

시훈민정음 교육시스템
프로젝트 페이지 등록 가이드

목차

1. 실행 방법
2. 파일 경로 규칙
3. API 요청 route 규칙
4. 스타일 추가 및 변경 방법

1. 실행 방법

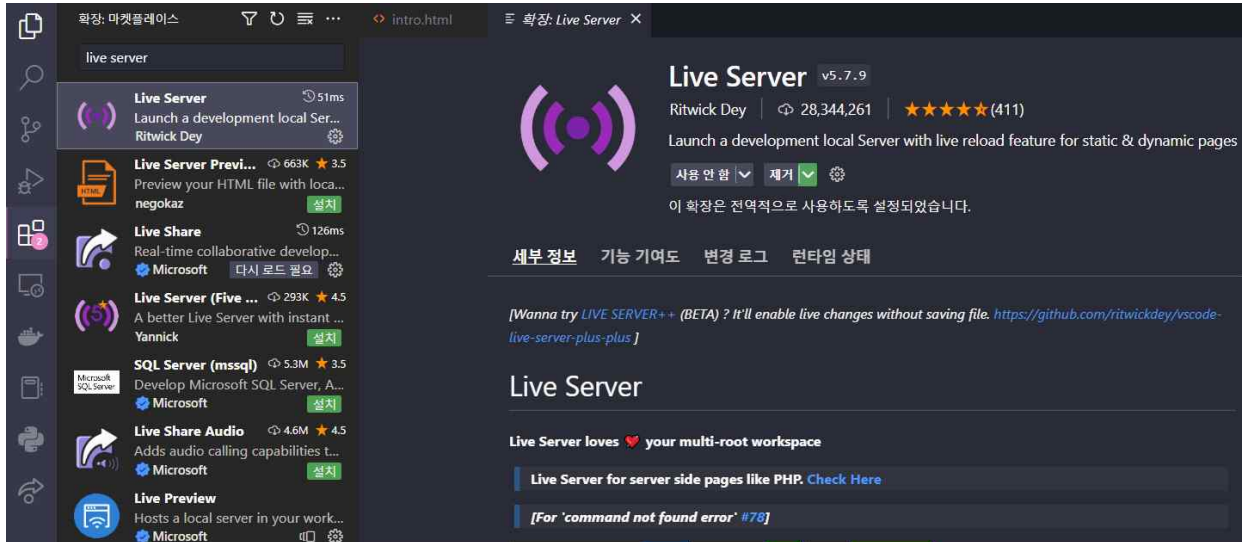
1. code editor

가. VSCode

2. 확장 기능 - Live Server

실행 환경은 Live Server로 대체한다.

가. 설치

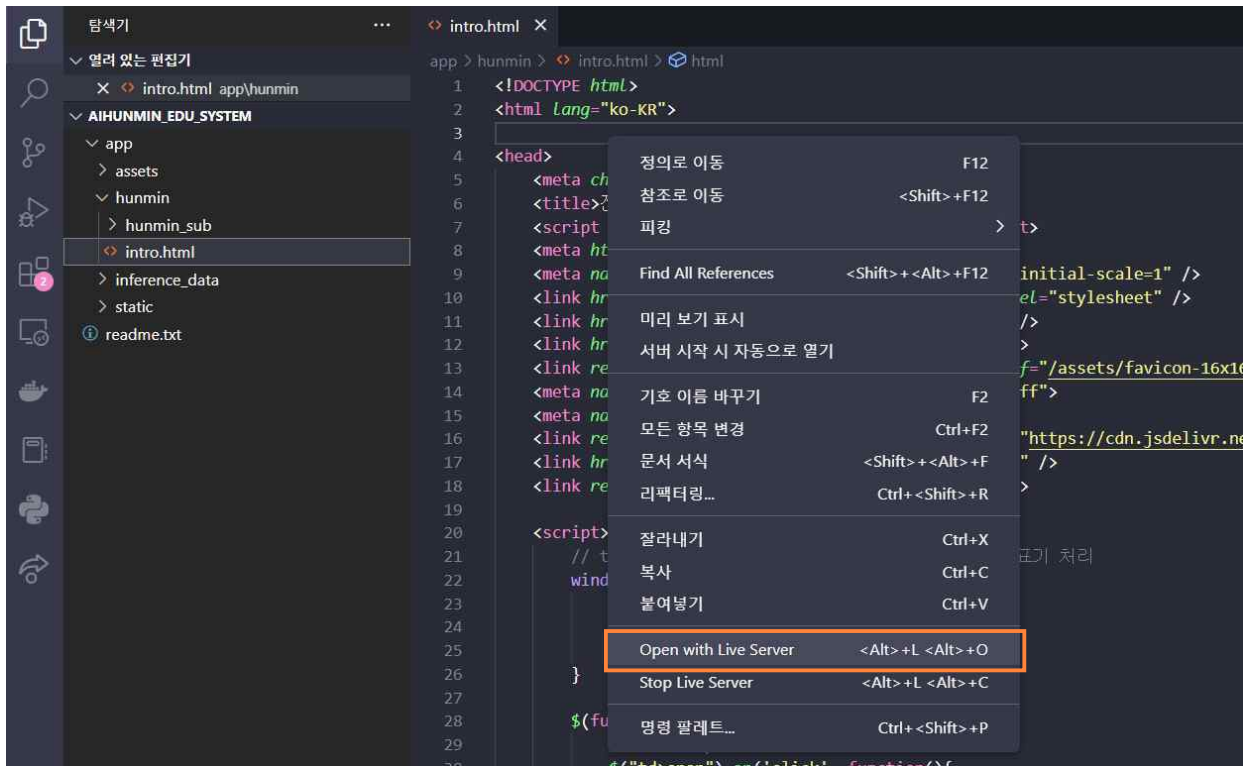


VSCode에서 ctrl + shift + x 단축키로 확장 기능 설치 메뉴로 이동한다.

live server를 검색하고 해당 기능을 설치한다.

나. 실행

전달받은 app폴더를 VSCode로 열고, 확인하고싶은 html파일을 연다.



선택된 html파일에서 우클릭하여 Open with Live Server를 실행한다.

다. 확인

Chrome 브라우저 기준으로 작성된 web이기 때문에 Chrome 브라우저를 설치해야 정상적인 페이지가 보인다.

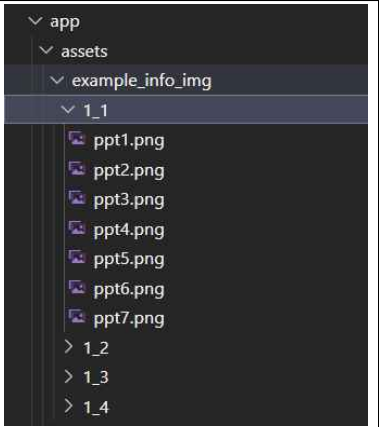
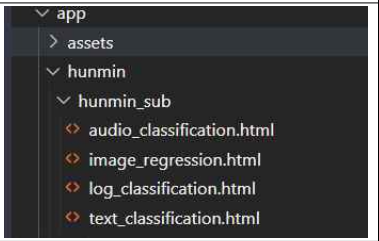
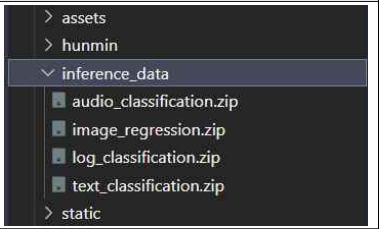
Open with Live Server를 실행한 상태라면 html 파일을 저장할 때마다 변경사항이 적용되어 웹에 표시된다.

2. 파일 경로 규칙

- 1. 시스템 구성
- 가. 전체 경로 구성

```
app
├── asserts
│   └── example_info_img
├── hunmin
│   └── hunmin_sub
├── intro.html
├── inference_data
└── static
```

나. 경로별 규칙

example_info_img	소개자료 ppt를 png형태로 변경하여 업로드한다. 프로젝트별 폴더를 만들고, png파일을 넣어놓는데 순서대로 ppt{n}.png 이름으로 저장한다.	
hunmin_sub	프로젝트별 페이지를 저장하는 경로 다른 프로젝트와 구분이 될 수 있도록 이름을 설정한다.	
inference_data	추론에 사용할 예시 데이터를 넣어두는 폴더로 압축파일을 hunmin_sub에 넣어둔 html파일 이름과 동일하게 설정한다.	

다. 주의사항

경로에 따라 코드에서 변경할 부분은 크게 두가지 인데, 소개자료(ppt)의 경로와 추론용 데이터(zip)의 경로다.

첫째, 소개자료의 적용에 따라 변경해야 할 사항(코드)

```
140 <div class="slider" >
141   <!-- 하위 input, li, label을 수정하여 작성한 소개자료 페이지에 맞도록 수정한다. -->
142   <input type="radio" name="slide" id="slide1" checked>
143   <input type="radio" name="slide" id="slide2">
144   <input type="radio" name="slide" id="slide3">
145   <input type="radio" name="slide" id="slide4">
146   <input type="radio" name="slide" id="slide5">
147   <input type="radio" name="slide" id="slide6">
148   <input type="radio" name="slide" id="slide7">
149   <ul id="imgholder" class="imgs">
150     <li></li>
151     <li></li>
152     <li></li>
153     <li></li>
154     <li></li>
155     <li></li>
156     <li></li>
157   </ul>
158   <div class="bullets">
159     <label for="slide1">&nbsp;&nbsp;&nbsp;</label>
160     <label for="slide2">&nbsp;&nbsp;&nbsp;</label>
161     <label for="slide3">&nbsp;&nbsp;&nbsp;</label>
162     <label for="slide4">&nbsp;&nbsp;&nbsp;</label>
163     <label for="slide5">&nbsp;&nbsp;&nbsp;</label>
164     <label for="slide6">&nbsp;&nbsp;&nbsp;</label>
165     <label for="slide7">&nbsp;&nbsp;&nbsp;</label>
166   </div>
```

- text_classification.html 참고

slider 클래스를 가지고있는 div부분이다.

div 하위의 input, li, label을 소개자료의 페이지 수에 맞게 늘리거나 줄이고, 경로설정을 변경한다.

둘째, 추론 데이터 적용에 따라 변경해야 할 사항(코드)

```
173 <div id="demo">
174   <h1>예제 실행해보기
175   <button class="download">
176     <a href="../../inference_data/text_classification.zip" download>추론 데이터 다운로드</a>
177   </button>
178 </h1>
```

적용 후 업로드한 파일이 다운로드 되는지 확인한다.

3. API 요청 route 규칙

1. 코드 주석 추가설명

API 요청 데이터와 응답 데이터에 대해 html 파일마다 주석으로 설명했으나 추가설명과 캡처다.

[AI훈민정음](#) / [추론플랫폼](#) / [추론API관리](#)

추론 API 상세

테스트

API URL

http://idro3vub.dl.nhnes.net/model/api/31dbc/inference

METHOD

POST

요청

```
{
  "data": "[['But there is no emotion in the movie.']]
}
```

API 호출

초기화

응답

```
{"inference":["neg"]}
```

가. 요청

T3Q dl플랫폼 추론 API 관리 메뉴의 요청 입력 부분의 형식이 위 캡처와 같을 때,

html 파일의 API 호출 부분에서 data.word는 'But there is no emotion in the movie.'가 입력되어야 한다.

```
if($form.find("input[name=file]").val() != null){
    $.ajax({
        url: "/inference/text_req_ajax", // 텍스트 입력이 들어가는 요청 route
        data: JSON.stringify({
            // 추론에 사용할 데이터가 word에 들어가야 한다.
            // 추론에 [['hi! there!']] 을 입력했다면
            // word에는 'hi! there!' 문자열이 들어간다.
            // 적절한 포매팅은 ajax요청의 url에 적어놓은 부분에서 실행된다.
            word: $("div.group_inputImg>span").html(),
            // 생성한 API URL이 들어가는 부분
            url: $form.find("input[name=url]").val()
        }),
        method: "POST", // HTTP 요청 메소드(GET, POST 등)
        dataType: "json" // 서버에서 보내줄 데이터의 타입
    })
    // HTTP 요청이 성공하면, 요청한 데이터가 done() 메소드로 전달됨.
```

나. 응답

```

.done(function(json) {
    // json은 요청의 응답인데, 보통 {"res" : "true", "response" : API응답 } 형식이다.
    // 해당 예시는 response안의 응답이 data 혹은 inference로 반환되는 것을 받아오는 코드
    // 예시의 json 변수는      {"res" : "true", "response" : {"inference" : inference_result }}
    // {"inference" : inference_result }가 API를 생성하고 테스트하였을때 결과로 확인되는 return이다.
    if (json.res == "true") {
        var response_data = json.response.data;
        if (response_data == null) {
            response_data = json.response.inference;
        }
        // 응답을 확인하여 화면에 적용
        if (response_data == "pos") {
            $(".div.wrap_next").addClass("show_alert_pass");
        } else {
            $(".div.wrap_next").addClass("show_alert_nonpass");
        }
    }
}

```

해당 코드는 json을 return으로 받고 response에 API 응답을 넣어두는데, 해당 예제에는 inference를 key로 하여 응답 받는다. 그래서 response_data = json.response.inference의 데이터가 웹에 적용된다.

플랫폼에서의 응답이 {"inference":["neg"]}일 때,

json 변수에는 {"res": "true", "response": {"inference": ["neg"]}}가 들어간다.

다. [참고] API 요청에 사용되는 route

```

# text data request
@router.post("/text_req_ajax")
async def text_req_ajax(request: Request):
    json_data = await request.json()
    data = { "data": str([[str(json_data["word"])]]) }
    return callApi(json_data["url"], data)

# file data request
@router.post("/file_req_ajax")
async def file_req_ajax(request: Request, file: UploadFile):
    form = await request.form()
    b64_up = base64.b64encode(file.file.read())
    data = { "data": "[" + b64_up.decode('utf-8') + "]" }
    return callApi(form["url"], data)

# log data request
@router.post("/log_req_ajax")
async def log_req_ajax(request: Request):
    json_data = await request.json()
    data = { "data": "[" + json_data["log_data"] + "]" }
    return callApi(json_data["url"], data)

```

API 요청시 위와 같은 포매팅을 거쳐 요청이 전송된다.

4. 스타일 추가 및 변경 방법

1. 스타일 변경

스타일 변경 혹은 스크립트 추가는 html파일 내에서만 하는 것을 권한다.

style태그 혹은 script는 head안의 현재 스크립트 아래에 붙여서 적용하는 것을 규칙으로 한다.

```
120         }  
121         $('#slide'+info_num).prop("checked",true);  
122     });  
123 </script>  
124  
125 <style>  
126 |  
127 </style>  
128 </head>  
129  
130 <body>  
131     <main class="main">  
132         <div class="container_main">  
133             <div class="inner_title">  
134                 <div height="25px">
```

추론데이터의 입력과 API호출 결과 출력은 수행중인 프로젝트에 어울리도록 코드를 수정한다.