Anyframe XPLATFORM Plugin

Version 1.6.0

I.Introduction

Installation

Command 창에서 다음과 같이 명령어를 입력하여 Anyframe XPLATFORM Plugin을 설치한다.

```
mvn anyframe:install -Dname=xplatform
```

installed(mvn anyframe:installed) 혹은 jetty:run(mvn clean jetty:run) command를 이용하여 설치 결과를 확인해볼 수 있다.

플러그인 설치 후 브라우저를 통해서 Anyframe XP Query Plugin을 확인 하기 위해서는 아래의 몇 가지 추가 작업이 필요하다.

- **TOBESOFT가 제공하는 UX Studio(UI 개발툴) 혹은 XPLATFORM RUNTIME client plugin(ver 9.2)을 설치한다.**
- Anyframe XPLATFORM Plugin을 동작시키기 위해서는 TOBESOFT가 제공하는 XPLATFORM_SERVER_License.xml, XPLATFORM_Client_License.xml 총 2가지 license가 필요하다. 현재 Anyframe XPLATFORM Plugin 내에는 두 가지 라이센스가 모두 포함 되어 있으며, license file의 상세 위치는 다음과 같다.
  - XPLATFORM_SERVER_LICENSE.xml : xplatform-api-x.x.x.jar 파일 내에 존재
  - XPLATFORM_CLIENT_LICENSE.xml : {PROJECT_HOME}\src\main\webapp\xplatform\SampleApplication\ 폴더에 존재

Anyframe XPLATFORM Plugin 개발 환경

Anyframe XPLATFORM Plugin에서 제공하는 예제 화면은 XPLATFORM Runtime 9.2 버전을 기준으로 개발 되었다. XPLATFORM Runtime client나 UX Studio가 상이한 버전으로 설치 된 경우, 비정상 동작 하거나 기동 되지 않는 문제가 생길 수 있으니 참고한다.

Dependent Plugins

Plugin Name	Version Range
Query [http://dev.anyframejava.org/docs/anyframe/plugin/optional/query/1.6.0/reference/htmlsingle/query.html]	2.0.0 > * > 1.4.0

II.XPLATFORM Integration

Anyframe에서는 XPLATFORM이 사용하는 PlatformRequest, PlatformResponse를 처리하기 위해 XPRequestHandlerUtil Class를 제공하고 있다. 또한 UI에서 서버로의 요청을 발생 시키기 위해 호출하는 transaction() 명령을 쉽게 사용할 수 있는 script를 제공한다.

1.XPRequestHandlerUtil

XPLATFORM을 사용하는 웹어플리케이션 개발 환경에서 Controller Class를 개발할 때, XPLATFORM에서 전용으로 사용하는 데이터 타입을 변환 하는 작업이 반드시 필요하다. Anyframe은 복잡한 변환 로직을 처리하고 개발자가 원하는 데이터 타입으로 사용할 수 있도록 XPRequestHandlerUtil class를 제공한다.

Spring MVC 아키텍쳐에 따라 Controller를 작성한 경우, XPLATFORM의 데이터 타입을 변환 하기 위해서 일반적으로 아래의 예제 코드와 같은 Request 변환, DataSet 및 VariableList 추출 등의 작업을 수행하는 코드를 메소드마다 작성 해야 한다.

```
@Controller
@RequestMapping("/xplatformMovie.do")
public class MovieController {

 @Inject
 @Named("xplatformMovieService")
 private MovieService movieService;

 @RequestMapping(params = "method=getList")
 public void getList(HttpServletRequest request, HttpServletResponse response)
   throws Exception {
  HttpPlatformRequest platformRequest = new HttpPlatformRequest(
  this.request, contentType, charset);
  platformRequest.receiveData();
  PlatformData inPlatformData = platformRequest.getData();

  DataSetList inDl = inPlatformData.getDataSetList();
  VariableList inVl = inPlatformData.getVariableList();

  // ...중략

  DataSetList outDl = new DataSetList();
  VariableList outVl = new VariableList();

  HttpPlatformResponse httpPlatformResponse = new HttpPlatformResponse(
   response, contentType, charset);
  PlatformData outPlatformData = new PlatformData();
  outPlatformData.setDataSetList(outDl);
  outPlatformData.setVariableList(outVl);
  httpPlatformResponse.setData(outPlatformData);
  httpPlatformResponse.sendData();
 }

// ...중략
}
```

또한 이 경우 XPLATFORM 전용 데이터 타입인 DataSet을 주로 사용하기 때문에 이를 Map 혹은 VO형태로 사용하기 위해서는 아래와 같은 추가 작업이 필요할 것이다.

```
DataSet dsGrdMovie = inDl.get("dsGrdMovie");
List<Movie> movieList = new ArrayList<Movie>();
for(int i = 0 ; i < dsGrdMovie.getRowCount() ; i++){
 Movie movie = new Movie();
 movie.setMovieId(dsGrdMovie.getString(i, "MOVIE_ID"));
 movie.setTitle(dsGrdMovie.getString(i, "TITLE"));

 //...중략

 movieList.add(movie);
```

```
}
```

위에서 살펴본 중복적인 변환 작업을 처리하기 위해 Anyframe에서 XPRequestHandlerUtil class를 제공하면 다음과 같이 코드를 단순화 시킬 수 있다.

```
@Controller
@RequestMapping("/xplatformMovie.do")
public class MovieController {

 @Inject
 @Named("xplatformMovieService")
 private MovieService movieService;

 private String contentType = PlatformType.CONTENT_TYPE_XML;

 private String charset = PlatformType.DEFAULT_CHAR_SET;

 @RequestMapping(params = "method=getList")
 public void getList(HttpServletRequest request, HttpServletResponse response)
   throws Exception {

  XPRequestHandlerUtil xpHandlerUtil = new XPRequestHandlerUtil(request,
    response, contentType, charset);
  try {
   List<Map<String, Object>> resultList = movieService
     .getList(xpHandlerUtil.getDataSetAsMap("dsSearch").get(0));

   xpHandlerUtil.setMapListToDataSetListWithCheck("dsGrdMovie",
     resultList);
   xpHandlerUtil.setResultMsg(0, "save successed");
  } catch (Exception e) {
   String errorMsg = e.getMessage();
   if ("".equals(errorMsg)) {
    errorMsg = "Fail to process client.";
   }
   xpHandlerUtil.setResultMsg(-1, errorMsg);
  } finally {
   xpHandlerUtil.sendData();
  }
 }
// ...중략
}
```

XPRequestHandlerUtil class가 생성될 때 입력 인자로 받는 request를 PlatformRequest로 변환하는 과정을 내부적으로 수행 한다. 또한 PlatformRequest에 저장되어 있는 DataSetList, VariableList를 멤버변수에 저장 해두고 사용자가 데이터를 요청할 때 Map 혹은 VO의 형태로 제공하는 메소드를 가지고 있다. 위 예제 코드에서 사용한 getDataSetAsMap method를 호출해서 UI에서 넘어온 "dsSearch"라는 이름의 DataSet을 Map 형태로 제공받아 검색조건 파라미터로 넘겨주는 것을 확인 할 수 있다.

2.gfnTransaction

XPLATFORM UI에서 서버로 요청을 발생 시키기 위해서는 transaction() 이라는 API를 사용해야 한다. transaction 호출을 위한 파라미터를 살펴보면 다음과 같다.

Parameter	Description
strSvcId	transaction을 구분하기 위한 ID값
strUrl	transaction을 요청할 주소
strInDatasets	transaction을 요청할 때 입력값으로 보낼 DataSet의 ID.
strOutDatasets	transaction 처리 결과를 받을 DataSet ID
strArgument	transaction을 위한 인자값.
strCallbackFunc	transaction 결과를 돌려줄 function 이름.
bAsync	비동기 여부
bBinary	Binary 형태로 전송할 지 여부
bCompress	통신시 PostData를 압축할지 여부

개발자가 transaction() 을 호출하기 위해서 사용해야 하는 parameter가 많아서 사용에 불편함이 있기 때문에 Anyframe에서 공통 스크립트 내에 gfnTransaction() 명령어를 이용해서 쉽게 서버 호출을 사용할 수 있도록 지원한다.

gfnTransaction() 명령을 수행하기 위해서는 반드시 dsTransaction 이라는 DataSet을 정의해야 한다. dsTransaction의 column 정보를 확인 해보면 다음과 같다.

	type	size	prop	sumtext
D	STRING	100		
TASET_...	STRING	256		
DATASE...	STRING	256		
	STRING	512		
BACK	STRING	100		
_YN	STRING	1		

D	IN_DATASET_LIST	OUT_DATASET_LIST	URL	CALL
tMovie	dsSearch=dsSearch	dsGrdMovie=dsGrdMovie	xplatformMovie.do?method=getList	
llMovie	dsGrdMovie=dsGrdMovie:U		xplatformMovie.do?method=saveAll	
tGenre		dsGrdGenre=dsGrdGenre	xplatformGenre.do?method=getList	

Columns	Description
SVC_ID	transaction을 구분하기 위한 ID값
IN_DATASET_LIST	transaction을 요청할 때 입력값으로 보낼 DataSet의 ID
OUT_DATASET_LIST	transaction 처리 결과를 받을 DataSet ID
URL	transaction을 요청할 주소
CALLBACK	transaction 결과를 돌려줄 function 이름.
bAsync	비동기 여부

스크립트에서 gfnTransaction()을 호출하는 방법은 다음과 같다.

```
gfnTransaction("getListMovie");
```

gfnTransaction() 메소드에서 인자값으로 받은 "getListMovie"는 dsTransaction에서 정의한 SVC_ID 값이다.