네이버 검색 API 활용

프로젝트명 : B16NaverSearchAPI

준비사항

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

네이버검색 API Key 발급받기 [바로가기]

뷰 : /B16NaverSearchAPI/src/main/webapp/WEB-INF/views/home.jsp

```
 9 <body>
10     <h2>Naver 검색API 활용</h2>
11     <ul>
12         <li><a href="/">루트</a></li>
13         <li><a href="/NaverSearchMain.do">검색바로가기</a></li>
14     </ul>
15 </body>
16 </html>
```

컨트롤러 : /B16NaverSearchAPI/src/main/java/com/edu/springboot/MainController.java

```
31     @RequestMapping("/")
32     public String home() {
33         return "home";
34     }
35 
36     @RequestMapping("/NaverSearchMain.do")
37     public String NaverSearchMain() {
38         return "SearchView";
39     }
40 
```

뷰 : /B16NaverSearchAPI/src/main/webapp/WEB-INF/views/SearchView.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>검색 API</title>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.5.1/j
<script>
// [검색 요청] 버튼 클릭 시 실행할 메서드를 정의합니다.
$(function() {
    $('#keyword').keydown(function(e) {
        if(e.keyCode==13){
            runAjax();
            e.preventDefault();
        }
        else{
            console.log(e.keyCode);
        }
    });
    $('#searchBtn').click(function() {
        runAjax();
    });
});
```

```jsp
function runAjax(){
    $.ajax({
        url : "./NaverSearchRequest.do",// 요청 URL
        type : "get",                   // HTTP 메서드
        data : {                        // 매개변수로 전달할 데이터
            keyword : $('#keyword').val(),
            startNum : $('#startNum').val()
        },
        dataType : "json",       // 응답 데이터 형식
        success : sucFuncJson,   // 요청 성공 시 호출할 메서드 설정
        error : errFunc          // 요청 실패 시 호출할 메서드 설정
    });
}
```

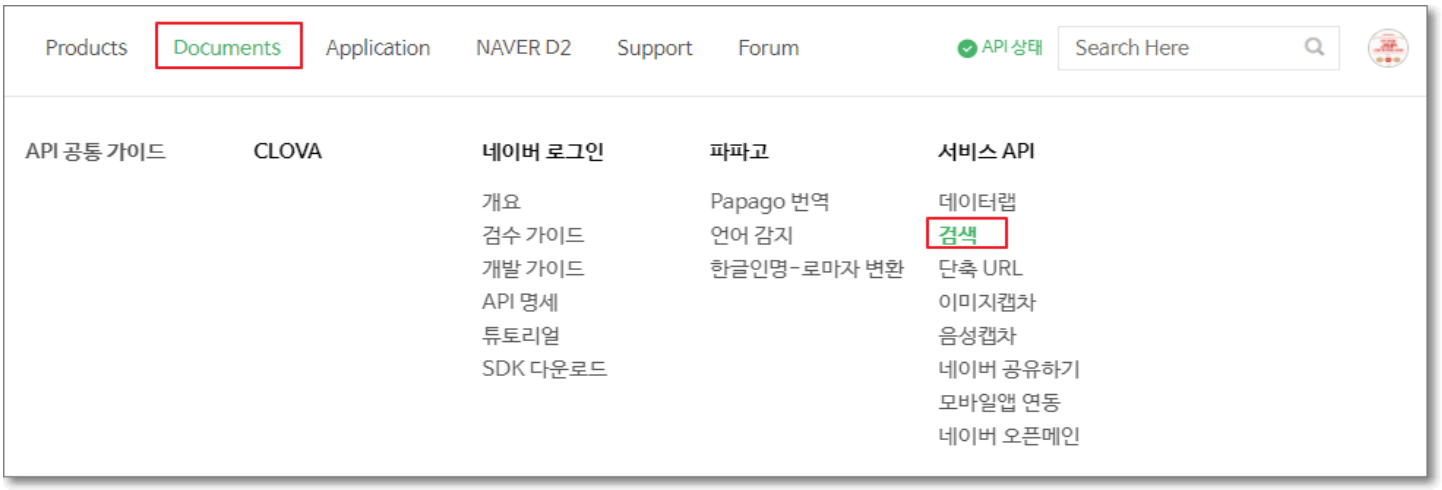
```
// 검색 성공 시 결과를 화면에 뿌려줍니다.
function sucFuncJson(d) {
    console.log("결과", d);
    var str = "";
    $.each(d.items, function(index, item) {
        str += "<ul>";
        str += "    <li>" + (index + 1) + "</li>        ";
        str += "    <li>" + item.title + "</li>         ";
        str += "    <li>" + item.description + "</li>   ";
        str += "    <li>" + item.bloggername + "</li>   ";
        str += "    <li>" + item.bloggerlink + "</li>   ";
        str += "    <li>" + item.postdate + "</li>      ";
        str += "    <li><a href='" + item.link + "'     ";
        str += "        target='_blank'>바로가기</a></li> ";
        str += "</ul>";
    });
    $('#searchResult').html(str);
}
// 실패 시 경고창을 띄워줍니다.
function errFunc(e) {
    alert("실패: " + e.status);
}
</script>
```

```
<style>
  ul{border:2px #cccccc solid;}
  #startNum{width:100px;}
</style>
</head>
<body>
<div>
  <div>
    <form id="searchFrm">
      한 페이지에 10개씩 출력됨 <br />
      <input type="number" id="startNum" step="10"
                          value="1">
      <input type="text" id="keyword"
                          placeholder="검색어를 입력하세요." />
      <button type="button" id="searchBtn">검색 요청</button>
    </form>
  </div>
  <div class="row" id="searchResult">
    여기에 검색 결과가 출력됩니다.
```

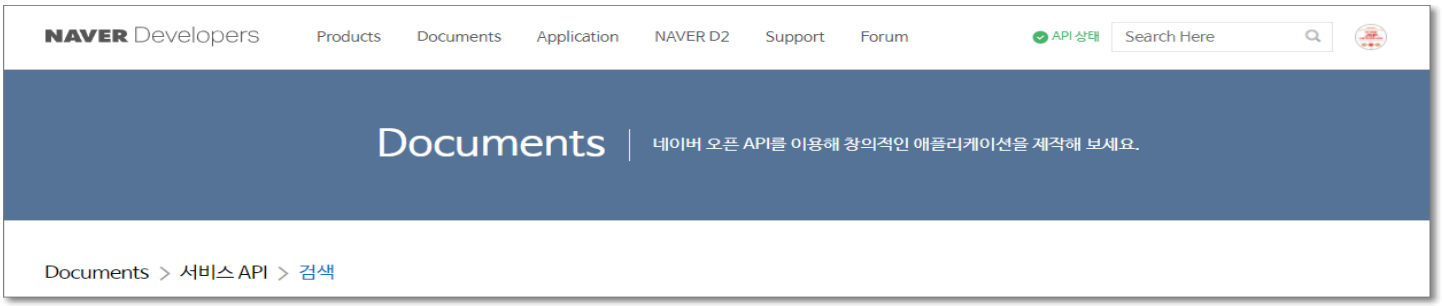
```
    </div>
</div>
</body>
</html>
```

여기까지 작성하세요.

개발문서를 보기위해 Documents 하위의 검색으로 이동합니다 .



지금부터 개발문서를 읽어본 후 네이버검색을 Spring 프로젝트에 추가해봅니다.



블로그

검색 > 블로그

네이버 블로그 검색 결과를 출력해주는 REST API입니다. 비로그인 오픈 API이므로 GET으로 호출할 때 HTTP Header에 애플리케이션 등록 시 발급받은 Client ID와 Client Secret 값을 같이 전송해 주시면 활용 가능합니다.

오픈 API 이용 신청 〉

0.API 호출 예제

예제 실행 전에 아래 *1.준비사항* 항목들을 꼭 체크하시길 바랍니다.

Java

```
네이버 검색 API예제는 블로그를 비롯 전문자료까지 호출방법이 동일하므로 blog검색만 대표로 예제를 올렸습니다.
// 네이버 검색 API 예제 - blog 검색
import java.io.*;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
```

컨트롤러 : /B16NaverSearchAPI/src/main/java/com/edu/springboot/MainController.java

```java
    @ResponseBody
    @RequestMapping("/NaverSearchRequest.do")
    public String NaverSearchRequest(HttpServletRequest req,
            HttpServletResponse resp) {

        // 1. 인증 정보 설정
        String clientId = "Ph_[illegible]";//클라이언트 아이디
        String clientSecret = "z[illegible]";//클라이언트 시크릿

        // 2. 검색 조건 설정
        int startNum = 0;     // 검색 시작 위치
        String text = null;  // 검색어
        try {
             startNum = Integer.parseInt(req.getParameter("startNum"));
             String searchText = req.getParameter("keyword");
             text = URLEncoder.encode(searchText, "UTF-8");
        } catch (UnsupportedEncodingException e) {
            throw new RuntimeException("검색어 인코딩 실패", e);
        }

        // 3. API URL 조합
        String apiURL = "https://openapi.naver.com/v1/search/blog?"
                + "query="+text+"&display=10&start="+startNum;//JSON콜백
        //String apiURL = "https://openapi.naver.com/v1/search/blog.xml?
        //      query=" + text; // XML콜백
```

```java
        // 4. API 호출
        Map<String, String> requestHeaders = new HashMap<>();
        requestHeaders.put("X-Naver-Client-Id", clientId);
        requestHeaders.put("X-Naver-Client-Secret", clientSecret);
        String responseBody = get(apiURL, requestHeaders);

        // 5. 결과 출력
        System.out.println(responseBody);  // 콘솔에 출력
        return responseBody;
    }
```

나머지는 그대로 사용한다.

```java

    private static String get(String apiUrl, Map<String, String> requestHeaders){
    private static HttpURLConnection connect(String apiUrl){
    private static String readBody(InputStream body){
}

```