네이버 Smart Editor

▣ 네이버 Smart Editor 란..??

- 네이버에서 개발한 WYSIWYG 웹 에디터
- 네이버 서비스에서 글쓰기가 제공되는 공간에서 만나볼 수 있다.
- 2008년 "똑똑한 웹 문서 편집기"라는 이름으로 오픈하여 네이버 블로그, 네이버 카페, 네이버 지식iN, 네이버 메일 등 네이버 내 웹 형식의 문서를 작성하는 공간인 게시판, 커뮤니티형 서비스 등에 적용되었다.
- 네이버에서 개발한 진도 자바스크립트 프레임워크를 사용한, 웹 문서 글쓰기에 특화된 에디터이다.

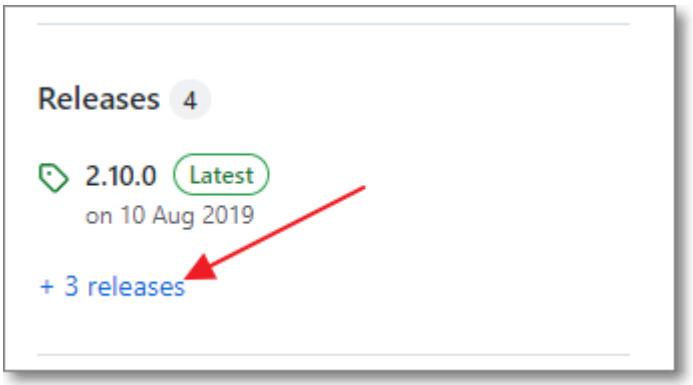
▣ WYSIWYG 이란..??

- WYSIWYG(What You See Is What You Get)
- UI 화면에서 보고 있는 내용이 실제 결과물과 동일하게 나타나는 방식을 의미
- 주로 텍스트 편집기, 웹 디자인 도구, 그래픽 소프트웨어 등에서 사용
- 사용자가 직접 작성한 내용이나 스타일이 미리보기와 동일한 형태로 출력되거나 저장된다.

▣ 다운로드

github에서 배포하고 있다.

- 링크1 : https://github.com/naver/smarteditor2
  - 2024년 8월 현재 최신버전은 2.10
  - 버전 2.9 까지 배포하고 있으나, 파일업로드 기능이 제외되어 있다.
  - 텍스트 에디터로만 사용한다면 이 버전을 다운로드한다.
  - 우측 하단에에 보면 Releases 라는 항목에 2.10.0 (Latest) 이 있다.



- 링크2 : https://github.com/naver/smarteditor2/releases/tag/v2.8.2.3
  - 버전 2.8 다운로드 링크
  - 2024년 9월 현재 이 링크는 숨겨진 상태
  - 파일업로드 기능이 필요하다면 이 버전을 다운로드한다.

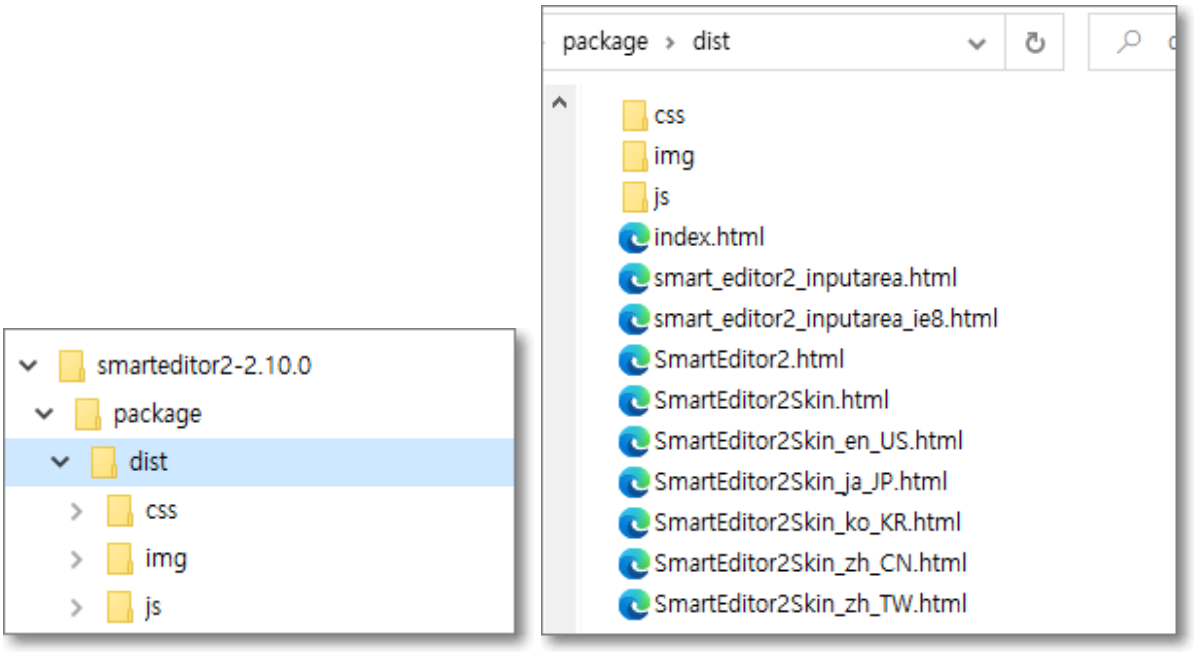
- 우리는 2.8.2 버전으로 학습한다.

프로젝트명 : B19WysiwygEditor

준비사항

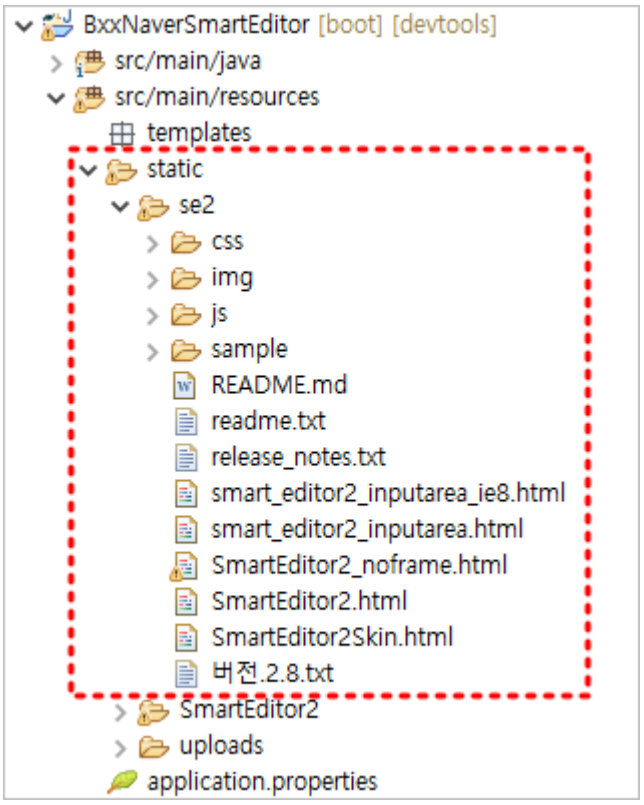
- 의존설정 : Spring Web, Lombok
- JSP 사용을 위한 설정을 한다.
- Refresh Gradle Project 를 눌러 적용한다.

다운로드 한 smarteditor2-2.8.2.3.zip 파일의 압축을 해제한다.(버전이 틀려도 세팅은 동일함)



static 디렉토리 하위에 SmartEditor2 폴더를 생성하고, 압축해제한 모든 폴더와 파일을 하위로 붙여넣기 한다. (공식문서에는 se2로 명시되어 있음)

폴더명은 본인이 원하는 데로 수정해도 된다.



▶요청명 : home.jsp

```
36      <h2>위지윅(WYSIWYG) 에디터</h2>
37      <li>
38          <a href="naverSmartEditor.do" target="_blank">
39              네이버 스마트 에디터
40          </a>
41      </li>
```

▶컨트롤러 : MainController.java

```
28      @RequestMapping("/naverSmartEditor.do")
29      public String naverSmartEditor() {
30          return "naverSmartEditor";
31      }
32      @RequestMapping(value="/naverSmartEditor.do",
33              method=RequestMethod.POST)
34      public String naverSmartEditorSubmit(HttpServletRequest req,
35              Model model) {
36          String title = req.getParameter("title");
37          String contents = req.getParameter("contents");
38          System.out.println("title="+ title);
39          System.out.println("contents="+ contents);
40
41          model.addAttribute("submit", "폼값전송됨");
42          return "naverSmartEditor";
43      }
44  }
```

▶뷰 : naverSmartEditor.jsp

JSP파일을 생성한 후 아래 내용을 작성한다. 작성시 아래 가이드 문서를 참조한다.

SmartEditor2 사용자 가이드 [바로가기]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>Insert title here</title>
8  <script type="text/javascript" src="./se2/js/HuskyEZCreator.js" char
9  <script type="text/javascript">
10 var oEditors = [];
11 window.onload = function(){
12     nhn.husky.EZCreator.createInIFrame({
13         oAppRef: oEditors,
14         elPlaceHolder: "contents",
15         sSkinURI: "./se2/SmartEditor2Skin.html",
16         fCreator: "createSEditor2"
17     });
18 }
```

```
19 function validateForm(f){
20     if(f.subject.value==''){
21         alert('제목을 입력하세요');
22         f.subject.focus();
23         return false;
24     }
25     // 에디터의 내용이 textarea에 적용된다.
26     oEditors.getById["contents"].exec("UPDATE_CONTENTS_FIELD", []);
27     // 에디터의 내용에 대한 값 검증은 이곳에서
28     let contents = document.getElementById("contents").value;
29     if(contents==''){
30         alert('내용을 입력하세요');
31         oEditors.getById["contents"].exec("FOCUS")
32         return false;
33     }
34     console.log("contents", contents);
35     return false;
36 }
37 </script>
38 </head>
```

위 코드는 모두 가이드 문서에 있는것을 사용했으므로 복사한 후 적절히 수정한다.

1. HuskyEZCreator.js 의 경로를 입력한다.
2. 스마트 에디터를 페이지에 적용하기 위한 코드이다. elPlaceHolder를 우리가 사용하는 contents 로 입력한다. 상황에 따라 달라질 수 있다.
3. 폼값 전송시 에디터에 입력한 내용을 <textarea> 태그에 적용하기 위해 필요한 코드이다.
4. validateForm 함수 마지막 부분의 return false;는 내용 확인을 위한 부분이므로 확인 후 주석처리한다.

<body> 부분은 아래 코드를 복사해서 사용한다.

```
<body>
	<h2>네이버 스마트 에디터</h2>
	<form method="post" onsubmit="return validateForm(this);">
	<span style="color:red;">${submit }</span>
	<table border="1" style="width:900px;">
	<colgroup>
		<col width="100px" />
		<col width="*" />
	</colgroup>
		<tr>
			<td>제목</td>
			<td><input type="text" name="subject" style="width:400px;"/></td>
		</tr>
```

```
            <tr>
                <td>내용</td>
                <td>
                    <!-- 에디터에 기본으로 삽입할 글(수정 모드)이 없다면
                    value 값을 지정하지 않으시면 됩니다. -->
                    <textarea name="contents" id="contents" rows="10"
cols="70"></textarea>
                </td>
            </tr>
            <tr>
                <td colspan="2">
                    <input type="submit" value="전송하기" />
                </td>
            </tr>
        </table>
        </form>
</body>
```

여기까지 작성하세요.

파일업로드 기능 추가

파일명 : static/se2/sample/photo_uploader/photo_uploader.html

```
            <div class="drag_area" id="drag_area">
                <ul class="lst_type" >
                </ul>
                <em class="blind">마우스로 드래그해서 이미지를 추가해주세요.</em><span
            </div>
            <div style="display:none;" id="divImageList"></div>
            <p class="dsc dsc_v1"><em>한 장당 10MB</em>의 이미지 파일을 1장씩<br>
            등록할 수 있습니다.(JPG, GIF, PNG, BMP)</p>
        </div>
        <!-- //content -->
    </div>
```

파일명 : static/se2/sample/photo_uploader/attach_photo.js

```
    function html5Upload() {

        var tempFile,
            sUploadURL;

        sUploadURL= '/file_uploader_html5.do';  //upload URL

        //파일을 하나씩 보내고, 결과를 받음.
        for(var j=0, k=0; j < nImageInfoCnt; j++) {
            console.log("하나씩??");
            tempFile = htImageInfo['img'+j];
            try{
                if(!!tempFile){
                    //Ajax통신하는 부분. 파일과 업로더할 url을 전달한다.
                    callAjaxForHTML5(tempFile,sUploadURL);
                    k += 1;
                }
            }catch(e){}
            tempFile = null;
        }
    }
```

```
    function callAjaxForHTML5 (tempFile, sUploadURL){
        xhr = new XMLHttpRequest();
        xhr.open("POST", sUploadURL, true);
        xhr.onreadystatechange = function(){
            var resText = xhr.responseText;
            console.log('xhr.readyState', xhr.readyState);
            if(xhr.readyState==4){
                console.log('resText', resText);
                makeArrayFromString(resText);
                //makeArrayFromString(res._response.responseText);
            }
```

```
            else{
                //onAjaxError();
            }
            //retrun;
        }

        xhr.timeout = 5000;
        xhr.ontimeout = function(){
            onAjaxError();
        }

        var form = new FormData();
        form.append("ofile", tempFile);
        xhr.send(form);
    }

```

여기까지 작성하세요.

컨트롤러 : com/edu/springboot/MainController.java

```
    @PostMapping("/file_uploader_html5.do")
    @ResponseBody
    public String fileUpload(
            @RequestParam("ofile") MultipartFile ofile,
            HttpServletRequest request,
            HttpServletResponse response)
                    throws IOException {
        String result = "";
        if (ofile.isEmpty()) {
            return "파일이 비어 있습니다.";
        }

        // 파일의 원본 이름을 가져옴
        String fileName = ofile.getOriginalFilename();
        // 저장 디렉토리의 물리적경로
        String uploadDir = ResourceUtils
            .getFile("classpath:static/uploads/").toPath()
            .toString();
        //System.out.println(uploadDir);
```

```
64          // 파일 저장
65          try {
66              byte[] bytes = ofile.getBytes();
67              System.out.println(Paths.get(uploadDir +
68                      File.separator + fileName));
69              Files.write(Paths.get(uploadDir +
70                      File.separator + fileName), bytes);
71          }
72          catch (IOException e) {
73              e.printStackTrace();
74              return "파일 처리 중 오류 발생";
75          }
76
77          //콜백 데이터
78          result = "&bNewLine=true"
79                  +"&sFileName="+ fileName
80                  +"&sFileURL=/uploads/"+ fileName;
81          return result;
82      }
```

여기까지 작성하세요.