eXbuilder6

전자정부 표준 프레임워크3.6 연동 가이드

목 차

1 개요

eXbuilder6는 웹 UI 저작도구로서 서버는 어떤 언어로 개발되었더라도 연동이 가능합니다. 가이드에서는 웹프로젝트에서 연동 방법을 설명합니다.

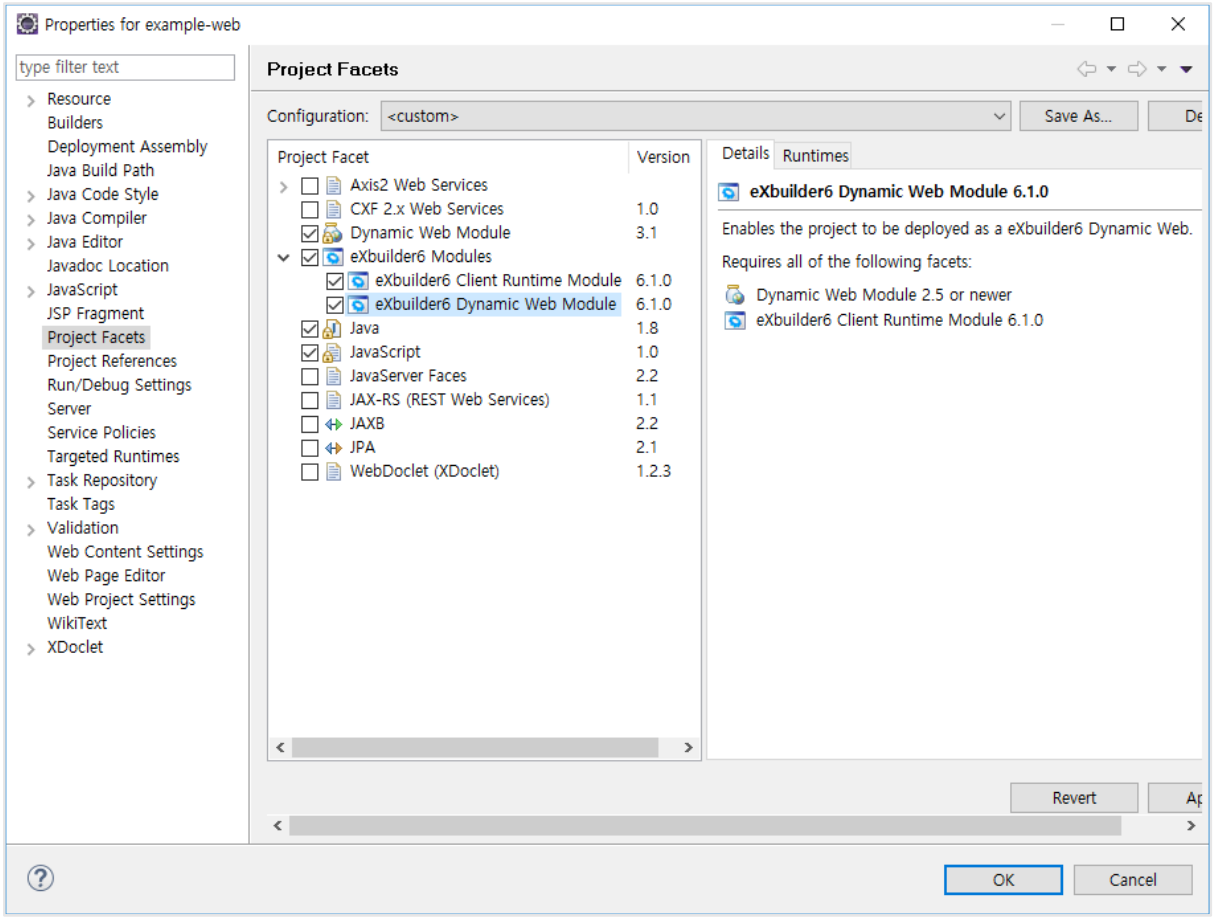
2 eXbuilder6 웹프로젝트 설정

eXbuilder6를 웹프로젝트에서 사용하기 위해 웹프로젝트에서 Facet으로 구성요소 설정이 필요하며, 별도로 작성된 eXbuilder6의 프로젝트를 배포 설정하여 clx파일을 접근할 수 있도록 설정합니다.

가) 모듈구성(Facet)

eXbuilder6 프로젝트는 eclipse에서 Java Project와 Dynamic Web Project와의 관계처럼 배포 설정을 통해 Web Project에 추가하여 배포할 수 있습니다. eXbuilder6 프로젝트의 추가 배포를 위해서 Dynamic Web Project는 eXbuilder6 프로젝트를 포함할 수 있는 facet의 구성이 필요합니다.

- Project Explorer에서 생성된 Dynamic Web Project를 선택한 후 마우스 우클릭 또는 Alt+Enter 키를 입력하면 Properties 다이얼로그가 출력됩니다.
- 좌측 트리에서 Project Facets를 선택하면 아래와 같은 창이 출력됩니다.



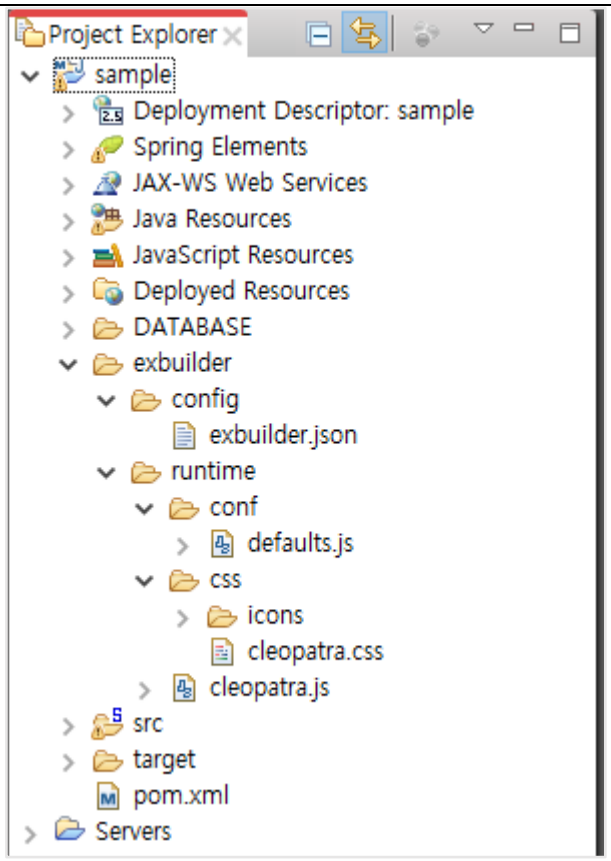
Project Facets의 eXbuilder6 Module

eXbuilder6 Facets의 모듈 설명

모듈명	설명
eXbuilder6 Client Runtime Module	eXbuilder6 실행환경이 브라우저에서 동작할 수 있도록 하는 JavaScript와 기본 Theme 관련된 파일의 설치 여부를 결정합니다.
eXbuilder6 Dynamic Web Module	eXbuilder6의 Protocol 처리 등을 도와줄 수 있는 서버 플러그인을 Build Path에 포함할지 여부를 결정합니다.

나) 전자정부프레임워크 웹프로젝트에 eXbuilder6 Facet 설정

Project Facets의 eXbuilder6 Modules에 있는 두 모듈을 모두 선택하고 [Apply]버튼을 클릭합니다. eXbuilder6 프로젝트를 연동하는데 필요한 구성요소가 웹 프로젝트에 설정됩니다.



eXbuilder6의 Facet이 적용된 웹 프로젝트의 폴더 구조

프로젝트에 생성되는 eXbuilder6 리소스

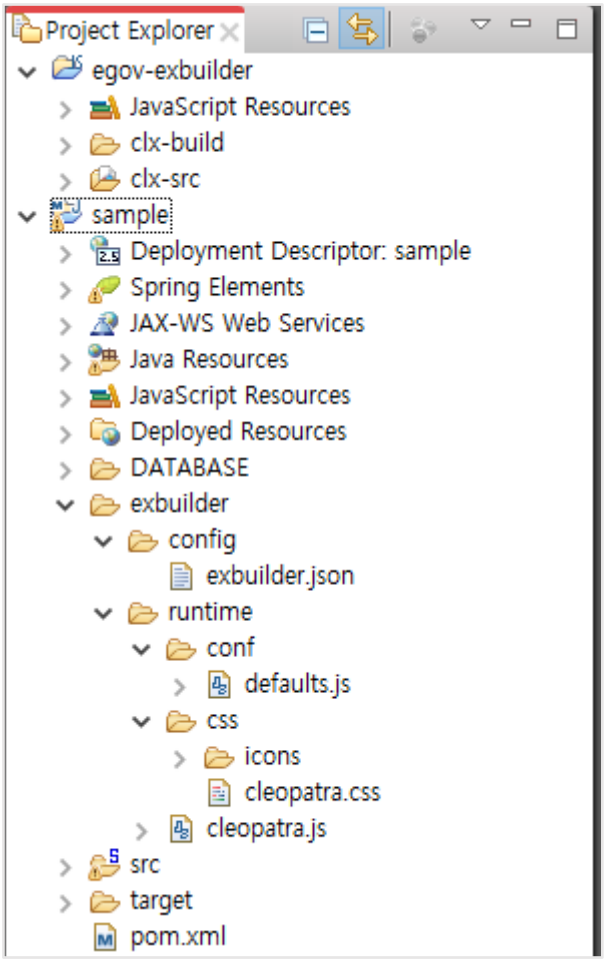
경로	설명
/exbuilder/config	eXbuilder6 서버 플러그인에서 처리할 설정파일이 위치하는 경로입니다. 실행환경에서 사용할 JavaScript와 CSS파일 목록과 FileUpload 처리의 기본 설정 그리고 함께 배포되는 eXbuilder6 프로젝트의 배포 경로 등이 저장되어 있습니다.
/exbuilder/runtime/conf	실행환경에서 사용하는 기본 설정이 저장된 경로입니다.
/exbuilder/runtime/css	eXbuilder6 기본 테마 CSS 파일과 관련 이미지 등이 위치하는 경로입니다. 변경이 필요할 경우 별도 경로에 CSS 파일을 설치한 후 서버 설정을 변경하기를 권장합니다.
/exbuilder/runtime/cleopatra.js	eXbuilder6의 실행환경 JavaScript 파일입니다.

다) 배포설정(Deployment Assembly)

배포설정은 Dynamic Web Project가 배포될 때 eXbuilder6 프로젝트를 내장하는 방법을 제공합니다. 이 과정은 Dynamic Web Project에 Java Project를 배포하는 과정과 유사하게 진행됩니다.

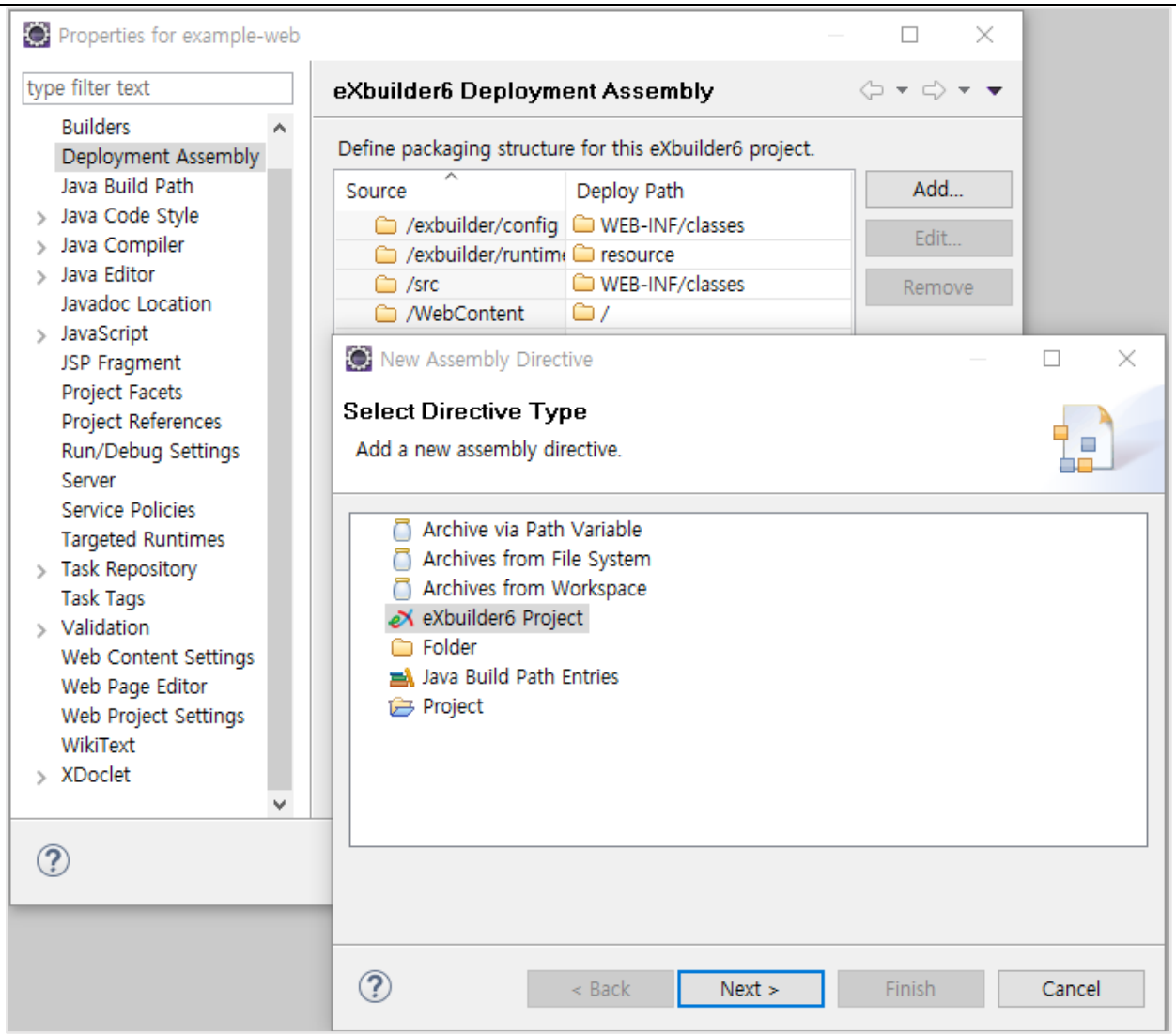
전자정부프레임워크 웹프로젝트에 Deployment Assembly 설정

상단메뉴에서 **File > New > eXbuilder 프로젝트**를 선택하고 프로젝트 이름을 "egov-exbuilder"로 입력하여 생성합니다.



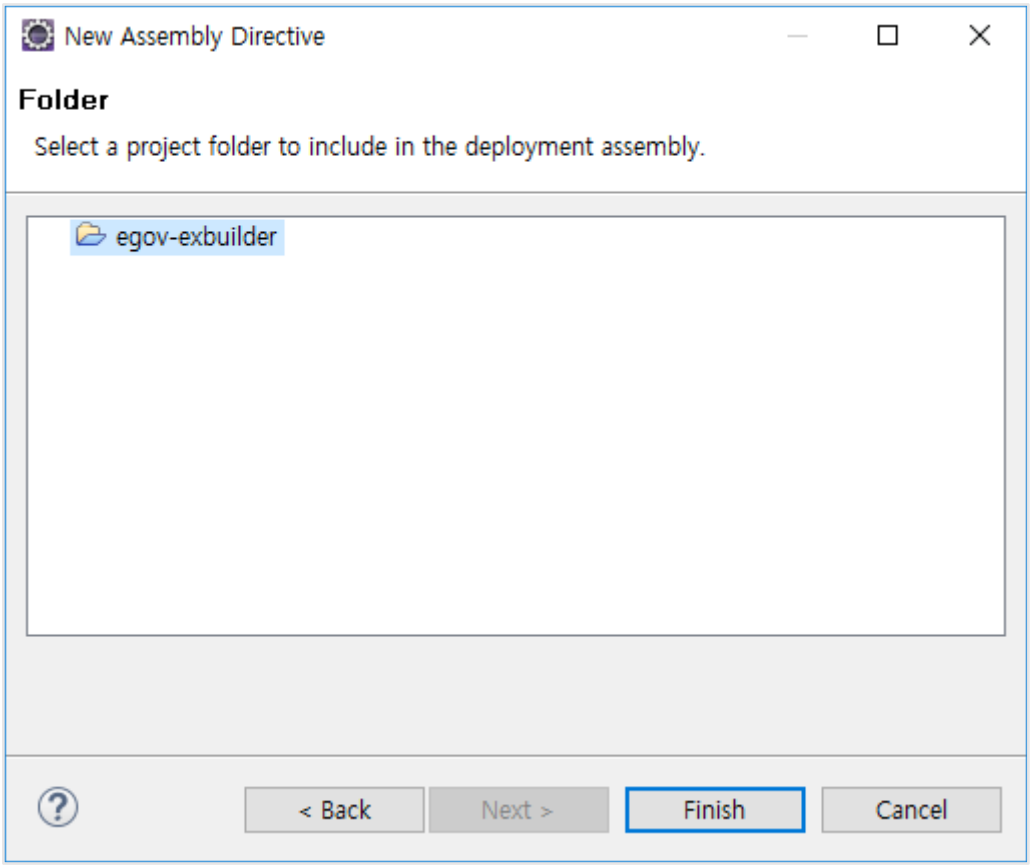
생성된 egov-exbuilder 프로젝트

Dynamic Web Project의 Properties 다이얼로그에서 Deployment Assembly를 선택합니다.



eXbuilder6 Project로 선택 후 [Next]버튼 클릭

우측 화면에서는 Project Facet 설정으로 인해 exbuilder 로 시작하는 배포 디렉토리가 추가된 것을 볼 수 있습니다. 우측 [Add]버튼을 클릭하면 New Assembly Directive 다이얼로그가 출력됩니다. 이 창에서 eXbuilder6 Project를 선택하고 [Next]버튼을 클릭합니다.



egov-exbuilder를 선택한 후 [Finish]버튼 클릭

프로젝트가 추가되면 위 화면과 같이 추가된 프로젝트가 목록에 출력됩니다. 우측 Deploy Path 부분을 편집하여 배포될 경로를 변경할 수 있습니다. 이렇게 추가된 배포 설정 정보는 서버 설정파일에 자동 반영됩니다.

3 서버 프레임워크 연동

서버에서 clx파일에 대한 요청이 오면 서버에서 이에 관련된 **HTML View페이지로 응답하는 클래스 작성**과 스프링에 eXbuilder6에서 제공하는 **DataResolver를 설정**합니다. 그리고 서버에서 eXbuilder6에 관련된 초기 설정하는 **XBInitializer을 설정**하면 연동작업이 완료됩니다.

가) web.xml 설정

웹프로젝트의 web.xml에서 XBInitializer를 설정하고, clx파일을 url에서 사용할 수 있도록 "*.clx"를 추가합니다.

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.5" xmlns="http://java.sun.com/xml/ns/javaee"
                    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
                    xmlns:web="http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
                    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd">
        <display-name>egovframework.sht</display-name>
        <listener>
                <listener-class>com.cleopatra.XBInitializer</listener-class>
        </listener>
...(생략)
        <servlet-mapping>
                <servlet-name>action</servlet-name>
                <url-pattern>*.do</url-pattern>
                <url-pattern>*.clx</url-pattern><!-- eXbuilder6 확장자 추가 -->
        </servlet-mapping>
...(생략)
```

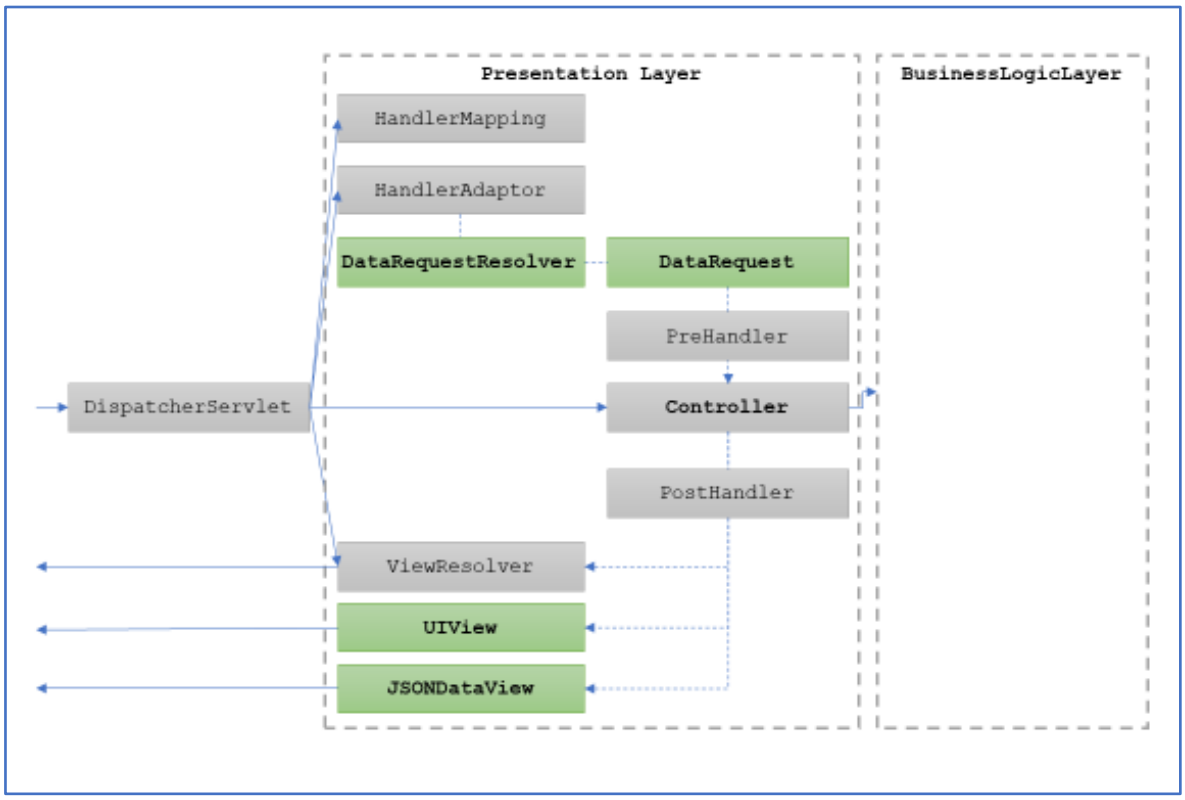
- com.cleopatra.XBInitializer 는 eXbuilder6 서버 플러그인의 초기 설정을 처리하는 Listener입니다.
- dispatcher servlet-mapping 부분에 *.clx 패턴을 처리할 수 있도록 url-pattern을 추가합니다.

나) spring 설정(egov-com-servlet.xml)

```
...(생략)
<!-- eXbuilder6의 데이터 라이브러리 사용 -->
<bean class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter">
    <property name="customArgumentResolvers">
        <list>
            <bean class="com.cleopatra.spring.DataRequestResolver"></bean>
        </list>
    </property>
</bean>
...(생략)
```

- com.cleopatra.spring.DataRequestResolver 는 eXbuilder6 프로토콜을 파싱하여 DataRequest 파라미터를 자동 생성해 주는 Resolver입니다.

다) eXbuilder6 연동 아키텍쳐



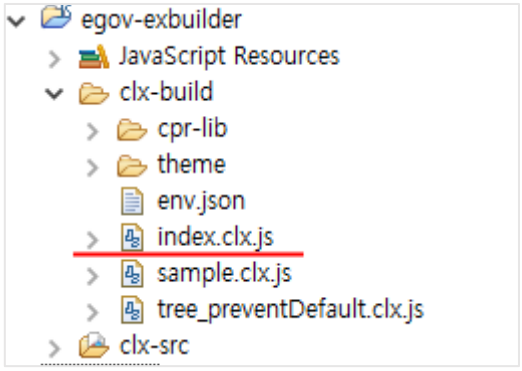
eXbuilder6에서 제공하는 DataRequest를 통하여 클라이언트에서 받은 가공된 데이터를 받거나 응답 데이터로 설정 할 수 있습니다. UIView로 페이지를 보여주며, JSONDataView는 JSON형식으로 클라이언트로 데이터를 전송합니다.

4 View

eXbuilder6는 자바스크립트 기반으로 되어 있어 어떤 환경에서도 사용이 가능합니다. 표준프레임워크에서 View를 생성하는 방법은 ViewResolver를 이용한 방법과 eXbuilder6에서 제공하는 어댑터를 사용하는 방법이 있습니다.

가) ViewResolver

표준프레임워크의 ViewResolver 를 이용하는 방법으로 Controller 과 jsp 페이지를 작성합니다. 작성 전에 eXbuilder6 프로젝트에서 생성되는 빌드 파일인 [파일명].clx.js 을 확인하시기 바랍니다.



eXbuilder6 프로젝트에서 View 에 보여줄 파일

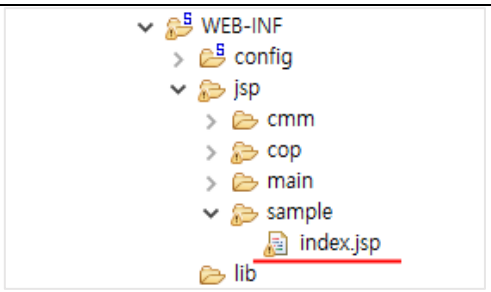
index.clx.js 에서 앱의 이름 확인

```
/*
 * App URI: index
 * Source Location: index.clx
 *
 * This file was generated at 2018. 2. 13 오후 4:19:20, Don't edit manually.
 */
(function(){
	var appDelegate = {
		onPrepare: function(loader){
		},

...(생략)

	var app = new cpr.core.App("index", appDelegate);
app.title = "index";
	cpr.core.Platform.INSTANCE.register(app);
})();
```

웹 페이지 작성



표준프레임워크에서 View 로 보여줄 파일

index.jsp 의 내용

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
	pageEncoding="UTF-8"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html lang="ko" style="width:100%;height:100%;">
<base href="/sht_webapp/ui/">
<meta http-equiv="cache-control" content="no-cache">
<meta http-equiv="expires" content="0">
<meta http-equiv="pragma" content="no-cache">
<meta charset="utf-8">
<meta name="viewport"
	content="user-scalable=1, initial-scale=1.0, width=device-width">
<title>index</title>
<script src="../resource/cleopatra.js"></script>
<script src="../resource/conf/defaults.js"></script>
<script src="cpr-lib/language.js"></script>
<script src="cpr-lib/user-modules.js"></script>
<script type="text/javascript" src="index.clx.js"></script>
<link rel="stylesheet" media="all" href="../resource/css/cleopatra.css">
<link rel="stylesheet" media="all" href="theme/cleopatra-theme.css">
<script>
	function doload() {
		cpr.core.Platform.INSTANCE.lookup("index").createNewInstance().run();
	}
</script>
</head>
<body onload="javascript:doload();" style="width:100%;height:100%;">

</body>
```

```
</html>
```

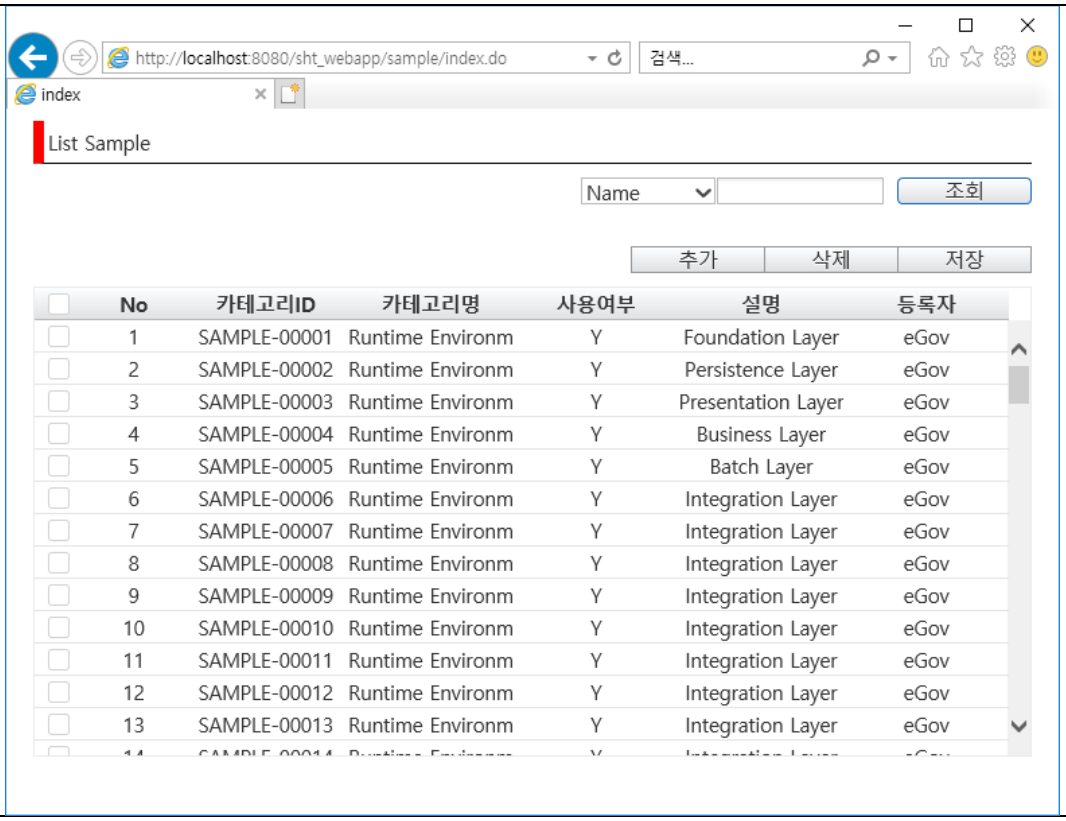
- base 엘리먼트의 href 속성은 웹프로젝트에서 Deployment Assembly 에서 설정한 경로를 설정합니다.
- index.clx.js 는 화면에 보여줄 페이지로 eXbuilder6 프로젝트에서 빌드 된 폴더에서 확인 할 수 있습니다.
- cpr.core.Platform.INSTANCE.lookup("index").createNewInstance().run(); 의 "index" 이름은 index.clx.js 파일의 내용 안에 앱의 아이디를 작성합니다.

컨트롤러에 View Resolver 작성

```
@RequestMapping(value = "/sample/index.do")
public String indexView(HttpServletRequest request, HttpServletResponse response, ModelMap model){
        return "sample/index";
}
```

Controller 에서 View 작성

설정된 내용으로 웹서버를 기동하여 확인하여 페이지를 확인 할 수 있습니다.



URL 실행 결과

화면이 출력되지 않는다면 index.jsp 에 작성한 내용 중에서 script 파일의 경로를 확인 바랍니다.

나) eXbuilder6 UI Adapter

View Resolver를 통한 방법은 각각의 페이지를 작성해야 하는 어려움이 있습니다.

이것을 해소하기 위해 eXbuilder6에서는 어댑터를 제공하고 있습니다. 어댑터는 파일명을 url로 받아 eXbuilder6의 프로젝트에 deployment된 경로의 페이지를 반환하여 HTML을 만들어주는 편의성을 제공합니다.

eXbuilder6 UIController 작성

initPageGenerator에서는 com.cleopatra.ui.PageGenerator를 이용하여 처리할 요청 URL 패턴을 정의합니다. 이후 *.clx 요청에 대해 UIView를 생성하고 리턴하면 eXbuilder6 플러그인에서는 배포된 eXbuilder6 프로젝트 자원 중 해당하는 파일을 찾아 화면을 출력할 수 있는 HTML을 생성하여 브라우저로 전송합니다.

```
package egovframework.com.cmm;

import java.io.IOException;
import javax.annotation.PostConstruct;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.servlet.View;
import com.cleopatra.spring.UIView;
import com.cleopatra.ui.PageGenerator;

@Controller
public class CleopatraUIController {
	public CleopatraUIController() {
    }

    @PostConstruct
    private void initPageGenerator() {
        PageGenerator instance = PageGenerator.getInstance();
        instance.setURLSuffix(".clx");
    }

    @RequestMapping("/**/*.clx")
    public View index(HttpServletRequest request, HttpServletResponse response) throws IOException {
        return new UIView();
    }
}
```

5. Model

eXbuilder6는 ajax통신을 사용하고 있습니다. 사용자에게 eXbuilder6와 서버간에 데이터 전송에 편의성을 제공하기 위해 JSONDataView를 제공하고 있습니다.

가) JSONDataView 사용

UIView 클래스를 통해 화면의 페이지를 생성한다면 JSONDataView는 eXbuilder6에 데이터를 전송하는 프로토콜입니다.

```
@RequestMapping("/selectList.do")
public View selectList(DataRequest dataRequest, HttpRequestServlet request, HttpResponseServlet response){

  //eXbuilder6에서 받은 데이터를 dataRequest를 통하여 파싱된 정보로 가져옵니다.
  ParameterGroup parameterGroup = dataRequest.getParameterGroup("dm_sample");

  String condition = parameterGroup.getValue("condition");
  Map<String,String> param = new HashMap<String,String>();

  List<Map<String,Object> list = service.selectList(param);
  //response에 exbuilder6의 데이터셋 ID와 데이터를 설정합니다.
  dataRequest.setResponse("ds_sample",list);

return new JSONDataView();
}
```