝트 폴더 오픈

Ex) C:\dev\quve

텍스트, 모니터, 스크린샷이(가) 표시된 사진

자동 생성된 설명

### 프로젝트 설정

* package.json

- scripts 아래에 dev, build 추가

- 참고 : https://quasar.dev/start/quasar-cli

텍스트이(가) 표시된 사진

자동 생성된 설명

* .prettierrc

- Prettier 라이브러리에서 사용하는 설정 파일로 프로젝트 root 에 파일을 생성한 후 아래와 같은 설정 입력

텍스트이(가) 표시된 사진

자동 생성된 설명

### 프로젝트 실행

* npm run dev

- 정상 실행 후 터미널창에 있는 주소(<http://localhost:9001>)를 브라우저에서 실행

|  |
| --- |
| PS C:\dev\quve> npm run dev  > quve@0.0.1 dev  > quasar dev  .d88888b.  d88P" "Y88b  888 888  888 888 888 888 8888b. .d8888b 8888b. 888d888  888 888 888 888 "88b 88K "88b 888P"  888 Y8b 888 888 888 .d888888 "Y8888b. .d888888 888  Y88b.Y8b88P Y88b 888 888 888 X88 888 888 888  "Y888888" "Y88888 "Y888888 88888P' "Y888888 888  Y8b  App • ⚠️ ️️Setting port to closest one available: 9001  » Reported at............ 2023. 2. 14. 오후 4:59:58  » App dir................ C:\dev\quve  » App URL................ http://192.168.0.67:9001/  http://localhost:9001/  » Dev mode............... spa  » Pkg quasar............. v2.11.5  » Pkg @quasar/app-vite... v1.2.0  » Browser target......... es2019|edge88|firefox78|chrome87|safari13.1  App • Opening default browser at http://localhost:9001/ |

* 브라우저 화면

텍스트이(가) 표시된 사진

자동 생성된 설명

## 기타 처리

### 404 Not Found

* Vue를 build 후 운영에 배포시에는 route경로의 파일잉 웹서버에는 실제로 존재하지 않으므로 404 Not Found가 발생한다.
* 이와 같은 경우에는 Main page 로 redirection을 시켜서 오류를 처리한다.

1. quasar.config.js

- route mode를 history로 설정한다. Hash 모드인 경우 url 앞에 #이 표시되나

Seo 등을 위해 history로 설정한다.

vueRouterMode: "history", // available values: 'hash', 'history'

1. /src/router/routes.js

- /err404 path를 routes 최상단으로 이동한다.

const routes = [

  {

    path: "/err404",

    name: "ERROR404",

    component: () => import("pages/ErrorNotFound.vue"),

  },

  //로그인 페이지

  {

    path: "/",

    component: () => import("layouts/LogIn.vue"),

  },

  //메인화면

  {

    path: "/main",

    name: "MainLayout",

    component: () => import("layouts/MainLayout.vue"),

    children: [

      {

        path: "home",

        name: "home",

        component: () => import("pages/SystemHomePage.vue"),

      },

    ],

  },

  // Always leave this as last one,

  // but you can also remove it

  {

    path: "/:catchAll(.\*)\*",

    name: "ERROR404",

    component: () => import("pages/ErrorNotFound.vue"),

  },

];

3. 웹서버 설정

- apach : /conf/httpd.conf에 FallbackResource 를 추가한다.

이는 404 등 없는 페이지 호출 시 “/”로 이동하게 한다.

<Directory "C:\dev\mobile\quasar\dist\spa">

# vue history mode 404 처리

FallbackResource /

….

</Directory>

- nginx : /etc/nginx/conf.d/default.config 에 아래 설정을 추가한다.

텍스트이(가) 표시된 사진

자동 생성된 설명

참고 : https://router.vuejs.org/guide/essentials/history-mode.html

4. 404 오류 처리 파일 생성

- 404 발생 시 /err404로 이동시킨다. 하지만, “/”로 해도 ErrorNotFound.vue 로 이동하기는 한다. Vue router와 웹서버 학습이 필요할 듯…

<script>

  // 운영 웹서버에서 vue history mode 404 발생 시 replace

  document.location.replace("/err404");

</script>

# Vue Llibrary

## Chart

차트 구현을 위한 라이브러리

\* chart.js 설치:

PS C:\dev\mobile\quasar> npm install chart.js

\* vue-chart-3 설치:

PS C:\dev\mobile\quasar> npm install vue-chart-3

## Vue Router

URL에 따라 어떤 페이지를 보여줄지 매핑해주는 라이브러리

\* 설치 :

PS C:\dev\mobile\quasar> npm install vue-router

### 설정

SEO 등의 처리를 위해 vue router는 history 모드로 설정한다.

\* quasar.config.js

   vueRouterMode: "history",

\* /src/router/routes.js

routes에는 로그인, 레이아웃, 404등 모든 사용자가 공통으로 사용하는 라우트 정보만 입력한다.

### 동적 라우트 설정

사용자 권한에 따라 페이지 접근이 달라지므로 로그인 후 메뉴정보를 DB에서 조회하여 동적으로 라우트 정보를 프레임워크에 세팅한다.

1. MainHeader.vue

- 사용자 권한별로 접속 가능 시스템을 조회한다.

// 시스템 리스트 조회

const appList = () => {

  useAxios("get", "/auth/appList", {}, false)

    .then((data) => {

      systemList.value = data; //[{app\_nm:'시스템', app\_cd:'system' }]

      //상위 메뉴가 존재하면 메뉴중 제일 첫번쩨

      changeMenu(systemList.value[0]);

      selSystem.value = systemList.value[0].app\_nm;

    })

    .catch((error) => {

      alert(error.message);

    });

};

- 사용자 시스템을 클릭하면 DB에서 하위 메뉴 리스트를 조회한다.

(\* 조회된 메뉴 리스트는 pinia에 등록하여 모든 컴포넌트에서 사용할 수 있다)

// 시스템 탭 클릭 시 하위 메뉴 리스트 변경

const changeMenu = (system) => {

  const param = {

    appCd: system.app\_cd,

  };

  useAxios("post", "/auth/menuList", param, true)

    .then((data) => {

      menuStore.setMenuList(convertMenu(data)); //메뉴 세팅

      // 라우터에 등록

      makeRoute(menuStore.getMenuList);

    })

    .catch((error) => {

      alert(error.message);

    });

};

- 조회된 메뉴 리스트를 vue router에 동적으로 등록한다.

// Vue Router에 routes 추가 (routes.js)

// MainLayout 하위의 children[] 에 추가함

// menu.component 는 컴포넌트의 절대경로로 입력해야 함.

// 재귀함수

const makeRoute = (menus) => {

  const modules = import.meta.glob("/src/pages/\*\*/\*.vue");

  menus.forEach((menu) => {

    if (menu.link) {

      // \_link : /formtest => formtest

      //let \_link = menu.link.substring(1, menu.link.length);

      let \_link = menu.link;

      let \_navi = menu.navi

        .replaceAll('"', "")

        .replaceAll("{", "")

        .replaceAll("}", "")

        .replaceAll(",", "|");

      router.addRoute("MainLayout", {

        path: \_link,

        name: \_link,

        meta: {

          breadcrumbs: \_navi,

        },

        //component: () => import(menu.component),

        component: modules[menu.component],

      });

    }

    // 하위 메뉴가 존재할 경우 라우터 추가 함수 실행

    if (menu.children) {

      makeRoute(menu.children);

    }

  });

};

## Pinia

시스템내의 컴포넌트에서 사용할 수 있는 상태관리 라이브러리

\* 설치 :

PS C:\dev\mobile\quasar> npm install pinia

### 설정

Vue App 부트시 초기화 파일인 appInit.js에 pinia를 사용처리한다.

\* /src/boot/appInit.js

import { boot } from "quasar/wrappers";

import { createPinia } from "pinia";

export default boot(({ app }) => {

  // Pinia 사용 등록

  app.use(createPinia());

});

### 상태 파일 생성

/src/stores에 공통으로 관리할 상태 파일을 생성한다.

\* /src/stores/munu.js

import { defineStore } from "pinia";

/\*

  Pinia Store 설정

  1. counterStore : Pinia Store 전역 ID명

  2. state : 상태값

  3. getters : component computed

  4. actions : component method

\*/

export const useMenuStore = defineStore("menuStore", {

  state: () => ({

    menuList: {},

  }),

  getters: {

    getMenuList: (state) => {

      return state.menuList;

    },

  },

  actions: {

    setMenuList(menuList) {

      this.menuList = menuList;

    },

  },

});

### 상태 등록

메뉴를 사용자 권한별로 조회 시 Pinia의 setMenuList를 이용해 menuList에 세팅한다.

\* MainHeader.vue

import { useMenuStore } from "stores/menu";

const menuStore = useMenuStore(); //시스템 하위 메뉴 스토어

// 시스템 탭 클릭 시 하위 메뉴 리스트 변경

const changeMenu = (system) => {

  const param = {

    appCd: system.app\_cd,

  };

  useAxios("post", "/auth/menuList", param, true)

    .then((data) => {

      menuStore.setMenuList(convertMenu(data)); //메뉴 세팅

};

### 상태 조회

Pinia에 등록된 메뉴를 이용해 메뉴를 구성한다.

\* MainMenu.vue

import { useMenuStore, useMdiStore } from "stores/menu";

import { route } from "quasar/wrappers";

const router = useRouter();

// 메뉴 스토어

const menuStore = useMenuStore();

const stMenuList = storeToRefs(menuStore).getMenuList;

      <!-----------------------------------------------------------------------

      MENU  START

    ------------------------------------------------------------------------>

      <q-tab-panel name="menus" class="q-pl-xs q-py-xs">

        <!-- Level 1 -->

        <q-expansion-item

          v-for="(level\_1, idx\_1) in stMenuList"

….

## Vue i18 (+ Quasar Language Pack)

사용자 정의 다국어 처리

\* 설치 :

PS C:\dev\mobile\quasar> npm install i18n

### 설정

\* /src/boot/appInit.js

import { boot } from "quasar/wrappers";

import { createI18n } from "vue-i18n";

export default boot(({ app }) => {

  // 다국어 i18n 사용 등록

  const i18n = createI18n({

    legacy: false,

    globalInjection: true,

    locale: "ko-KR", // 기본 언어

  });

app.use(i18n);

  // 퀘이사 language pack 로딩 후 전역 변수 등록

  const qLangList = import.meta.glob("../../node\_modules/quasar/lang/\*.mjs");

  app.config.globalProperties.qLangList = qLangList;

});

### 사용

\* Patter01.vue

import { useI18n } from "vue-i18n";

/\*

  다국어 처리 방법

  vue i18n의 경우 기본적인 다국어 처리 방법은 각 로케일별 메세지를 js로 만들어놓고

  로케일만 변경시켜 메세지를 변경시키는 방법을 사용한다.

  QV 프레임워크에서는 로케일별 메시지를 실시간으로 DB에서 조회한 후

  세팅하는 방법으로 진행한다.

  0. boot(appinit.js) 에 createI18n 과 퀘이사 language pack 를 로드한다.

  1. 로그인시 디폴트 로케일 또는 선택된 로케일을 가져온다.

  2. 이 로케일로 아래와 같은 형식으로 DB에서 message를 가져온다.

  3. 가져온 메시지는 아래 두단계로 변경한다.

     3.1 i18n.locale.value = 'ko-KR';

     3.2 i18n.setLocaleMessage("ko-KR", i18nMessage);

\*/

const i18n = useI18n({ useScope: "global" });

let i18nMessage\_ko\_KR = {

  isoName: "ko-KR",

  nativeName: "한국어",

  prodName: "퀘이사!!",

  label: {

    clear: "초기화",

    ok: "확인",

    cancel: "취소",

  },

};

let i18nMessage\_en\_US = {

  isoName: "en-US",

  nativeName: "English",

  prodName: "Quasar!!",

  label: {

    clear: "clear",

    ok: "ok",

    cancel: "cancel",

  },

};

const appMode = import.meta.env.MODE; // 운영 모드 (development, production)

const qLangList = proxy.qLangList; // 퀘이사 language pack (boot 시 등록)

let quasarLangDir = ""; // 퀘이사 language pakc dirertory

// 언어변경을 감지하여 동적으로 퀘이사 언어팩 세팅

// 개발모드와 운영모드 방식이 다름(Vite)

watch(

  () => formVal.value.langSelected,

  (val) => {

    // 개발 모드

    if (appMode === "development") {

      quasarLangDir = "/node\_modules/quasar/lang/";

      // quasar language 동적 import

      import(quasarLangDir + val + ".mjs").then((lang) => {

        // i18n locale 세팅

        i18n.locale.value = val;

        if (val === "ko-KR") i18n.setLocaleMessage(val, i18nMessage\_ko\_KR);

        else if (val === "en-US") i18n.setLocaleMessage(val, i18nMessage\_en\_US);

        // quasar language 세팅

        $q.lang.set(lang.default);

      });

      //운영

    } else if (appMode === "production") {

      quasarLangDir = "../../node\_modules/quasar/lang/";

      qLangList[quasarLangDir + val + ".mjs"]().then((lang) => {

        // i18n locale 세팅

        i18n.locale.value = val;

        if (val === "ko-KR") i18n.setLocaleMessage(val, i18nMessage\_ko\_KR);

        else if (val === "en-US") i18n.setLocaleMessage(val, i18nMessage\_en\_US);

        // quasar language 세팅

        $q.lang.set(lang.default);

      });

    }

  }

);

                      <QVInput

                        v-model="formVal.testInput1"

                        :title="$t('prodName')"

                        mandatory

                      />

# 프로젝트 구조

## File System

모듈, 컴포넌트 및 스크립트등의 파일 위치 등 구조를 설명한다.

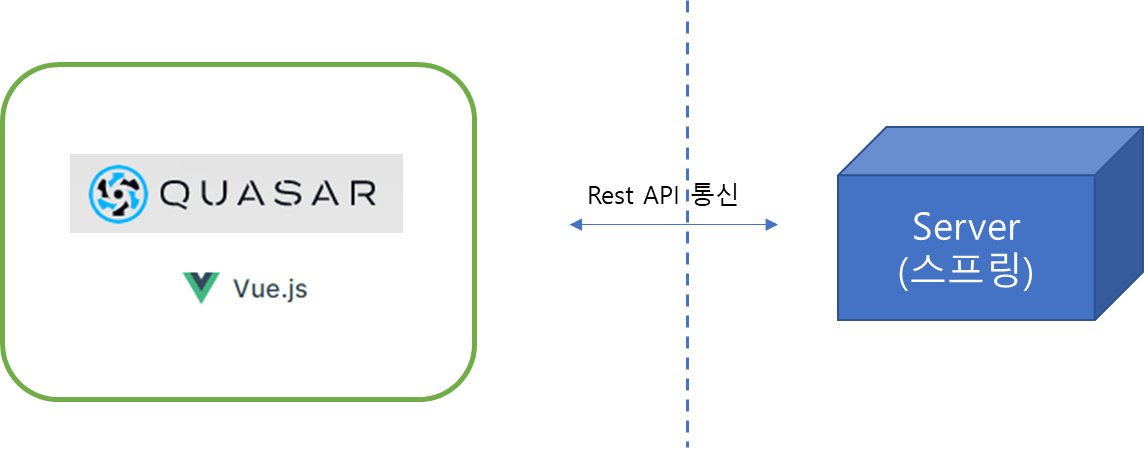
### 프로젝트 구조

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **.quasar** |  | | | 퀘이사 기본 디렉토리로 변경 금지 npm run dev로 자동생성됨 |
| **dist** |  | | | 앱 빌드시 빌드된 파일이 위치함. (npm run build) |
| **node\_modules** |  | | | 개발, 운영에 필요한 모듈 |
| **public** |  | | | 정적 파일(이미지 등) 위치 |
| **src** |  | | | 개발 소스 파일 위치 |
| **assets** | | | 정적 파일(이미지 등) 위치 |
| **boot** |  | | Vue App 초기화 실행 파일 위치 |
| **appInit.js** | | 글로벌 컴포넌트 등록(폼, 페이지 등) 글로벌 변수 등록(프로젝트명 등) |
| **components** |  | | 공통 컴포넌트 파일 위치 |
| **dialog** |  | 다이어로그 컴포넌트 파일 위치 |
| QVAlert.vue | Alert창 |
| QVConfirm.vue | Confirm창 |
| **page** |  | 화면 관련 컴포넌트 파일 위치 |
| QVPageHeader.vue | 화면 헤더의 경로 표시 컴포넌트 |
| **popup** |  | 모달창등 팝업 컴포넌트 파일 위치 |
| QVModal.vue | 이동 기능 있는 모달창 |
| **search** |  | 검색창 아이템 컴포넌트 파일 위치 |
| QVCalendar.vue | 달력(단일) |
| QVCalendarPeriod.vue | 달력(기간) |
| QVCheckbox.vue | 체크박스 |
| QVInput.vue | input, Textarea |
| QVRadio.vue | 라디오 버튼 |
| QVSelect.vue | 셀렉트박스 |
| QVToggle.vue | 토글 |
| **scripts** |  | | 공통 스트립트 파일 위치 |
| commonFuncions.js | | 공통 함수 파일 |
| golobalComponents.js | | 글로벌 컴포넌트 등록(폼, 페이지 등) |
| **css** |  | | 스타일시트 파일 위치 |
| app.scss | | global css in SCSS form |
| quasar.variables.scss | | Quasar SCSS (& Sass) 변수 설정 |
| **layouts** |  | | 메인 레이아웃 구성 컴포넌트 위치 |
| MainLayout.vue | | 헤더, 메뉴, 페이지 영역 |
| MainPage.vue | | 화면 영역 구성 |
| MenuLink.vue | | 메뉴 구성 컴포넌트 |
| **pages** |  | | 업무 화면 파일 위치 |
| dev | | 개발, 참고용 화면 |
| NotFoundView.vue | | 404 에러시 파일 |
| SystemHomePage.vue | | 홈 화면 |
| **router** |  | | Vue Router 컴포넌트 설정 파일 |
| index.js | | Vue Router 컴포넌트 설정 파일 |
| routes.js | | 라우트 설정 파일 |
| App.vue |  | | Vue 프로젝트 Root 컴포넌트 |
| .env |  | | | VITE 환경설정 파일(공통) |
| .env.development |  | | | VITE 환경설정 파일(개발 모드) |
| .eslintignore |  | | | Eslint 제외할 디렉토리 및 파일 |
| .eslintrc.js |  | | | Eslint 설정파일 |
| index.html |  | | | 프로젝트 시작 html(quasar:entry-point) |
| jsconfig.json |  | | | vscode 설정파일(디렉토리 경로 등) |
| package-lock.json |  | | |  |
| package.json |  | | |  |
| postcss.config.js |  | | |  |
| quasar.config.js |  | | | 퀘이사 프레임워크 설정 파일 |

# 프레임워크 구조

## 프레임워크 기반 기술

Vue quasar, vscode 등 설명 후 도식 화 필요



|  |  |  |
| --- | --- | --- |
| **No** | **기반 기술** | **내용** |
| 1 | CSS, HTML | Quasar 프레임워크  - 폼 아이템, 색상, 레이아웃 구성 등 |
| 2 | Javascript | Vue 3  - 통신, 폼 아이템 컨트롤 등 |
| 3 | Server | 스프링 프레임워크  - 프론트와는 Rest API 통신을 함 |

### Vue 3

- 공식 문서 : <https://v3-docs.vuejs-korea.org/>

웹 사용자 인터페이스를 만들기 위한 쉽고 강력하며 다재다능한 프레임워크

### Quasar 2

- 공식 문서 : <https://quasar.dev/>

Quasar 프레임워크는 단일 코드베이스로 앱을 빌드하고 SPA, PWA, SSR로 웹에 배포하고 iOS 및 Android용 Cordova를 사용하여 모바일 앱에 배포하기 위한 오픈 소스 Vue.js 기반 프레임워크입니다. - 위키백과

# 프레임워크 컴포넌트

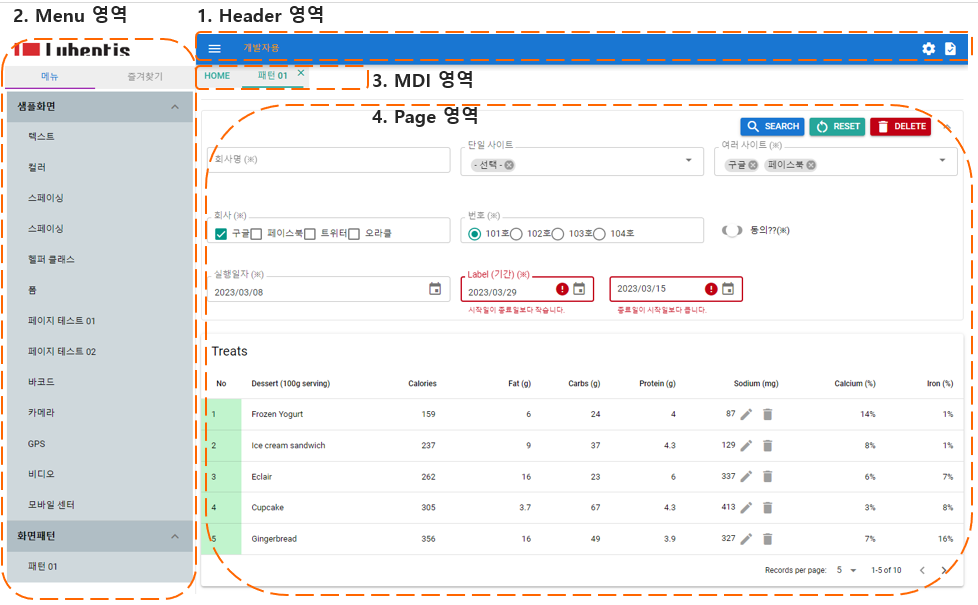
## Main Layout

\* 컴포넌트 구조도

도표이(가) 표시된 사진

자동 생성된 설명

\* 레이아웃 구조도



### Header 영역

시스템의 상태나 개인정보 등을 관리할 수 있는 영역

- 햄버거 메뉴 : 좌측메뉴를 온/오프할 수 있는 버튼

- 시스템 탭 : 대메뉴(시스템) 탭으로 클릭 시 하위의 메뉴가 좌측에 보여진다.

- 설정 : 개인 프로필, 로그아웃 버튼 등 시스템과 관련된 기능이 드롭다운으로 열린다.

- 열어본 페이지 : MDI를 대체할 수 있는 기능으로 열어본 페이지가 리스트 형식으로   
 보여지며 클릭 시 해당 화면으로 이동된다.

### Menu 영역

시스템의 메뉴 구성이 드롭다운으로 보여지게 되며 최대 3단계까지 구성할 수 있다.

- 시스템명 : 아이콘, 시스템명으로 구성할 수 있다.

- 메뉴 : 대메뉴(시스템)의 하위 메뉴

- 즐겨찾기 : 각 페이지에서 즐겨찾기를 선택한 경우 보여진다.

- Open : 모든 메뉴가 보여지거나 접혀진다.

### MDI영역

열어본 페이지를 탭으로 보여주며 이 MDI영역의 Page는 재랜더링되지 않고 상태값을 저장한다. 단 MDI탭을 닫게 되면 캐쉬된 페이지는 삭제된다.

‘홈’화면의 MDI탭은 닫히지 않는다.

/src/boot/appinit.js에서 전역설정으로 통해 사용, 미사용 설정을 할 수 있다.

예) app.config.globalProperties.useMdi = true;

### Page영역

개발화면이 랜더링되는 영역이다.

## 검색영역 컴포넌트

검색영역의 각 아이템 컴포넌트들은 퀘이사의 각 폼 컴포넌트를 재컴포넌트화하였으며

아래의 공통기능을 제공하는 목적으로 생성하였다.

- 디자인 통일성 제공

- 검색값 저장 기능 : ‘searchName’ 요소에 이름 지정 시 브라우저의 Local스토리지에 값을

저장하여 같은 이름의 검색 아이템에는 같은 검색값을 자동 세팅함.

- 필수값 지정 기능 : ‘mandatory’ 요소 지정 시 라벨 표시 변경 및 필수 값 rule 적용

|  |  |  |
| --- | --- | --- |
| **No** | **컴포넌트** | **내용** |
| 1 | QVInput | QInput 구현 |
| 2 | QVSelect | QSelect 구현 |
| 3 | QVCheckbox | QCheckbox 구현 |
| 4 | QVRadio | QRadio 구현 |
| 5 | QVToggle | QToggle 구현 |
| 6 | QVCalendar | QInput, QDate 구현 |
| 7 | QVCalendarPeriod | QInput, QDate 구현 |

### QVInput

Input box 컴포넌트이며 Quasar의 QInput을 재컴포넌트화하였다.

* 레이아웃



* 코드 예시

<QVInput

  v-model="formVal.testInput1"

  title="회사명"

  searchName="TEST"

  mandatory

  :validate="[

    (val) =>

      (val !== null && val.length <= 10) ||

      '10자까지만 입력해주세요.',

  ]"

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- searchName : 검색값 저장 이름

이 값이 설정되어 있을 경우 브라우져의 Local storage에 사용자가

검색한 값이 저장되며 이 값은 같은 화면, 또는 다른 화면에서라도

같은 이름이 있을 경우 자동세팅하여 편의기능을 제공한다.

예를 들어 센터코드를 저장하게 되면 다른 화면에서도 최근에 검색한

센터코드를 자동으로 세팅하여 업무에 편의를 줄 수 있다.

* Local Storage 저장 예시

테이블이(가) 표시된 사진

자동 생성된 설명

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

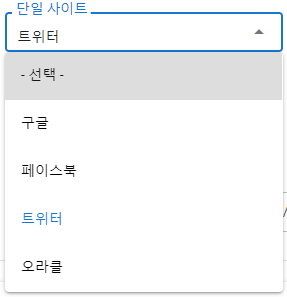
- validate : 입력 폼의 유효성 체크를 설정한다.

\* 참고 : <https://quasar.dev/vue-components/input#internal-validation>

### QVSelect

Select box 컴포넌트이며 Quasar의 QSelect를 재컴포넌트화하였다.

* 레이아웃



* 코드 예시

<QVSelect

  v-model="formVal.siteSelected2"

  title="여러 사이트"

  :options="siteOptions2"

  mandatory

  headerOption="ALL"

  multiple

  searchName="TEST\_SELECT2"

  :validate="[

    (val) =>

      (val !== null && val.length <= 2) ||

      typeof val + ' 2개까지만 선택 가능합니다.',

  ]"

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- options : 셀렉트박스 옵셥을 구성할 배열 데이터

const siteOptions1 = ref([

  {

    label: "구글",

    value: "Google",

  },

  { label: "페이스북", value: "Facebook" },

  { label: "트위터", value: "Twitter" },

  { label: "오라클", value: "Oracle" },

]);

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

- headerOption : 이 요소에 값이 있을 경우 셀렉트박스 옵션 0번째에 배열을 추가한다.

\* SELECT : - 선택 – (값 : “”)

\* ALL : - 전체 – (값 : “”)

- searchName : 검색값 저장 이름

이 값이 설정되어 있을 경우 브라우져의 Local storage에 사용자가

검색한 값이 저장되며 이 값은 같은 화면, 또는 다른 화면에서라도

같은 이름이 있을 경우 자동세팅하여 편의기능을 제공한다.

예를 들어 센터코드를 저장하게 되면 다른 화면에서도 최근에 검색한

센터코드를 자동으로 세팅하여 업무에 편의를 줄 수 있다.

- multiple : 다중 선택 가능

- validate : 입력 폼의 유효성 체크를 설정한다.

### QVCheckbox

check box 컴포넌트이며 Quasar의 QCheckbox를 재컴포넌트화하였다.

* 레이아웃



* 코드 예시

<QVCheckbox

  title="회사"

  v-model="formVal.companyChecked"

  :options="companyOptions"

  mandatory

  searchName="TEST\_CHECKBOX"

  :validate="[

    (val) =>

      (val !== null && val.length <= 2) ||

      '2개까지만 선택 가능합니다.',

  ]"

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- options : 체크박스 옵션을 구성할 배열 데이터

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

- searchName : 검색값 저장 이름

- validate : 입력 폼의 유효성 체크를 설정한다.

### QVRadio

radio box 컴포넌트이며 Quasar의 QRadio를 재컴포넌트화하였다.

* 레이아웃



* 코드 예시

<QVRadio

  title="번호"

  v-model="formVal.numberChecked"

  :options="numberOptions"

  searchName="TEST\_RADIO"

  mandatory

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- options : 라디오박스 옵션을 구성할 배열 데이터

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

- searchName : 검색값 저장 이름

- validate : 입력 폼의 유효성 체크를 설정한다.

### QVCalendar

단일 일자 컴포넌트이며 Quasar의 QInput과 QDate를 재컴포넌트화하였다.

* 레이아웃



* 코드 예시

<QVCalendar

  v-model="formVal.searchDate"

  title="실행일자"

  searchName="TEST\_CALENDAR"

  :validate="['date' || '날짜형식 오류']"

  mandatory

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- searchName : 검색값 저장 이름

- validate : 입력 폼의 유효성 체크를 설정한다.

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

### QVCalendarPeriod

기간 일자 컴포넌트이며 Quasar의 QInput과 QDate를 재컴포넌트화하였다.

* 레이아웃

텍스트이(가) 표시된 사진

자동 생성된 설명

* 코드 예시

<QVCalendarPeriod

  v-model="formVal.searchDate"

  title="실행일자"

  searchName="TEST\_CALENDAR"

  :validate="['date' || '날짜형식 오류']"

  mandatory

/>

- v-model : 폼 input과 스크립트간 양방향 데이터 바인딩

- title : 타이틀

- searchName : 검색값 저장 이름

- validate : 입력 폼의 유효성 체크를 설정한다.

- mandatory : 이 요소가 있을 경우 타이틀에 “(※)”를 자동으로 붙여주며, 필수체크 룰도 자동으로 추가해준다.

- 기간검색기는 기간의 선후를 기본적으로 체크한다.

## 다이얼로그 컴포넌트

사용자와의 알림, 결정 등을 대화하기 위한 다이얼로그 창이며 디자인과 기능의 공통화를 위해 생성하였다.

- 아래 컴포넌트 대신 “$q.dialog”로 대체함.

- 참고 : https://quasar.dev/quasar-plugins/dialog

|  |  |  |
| --- | --- | --- |
| **No** | **컴포넌트** | **내용** |
| 1 | QVAlert | QDialog 구현  - 닫기버튼 또는 Esc 키로 창 닫기 기능 제공  - 헤더와 내용 Slot으로 상황에 맞게 사용 가능 |
| 2 | QVConfirm | QDialog 구현  - 닫기버튼 또는 Esc 키로 창 닫기 기능 제공  - 헤더와 내용 Slot으로 상황에 맞게 사용 가능  - “OK”클릭 시 confirmOK 이벤트 발생 |

## 페이지 컴포넌트

페이지 헤더에 공통기능을 위한 컴포넌트이다.

|  |  |  |
| --- | --- | --- |
| **No** | **컴포넌트** | **내용** |
| 1 | QVPageHeader | QBreadcrumbs구현  - 메뉴의 breadcrumbs 에 의해 메뉴 경로 표시  - 즐겨찾기 설정 기능 |
|  |  |  |

## 팝업 컴포넌트

모달창을 위한 공통 컴포넌트이다.

|  |  |  |
| --- | --- | --- |
| **No** | **컴포넌트** | **내용** |
| 1 | QVModal | QDialog 구현  - 모달창 이동 기능  - 오픈 위치 지정 기능  - 창 크기 지정 기능  - 헤더와 내용 Slot으로 상황에 맞게 사용 가능 |
|  |  |  |

# 프레임워크 기능 사용

## 프레임워크 기능 사용

### 클라이언트, 서버 데이터 통신 – Axios

캡쳐 후 각 영역 및 기능에 대한 설명

### 페이지 이동 – Vue router

Router를 이용하여 페이지간 네비게이션을 구현하며, 메뉴는 사용자 권한별로 달라질 수 있으므로 아래의 2단계로 처리한다.

1. routes.js : 메인레이아웃 ,홈 화면, 에러 화면 route 설정

텍스트이(가) 표시된 사진

자동 생성된 설명

1. MainLayout 컴포넌트 오픈 시 메뉴정보를 DB에서 조회하여 menuList를 세팅한다. 세팅된 메뉴는 동적으로 Router에 추가된다.  
   - title : 메뉴 이름  
   - explantation : 메뉴 설명으로 좌측 메뉴 하단에 보여진다.  
   - link : route path 설정  
   - component : 랜더링될 Vue 페이지로 절대경로로 설정한다.  
   - breadcrumbs : 메뉴 경로로 ‘|’로 구분한다.  
   - subMenuList : 하위 메뉴가 있을 경우 추가한다.

### 개인 편의성 데이터 저장 – Local Storage

캐쉬와는 다르게 사용자가 삭제하거나 프로그램으로 삭제하지 않는 한 유효기간이 없으며 저장용량도 크다. 이 로컬 스토리지에는 검색값 저장 등 보안과 관련없는 편의를 위한 값이 저장되어야 하며 일반적으로 로컬 스토리지는 문자열 형태로 저장되지만, 퀘이사에서는 각 변수 타입을 고려하여 저장된다

### 컴포넌트용 전역 상태 관리 – Piana

캡쳐 후 각 영역 및 기능에 대한 설명

### Vue App 초기화 실행 – Quesar boot

퀘이사에서는 App초기화를 위하여 boot 모듈을 이용한다.

- /src/boot/ 하위에 초기화할 스크립트를 생성한다.

- quasar.config.js 의 boot: [""] 에 확장자를 제외한 파일명을 추가한다.

### 공통 Javascript 호출 및 실행

각 페이지에서 사용할 수 있는 공통함수를 생성하고 호출할 수 있다.

- /src/scripts/ 에 공통함수 스크립트를 생성한다.

- 페이지에서 해단 스크립트를 import하여 함수를 호출한다.

### 전역 컴포넌트 등록

자주 사용되는 폼 컨트롤 컴포넌트 등은 페이지에서 별도로 import하지 않게 전역 컴포넌트로 등록한다.

- 퀘이사 boot를 이용하여 등록함.

### 검색 실행(<q-form>)

QVInput 등과 같은 검색 아이템 컴포넌트가 하위에 위치한다.

          <q-form

            class="column q-gutter-md"

            ref="searchForm"

            greedy

            autofocus

            @submit="onSubmit"

          >

- class : css class 명

- ref : form 컨트롤을 위한 폼 명

- greedy : 폼의 모든 필드 유효성 검사

- autofocus : 첫번째 검색 컴포넌트에 포커스 위치

- submit : Submit 버튼 클릭 시 이벤트 핸들러

### 페이지 내 영역 분리(<q-splitter>)

페이지 컨텐츠가 상하, 좌우 또는 조합되어 분리가 필요한 경우 QSpliitter를 이용하여 분리한다.

Slot으로 컨텐츠를 분리하고 아이콘을 추가하여 분리영역을 나타낼 수 있다.

    <q-splitter v-model="splitterModel" style="height: 500px">

      <template v-slot:before>

        <Page02 />

      </template>

      <template v-slot:separator>

        <q-avatar size="40px" icon="more\_vert" />

      </template>

      <template v-slot:after>

         <Page02 />

       </template>

    </q-splitter>

### 그리드 (<q-table>)

그리드 형태의 데이터를 화면에 제공할 경우 QTable을 이용하여 구현한다.

Quesar 공식문서를 참조한다.

    <q-table

      class="my-sticky-header-column-table"

      title="Treats"

      :rows="rows"

      :columns="columns"

      row-key="name"

      @row-click="rowClick"

    >

### 파일 업로드 (<q-uploader>, <q-file>)

단일, 다중 파일 업로드를 할 경우 QUploader 또는 QFile을 각 화면의 디장인에 맞게 이용한다.

### 파일 다운로드

캡쳐 후 각 영역 및 기능에 대한 설명

## 프레임워크 개발 규칙

### 파일명

캡쳐 후 각 영역 및 기능에 대한 설명

### 코드 내 규칙

캡쳐 후 각 영역 및 기능에 대한 설명

### 변수명 규칙

캡쳐 후 각 영역 및 기능에 대한 설명

### 객체 및 함수 규칙

캡쳐 후 각 영역 및 기능에 대한 설명

| 용어 | | 내용 |
| --- | --- | --- |
| 단어 | | 테이블과 컬럼의 한글명 및 영문명을 구성하는 최소한의 원소를 가리키며 용어는 반드시 이러한 단어의 조합으로 이루어 짐 |
|  | 단일어 | 다른 단어와의 조합이 아닌 순수한 단어 |
| 합성어 | 일반단어나 접두[미]사의 조합으로 이루어진 단어 |
| 유사어 | 표준 한글단어와 함께 사용이 허락된 단어. 즉 사용 할 수는 있으나 제시된 표준단어를 사용하도록 권장 함 |
| 금칙어 | 사용이 허락되지 않는 단어 |
| 용어 | | 단어의 조합으로 이루어져 테이블 및 컬럼의 명칭으로 사용되는 용어 |
| 도메인 | | 속성의 일반적인 특징을 나타내는 데이터의 성격을 분류한 것으로써 속성이 취할 수 있는 값의 집합이라 할 수 있다. |

## 한글 용어(논리명) 표준정의 원칙

### 한글용어(논리명) 기본원칙

* 누구나 이해하기 쉽도록 간결하고 명확한 표현을 사용한다.

(예) 결정기준(X) 🡪 가격결정기준(O)  
 계좌표기여부(X) 🡪 계좌번호표기여부(O)

* 반드시 단어사전에 등록된 단어를 사용하며 단어사전에 등록되어 있지 않은 경우, 단어 등록이 선행되어야 한다.
* 한글 및 영문 약어 사용시는 띄어쓰기를 하지 않고 Under Bar(\_) 및 특수문자(/, \_, -, +, (, ))를 사용하지 않는다. 단 ‘/’가 포함된 영문단어를 사용할 경우만 ‘/’를 허용한다.

(예) 거래 일자(X), 거래\_일자(X) 🡪 거래일자(O)  
P/L 코드(X), P/L\_코드(X) 🡪 P/L코드(O)

* 한글사용을 원칙으로 하며, 불가피한 경우 영문을 사용한다. 영문만으로 용어를 구성할 경우에만 Space를 허용하며 대문자로 표현한다.

(예) Campaign Type Code(X) -> CAMPAIGN TYPE CODE(O)

* 하나의 용어가 여러 의미를 가지는 용도로 사용하지 않음을 원칙으로 하나, 불가피할 경우 ‘및’을 사용하여 명명할 수 있다. 영문명 변환시 ‘및’은 변환하지 않는다. 3개 이상의 의미를 가지는 속성명의 경우 주로 엔티티을 구분하는 식별번호성으로 사용되므로 도메인을 “식별번호”로하여 용어를 정의한다.

(예) 하나의 컬럼에 주민번호, 사업자번호가 관리되는 경우 관용적으로 사용되는 “주민사업자번호”라고 명명하여도 무방하다.

* 한글 글자수는 영문전환시 30자 이내가 되도록 정의 한다.  
  기본 영문약어의 수를 6자로 정의하며, 5개까지의 단어를 조합하여 용어를 생성 할 수 있다.  
   = 6\*5+4 (약어기본수 6, 5개단어조합, ‘\_’4개)

(예) 구매+업체+은행+지점+명 -> Purchase+Enterprise+Bank+Branch Office+Name  
=> PCH\_ENTR\_BANK\_BROFC\_NM

고가품 + 구분 + 코드 -> High-priced Article + Section + Code   
=> HPRCAT + SCTN + CD

* 특정한 용어를 제외한 일반적인 명사를 사용하는 경우 반드시 업무적인 의미를 부여한다.

(예) 상품구분 -> 상품종류구분, 상품거래방법

## 영문 용어(물리명) 표준정의 원칙

### 영문용어(물리명) 기본원칙

* 정의된 한글명에 대하여 등록된 단어단위로 영문약어를 사용하여 자동 전환하며 단어구분자로 Under Bar(\_)를 사용한다.
* 합성어를 단일어 보다 우선 적용한다

### 테이블명 명명규칙

* 테이블명은 한글엔티티명에서 변환된

‘업무구분 Level 2’ + ‘\_’ + 업엔티티명 + [테이블유형영문약어](영문 1자리)으로 정의한다

(예) 운송관리\_고객기본 : TM\_고객정보이력(엔티티명) -> TM\_CUST\_H(테이블명)

* ‘\_’를 포함하여 30Byte이내가 되도록 한다.

### 컬럼명 명명규칙

* 전환된 물리명은 Under Bar(\_)를 포함하여 30Byte 이내로 구성한다.
* 생성된 영문명이 30자를 넘을 경우에는 컬럼생성용영문명을 우선 적용한다.
* 컬럼명은 용어사전에 등록된 용어로 유일성을 유지해야 한다.  
  해당 한글 속성명에 대하여 영문 컬럼명은 반드시 하나로 정의되어야 한다.

### 기타 물리 데이터 오브젝트 명명규칙

기타 물리데이터 오브젝트는 DBMS별로 정의 되는 형식이 다를 수 있으며, 아래의 명명규칙은 특정 DBMS에 관계없이 정의된 것이다. 실제 물리DB에서 정의할 때, DBMS의 특징에 의하여 추가 정의할 필요가 있는 경우 [표4-5]에 추가 구분으로 등록하여 명명 규칙을 정의한 후에 사용한다.

| 구분 | 설명 | | | |
| --- | --- | --- | --- | --- |
| DB명 | DB \_ A  ① ② | | | |
| ① | 약칭 | Data Base의 ‘DB’사용 | |
| ② | DB명 | DB명은 각 시스템별로 DB가 하나일 경우 가급적 시스템의 3자리 영문 약어를 사용한다.(1개 이상일 경우 해당 DB의 특성에 맞는 영문 약어를 사용한다.) | |
| 예 | DB\_XXX | | |
| Table Space 명 | TS A \_ B #  ① ② ③ ④ | | | |
| ① | 약칭 | Table space의 ‘TS’사용 | |
| ② | Table space 구분 | Data 용: D, Index 용: I, Temp용:T | |
| ③ | 주제영역영문약어명 | 주제영역영문약어명 | |
| ④ | 일련번호 | 두자리 숫자를 사용 | |
| 예 | 상품관리의 데이터 테이블 스페이스: TSD\_TMO01  예외사항 : 크기가 큰 Table은 별도의 Table space에 저장하며  이런 경우는Tablespace ‘③ 업무영역코드’ 는 Table명에서 ‘TB’와 ‘\_’를 제외한 명과 같다.  (예) TB\_XXXX\_CUST\_M 을 저장하는 Tablespace : TSD\_XXCUSTM01 | | |
| Data File 명 | df A \_ B \_ # . dbf  ① ② ③ ④ | | | |
| ① | 약칭 | Data File의 ‘df’ 사용 | |
| ② | Table space 구분 | Data 용 : d , Index 용 : I, Temp용:t | |
| ③ | Table space명의Second Node | Table space명의‘TSD\_’및’TSI\_’를 제외한  부분 사용 | |
| ④ | 일련번호 | 두자리 숫자를 사용 | |
| 예 | dfd\_afc01\_01.dbf : TSD\_업무약칭01의 첫번째 데이터 파일명 | | |
| Cluster 명 | CL \_ A #  ① ② ③ | | | |
| ① | 약칭 | Cluster ‘CL’ 사용 | |
| ② | 주제영역영문약어명 | Cluster가 속한 주제영역영문약어명 | |
| ③ | 일련번호 | 두자리 숫자를 사용 | |
| 예 | XXXX업무에서 사용하는 Cluster : CL\_업무약칭O01 | | |
| Partition Table Segment 명 | PT \_ A \_ #  ① ② ③ | | | |
| ① | 약칭 | Partition Table의 ‘PT’ 사용 | |
| ② | Table 명 | Table명에서 ‘TB\_’를 제외한부분 | |
| ③ | 일련번호 | 두자리 숫자를 사용 | |
| 예 | PT\_XXXX\_CUST\_M\_01 : TB\_XXXX\_CUST\_M 테이블의 첫번째 Partition Table 명 | | |
| Lob Segment 명 | LS \_ A \_ #  ① ② ③ | | | |
| ① | 약칭 | Lob Segment의 ‘LS’ 사용 | |
| ② | Table 명 | Table명에서 ‘TB\_’를 제외한부분 | |
| ③ | 일련번호 | 두자리 숫자 | |
| 예 | LS\_XXXX\_CUST\_M\_01 : TB\_XXXX\_CUST\_M 테이블의 Lob Segment 명 | | |
| Index 명 | A \_ C \_ # D   1. ② ③ | | | |
| ① | 인덱스 구분 | Primary Key : PK  Alternative Key : AK+일련번호  Inversion Key : IE+일련번호 | |
| ② | Table 명 인덱스 구분 | Table 명 | |
| ③ | Partition 구분 | Partition Table의 인덱스인 경우에 만 사용  L : Local Index, G : Global | |
| 예 | TB\_XXXX\_CUST\_M 의 Primary Key : PK\_TB\_XXXX\_CUST\_M  TB\_XXXX\_CUST\_M 의 Index : IX01\_TB\_XXXX\_CUST\_M | | |
| Partition Index Segment 명 | A \_ B \_ #  ① ② | | | |
| ① | Partition Table Segment 명 | | Partition Table Segment 명 |
| ② | 인덱스 구분 | | Primary Key : PK,  Index : IX+’\_’+일련번호 |
| 예 | TB\_XXXX\_CUST\_M PK의 Partition Index Segment명: PT\_XXXX\_CUST\_M\_01\_PK명 | | |
| Foreign Key 명 | A \_ B \_ F#  ① ② ③ | | | |
| ① | Table 명 | Table 명 | |
| ② | Parent Table 명 | ‘TB\_’를 제거한Parent Table 명 | |
| ③ | 구분 | F + 일련번호(한자리 숫자) | |
| 예 | TB\_XXXX\_CUSTHIST\_H 이 TB\_XXXX\_CUST\_M 를 참조할 경우:  TB\_XXXX\_CUSTHIST\_H\_BFTO\_CUST\_M\_F1 | | |
| Check Constraint 명 | A \_ B \_ CK  ① ② ③ | | | |
| ① | Table명 | Table명 | |
| ② | Column명 | Constraint를 거는 컬럼 이름을 사용 | |
| ③ | 약칭 | Check Constraint 의 ‘CK’ | |
| 예 | TB\_XXXX\_CUST\_M 의 상태 컬럼의 Check Constraint: TB\_XXXX\_CUST\_M\_STTS\_CK | | |
| Trigger 명 | TG \_ A \_ B C D  ① ② ③ ④ ⑤ | | | |
| ① | 약칭 | Trigger의 ‘TG’ 사용 | |
| ② | Table 명 | Table명에서 ‘TB\_’를 제외한부분 | |
| ③ | Time 구분 | After : A , Before : B | |
| ④ | Transaction 구분 | Insert : I, Update : U, Delete : D | |
| ⑤ | Access Level 구분 | Row : R, Statement : S | |
| 예 | TG\_XXXX\_CUST\_M\_AIR : TB\_XXXX\_CUST\_M 의 Insert 후 Row Level Trigger | | |
| Sub Program 명 | AB \_ C \_ E  ①② ③ ④ | | | |
| ① | suffix | Function: ‘FN’, Procedure: ‘ SP’  DB IO용 Package: ‘DBP’,  Package : ‘PA’, Package Body : ‘PB’ | |
| ② | 구분 | 업무공통모듈: ‘CM’,  RuleBase: ‘RB’, DB IO Module : ‘DC’ | |
| ③ | 주제영역영문약어명 | 주제영역영문약어명 | |
| ④ | 명 | Free Name | |
| 예 | 고객여부 확인 Procedure : SP\_ CM\_XXXX\_CUSTCHK | | |
| VIEW 명 | VW \_ A \_ B  ① ② ③ | | | |
| ① | 약칭 | View의 ‘VW’ 사용 | |
| ② | 주제영역영문약어명 | 주제영역영문약어명 | |
| ③ | 명 | 영문 뷰명 | |
| 예 | 계정계외환고객\_고객뷰 : VW\_XXXX\_CUST | | |
| Sequence 명 | SQ \_ A \_ B  ① ② ③ | | | |
| ① | 약칭 | Sequence의 ‘SQ’ 사용 | |
| ② | Table 명 | Table명에서 ‘TB\_’를 제외한부분 | |
| ③ | Column명 | 컬럼 명 | |
| 예 | SQ\_XXXX\_CUST\_M\_CUST\_NO : TB\_XXXX\_CUST\_M의 고객번호에 대한 Sequence 명 | | |
| Synonym 명 | SYN\_A  ① | | | |
| ① | Synonym 명 | 소유자를 제외한 Object명(테이블,procedure등) | |
| 예 | Scott 사용자가 생성한 emp 테이블을 Tiger 사용자가 사용할 경우 Tiger도 emp 테이블 명으로 사용할 수 있게 하기 위해 Synonym을 생성  Create synonym SYN\_emp for ‘scott.emp’ ; | | |
| TYPE 명 | TP \_ A #  ① ② ③ | | | |
| ① | 약칭 | Type의 ‘TP’ 사용 | |
| ② | 주제영역영문약어명 | 주제영역영문약어명 | |
| ③ | 일련번호 | 두자리 숫자 | |
| 예 | 업무에서 에서 Type을 정의한 경우 : TP\_ XXX01 | | |
| DB Link 명 | DL \_ A  ① ② | | | |
| ① | 약칭 | Database Link의 “DL’ 사용 | |
| ② | DB명의 Second Node | 연결하고자 하는 DB 중 ‘DB\_’를 제외한 부분 사용 | |
| 예 | 특정업무DB를 Link하는 경우 : DL\_XXXX | | |
| Snapshot 명 | SN \_ A  ① ② | | | |
| ① | 약칭 | Snapshot의 ‘SN’ 사용 | |
| ② | Table 명 | Table명에서 ‘TB\_’를 제외한부분 | |
| 예 | SN\_XXXX\_CUST\_M : TB\_XXXX\_CUST\_M 의 Snapshot 명 | | |

[표 4‑3] 물리 오브젝트 별 명명 규칙

## 요약

### 테이블명 등 예시 요약

| 구분 | 설명 |
| --- | --- |
| 테이블 | 업무구분(2) + 엔티티명 + [유형요약(1)]  MST : 마스터 / DTL : 상세 |
| 예) 오더관리 이력 : OM\_ORDER\_MST\_H |
| 컬럼 | XXXX\_CD (코드), XXXX\_NM(코드명), XXX\_YN(NULL사용금지),  XXXX\_YYYYMMDD(년월일), XXXX\_SEQ(순번),XXXX\_DATE(DATE), |
| 예) 아이템 순번 : ITEM\_SEQ |
| PK / INDEX | 테이블명 + PK [IX+순번(2)] |
| 예) OM\_ORDER\_MST\_ IX01 |
| 트리거 | TR\_테이블명 + [작업요약(3)]  Time 구분 After : A , Before : B  Transaction 구분 Insert : I, Update : U, Delete : D  Access Level 구분 Row : R, Statement : S |
| 예) TR\_OM\_ORDER\_MST\_AIR : OM\_ORDER\_MST 의 Insert 후 Row Level Trigger |
| 프로시져/펑션/패키지 | 오브젝트구분(2) + 업무구분(2) +작업명  Function: 'FN', Procedure: ' SP'  DB IO용 Package: 'DBP',  Package : 'PA', Package Body : 'PB' |
| 예) 오더관리 작업 프로시져 : SP\_OM\_ORDER\_UPDATE |
| 뷰 | VW+ 업무구분(2) + 작업명 |
| 예) 오더관리 VIEW : VW\_OM\_ORDER\_MGT |
| 시퀀스 | SQ+ 테이블명 + 컬럼명 |
| 예) 오더관리 순번: SQ\_OM\_ORDER\_MST\_ORDER\_SEQ |
| Synonym | SYN + 테이블명 |
| 예) SYN\_OM\_ORDER\_MST |
| DB링크 | DL + DB명 |
| 예) KRCC DB LINK : DL\_KRCC |
|  |  |
|  |
|  |  |
|  |

1. **프로그래밍 표준**

**2.1.** **포맷 규칙(Formatting rule)**

코드표준화는 코드명, 코드값의 표준화를 의미한다.

코드표준화는 코드와 관련된 기본적인 부여방식이나 코드의 유형들을 분류하고 정의하며 중복되는 코드와 코드 부여방식의 원칙에 위배되는 사항을 표준화 하고 유사한 코드를 추출해서 코드 표준화를 한다.

* + 1. 들여쓰기

들여쓰기는  코드에서  블록의  시작과  끝을  쉽게  알아볼  수  있게  해준다.  들여쓰기  되었을  때,  더욱  
읽기  좋고,  더  많은  단계의  블록이  다른  블록  내부에  포함되어  있을  때,  들여쓰기의  필요성은  더욱  
강조된다. 들여쓰기  및  탭  크기는  기본적으로  4  글자로 한다.  탭  문자는  사용하지 않는  것을  원칙으  
로 한다. 다음은 들여쓰기 예이다.

**[출처]** [[펌] Java & JSP 코딩 가이드 (헤르메스의 날개)](http://cafe.naver.com/hermeswing/507) |**작성자** [헤르메스](http://cafe.naver.com/hermeswing.cafe?iframe_url=/CafeMemberNetworkView.nhn%3Fm=view%26memberid=hermeswing)

들여쓰기는 코드에서 블록의 시작과 끝을 쉽게 알아볼 수 있게 해준다. 들여쓰기 되었을 때, 더욱 읽기 좋고, 더 많은 단계의 블록이 다른 블록 내부에 포함되어 있을 때, 들여쓰기의 필요성은 더욱 강조된다. 들여쓰기 및 탭 크기는 기본적으로 1탭(4공백)으로 한다. 다음은 들여쓰기 예이다.

1. main() {  
           for( ; ; ) {  
           }  
       }

**[출처]** [[펌] Java & JSP 코딩 가이드 (헤르메스의 날개)](http://cafe.naver.com/hermeswing/507) |**작성자** [헤르메스](http://cafe.naver.com/hermeswing.cafe?iframe_url=/CafeMemberNetworkView.nhn%3Fm=view%26memberid=hermeswing)

main()

{

For( ; ; )

{

For( ; ; )

{

}

}

}

* + 1. 라인 작성 기준

1024\*768 화면 기준으로 각 라인은 한 화면에 가급적 전부 보이도록 100 컬럼이내로 짧게 작성하며, 코드의 길이가 긴 라인은 두 줄에 걸쳐서 작성한다. 한 줄에 꼭 하나의 명령문만을 기술하여야 한다.

복잡한 수식을 가진 긴 라인이라면 여러 개의 작은 수식으로 분할하여 작성한다.

라인 뒤쪽은 공백이 없게 한다.

double length = Math.sqrt( Math.pow(Math.random(),2.0) + Math.pow(Math.randown(),2,0) ) ; 은 아래와 같이 분리한다.

1. double xSquared = Math.pow(Math.random(), 2.0 ) ;

double ySquared = Math.pow(Math.random(), 2, 0 ) ;

double length = Math.sqrt(xSquared + ySquared) ;

* 1. **클래스 명명 규칙**
     1. 접미사
  2. Service : XxxxService.java
  3. VO : XxxxVO.java
  4. Implement : XxxxImpl.java
  5. DAO : XxxxDAO.java
  6. **메소드 명명 규칙**
     1. 메소드 명명 규칙
  7. **변수 명명 규칙**
     1. 상수

상수 이름은 항상 대문자를 사용하되 각 단어 혹은 어절의 사이에는 밑줄(\_)로 연결한다. 상수에 대문자를 사용하는 것은 다른 종류의 명칭들과 구별하기 위함이다.

1. **소스 파일 작성 규칙** 
   1. HTML, JS, JAVA

.html, .js, .java 등 소스 파일은 파일의 최상위 부분에 그 파일에 대한 정보를 표시해주어야 하며 다음과 같은 주석을 작성한다.

/\*

\* 작성일자 : 2001-05-25

\* 작 성 자 : 홍길동

\* 파일이름 : file.js

\* 수정이력 : 2001-05-25 최초 작성

\* 설 명 : 프로젝트명, 프로그램명, 소스 용도, 공통 시스템 혹은 단위 시스템 설명.

\*/

* 1. **주석**

주석은 시스템 유지보수시 빠르게 현황을 파악할 수 있는 중요한 도구이므로

가능한 자세한 설명을 첨부한다.

Jsp의 경우 주석은 jsp 주석안에 기술한다.

<%-- jsp 주석 기술 부분 --%>

* + 1. 한줄 주석

소스 한 라인에 대한 주석문 혹은 변수의 선언 후에 변수에 대한 설명을 짧게 적을 경우에 문자의 오른편에 기술한다.

private PageContext pageContext; // JSP페이지의 모든 객체정보를 가지고 있다.

private long startTime; // 시스템의 현재의 시간을 가져온다.

* + 1. 여러줄 주석

주석을 항목 위에 붙이는 경우는 두 라인 이상의 명령문에 대한 해석이나 메소드에 대한 설명을 붙이는 경우이다. 주석 내용의 위 아래에 단일선으로 시작과 끝을 표시하도록 한다.

./\*\*

\* 디버그 빈이 실행된 시간을 가져온다.

\*/

public DebugBean()

{

startTime = System.currentTimeMillis();

}

* + 1. 코드명 부여원칙 예외

기존 As-Is에서 특별한 의미를 가지는 코드로 사용되었을 경우 그대로 채택한다.

| **코드명** | **코드값** |
| --- | --- |
| KRW | 원화 |
| USD | 미국 달러 |
| HKD | 홍콩달러 |
| DEM | 독일마르크 |

[표 3‑4] 코드부여원칙 예외: 통화코드

* 목록성 코드의 경우 표준화 대상에서 제외
* 외부기관에서 정의되어진 코드의 경우 표준화 대상에서 제외
* 현재 사용중인 코드 중 향후 전사적 차원에서 표준화가 논의 되어야 할 코드의
* 경우 표준화를 하지 않는다.
  1. **선언부**
     1. import

자바 표준 API 패키지를 먼저 기술하고, 커스텀 패키지를 나중에 기술한다.

표준 API외에는 원칙적으로 class는 full path를 기술한다.

import java.util.\*;

import javax.servlet.\*;

import javax.servlet.http.\*;

import javax.servlet.jsp.\*;

import daelim.logis.tms.basic.HBLManagement;

[출처] [펌] Java & JSP 코딩 가이드 (헤르메스의 날개) |작성자 헤르메스

* + 1. 대소문자 규칙
* SQL 문은 대문자로 기술하며, 할당되는 변수는 PARAMETER 와 일치시킨다.
* HTML TAG는 대문자로 기술한다.

1. **산출물**
   1. 디렉토리 정의

산출물은 프로젝트 단계별로 구분하며 각 단계별 산출물 디렉토리 구조는 아래와 같다.

**├─0. 프로젝트 관리**

│ ├─10. 착수

│ │ ├─10. 계약관리

│ │ └─20. 프로젝트 수행환경 구성

│ ├─20. 계획 수립

│ │ ├─10. 사업수행계획 수립

│ │ ├─20. WBS

│ │ └─30. 산출물 목록

│ ├─30. 실행 및 통제

│ ├─40. 보고회

│ ├─50. 적용

│ └─60. 종료

**├─1. 현황분석 및 개발환경구성**

│ ├─10. 개발환경구성

│ ├─20. 요구사항 정의

│ └─30. 현황분석

**├─2. 설계**

│ ├─10. 프로세스 설계

│ ├─20. 프로그램 설계

│ ├─30. 인터페이스 설계

│ └─40. 데이터모델 설계

**├─3. 구축**

│ ├─10. 서버프로그램구축

│ ├─20. 모바일프로그램구축

│ └─30. 인터페이스프로그램구축

**├─4. 테스트**

│ ├─10. 단위테스트 계획수립

│ ├─20. 통합테스트 계획수립

│ ├─30. 단위테스트

│ └─40. 통합테스트

**├─5. 이행 및 적용**

│ ├─10. 전확계획수립

│ ├─20. 운영적용

│ ├─30. 교육준비

│ └─40. 운용자교육

**├─6. 안정화**

│ └─10. 시스템운용

* 1. **명명 규칙**

산출물 파일명은 아래와 같이 규정하여 체계화하며 이를 통해 개발자간 공유가 원활하도록 한다.

* + 1. 명명 규칙

구조) 산출물 ID + 산출물명 + 버전 + 확장자

예) 110-01 개발표준 가이드 v0.5.docx

* 산출물 ID : 디렉토리 ID + 순번

예) 1. 현황분석 및 개발환경구성\10. 개발환경구성\첫번째 산출물

=> 110-01

* 산출물명 : 산출물명

예) 개발표준 가이드

* 버전 : 산출물 버전
* 예) v0.5

Boot

앱 초기 변수등 추가

/src/boot/아래에 js추가

추가한 파일명은 quasar.config.js의 boot[] 에 추가

참고 : 프로젝트 구조 - Scaffolding 된 구조 살펴보기 (04:00) ~

환경변수 파일

참고 : [https://gymcoding.teachable.com/courses/vue3-practice/lectures/40756999 (2](https://gymcoding.teachable.com/courses/vue3-practice/lectures/40756999%20(2)  
:25)~

아이콘 :

[https://gymcoding.teachable.com/courses/quasar/lectures/43736664 1:50](https://gymcoding.teachable.com/courses/quasar/lectures/43736664%201:50)

공통 컴포넌트 등록

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40757254>

퀘이사 로컬 스토리지

[https://gymcoding.teachable.com/courses/quasar/lectures/43736664 11](https://gymcoding.teachable.com/courses/quasar/lectures/43736664%20%2011)분

퀘이사 부트파일

[https://gymcoding.teachable.com/courses/quasar/lectures/43736606 3](https://gymcoding.teachable.com/courses/quasar/lectures/43736606%203)분30초

@:keyup:enter=”submit”

<https://bangj.tistory.com/78>

히스토리 모드

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40756930>

PINIA

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40757258>

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40757257>

scss 변수

<https://quasar.dev/style/sass-scss-variables>

[https://gymcoding.teachable.com/courses/quasar/lectures/43736658 8](https://gymcoding.teachable.com/courses/quasar/lectures/43736658%208)분

퀘이사 파일 디렉토리 알리아스

[https://gymcoding.teachable.com/courses/quasar/lectures/43736606 9](https://gymcoding.teachable.com/courses/quasar/lectures/43736606%209)분 30초

공통함수 (Composable 함수)

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40757249>

알림 등 공통 컴포넌트 위치

<https://gymcoding.teachable.com/courses/vue3-practice/lectures/40757249>

8분 30초