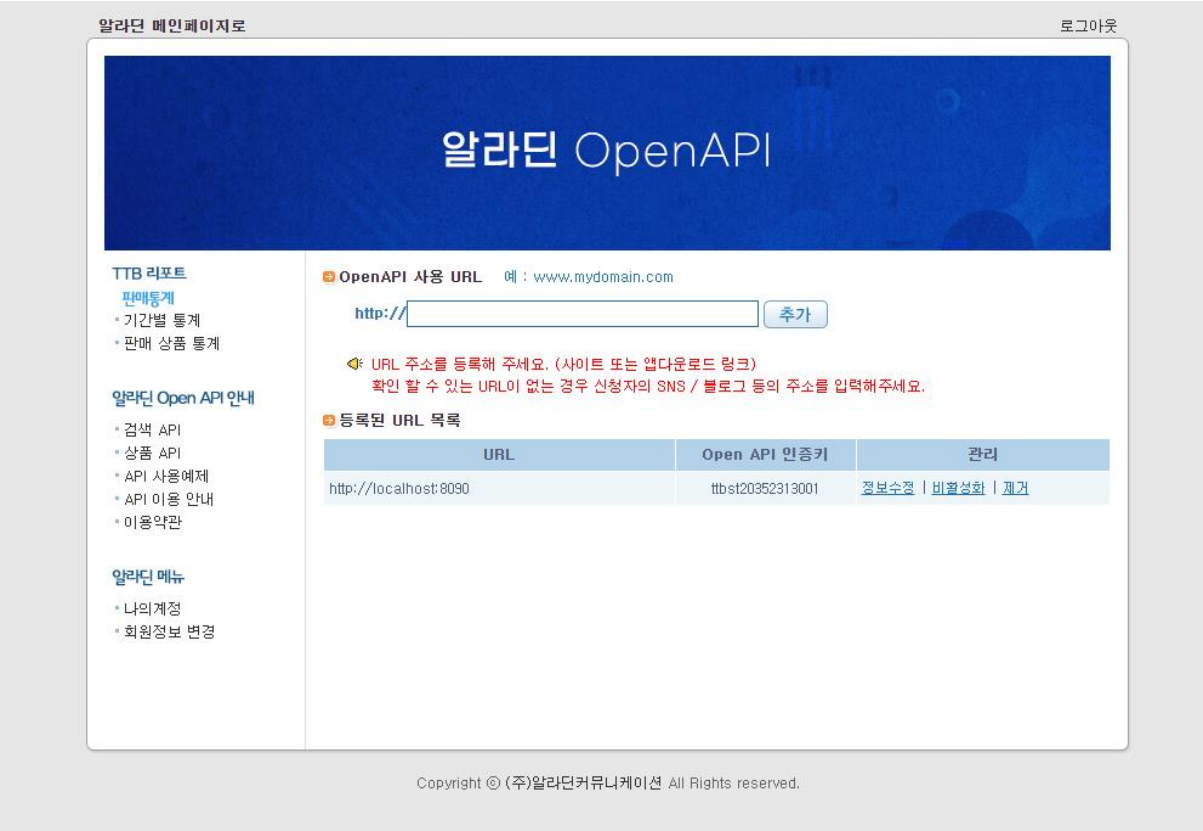
알라딘 Open API 설명서

공식 매뉴얼

https://docs.google.com/document/d/1mX-WxuoGs8Hy-QalhHcvuV17n50uGI2Sg_GHofgiePE/edit

인증키 발급 페이지 (회원가입 및 로그인 후 사용 가능)

https://www.aladin.co.kr/ttb/wblog_manage.aspx



프로젝트에 사용된 API 목록

- 1) 검색어를 통한 도서 검색 리스트
- 2) 신간 전체 리스트 / 주목할 만한 신간 리스트 / 베스트 셀러 리스트
- 3) 도서 고유 번호(ISBN)를 통한 도서 1권에 대한 상세 정보

- 매뉴얼 간략 정리 -

1) 상품 검색 API

- 요청 방법
  - 요청 URL : http://www.aladin.co.kr/ttb/api/ItemSearch.aspx
  - 요청 URL 샘플 : http://www.aladin.co.kr/ttb/api/ItemSearch.aspx?ttbkey=[TTBKey]&Query=aladdin&QueryType=Title&MaxResults=10&start=1&SearchTarget=Book&output=xml&Version=20131101
  - 결과샘플
    - XML 형식 : http://www.aladin.co.kr/ttb/api/test/ItemSearch_20131101.xml
    - javascript 형식 : http://www.aladin.co.kr/ttb/api/test/ItemSearch_20131101.js
  - 한 페이지에 최대 50 개, 총 결과는 200 개까지만 검색 가능

필수	TTBKey	문자열	부여받은 TTBKey 값
	Query	문자열	검색어

2) 상품 리스트 API

- 요청 방법
  - 요청 URL : http://www.aladin.co.kr/ttb/api/ItemList.aspx
  - 요청 URL 샘플 : http://www.aladin.co.kr/ttb/api/ItemList.aspx?ttbkey=[TTBKey]&QueryType=ItemNewAll&MaxResults=10&start=1&SearchTarget=Book&output=xml&Version=20131101
  - 결과샘플
    - XML 형식 : http://www.aladin.co.kr/ttb/api/test/ItemList_20131101.xml
    - javascript 형식 : http://www.aladin.co.kr/ttb/api/test/ItemList_20131101.js
  - 한 페이지에 최대 50 개, 총 결과는 200 개까지만 조회 가능

필수	TTBKey	문자열	부여받은 TTBKey 값
	QueryType	ItemNewAll : 신간 전체 리스트 ItemNewSpecial : 주목할 만한 신간 리스트 ItemEditorChoice : 편집자 추천 리스트 (카테고리로만 조회 가능 - 국내도서/음반/외서만 지원) Bestseller : 베스트셀러 BlogBest : 블로거 베스트셀러 (국내도서만 조회 가능)	리스트 종류

3) 상품 조회 API

- 요청 방법
  - 요청 URL : http://www.aladin.co.kr/ttb/api/ItemLookUp.aspx
  - 요청 URL 샘플 : http://www.aladin.co.kr/ttb/api/ItemLookUp.aspx?ttbkey=[TTBKey]&itemIdType=ISBN&ItemId=[도서의 ISBN]&output=xml&Version=20131101&OptResult=ebookList,usedList,reviewList
  - "상품 조회 응답(Reponse) 결과값"의 스펙은 "검색 응답(Reponse) 결과값"과 동일한 결과에 단순히 부가정보가 추가되어지는 것임.
  - 분류 ID 값은 알라딘 모든 분야 카테고리 엑셀문서를 참고하십시오.
  - 결과샘플
    - XML 형식 : http://www.aladin.co.kr/ttb/api/test/ItemLookUp_20131101.xml
    - javascript 형식 : http://www.aladin.co.kr/ttb/api/test/ItemLookUp_20131101.js

필수	TTBKey	문자열	부여받은 TTBKey 값
	ItemId	문자열/숫자	상품을 구분짓는 유일한 값 (ItemIdType 으로 정수값과 ISBN 중에 택일)

API PARSING

공통적으로 쓰이는 메소드(API URL 유효성 검사 및 버퍼리더 사용)

```
private static String get(String apiUrl, Map<String, String> requestHeaders) {
                HttpURLConnection con = connect(apiUrl);
                try {
                        con.setRequestMethod("GET");
                        for (Map.Entry<String, String> header : requestHeaders.entrySet()) {
                                con.setRequestProperty(header.getKey(), header.getValue());
                        }

                        int responseCode = con.getResponseCode();
                        if (responseCode == HttpURLConnection.HTTP_OK) { // 정상 호출
                                return readBody(con.getInputStream());
                        } else { // 오류 발생
                                return readBody(con.getErrorStream());
                        }
                } catch (IOException e) {
                        throw new RuntimeException("API 요청과 응답 실패", e);
                } finally {
                        con.disconnect();
                }
        }

private static HttpURLConnection connect(String apiUrl) {
                try {
                        URL url = new URL(apiUrl);
                        return (HttpURLConnection) url.openConnection();
                } catch (MalformedURLException e) {
                        throw new RuntimeException("API URL이 잘못되었습니다. : " + apiUrl, e);
                } catch (IOException e) {
                        throw new RuntimeException("연결이 실패했습니다. : " + apiUrl, e);
                }
        }

        //
private static String readBody(InputStream body){
       InputStreamReader streamReader = new InputStreamReader(body);

       try (BufferedReader lineReader = new BufferedReader(streamReader)) {
          StringBuilder responseBody = new StringBuilder();

          String line;
          while ((line = lineReader.readLine()) != null) {
             responseBody.append(line);
          }

          return responseBody.toString();
       } catch (IOException e) {
          throw new RuntimeException("API 응답을 읽는 데 실패했습니다.", e);
       }
    }
```

1) API DAO 도서 검색 리스트 메소드

```
public static String getSearch(int start,int categoryId,String query) {

		String apiURL = "http://www.aladin.co.kr/ttb/api/ItemSearch.aspx"
				+ "?"
				+ "ttbkey=_________________"
				+ "&"
				+ "Query="+query
				+ "&"
				+ "QueryType=Title"
				+ "&"
				+ "MaxResults=50"
				+ "&"
				+ "start="+start
				+ "&"
				+ "SearchTarget=Book"
				+ "&"
				+ "output=xml"
				+ "&"
				+ "Version=20131101"
				+ "&"
				+ "Cover=Big"
				+ "&"
				+ "CategoryId="+categoryId
				;

		// 결과
		Map<String, String> requestHeaders = new HashMap<String, String>();
		String responseBody = get(apiURL, requestHeaders);

		//XML -> JSONObecjt
		JSONObject resultObj = XML.toJSONObject(responseBody);

		// "object" key를 JSONObjext 객체로 생성
		JSONObject result = resultObj.getJSONObject("object");

		return result.toString();
	}
```

2) API DAO 베스트셀러 리스트 메소드

```
public static String getBestseller(int start,int categoryId) {

		String apiURL = "http://www.aladin.co.kr/ttb/api/ItemList.aspx"
				+ "?"
				+ "ttbkey=_________________"
				+ "&"
				+ "QueryType=Bestseller"
				+ "&"
				+ "MaxResults=50"
				+ "&"
				+ "start="+start
				+ "&"
				+ "SearchTarget=Book"
				+ "&"
				+ "output=xml"//xml or JS
				+ "&"
				+ "Version=20131101"
				+ "&"
				+ "Cover=Big"
				+ "&"
				+ "CategoryId="+categoryId
				;


		// 결과
		Map<String, String> requestHeaders = new HashMap<String, String>();
		String responseBody = get(apiURL, requestHeaders);

		//XML -> JSONObecjt
		JSONObject resultObj = XML.toJSONObject(responseBody);

		// "object" key를 JSONObjext 객체로 생성
		JSONObject result = resultObj.getJSONObject("object");
		//System.out.println(result);


		return result.toString();
	}
```

3) API DAO 도서 상세 검색 메소드

```
public static String getDetail(String isbn) {

                        String apiURL = "http://www.aladin.co.kr/ttb/api/ItemLookUp.aspx"
                                        + "?"
                                        + "ttbkey=_________________"
                                        + "&"
                                        + "itemIdType=ISBN"
                                        + "&"
                                        + "ItemId="+isbn
                                        + "&"
                                        + "output=xml"
                                        + "&"
                                        + "Version=20131101"
                                        + "&"
                                        + "OptResult=ebookList,usedList,reviewList"
                                        + "&"
                                        + "Cover=Big"
                                        ;


                        // 결과
                        Map<String, String> requestHeaders = new HashMap<String, String>();
                        String responseBody = get(apiURL, requestHeaders);

                        //XML -> JSONObecjt
                        JSONObject resultObj = XML.toJSONObject(responseBody);

                        // "object" key를 JSONObjext 객체로 생성
                        JSONObject result = resultObj.getJSONObject("object");

                        return result.toString();
                }
```

fromJSONtoItems 메소드

```
public static HashMap<String , Object> fromJSONtoItems(String result){

                JSONObject rjson = new JSONObject(result);
                 //System.out.println("rjson = "+ rjson);

                 int count  =rjson.getInt("totalResults");
                 //System.out.println("count = "+ count);

                JSONArray item = rjson.getJSONArray("item");
                //System.out.println("item = "+ item);

                HashMap<String , Object>  map = new HashMap<>();

                ArrayList<BooksDto> booksDtoList = new ArrayList<BooksDto>();
                for(int i = 0 ; i < item.length();i++) {
                        JSONObject json = item.getJSONObject(i);
                        BooksDto booksDto = new BooksDto(json);
                        //System.out.println(booksDto);

                        booksDtoList.add(booksDto);
                }

                map.put("list", booksDtoList);
                map.put("totalCnt" , count);


                return map;
        }
```

필요한 데이터가 두 가지 존재

<object>태그 안 totalResults 값 , <item> 태그들

<object>태그는 위 1)2)메소드에서 벗겨냈고 fromJSONtoItems 메소드에서는 위 두가지 값을 HashMap 에 저장해서 리턴하는 메소드

fromJSONtoItems2 메소드

```
public BooksDetailDto fromJSONtoItems2(String result){

		JSONObject rjson = new JSONObject(result);

		JSONObject item = rjson.getJSONObject("item");
		//System.out.println(item);

		BooksDetailDto booksDetail = new BooksDetailDto(item);

		return booksDetail;

	}
```

fromJSONtoItems2 메소드는 3)도서 상세 검색을 위한 메소드
HashMap 이 아닌 DTO 를 return

Controller

도서 검색

```
	@ResponseBody
	@RequestMapping(value = "/books/Search", method = RequestMethod.GET)
	public HashMap<String, Object> getSearch(int start, int categoryId, String query) {

		String result = apiBooks.getSearch(start, categoryId, query);

		HashMap<String, Object> map = apiBooks.fromJSONtoItems(result);

		return map;
	}
```

베스트 셀러

```
	@ResponseBody
	@RequestMapping(value = "/books/Bestseller", method = RequestMethod.GET)
	public HashMap<String, Object> getBestsellerListView(int start, int categoryId) {

		String result = apiBooks.getBestseller(start, categoryId);

		HashMap<String, Object> map = apiBooks.fromJSONtoItems(result);

		return map;
	}
```

도서 상세 페이지

```
	@RequestMapping(value = "/books/Detail", method = RequestMethod.GET)
	public String booksDetailView(String isbn, Model m) {

		String result = apiBooks.getDetail(isbn);

		BooksDetailDto booksDetail = apiBooks.fromJSONtoItems2(result);

		m.addAttribute("title", booksDetail.getTitle());
		m.addAttribute("link", booksDetail.getLink());
		m.addAttribute("author", booksDetail.getAuthor());
		m.addAttribute("pubDate", booksDetail.getPubDate());
		m.addAttribute("description", booksDetail.getDescription());
		m.addAttribute("isbn", booksDetail.getIsbn());
		m.addAttribute("isbn13", booksDetail.getIsbn13());
		m.addAttribute("itemId", booksDetail.getItemId());
		m.addAttribute("priceSales", booksDetail.getPriceSales());
		m.addAttribute("priceStandard", booksDetail.getPriceStandard());
		m.addAttribute("mileage", booksDetail.getMileage());
		m.addAttribute("cover", booksDetail.getCover());
		m.addAttribute("categoryId", booksDetail.getCategoryId());
		m.addAttribute("categoryName", booksDetail.getCategoryName());
		m.addAttribute("publisher", booksDetail.getPublisher());

		return "/books/detail";
	}
```