

WebSquare5 기본

- WebSquare5 특징
- WebSquare5 Studio
- 컴포넌트 설명
- UI Design
- 컴포넌트와 Data 연동
- 데이터 객체와 Submission
- 디버깅

| WebSquare5 특징

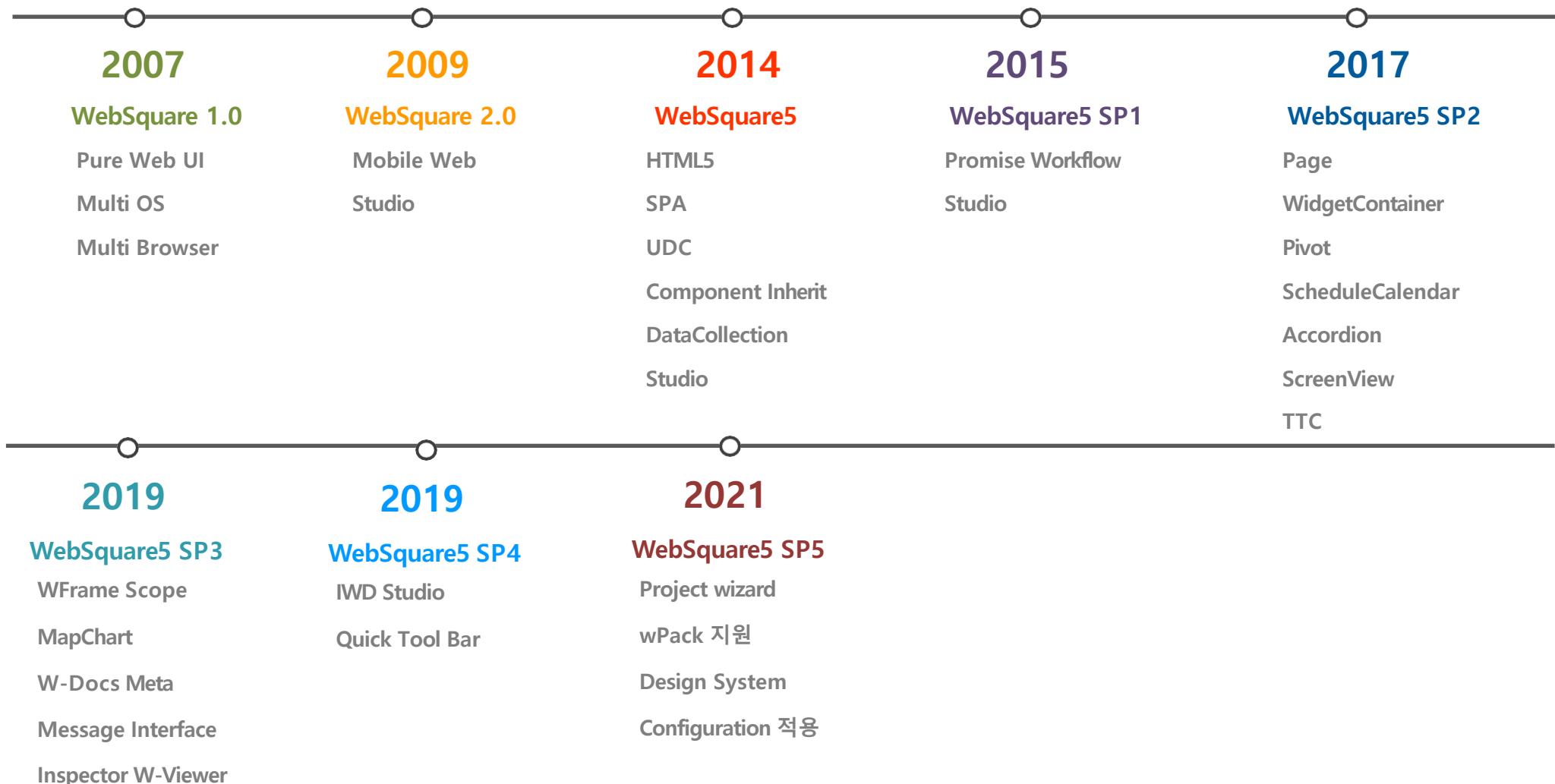
- 소개
- 주요기술
- 지원사양
- 실행환경 Architecture
- 화면호출
- 서버통신
- WebSquare5 소스 구조
- WebSquare5 Data Model
- WebSquare5 Script 코딩
- 프로젝트 시작 전 최소 체크 사항
- 개발 참고 사이트

WebSquare5는 웹 표준을 준수한 UI 프레임워크로 UI 컴포넌트와 Util API, 통합 개발 도구를 제공한다.

2007년 국내 최초 웹 표준 UI 플랫폼인 WebSquare1.0을 시작으로,

2014년 HTML5가 웹 표준으로 권고됨에 맞춰 같은 해 대규모 업그레이드된 WebSquare5를 출시하였습니다.

이후 ServicePack(SP)을 통해 신규 컴포넌트 및 성능 향상을 위한 기능과 개발 도구의 편이성 등을 제공하고 있습니다.



| 구 분 | 설 명 |
|------------|---|
| HTML | 사용자(Client)가 최종적으로 보는 화면(브라우저)에 표현되는 결과물 |
| JavaScript | WebSquare Libray 화면 내 로직 구성 |
| CSS | 화면 디자인 |
| XHR | 데이터 통신 모듈 (submission과 ajax util) JSON, XML 형식의 데이터 (그 외 암복호화 문자열 등의 문자열) |
| XML | WebSquare 화면 파일 (소스 파일) |
| wPack(js) | 화면 xml을 js로 compile (배포용 소스) |
| JAVA | WebSquare Libray 라이선스 체크 UI와 연결 된 서버 기능(파일 업로드, 그리드뷰의 엑셀 업/다운로드, 엔진 다운로드 등) |
| XForms | 화면 소스 구성 스펙 |

웹스퀘어5는 화면 소스를 JavaScript로 빌드할 수 있는 W-Pack을 제공합니다.

스튜디오에서 개발한 화면 파일은 XML 형식으로 생성되며, W-Pack은 이를 JavaScript 파일로 변환합니다.



파일 크기 감소

압축 및 최소화(Minify) 기능을 제공하여, 기존 XML 형식과 비교하여 파일 크기가 감소되었습니다.

보안성 개선

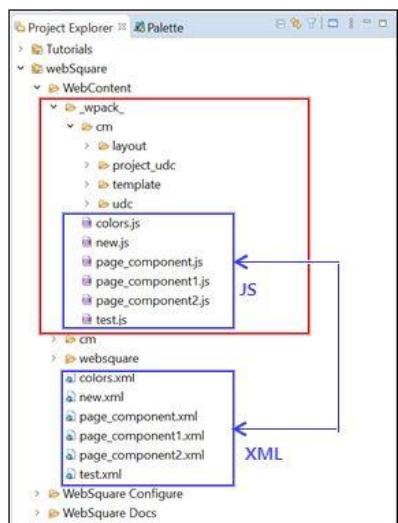
난독화(Obfuscation)를 통해 소스 코드의 보안성을 높였습니다.

속도 개선

웹 환경에 최적화된 JS 파일은 XML과 비교하여 로딩 시간이 짧고 메모리 사용량이 적어, 웹스퀘어 엔진은 JS 파일을 이용하여 더욱 빨리 화면을 브라우저에 그릴 수 있습니다.

XML 화면 파일은 JS 파일로 변환되어 [/WebContent/_wpack_](#) 폴더 아래에 저장됩니다.

화면 파일 개발에 사용한 공통 리소스도 JS 파일로 변환되어 [/WebContent/_wpack_cm/](#) 폴더 아래에 저장됩니다.



WebSquare Studio에서 작성한 화면은 XML 파일로 생성되며, W-Pack에 의해 JS 파일로 변환되며,

브라우저는 XML 파일을 표시하나 화면을 표시할 때 실제로 사용하는 파일은 JS 파일입니다.



WebSquare5로 개발한 화면의 브라우저 호출 순서는 아래와 같습니다.

1. 브라우저에서 화면을 호출합니다.
2. 서버 쪽 WebSquare5 엔진의 **websquare.html** 파일이 호출됩니다.
3. WebSquare5 엔진이 구동됩니다.
4. 화면에 해당하는 XML 파일을 URL로 호출합니다.
5. W-Pack에 의해 변환된 JS 파일이 로딩됩니다.
6. 해당 화면이 브라우저에 표시됩니다.

WebSquare browser output process:

- Web browser address bar: 127.0.0.1:60685/websquare/websquare.html?w2xPath=/edu/test.xml
- Developer tools Network tab: Shows the loaded JS file: test.js?postfix=0.037491119023 javascriptLoader.wq
- Developer tools Source tab: Shows the loaded JS code.
- Bottom right callout: 웹브라우저에는 호출한 화면의 XML 파일 명이 표시됩니다. (The XML file name of the called screen is displayed in the web browser.)
- Bottom right callout: 실제 브라우저에 구동되는 파일은 이와 같이 JS 파일입니다. (The file actually executed in the browser is a JS file like this.)
- Bottom right callout: 브라우저에서 (F12) 실행 개발자 도구>소스에서 확인 (Check in the browser (F12) execution developer tools > source.)

OS 지원 범위

| 구 분 | 지원 범위 |
|-------|---|
| 설치 서버 | Unix, Linux, Windows, z/OS 지원 |
| 클라이언트 | Linux, Windows (XP/7/8/8.1/10), Mac OS X, iOS, Android 지원 |
| 개발 환경 | Linux, Windows, Mac OS X 지원 |

JVM 지원 범위

| 구 분 | 지원 범위 |
|-------|--|
| 설치 서버 | JVM 1.6 이상 WAS에서 실행되어 WAS의 JVM 지원 범위와 동일함 |
| 클라이언트 | N/A |
| 개발 환경 | JDK 1.6 이상 |

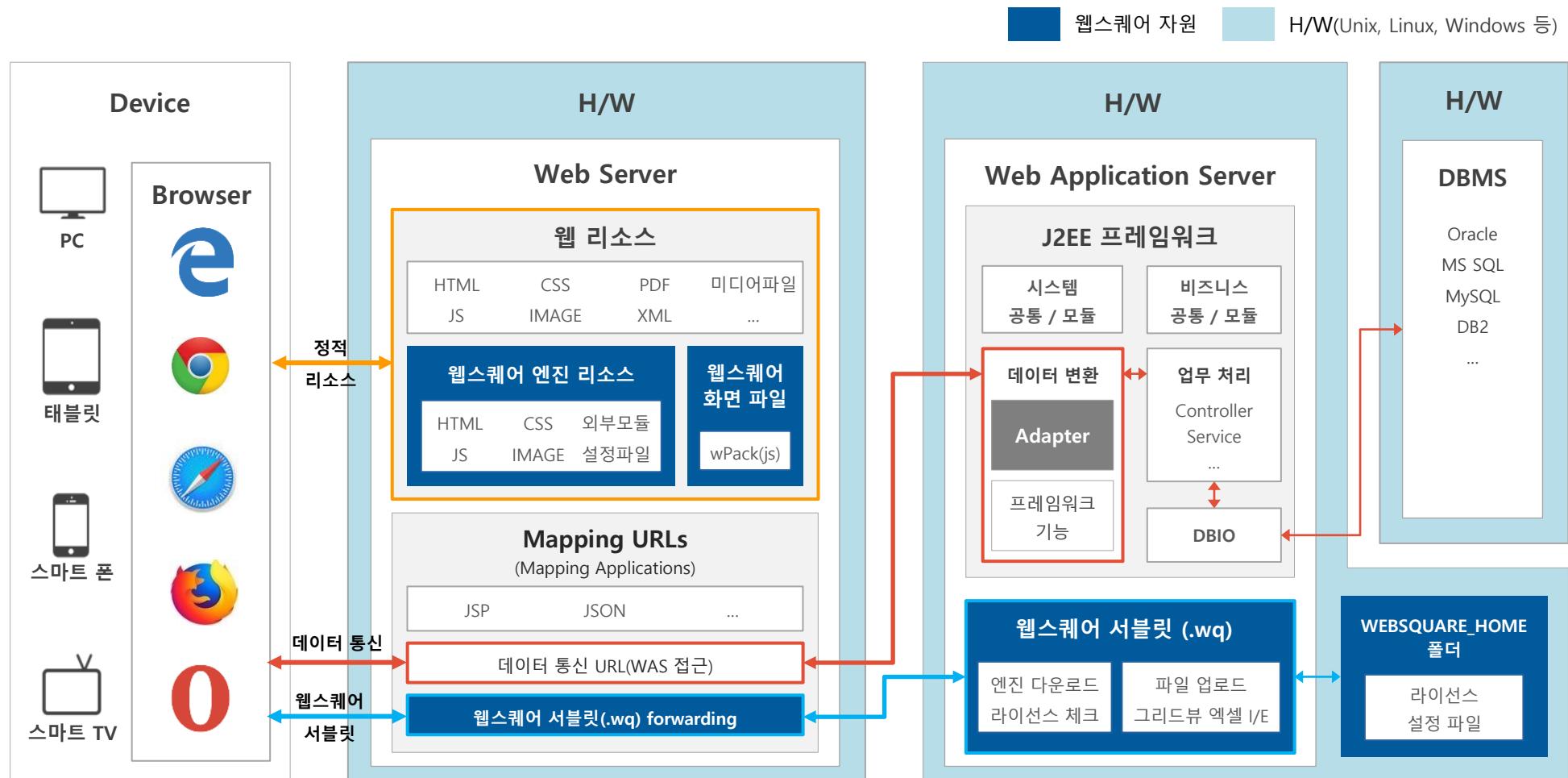
Studio(개발 툴)의 Eclipse 지원 범위

| 구 분 | 지원 범위 |
|--------------|---|
| ServicePack1 | Stand-alone : 3.7(Indigo) 기반 Plug-in : 4.4(Luna) 이하 |
| ServicePack2 | Stand-alone : 4.3(kepler) 기반 Plug-in : 4.7(Oxygen) 이하 |
| ServicePack3 | Stand-alone : 4.7(Oxygen) 기반 Plug-in : 4.7(Oxygen) 이하 |
| ServicePack4 | Stand-alone : 4.7(Oxygen) 기반 Plug-in : 4.5(Mars) ~ 4.16 이하 |
| ServicePack5 | Stand-alone : 4.15 기반 Plug-in : 4.15 ~ 4.19 이하 |

Studio(개발 툴)에 대한 로컬 PC 최소 사양 및 권장 사양

| 구 분 | 지원 범위 |
|-------|--|
| 최소 사양 | CPU : 팬티엄 4 이상, RAM : 2GB 이상 OS : Windows XP/Vista/7/8/10 32bit or 64bit HDD: 1기가 이상의 여유 공간, 디스플레이 : 1024 * 768 이상 |
| 권장 사양 | CPU : 팬티엄 4 2.8Ghz이상, RAM : 4GB 이상 OS : Windows 7/8/10 32bit or 64bit HDD: 10기가 이상의 여유 공간, 디스플레이 : 1280 * 1024 이상 |

| 구 분 | 확장자 예시 | 설 명 |
|----------|----------------------|--|
| 정적 리소스 | .html, .js, .xml ... | 웹스퀘어 엔진 리소스, 웹스퀘어 화면 파일. |
| 웹스퀘어 서블릿 | .wq | - wq 확장자를 서블릿으로 등록. - JS 엔진 요청, UI 기능(파일 업로드, 그리드뷰 엑셀 업/다운로드) 요청. |
| 데이터 통신 | .do 등 | - Form 통신이 아닌 XHR 통신(데이터 String을 주고 받음). 데이터 포맷 : JSON / XML. |



웹스퀘어로 개발 된 화면 파일(이하 화면파일)은 단독으로 브라우저에서 실행할 수 없습니다.

화면 파일은 웹스퀘어 JavaScript 엔진(이하 웹스퀘어 엔진)를 통해 실시간 해석되어 브라우저에 표현됩니다.

웹스퀘어 엔진은 서버에 배포된 websquare.html 파일을 통해 호출되고 엔진 로딩이 끝나면 실행됩니다.

즉, 모든 웹스퀘어 화면 파일은 websquare.html을 통해 실행됩니다.

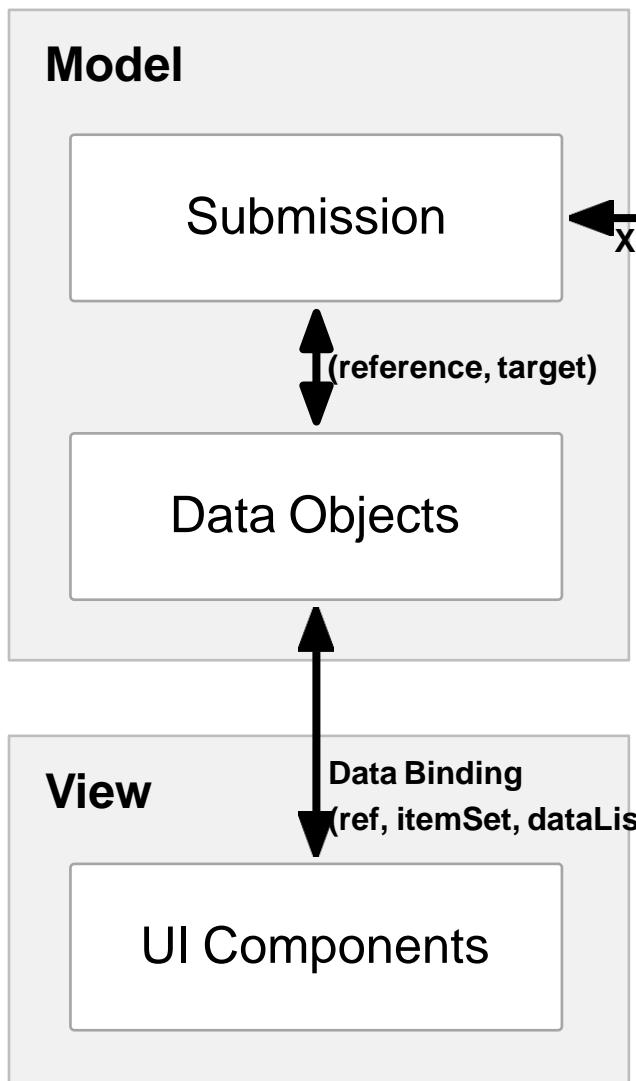
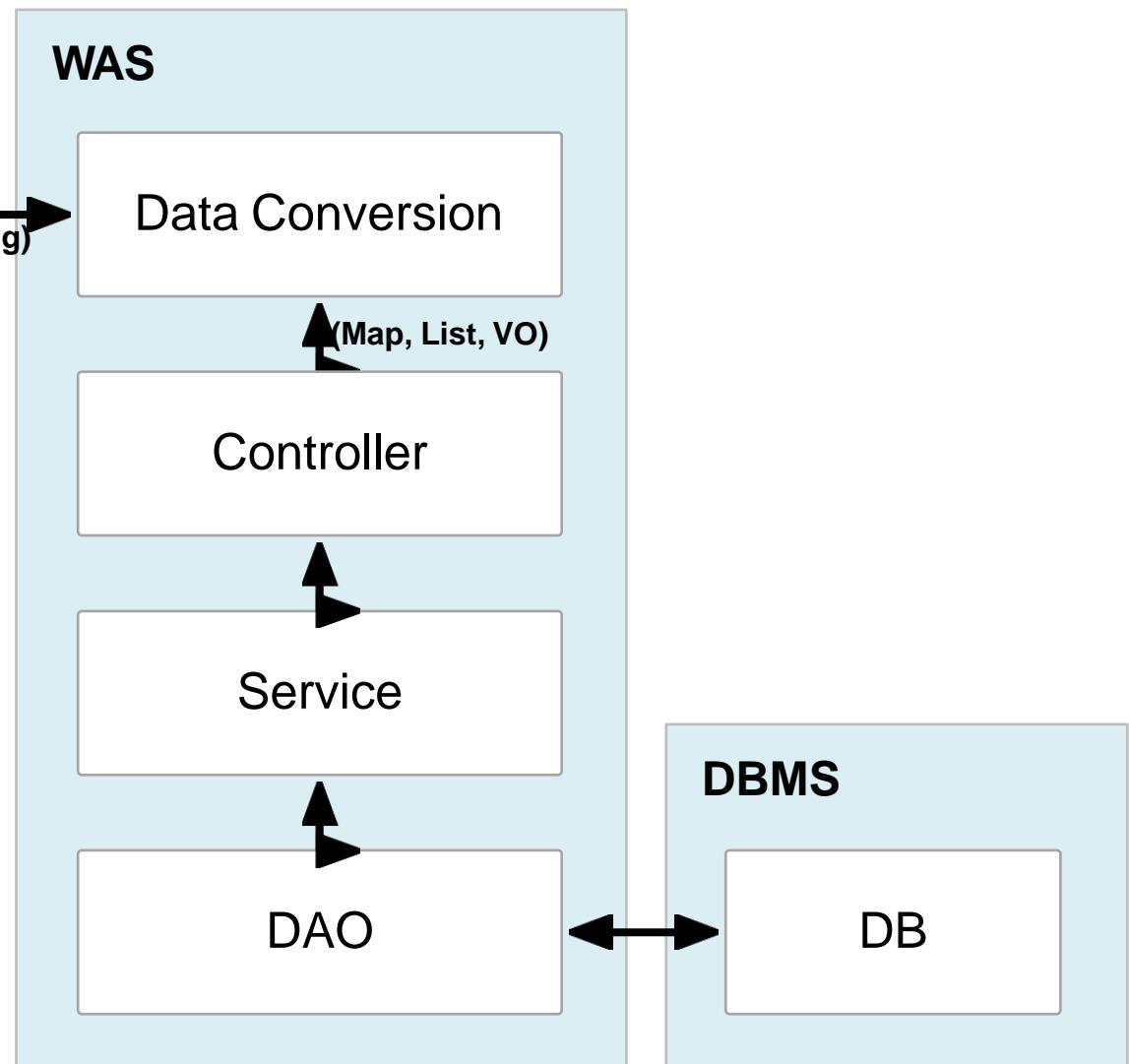
일반적으로 화면 파일의 정보는 Get방식의 파라메터를 통해 웹스퀘어 엔진에 전달됩니다.

다음은 웹스퀘어 화면을 브라우저에서 호출하는 URL의 예시와 URL을 분리하여 설명한 표 입니다.

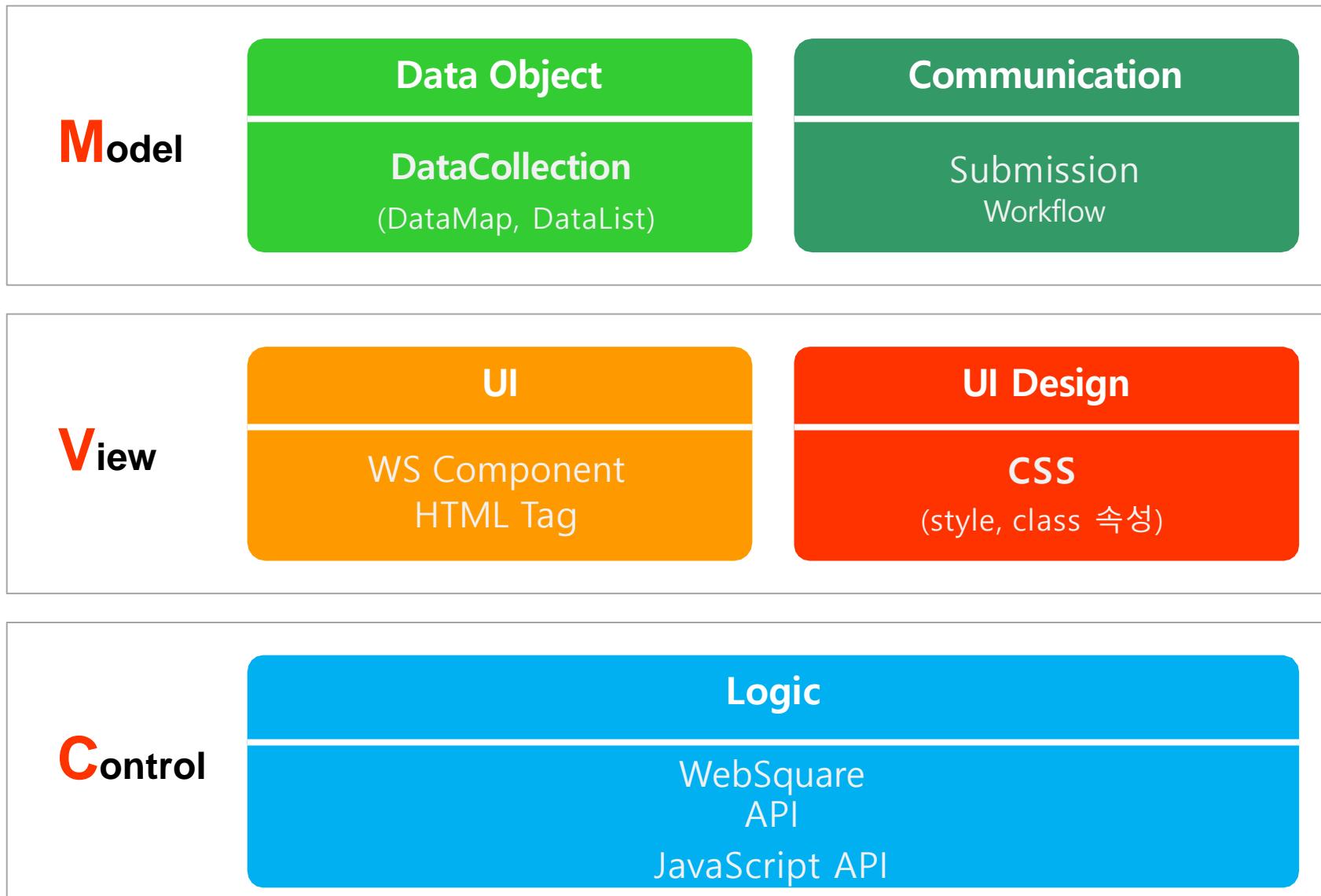
예시) **http://domain.com/websquare/websquare.html?w2xPath=/MA/MA01M01.xml**

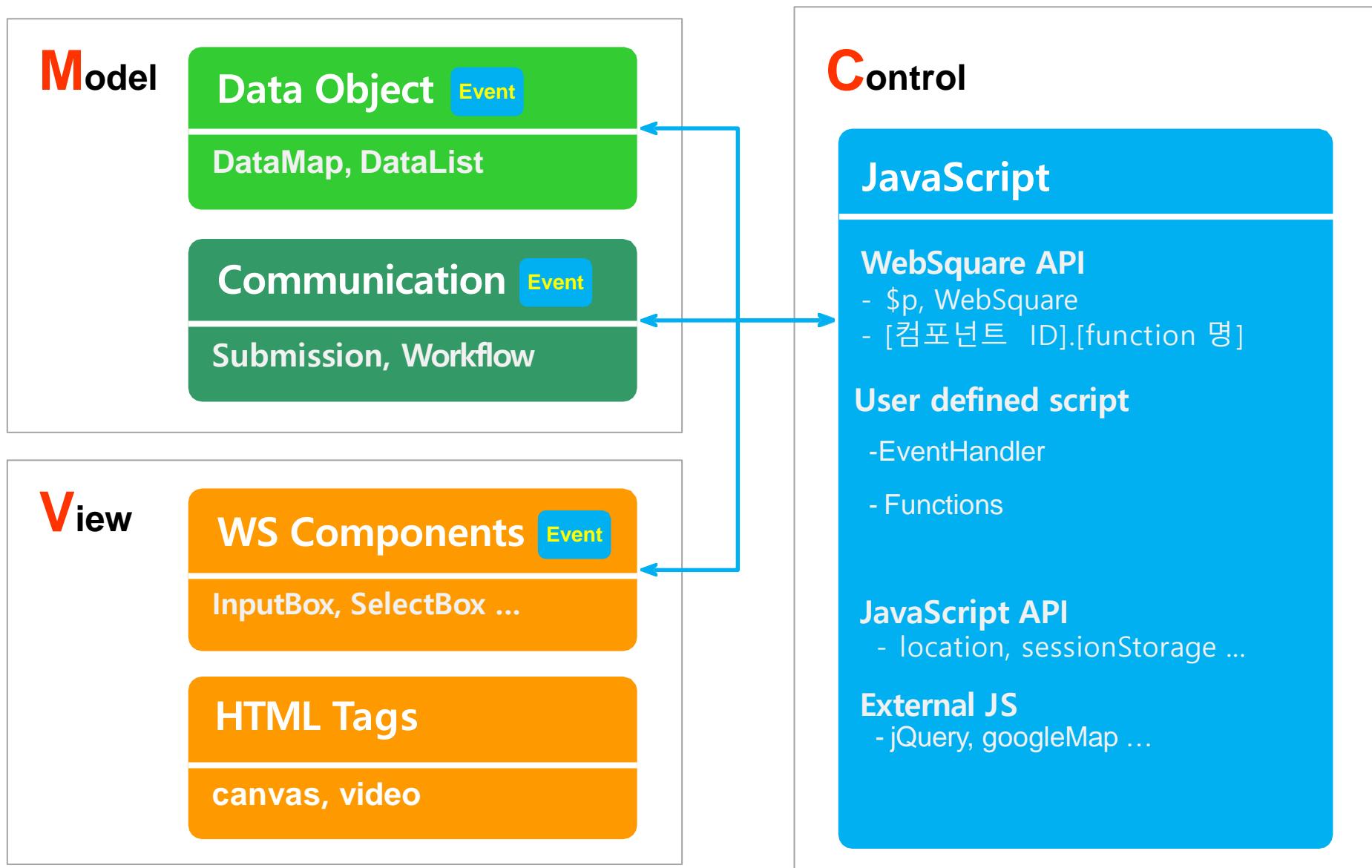
| 구 분 | 설 명 |
|----------------------------------|---|
| http://domain.com | 서버 도메인. |
| /websquare/websquare.html | -웹스퀘어 엔진 로딩 및 실행 파일. - HTML의 DocType, Meta 등을 정의할 수 있다. - 파일명 변경이 가능하다.(용도에 따라 파일을 여러 개 사용할 수 있다.) - 파일의 확장자를 jsp로 변경할 수 있다. |
| ?w2xPath=/MA/MA01M01.xml | - 웹스퀘어 화면 파일 경로. - 웹스퀘어 엔진이 파일을 해석하여 HTML과 JavaScript을 생성하고 실행한다. |



Client(WebSquare)**Server**

웹스퀘어 화면 소스는 XForms 스펙을 확장하여 MVC 패턴 코딩을 제공하며,
기존 HTML 코딩과 가장 큰 다른 점은 View와 Model이 분리되어 있다는 점입니다.





HTML DOCTYPE 정의

html

head

meta

title

External CSS

Internal Style

External JS

Script

업무 로직 구성

body

UI

HTML UI 태그

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>HTML</title>
<link rel="stylesheet" type="text/css" href="common.css">
<style type="text/css">
    .lbl01{
        color:steelblue;
    }
</style>
<script type="text/javascript" src="common.js"></script>
<script type="text/javascript">
    var tmpObj = {};

    tmpObj.fn_submit = function(){
        var tmpObj = document.getElementById("ui_name");
        var tmpValue = tmpObj.value;
        if(confirm(tmpValue+"로 검색하시겠습니까?")){
            document.getElementById("frm01").submit();
        }
    };

    tmpObj.fn_init = function(){
        var tmpObj = document.getElementById("ui_name");
        tmpObj.value = "HTML Test"
    };

    window.onload = tmpObj.fn_init;
</script>
</head>
<body>
    <form id="frm01" name="frm01" action="/search/codeSearch.do" method="post" >
        <div style="padding:10px;" class="">
            <label for="ui_name" class="lbl01" >검색어 : </label>
            <input id="ui_name" name="searchKey" type="text">
            <input name="searchType" type="hidden" value="C001">
            <input type="button" value="click" class="lbl01" onclick="javascript:tmpObj.fn_submit()">
        </div>
    </form>
</body>
</html>
```

XML 정의**External css link****html****head****Model**

- dataCollection
- workflowCollection
- submission

External JS**Script**

업무 로직 구성

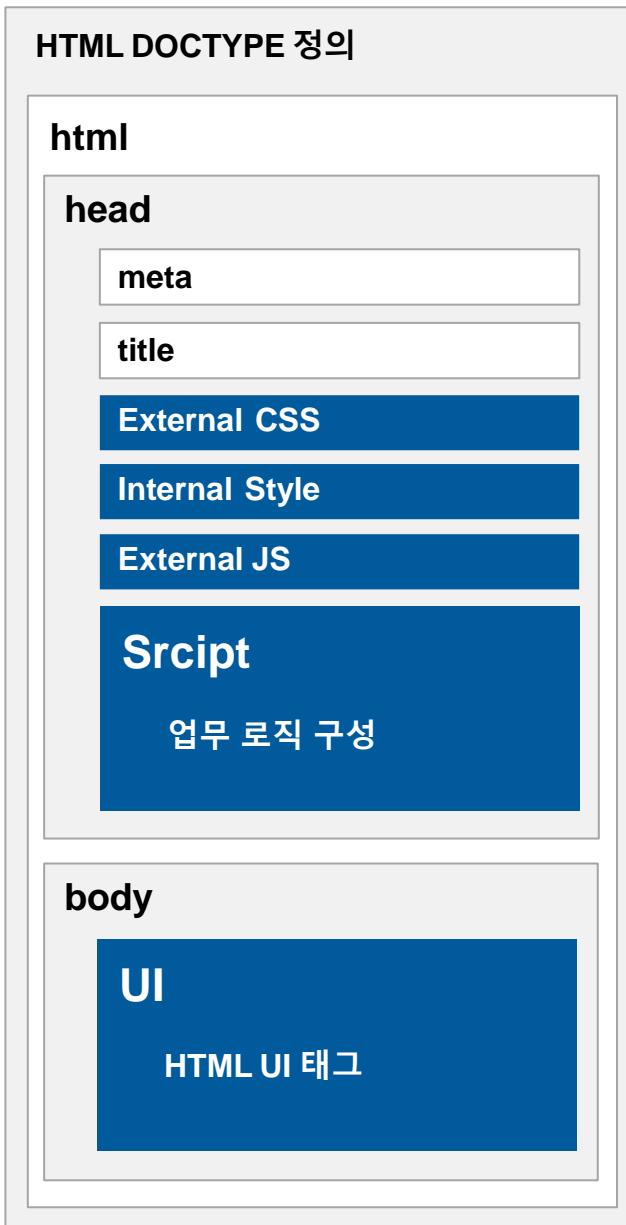
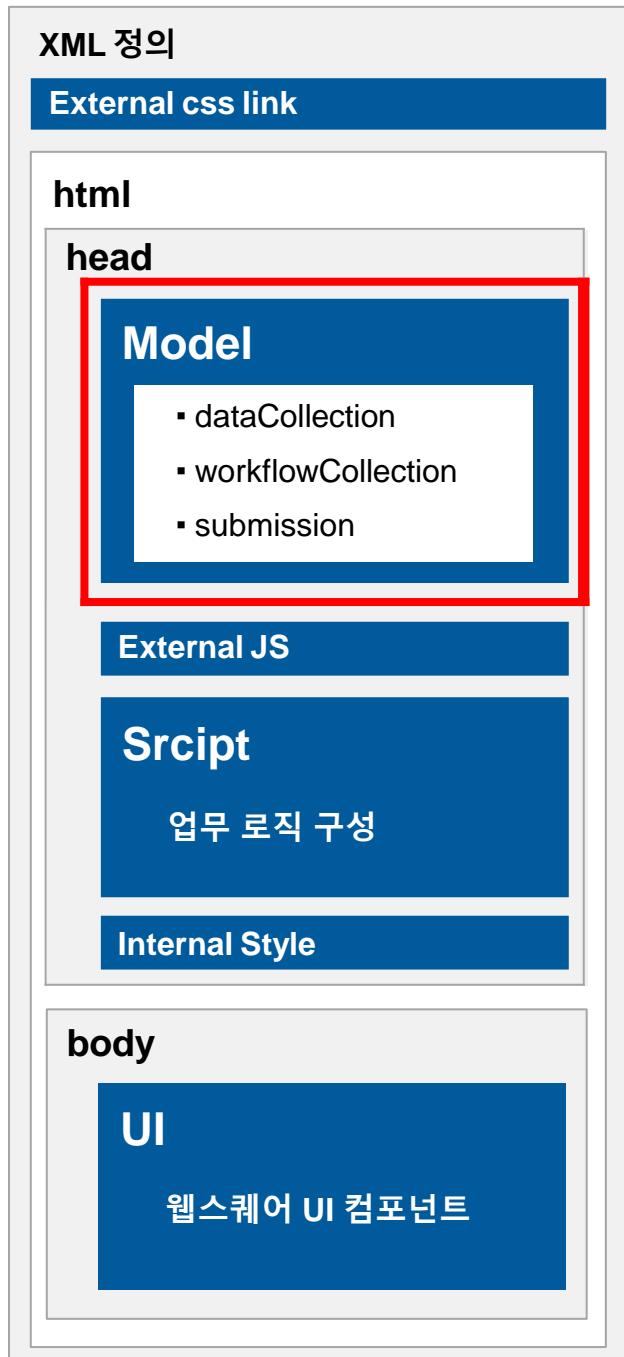
Internal Style**body****UI**

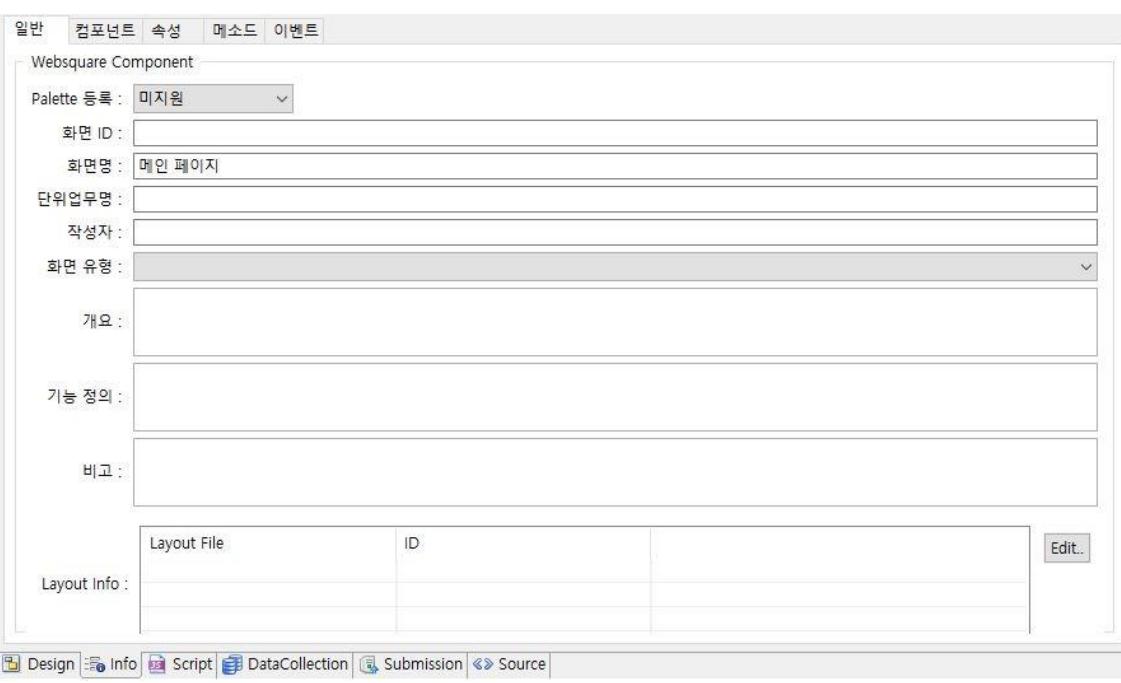
웹스퀘어 UI 컴포넌트

```

<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet href="/ws5/css/color.css" type="text/css"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:ev="http://www.w3.org/2001/xml-events"
      xmlns:w2="http://www.inswave.com/websquare" xmlns:xf="http://www.w3.org/2002/xforms">
<head>
    <w2:type>DEFAULT</w2:type>
    <w2:buildDate />
    <xf:model>
        <xf:instance>
            <data xmlns="" />
        </xf:instance>
        <w2:dataCollection baseNode="map">
            <w2:dataMap baseNode="map" id="dataMap1">
                <w2:keyInfo>
                    <w2:key id="searchType" name="검색종류" dataType="text"></w2:key>
                    <w2:key id="searchKey" name="검색어" dataType="text"></w2:key>
                </w2:keyInfo>
                </w2:dataMap>
            </w2:dataCollection>
            <w2:workflowCollection></w2:workflowCollection>
            <xf:submission id="submission1" ref="data:json,dataMap1" target="" action="/search/codeSearch.do" method="post"
                           mediatype="application/json" encoding="UTF-8" instance="" replace="" errorHandler="" customHandler="" mode="asynchronous"
                           processMsg="" ev:submit="" ev:submitdone="" ev:submiterror="" abortTrigger="">
                </xf:submission>
            </xf:model>
            <script type="text/javascript" src="/ws5/js/holiday.js"></script>
            <script type="text/javascript" lazy="false"><![CDATA[
                scwin.onpageload = function() {
                    ui_name.setValue("Websquare Test");
                };
                scwin.onpageunload = function() {
                };
                scwin.btn_search_onclick = function(e) {
                    $p.executeSubmission("submission1");
                };
            ]]></script>
            <style type="text/css"><![CDATA[
                .lbl01 { color : steelblue; }
            ]]></style>
        </head>
        <body ev:onpageload="scwin.onpageload" ev:onpageunload="scwin.onpageunload">
            <xf:group id="" style="padding: 5px;">
                <w2:textbox for="ui_name" style="width: 100px;float:left;clear:none;margin-right: 10px;" label="이름:" id="" class="lbl01"></w2:textbox>
                <xf:input adjustMaxLength="false" style="width: 144px;height: 21px;float:left;clear:none;margin-right: 10px;" id=""></xf:input>
                <xf:trigger style="width: 80px;height: 23px;" id="btn_search" type="button" ev:ondblclick="" ev:onclick="scwin.btn_search_onclick">
                    <xf:label><![CDATA[조회]]></xf:label>
                </xf:trigger>
            </xf:group>
        </body>
    </html>

```



| | |
|-------------------------|--|
| 설명 | <ul style="list-style-type: none"> - 화면정보를 기술하는 영역입니다. - sp5에서는 개발된 개별 페이지를 module화 시켜줄 수 있습니다. (개발된 단위화면을 마치 하나의 컴포넌트처럼 이용할 수 있는 기능이 있습니다.) |
| 소스 예시 |  <pre> 1 <?xml version="1.0" encoding="UTF-8"?> 2 <html xmlns="http://www.w3.org/1999/xhtml" 3 xmlns:ev="http://www.w3.org/2001/xml-events" 4 xmlns:w2="http://www.inswave.com/websquare" xmlns:xf="http://www.w3.org/2002/xforms"> 5 <head meta_screenName="메인 페이지" meta_vertical_guides="" meta_horizontal_guides=""> 6 <w2:type>COMPONENT</w2:type> 7 <w2:buildDate /></pre> |
| 스튜디오 예시 (일부) |  <p>The screenshot shows the 'Component' configuration dialog in the Websquare Studio. The tabs at the top are '일반', '컴포넌트', '속성', '메소드', and '이벤트'. The '컴포넌트' tab is selected. The form contains the following fields:</p> <ul style="list-style-type: none"> Palette 등록 : 미지원 화면 ID : 화면명 : 메인 페이지 단위업무명 : 작성자 : 화면 유형 : 개요 : 기능 정의 : 비고 : Layout File : ID <p>At the bottom, there are tabs for Design, Info, Script, DataCollection, Submission, and Source. The 'Info' tab is currently selected.</p> |

| 설명 | <ul style="list-style-type: none"> - 데이터 객체들을 정의하는 영역입니다. - 서버 통신을 위한 request, response 데이터와 화면에서 사용할 데이터를 정의합니다. - 데이터 객체의 타입은 DataMap, DataList, LinkedDataList, AliasDataMap, AliasDataList가 있습니다. | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------|--|-------|----------|------|----------|---|--------|----|------|---|--------|----|------|---|-----------|----|------|---|-------|-----|------|---|------------|-------|------|---|--------------|------|------|
| 소스 예시 | <pre> <w2:dataCollection baseNode="map"> <w2:dataList id="dlt_Member" baseNode="list" saveRemovedData="true" repeatNode="map" ev:ondataload="scwin.dlt_Member_ondataload" ev:oninsertrow="scwin.dlt_Member_oninsertrow" ev:onremoverow="scwin.dlt_Member_onremoverow"> <w2:columnInfo> <w2:column id="EMP_CD" name="코드" dataType="text"></w2:column> <w2:column id="EMP_NM" name="성명" dataType="text"></w2:column> <w2:column id="GENDER_CD" name="성별" dataType="text" defaultValue=""></w2:column> <w2:column id="EMAIL" name="이메일" dataType="text"></w2:column> <w2:column id="IMAGE_PATH" name="이미지경로" dataType="text"></w2:column> <w2:column id="CREATED_DATE" name="생성일자" dataType="text"></w2:column> </w2:columnInfo> </w2:dataList> <w2:dataProvider baseNode="map" id="dma_SearchParam"> <w2:keyInfo> <w2:key id="count" name="화면 향수" dataType="text"></w2:key> <w2:key id="page" name="페이지" dataType="text"></w2:key> <w2:key id="prePage" name="이전 페이지" dataType="text"></w2:key> <w2:key id="searchType" name="검색 타입" dataType="text"></w2:key> <w2:key id="searchParam" name="검색 파라미터" dataType="text"></w2:key> </w2:keyInfo> <w2:data use="true"> <page><![CDATA[1]]></page> </w2:data> </w2:dataProvider> </w2:dataCollection> </pre> | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 스튜디오 예시 (일부) | <p>Column Info</p>  <table border="1"> <thead> <tr> <th>No</th> <th>id</th> <th>name</th> <th>dataType</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>EMP_CD</td> <td>코드</td> <td>text</td> </tr> <tr> <td>2</td> <td>EMP_NM</td> <td>성명</td> <td>text</td> </tr> <tr> <td>3</td> <td>GENDER_CD</td> <td>성별</td> <td>text</td> </tr> <tr> <td>4</td> <td>EMAIL</td> <td>이메일</td> <td>text</td> </tr> <tr> <td>5</td> <td>IMAGE_PATH</td> <td>이미지경로</td> <td>text</td> </tr> <tr> <td>6</td> <td>CREATED_DATE</td> <td>생성일자</td> <td>text</td> </tr> </tbody> </table> | No | id | name | dataType | 1 | EMP_CD | 코드 | text | 2 | EMP_NM | 성명 | text | 3 | GENDER_CD | 성별 | text | 4 | EMAIL | 이메일 | text | 5 | IMAGE_PATH | 이미지경로 | text | 6 | CREATED_DATE | 생성일자 | text |
| No | id | name | dataType | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | EMP_CD | 코드 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | EMP_NM | 성명 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | GENDER_CD | 성별 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | EMAIL | 이메일 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | IMAGE_PATH | 이미지경로 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | CREATED_DATE | 생성일자 | text | | | | | | | | | | | | | | | | | | | | | | | | | | |

| | |
|-----------------|--|
| 설명 | <ul style="list-style-type: none"> - 서버 통신을 위한 인터페이스로 용도별로 여러 개를 정의할 수 있습니다. - 통신에 결과에 대해 비동기/동기 방식을 선택할 수 있습니다. - 통신 실행 전, 후에 실행할 함수(이벤트)를 정의할 수 있습니다. - request 데이터는 ref 속성에 정의된 데이터 객체의 정보로 생성합니다. - response 데이터는 target 속성에 정의된 데이터 객체의 정보를 참조하여 설정 합니다. |
| 소스 예시 | <pre> <xf:submission id="sbm_select" ref='data:json,[{"id":"dma_SearchParam"}]' target='data:json,["dlt_Member", "dma_SearchResult"]' action="/edu/selectMemberList.do" method="post" mediatype="application/json" encoding="UTF-8" customHandler="" mode="asynchronous" processMsg="처리중입니다." ev:submit="scwin.sbm_select_submit" ev:submitdone="scwin.sbm_select_submitdone"> </xf:submission> <xf:submission id="sbm_saveMember" ref='data:json,[{"id":"dlt_Member", "action": "modified"}, "dma_SearchParam"]' target='data:json,["dlt_Member", "dma_SearchResult"]' action="/edu/saveAndSelectMember.do" method="post" mediatype="application/json" encoding="UTF-8" mode="asynchronous" ev:submitdone="scwin.sbm_saveMember_submitdone"> </xf:submission></pre> |
| 스튜디오 예시 (일부) | <p>The screenshot shows the configuration dialog for a submission component. The fields are as follows:</p> <ul style="list-style-type: none"> ID*: sbm_select Reference: data:json,[{"id":"dma_SearchParam"}] Target: data:json,["dlt_Member", "dma_SearchResult"] Mode: asynchronous Method*: post Media Type: application/json Encoding: UTF-8 Instance: (empty) Replace: (empty) Error Handler: (empty) Custom Handler: (empty) Process Message: 처리중입니다. |

| 설명 | <ul style="list-style-type: none"> - Workflow들을 정의하는 영역입니다. - 여러 개의 Submission을 실행할 경우 실행할 Submission들을 하나의 Workflow에 등록하여 사용합니다. - Workflow에 정의한 Submission 실행 순서와 결과 처리 순서, 결과에 따른 이후 Submission의 실행 여부 등을 정의할 수 있습니다. - 조회(Select)용도의 통신에 사용할 것을 권장 합니다. | | | | | | | | | | | | | | | | | | | | | | | | | |
|-----------------|--|---------------------|--------|-------------|------|-------------|---|-----------|--|--|--|---|-----------|--|--|--|---|-----------|--|--|--|---|-----------|---------------------|--|--|
| 소스 예시 | <pre><w2:workflowCollection> <w2:workflow id="workflow1" resolveCallback="" rejectCallback=""> <w2:step type="submit" action="sbm_codeA"></w2:step> <w2:step type="submit" action="sbm_codeB"></w2:step> <w2:step type="submitDone" action="sbm_codeA"></w2:step> <w2:step type="submitDone" action="sbm_codeB" post="scwin.post_sbm_codeB"></w2:step> <w2:step type="submit" action="sbm_codeC" pre="scwin.pre_sbm_codeC"></w2:step> <w2:step type="submitDone" action="sbm_codeC"></w2:step> <w2:step type="submit" action="sbm_codeD"></w2:step> <w2:step type="submitDone" action="sbm_codeD"></w2:step> </w2:workflow> </w2:workflowCollection></pre> | | | | | | | | | | | | | | | | | | | | | | | | | |
| 스튜디오 예시 (일부) | <table border="1"> <thead> <tr> <th>Order</th> <th>Action</th> <th>Pre</th> <th>Post</th> <th>Description</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>sbm_codeB</td> <td></td> <td></td> <td></td> </tr> <tr> <td>2</td> <td>sbm_codeA</td> <td></td> <td></td> <td></td> </tr> <tr> <td>3</td> <td>sbm_codeB</td> <td></td> <td></td> <td></td> </tr> <tr> <td>4</td> <td>sbm_codeC</td> <td>scwin.pre_sbm_codeC</td> <td></td> <td></td> </tr> </tbody> </table> <p>Type : submit Action : sbm_codeC Description :</p> <p>Pre : scwin.pre_sbm_codeC Post :</p> <p>선행 Step 선택 Serial Step Parallel Step</p> <p>Done (1) Done (2) sbm_codeA (1) Done (3) sbm_codeB (1) Done (4) Done (5) Done (6) Done (7) sbm_codeC (4) Done (7) Done (8) Done (9) sbm_codeD (6)</p> | Order | Action | Pre | Post | Description | 1 | sbm_codeB | | | | 2 | sbm_codeA | | | | 3 | sbm_codeB | | | | 4 | sbm_codeC | scwin.pre_sbm_codeC | | |
| Order | Action | Pre | Post | Description | | | | | | | | | | | | | | | | | | | | | | |
| 1 | sbm_codeB | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | sbm_codeA | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | sbm_codeB | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | sbm_codeC | scwin.pre_sbm_codeC | | | | | | | | | | | | | | | | | | | | | | | | |

| 구 분 | 설 명 |
|-----------|--|
| Browser | <p>Browser에서 제공하는 객체.</p> <ul style="list-style-type: none"> - document, window 등 <p>ex) window.open('http://');</p> |
| 사용자 정의 | <p>사용자(개발자)가 script블록 또는 js파일에 정의한 변수, 함수, 객체들.</p> <p>ex)</p> <pre>var userName = "WebSquare"; function.getUserName(){ return userName; } var info = { name : "WebSquare", city : "Seoul" }; var common = { isNumber: function(tmpVal){ return Number(tmpVal); } };</pre> |
| WebSquare | <p>WebSquare에서 제공하는 객체.</p> <p>전역 공간에 정의되기 때문에 Browser 객체명, 사용자 정의 객체명과 중복되면 안 됩니다.</p> <ul style="list-style-type: none"> - Util : \$p, WebSquare <ul style="list-style-type: none"> ex) var curDate = \$p.getCurrentServerDate('yyyy-MM-dd'); //WAS에서 오늘 날짜를 꺼내 옵니다. - UI 객체: WebSquare UI 컴포넌트의 ID <ul style="list-style-type: none"> ex) inputbox1.setValue("WebSquare"); //inputbox1의 ID를 가진 컴포넌트의 value를 설정 - Data 객체 : DataCollection에 정의된 Data객체의 ID <ul style="list-style-type: none"> ex) curRowDataObj = dataList1.getRowJSON(0); //0번째 행의 데이터를 꺼내 옵니다. |

- **브라우저의 지원 범위 확인**

- ActiveX 모듈의 확인 (ActiveX는 IE만 지원)
- 지원하려는 하위 IE의 버전 선정
- HTML5의 MS 공식 지원은 IE10 부터 지원
- Safari 브라우저는 Mac OS에서 확인 권장

- **화면 레이아웃 구성 범위 확인**

- PC를 기준으로 모니터 해상도까지 지원
 - 가변형 화면 레이아웃의 경우 최소 IE8 이상부터 유연하게 개발 가능
- 모바일까지 지원
 - CSS의 지식이 고급 이상

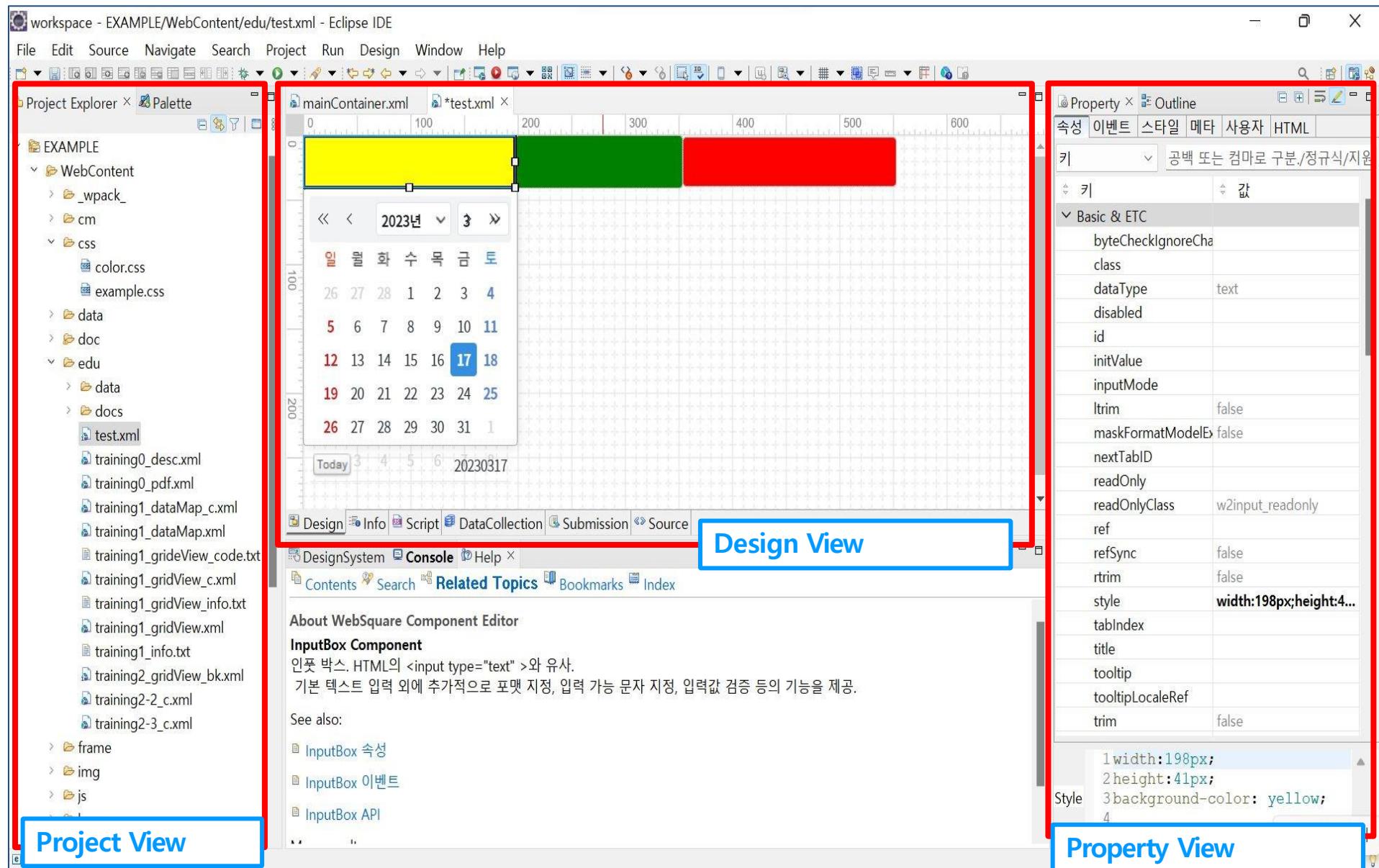
- **대용량 데이터 화면 확인**

- **다국어 지원 여부 확인**

- **웹접근성 지원 여부 확인**

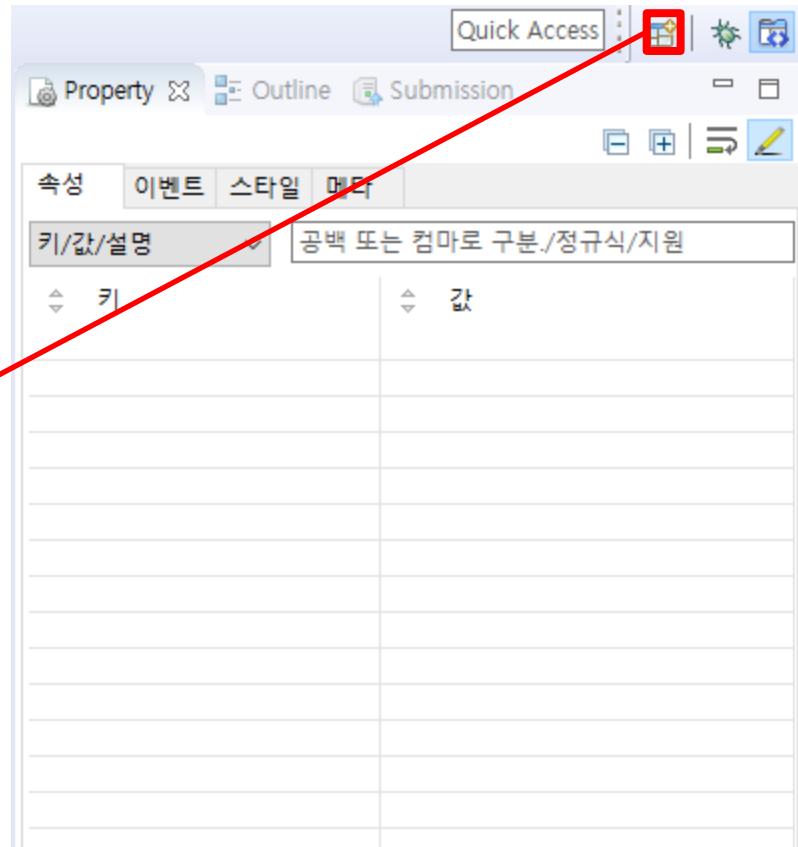
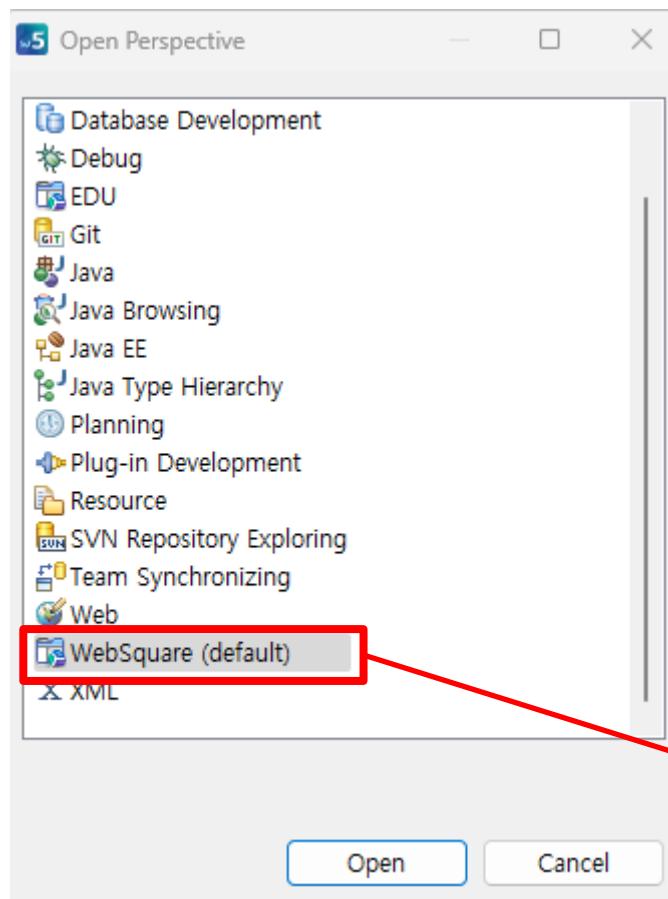
| WebSquare5 Studio

- WebSquare5 스튜디오 View
- Perspective View 사용하기
- WebSquare5 프로젝트 및 파일생성
- WebSquare5 브라우저 테스트
- WebSquare5 컴포넌트
- WebSquare5 Change Draw Mode
- WebSquare5 CSS 적용
- WebSquare5 Design System
- WebSquare5 Component Viewer



▪ Eclipse Perspective view

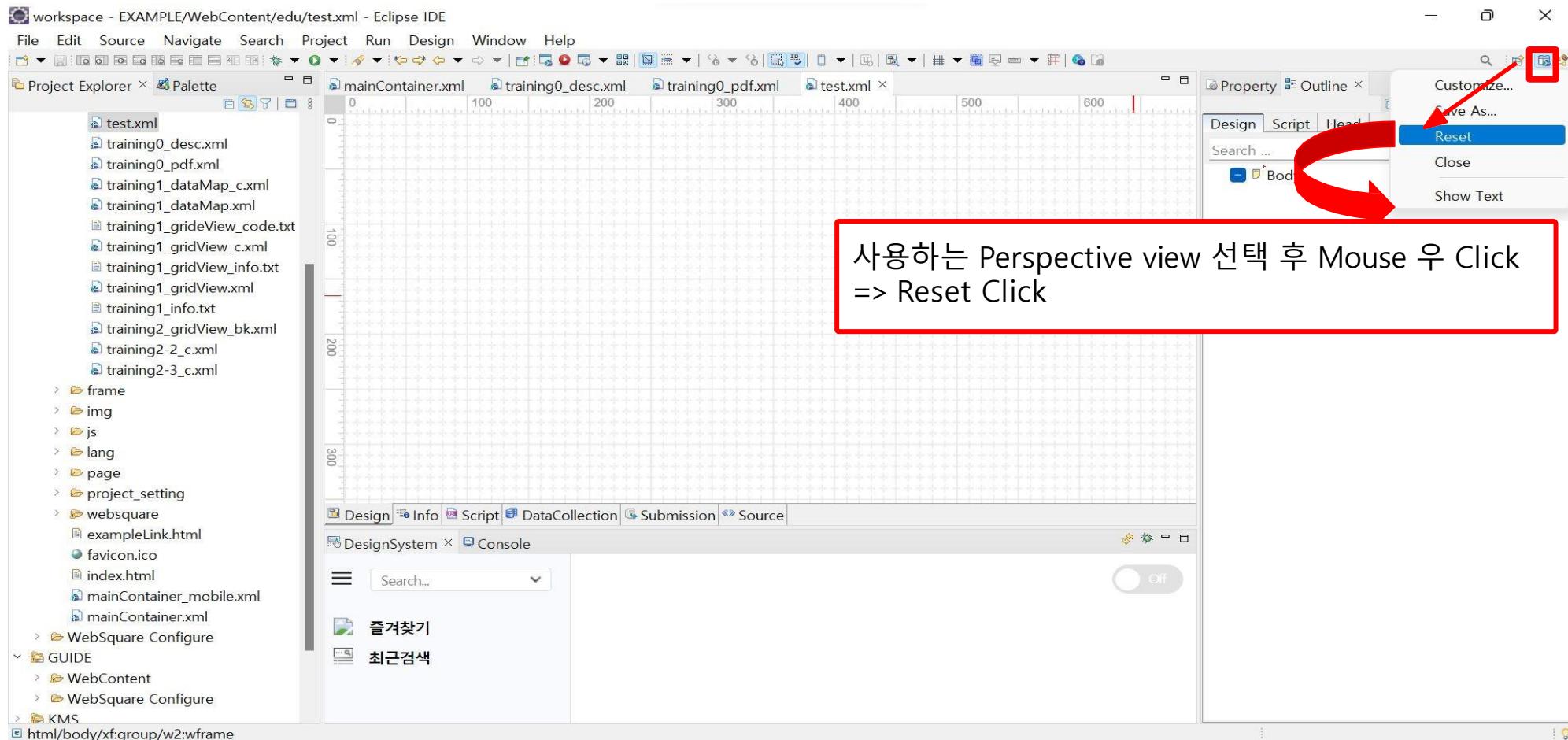
Eclipse 우측 상단의 Open Perspective 확인



WebSquare5 plugin이 Eclipse에 정상 설치되어 있으면
개발자 기본(default) – **WebSquare Developer**
Perspective view 가 보여집니다.

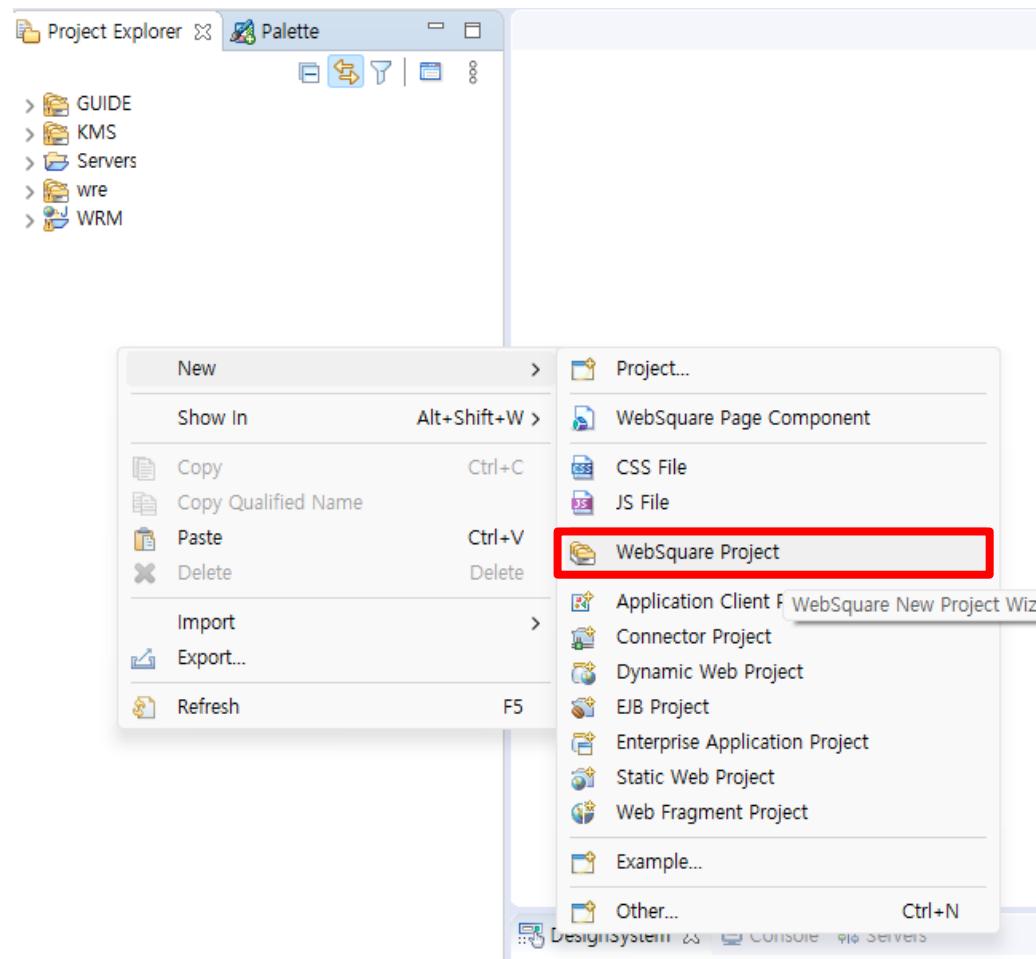
▪ Eclipse Perspective view Reset 기능

- 선택된 Perspective view에서 Mouse 우 click 후 Reset click
: 최초 구성한 Perspective 형태로 되돌아 가는 기능입니다.

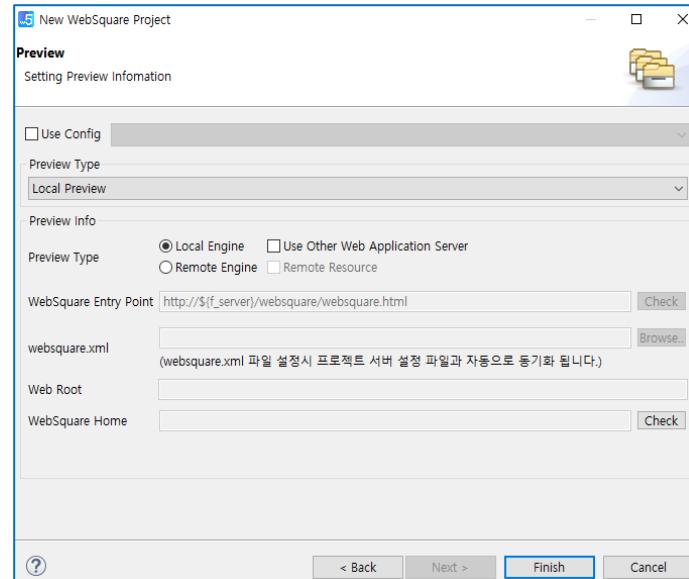
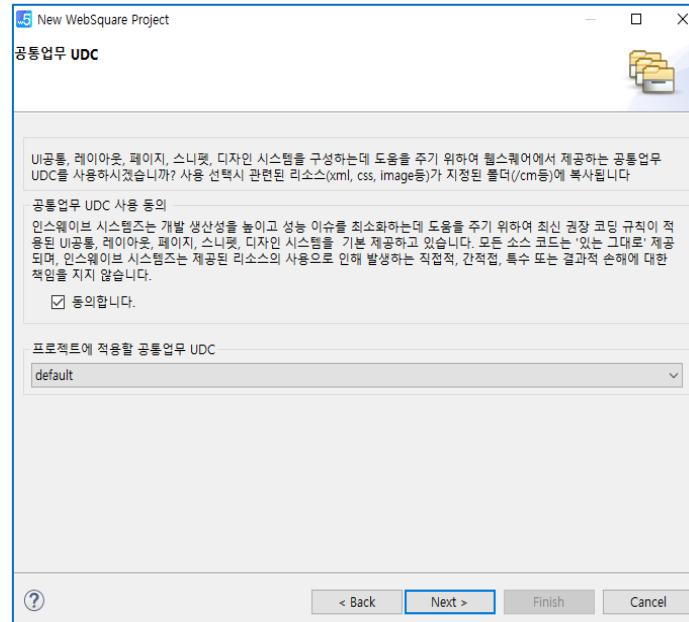
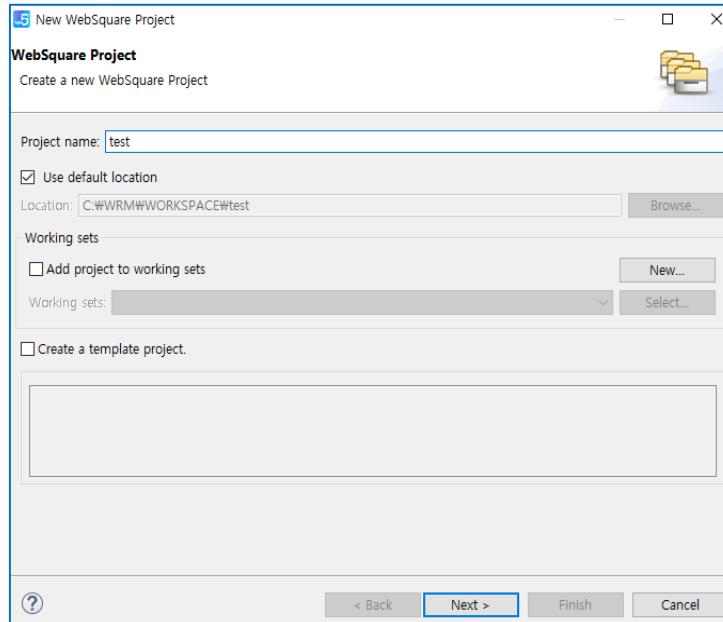


▪ WebSquare Project 생성

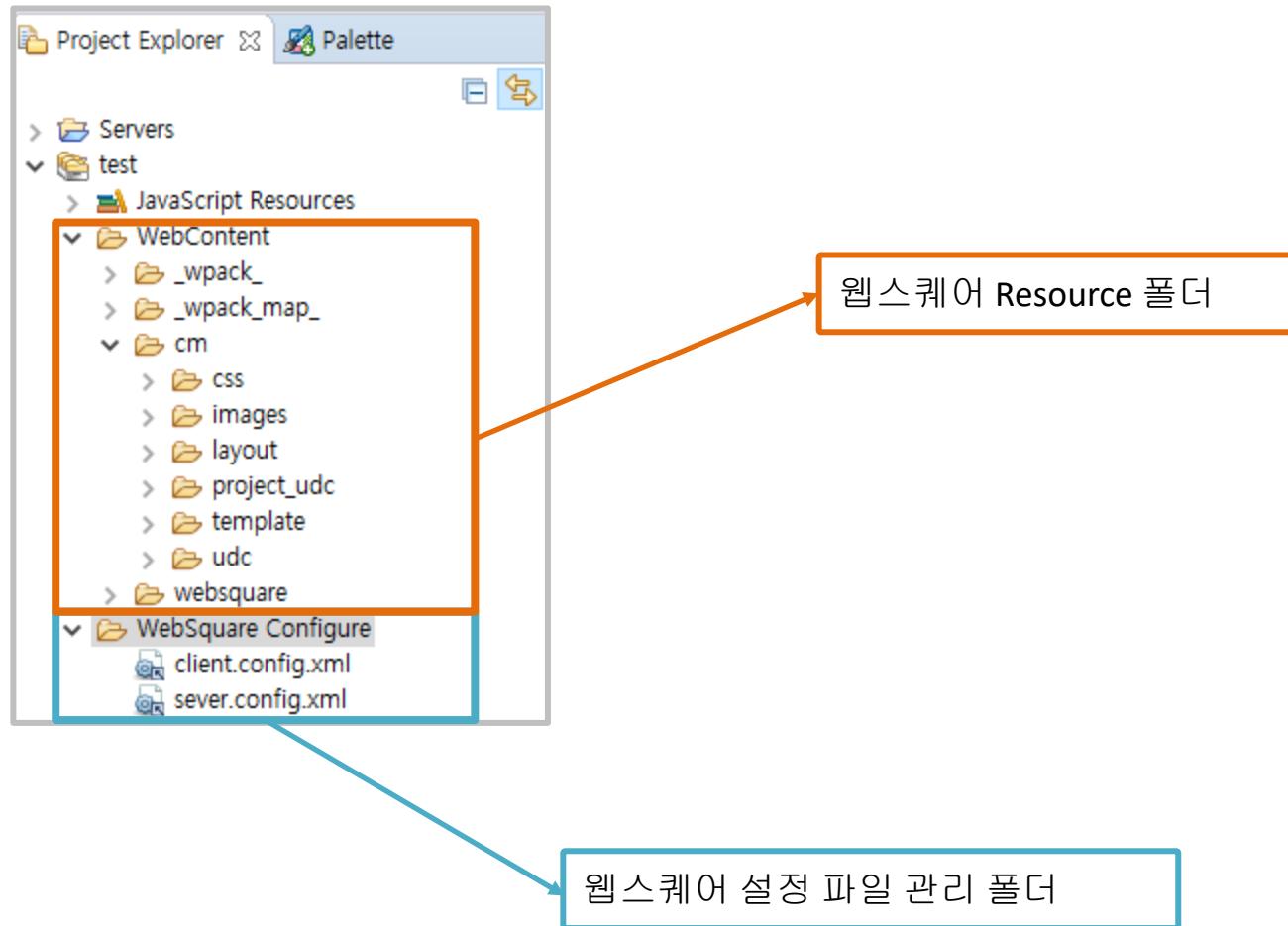
Project Explorer => new => **WebSquare Project** 선택



▪ WebSquare Project 생성

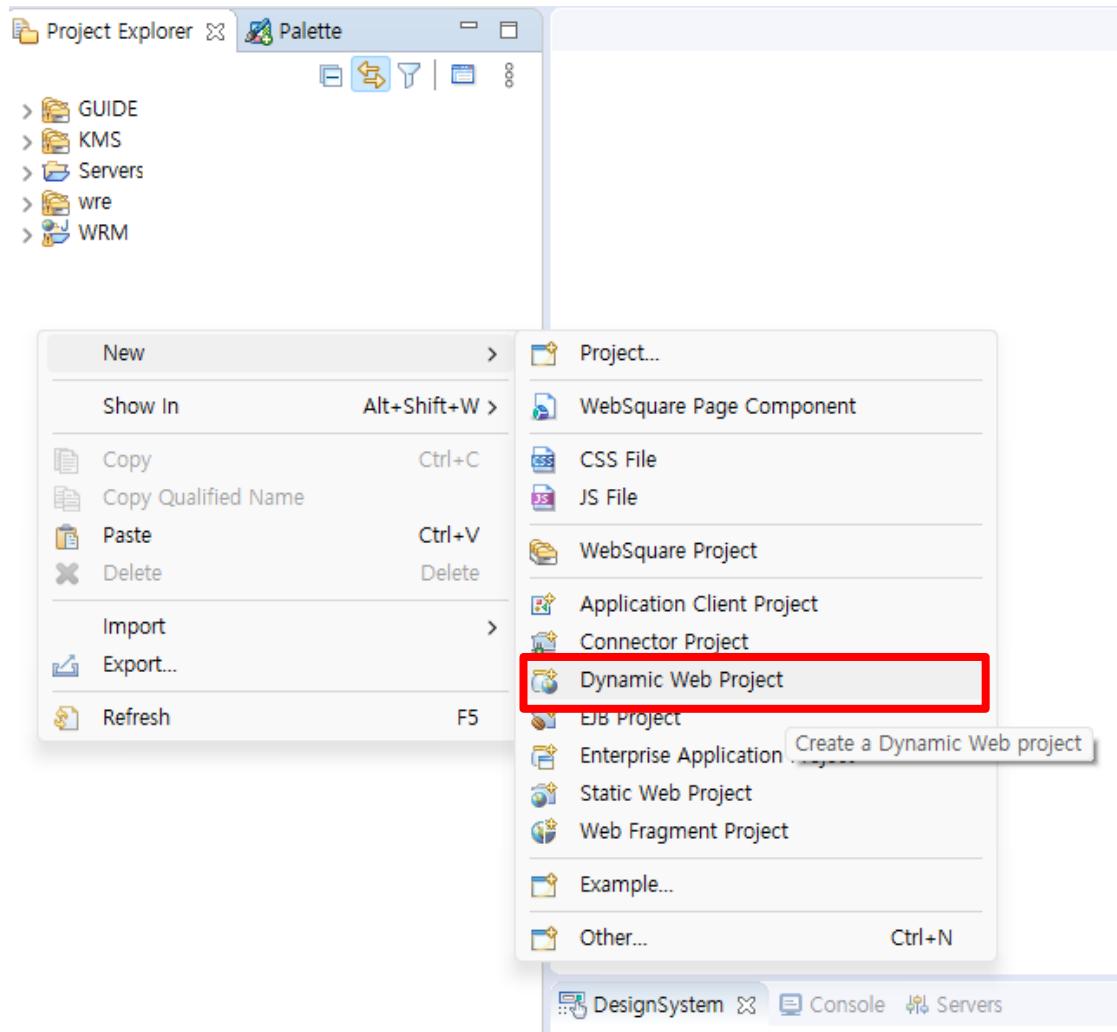


▪ WebSquare Project 생성

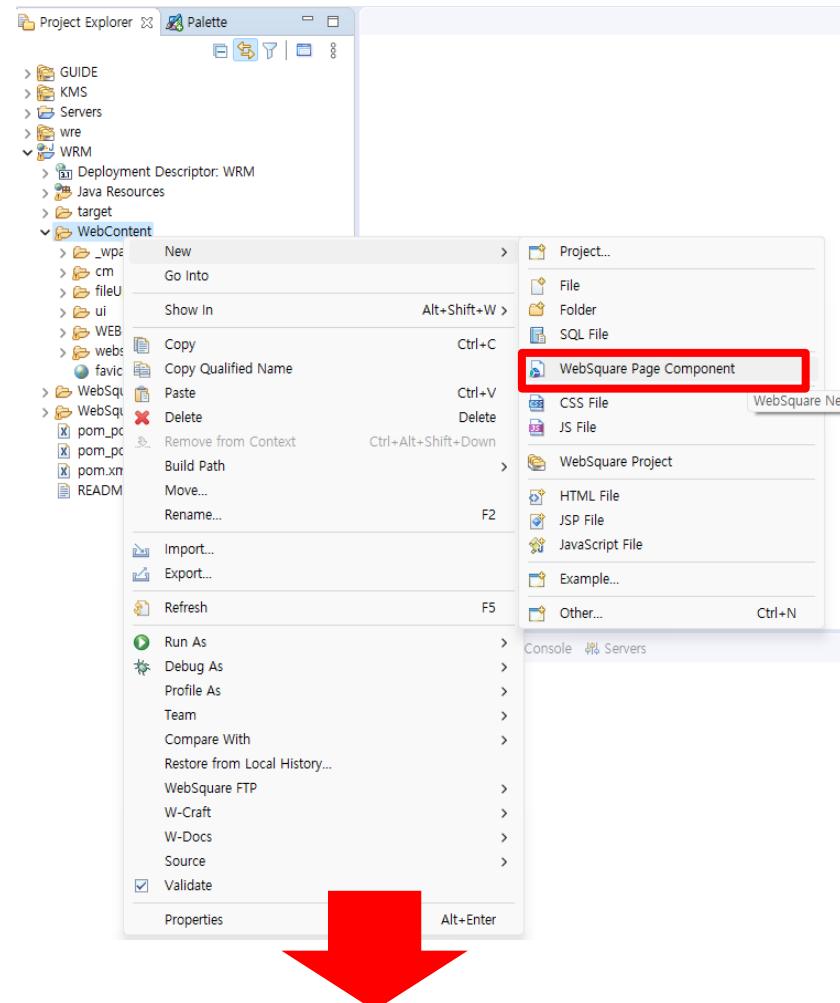


- Dynamic Web Project 생성

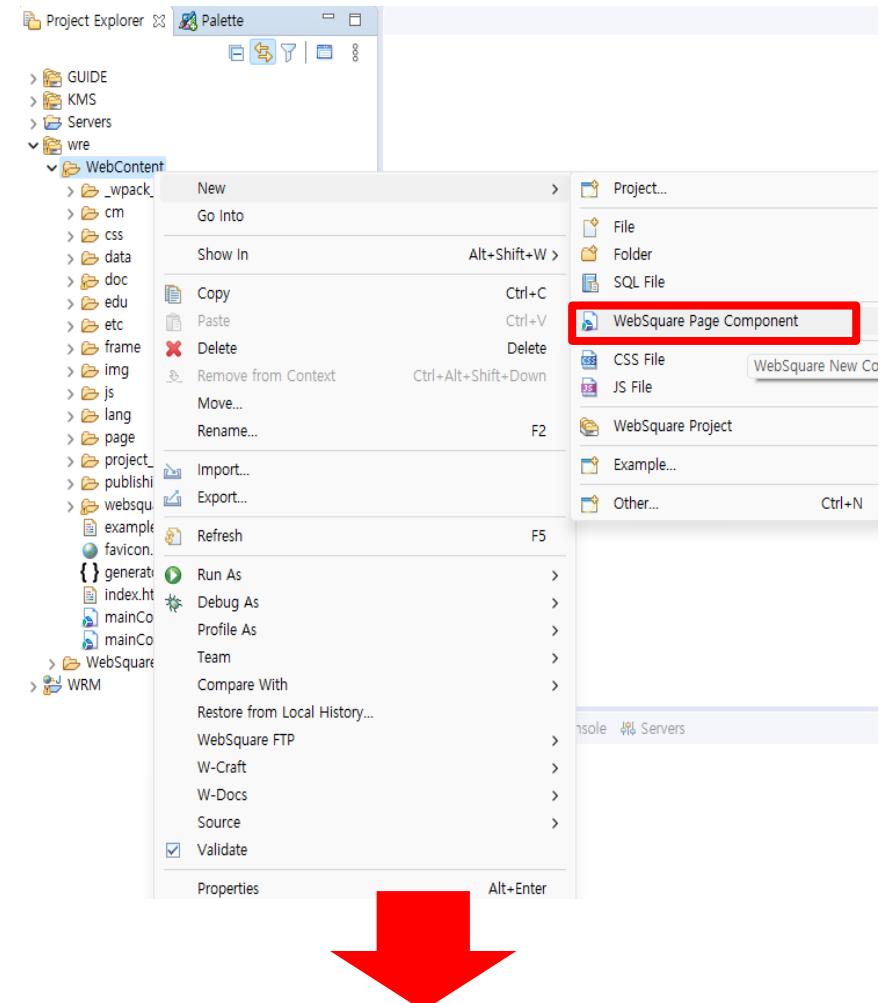
- Project Explorer => new => **Dynamic Web Project 선택**



▪ WebSquare File 생성



Dynamic Web Project 인 경우
생성된 Project에서 WebRoot 이하 경로 선택 후 우클릭
=> New => WebSquare Page Component 선택



WebSquare Project 인 경우
프로젝트 이하 경로 선택 후 우클릭
=> new => WebSquare Page Component 선택

▪ 설정 파일 – config.xml

client.config.xml X

overview

W-Pack

| | | |
|-----------------|---------|---|
| enable : | true | W-Pack을 활성화할지 설정합니다. 값이 지정되지 않은 경우 true로 동작합니다. |
| contextRoot : | / | 프로젝트의 contextRoot를 설정합니다. 값이 지정되지 않은 경우 root로 동작합니다. |
| srcExtension : | xml | W-Pack 빌드 대상이 될 화면 파일의 확장자를 설정합니다. 값이 지정되지 않은 경우 "xml"로 동작합니다. |
| destExtension : | js | W-Pack 빌드 결과물이 될 파일의 확장자를 설정합니다. 값이 지정되지 않은 경우 "js"로 동작합니다. |
| destRoot : | _wpack_ | W-Pack 빌드 결과물이 생성될 폴더이름을 지정합니다. 값이 지정되지 않은 경우 "_wpack_"를 사용합니다. |

strictMode

| | | |
|--------------------|------|--|
| strictMode : | true | wframe를 사용하는 경우 자신의 프레임 영역을 벗어나는 자원들을 암시적으로 접근하는 것을 차단한 설정입니다. true 설정 시 모든 화면이 눈에 보이지 않는 wframe으로 감싸지게 된다. wframe 내부 화면과 wframe 외부 화면의 코딩 방식을 통일하기 위해 도입한 설정이다. 즉 wframe 안에 들어가는 page와 일반 화면의 없애고 모든 화면이 page가 되도록 하는 설정입니다. SP3부터는 "true"로 설정할 것을 권장합니다. |
| scope variable : | mf | strict mode에서 사용할 TOP 스코프의 scope variable 이름입니다. 기본값은 mf입니다. |
| scriptPrecedence : | true | initScript, onpageload, postScript의 이벤트 순서를 initScript->onpageload->postScript 순서로 실행하는 옵션입니다. SP3부터는 "true"로 설정할 것을 권장합니다. |

page(SP3)/SPA(SP2)

| | | |
|------------|-------|--|
| variable : | scwin | SP2 : spa로 페이지 이동 시 자동으로 삭제할 객체를 string형태로 등록합니다. SP3 이후 : 페이지에서 사용할 변수명을 정의합니다. 기본값은 scwin입니다. |
| clone : | | SP3 : js>script>external에 설정된 js 파일의 공통 모듈 중 page scope를 적용할 객체를 지정합니다. |
| cna : | | 페이지 이동 시 뷰라우저를 개시하지 않고 body 여백만 새로 그려서 마지막 페이지가 실제로 이동되는 |

overview Common dev js UI/UX comm data resource notRecommended 컴포넌트(config 전용) deprecated Source

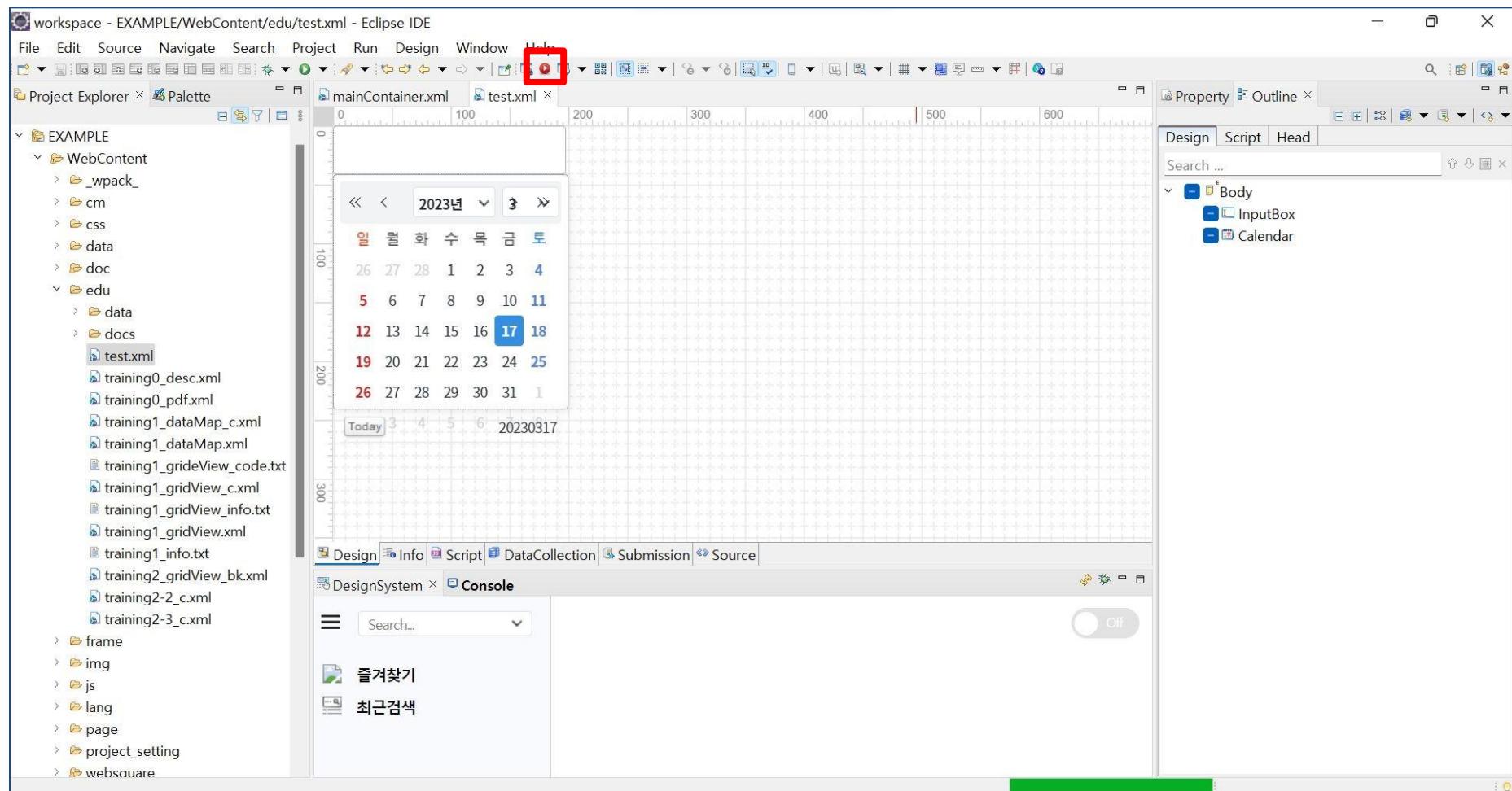
▪ 설정 파일 – websquare.xml

The screenshot shows the 'sever.config.xml' configuration interface in the WebSquare5 environment. The main title bar says 'sever.config.xml'. Below it, the title 'overview' is displayed. The configuration is organized into three main sections: 'encoding', 'engine', and 'date'. The 'encoding' section contains two fields: 'default_encoding' set to 'UTF-8' (described as the basic encoding used within the engine) and 'encoding' set to 'ISO8859-1' (described as setting the encoding for parameters sent via POST form). The 'engine' section contains three fields: 'engineType' set to '6' (described as the engine's normalization and compression settings), 'etag' set to 'support' (described as using 'support' by default), and 'wqDefiner' (an empty field described as calling the doHandle method of a specified class when a wqDefiner.wq request is made). The 'date' section contains one field: 'encoding' set to 'UTF-8' (described as handling Korean date patterns like '2016년 11월 11일' based on the server's encoding settings). At the bottom, there are navigation links: 'overview', 'upload', 'excel', 'csv', 'i18n', and 'Source'.

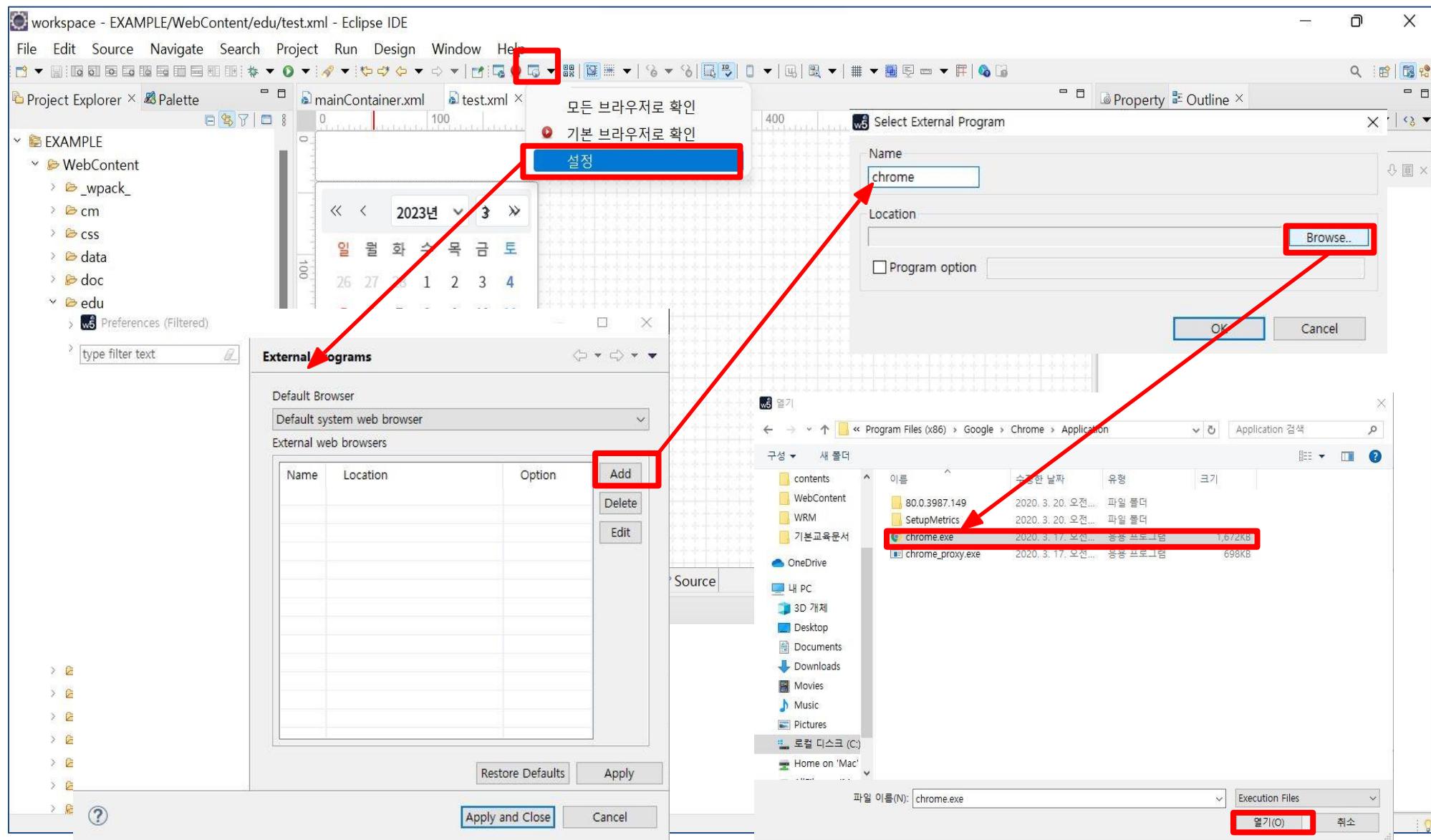
| Section | Setting | Description |
|----------|------------------|--|
| encoding | default_encoding | UTF-8 엔진 내부에서 사용되는 기본 인코딩 값입니다. |
| | encoding | ISO8859-1 post 태그를 통하여 form을 통해 post 방식으로 전달되는 파라메터의 인코딩 값을 설정합니다. |
| engine | engineType | 6 웹스퀘어 엔진의 난독화 및 압축 수준을 설정하는데 사용됩니다. |
| | etag | support etag를 사용하지 유무, 기본값은 "support" |
| | wqDefiner | 사용자 정의 wq의 구현체 http요청시 wqDefiner.wq를 요청하게 되면 사용자가 정의한 class의 doHandle를 호출한다. |
| date | encoding | UTF-8 pattern:yyyyMMdd 인 경우 특이사항이 없지만, "2016년 11월 11일"와 같이 한글형식의 날짜를 반 환 하는 경우 서버 encoding설정에 따라 한글깨지는 현상이 있을수 있으며 encoding설정을 변경하 여 인코딩을 맞춥니다. |

▪ 테스트 브라우저 실행

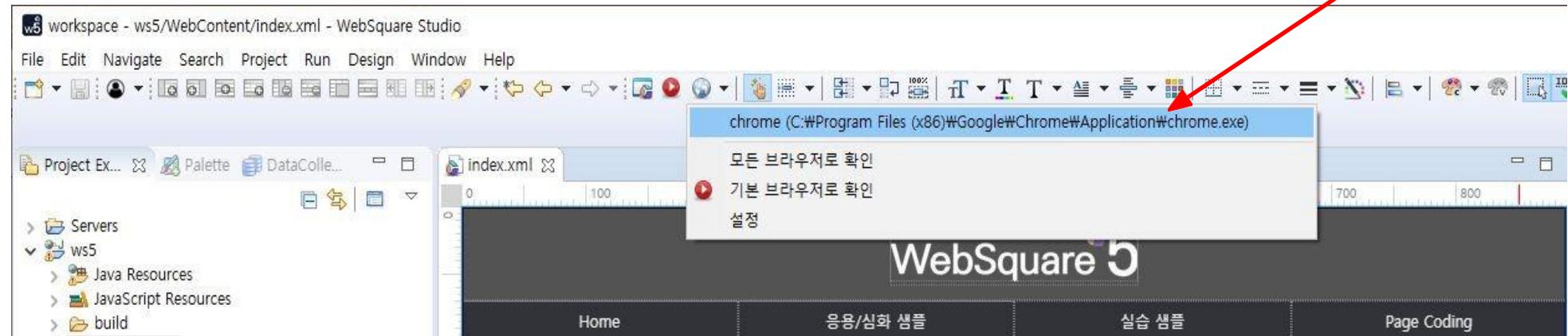
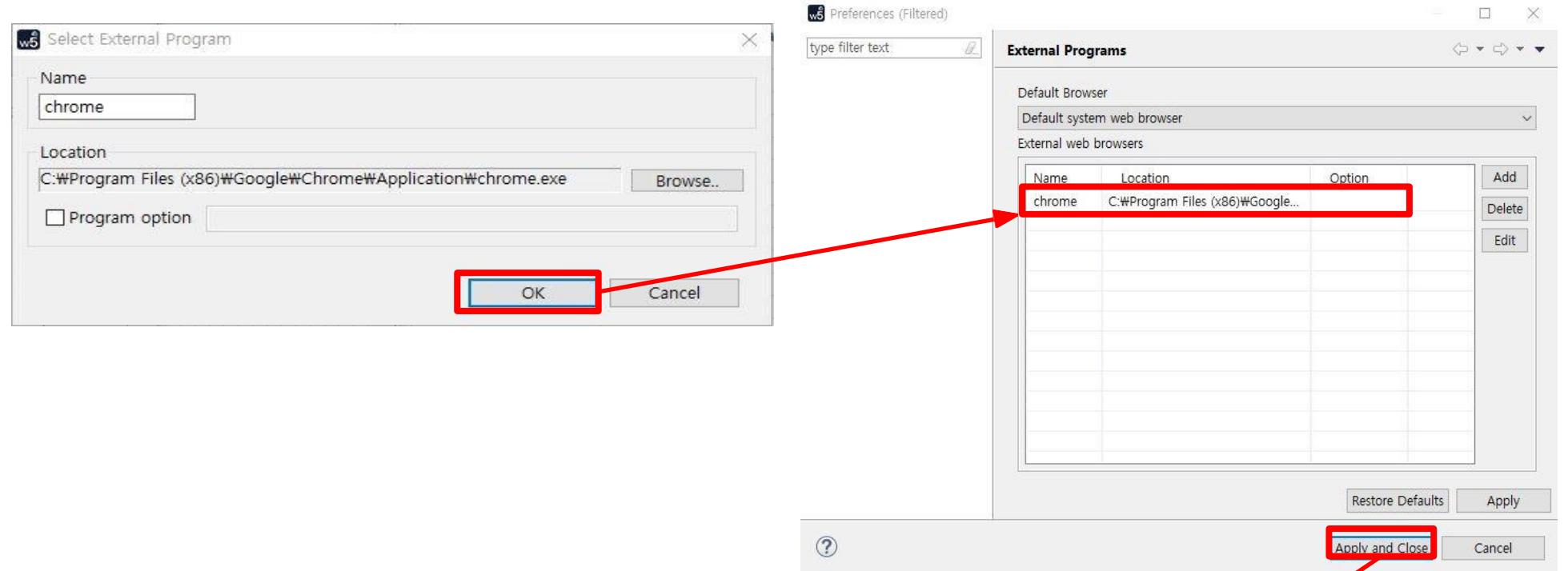
- 상단의 붉은색 실행 버튼 클릭(혹은 단축키 F7 클릭)



▪ 테스트 브라우저 등록 및 변경

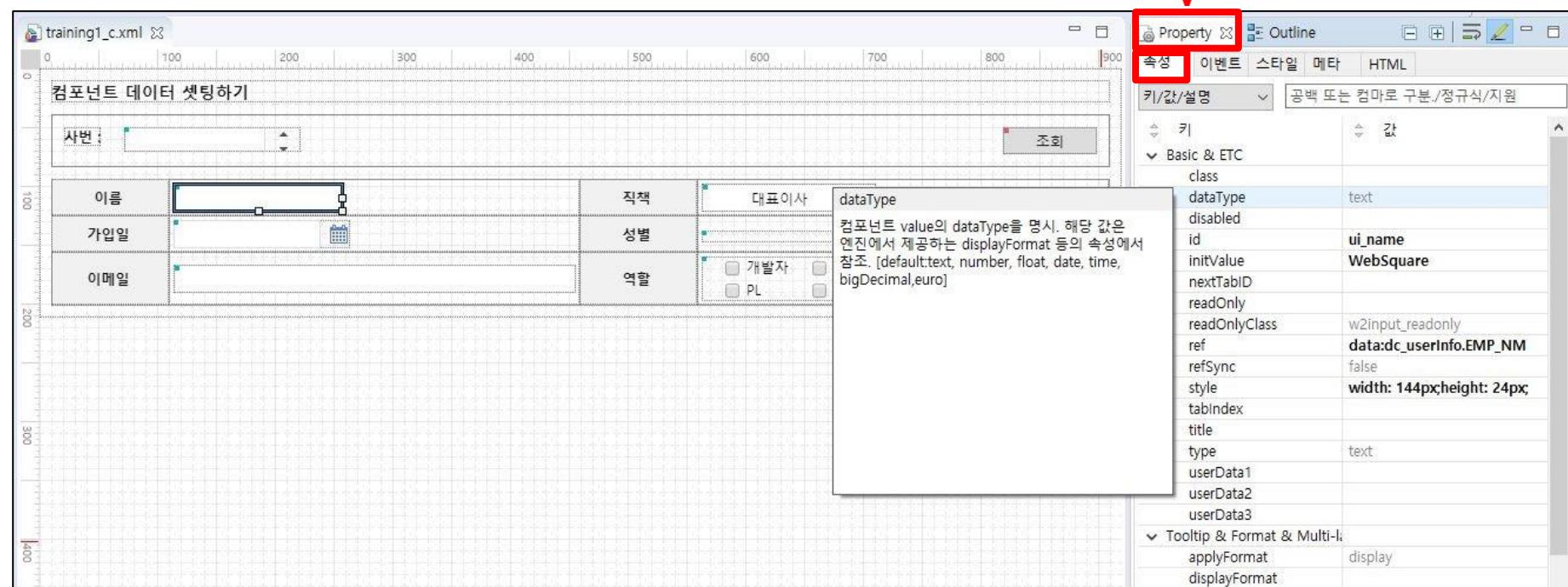


▪ 테스트 브라우저 등록 및 변경



▪ 속성

- Property view의 '속성'을 이용한다.
- 컴포넌트별 속성을 통하여 기능을 추가 합니다.
- 각 속성에 Mouse Over 하면, Tooltip 기능을 통해 상세 기능을 알 수 있습니다.



▪ 이벤트

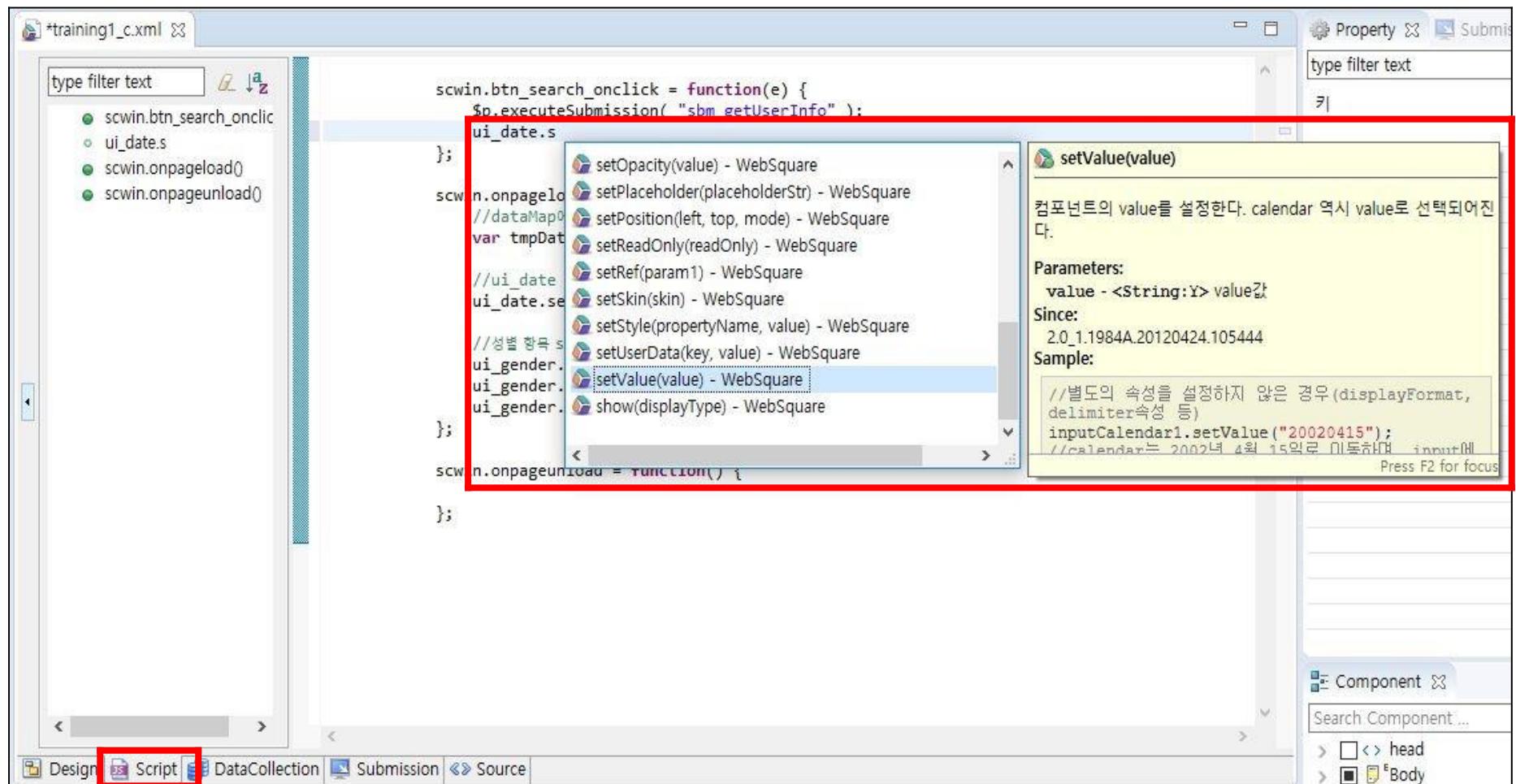
- Property의 '이벤트' 부분 혹은 컴포넌트 클릭 후 마우스 우 클릭에서 '이벤트' 사용
- Mouse over 시 속성에서 제공되던 tooltip 기능을 제공 합니다.
(상세기능은 Help 메뉴를 통해 확인)

The screenshot shows the WebSquare5 IDE interface. On the left, there is a design view of a form with various input fields and a button labeled '조회'. Below the design view, there are tabs for Design, Script, DataCollection, Submission, and Source. Under the Design tab, there are links for Help, Servers, Console, Snippets, Contents, Search, Related Topics, Bookmarks, and Index.

On the right, there is a Properties panel with several tabs: 속성 (Properties), 이벤트 (Events) (which is highlighted with a red box), 스타일 (Style), 메타 (Meta), and HTML. The '이벤트' tab is currently active, showing a list of events such as onblur(e), onchange, onclick(e), ondblclick(e), oneditkeyup(info), onfocus(e), oninputinvalid(info), onkeydown(e), onkeypress(e), onkeyup(e), onmousedown(e), onmousemove(e), onmouseout(e), onmouseover(e), onmouseup(e), ontooltiphide, and ontooltipshow. The 'onviewchange(info)' event is selected and highlighted with a blue box. A tooltip for this event is displayed below the list, stating: '키보드 또는 마우스 조작을 통해 값이 변경된 경우 발생. 스크립트를 통해 값이 변경된 경우에는 발생하지 않음. (native onchange 이벤트와 유사함.)'

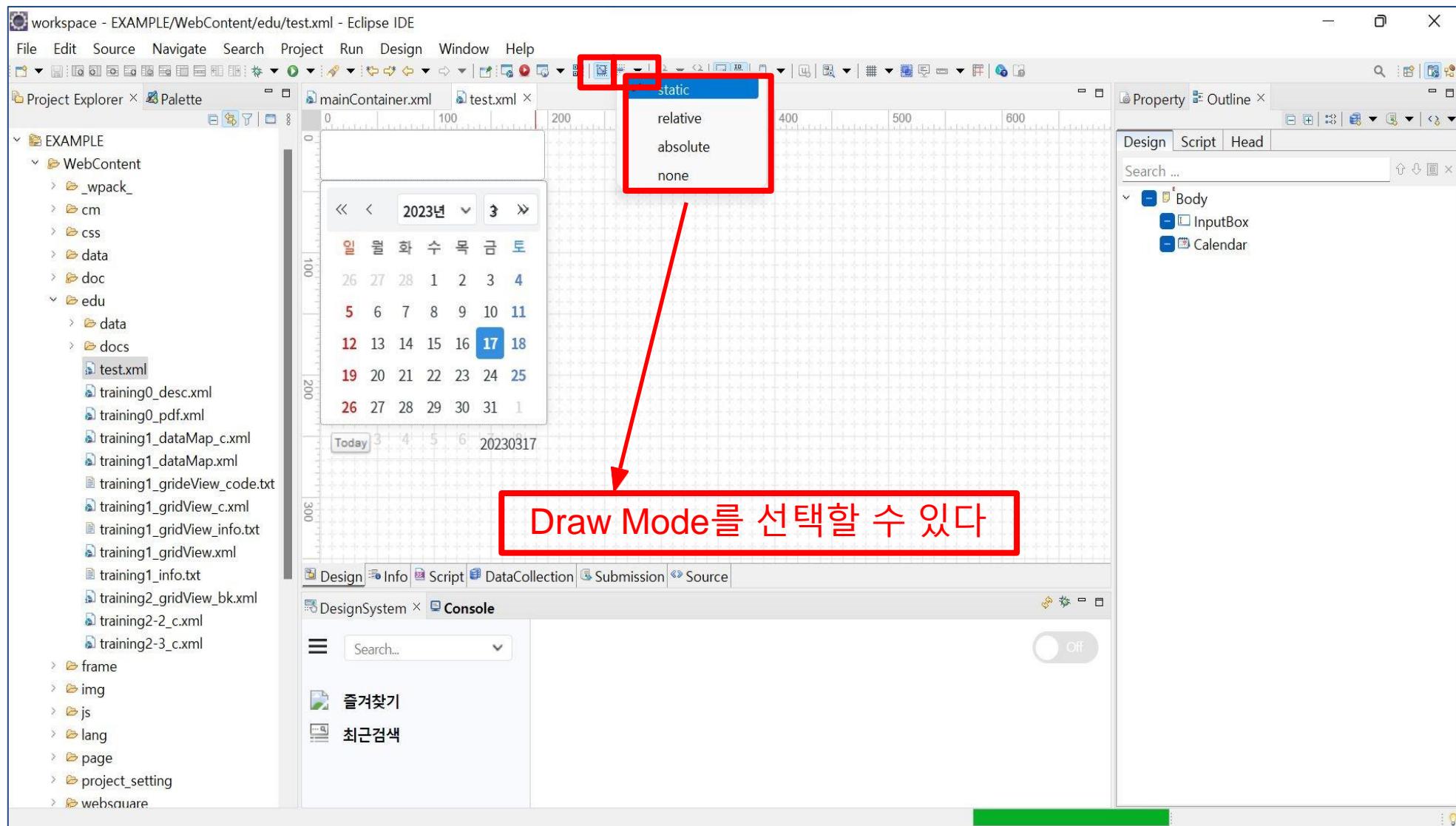
▪ API

- 컴포넌트의 id 기준으로 접근하여 사용하는 웹스퀘어 제공 API
- Script 탭에서 'ctrl+space' 누르면 선택된 객체에서 사용 할 수 있는 API 목록을 확인할 수 있습니다.



▪ Change Draw Mode

- 화면 구성 방법을 변경할 수 있는 기능
- **static, relative, absolute** Mode 제공 합니다.



▪ Inline CSS

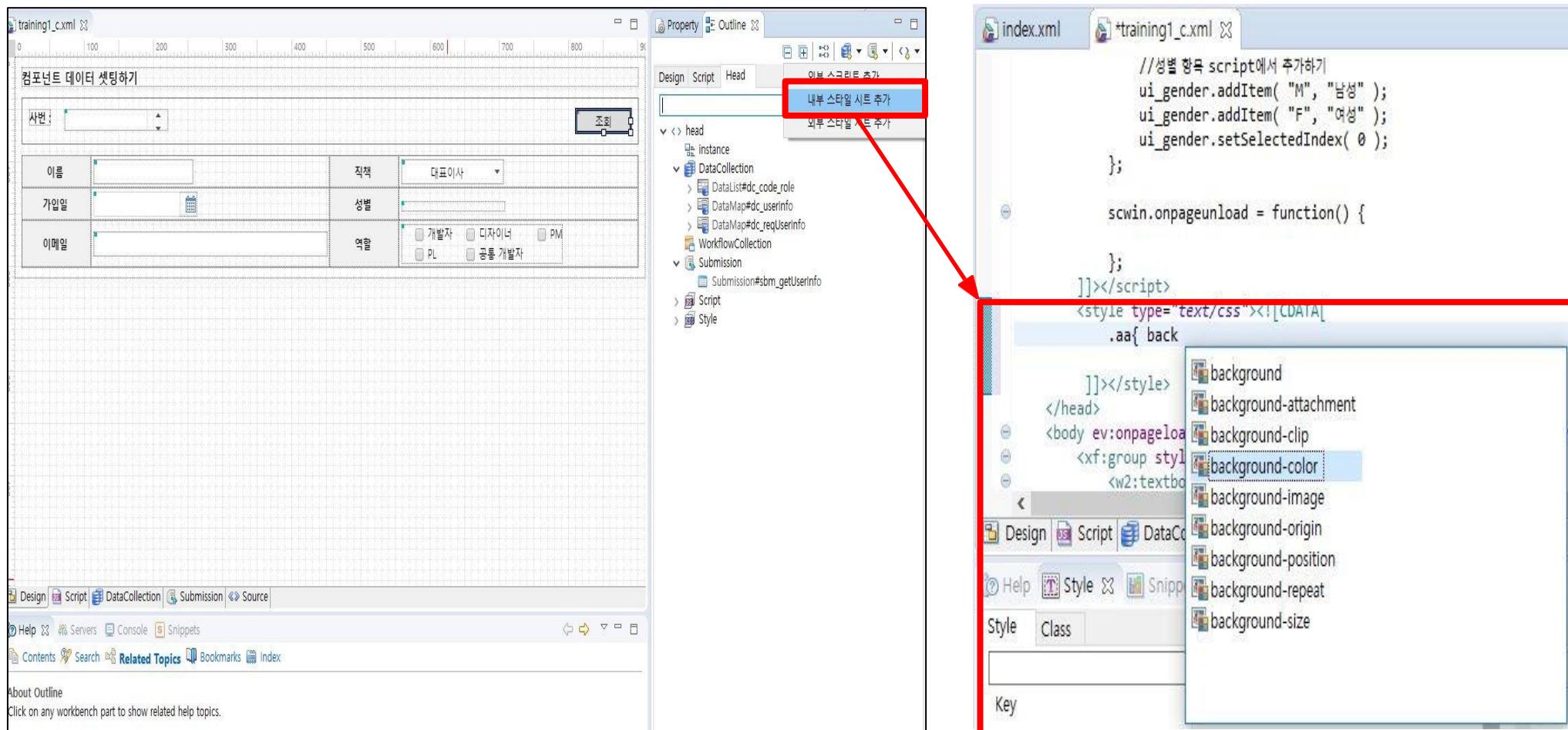
- 컴포넌트에 직접 적용하는 CSS
- Property '속성' 하단에서 적용할 수 있습니다.

The screenshot shows the WebSquare5 IDE interface. On the left is the 'Design' tab, displaying a form with several input fields and dropdown menus. On the right is the 'Property' panel, which lists various properties for the selected component. A red box highlights the 'Style' section at the bottom of the property list, which contains the inline CSS rule: `width: 144px; height: 24px; speech-rate:0;`.

| 속성 | 값 |
|-------------------------------|---|
| key | |
| dataType | text |
| disabled | |
| id | ui_name |
| initValue | WebSquare |
| nextTabID | |
| readOnly | w2input_READONLY |
| readOnlyClass | <code>data:dc.userInfo.EMP_NM</code> |
| ref | false |
| refSync | |
| style | <code>width: 144px; height: 24px; speech-rate:0;</code> |
| tabIndex | |
| title | |
| type | text |
| userData1 | |
| userData2 | |
| userData3 | |
| Tooltip & Format & Multi-lang | |
| applyFormat | display |
| displayFormat | |
| escape | false |
| euroMask | |
| imeMode | |
| ioFormat | |
| maskFormat | |
| maskFormatModelExcludeChar | |
| numberMask | |
| placeholder | |
| rupeeMask | |
| showPlaceHolderOnReadOnly | false |
| tengeMask | |
| toolTip | |
| useMonthYearFormat | false |
| Action | |
| autoFocus | false |
| Style | <code>width: 144px; height: 24px; speech-rate:0;</code> |

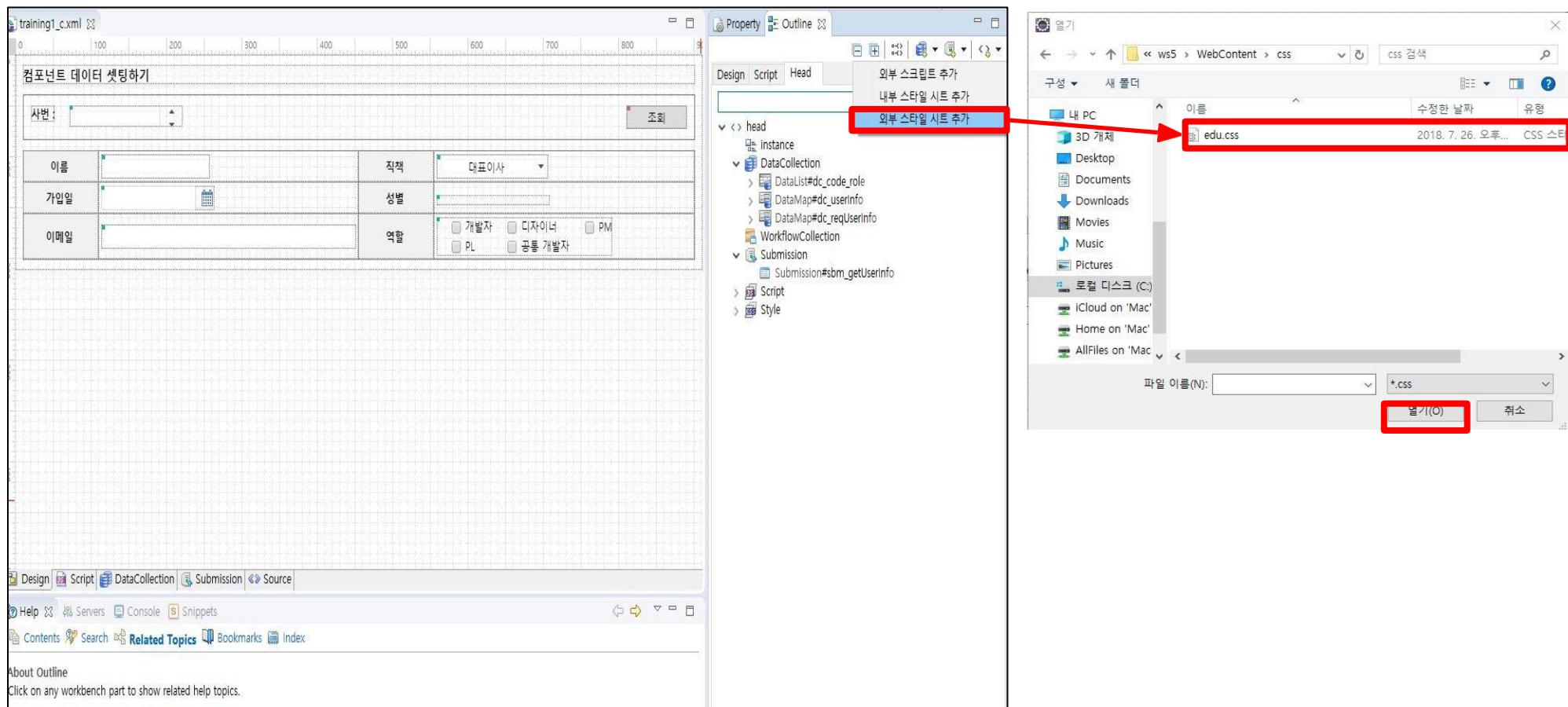
▪ Internal CSS

- 파일에서 적용하는 CSS
- outline의 ‘내부 스타일시트 추가’를 클릭하면, 소스 탭으로 이동하여 파일 내에 CSS를 정의하여 사용할 수 있습니다. (Code Assist 기능 사용 가능)



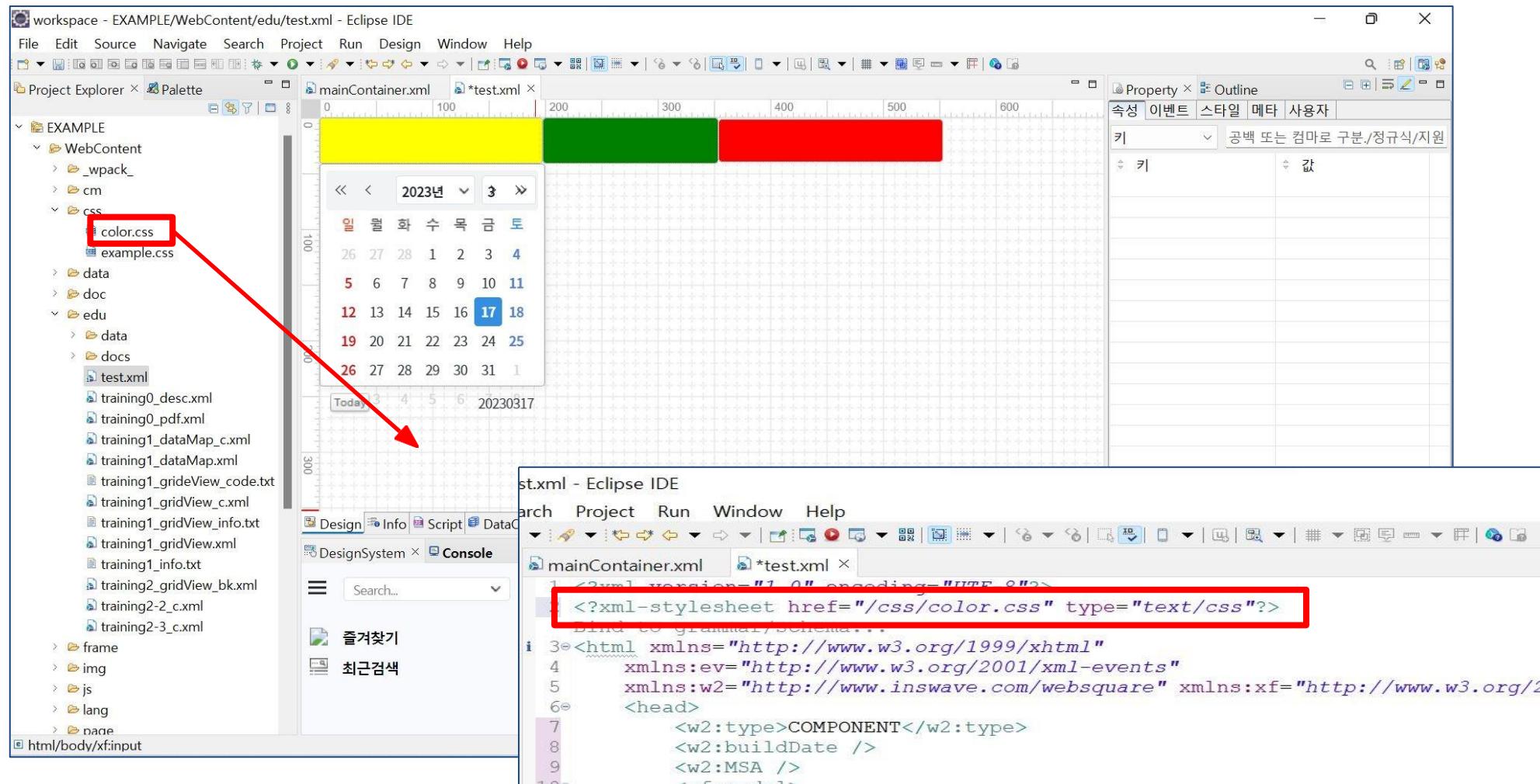
▪ External CSS

- Property의 '외부 스타일시트 추가' 클릭으로 적용 합니다.
(사용할 css file을 선택하여 적용 합니다.)



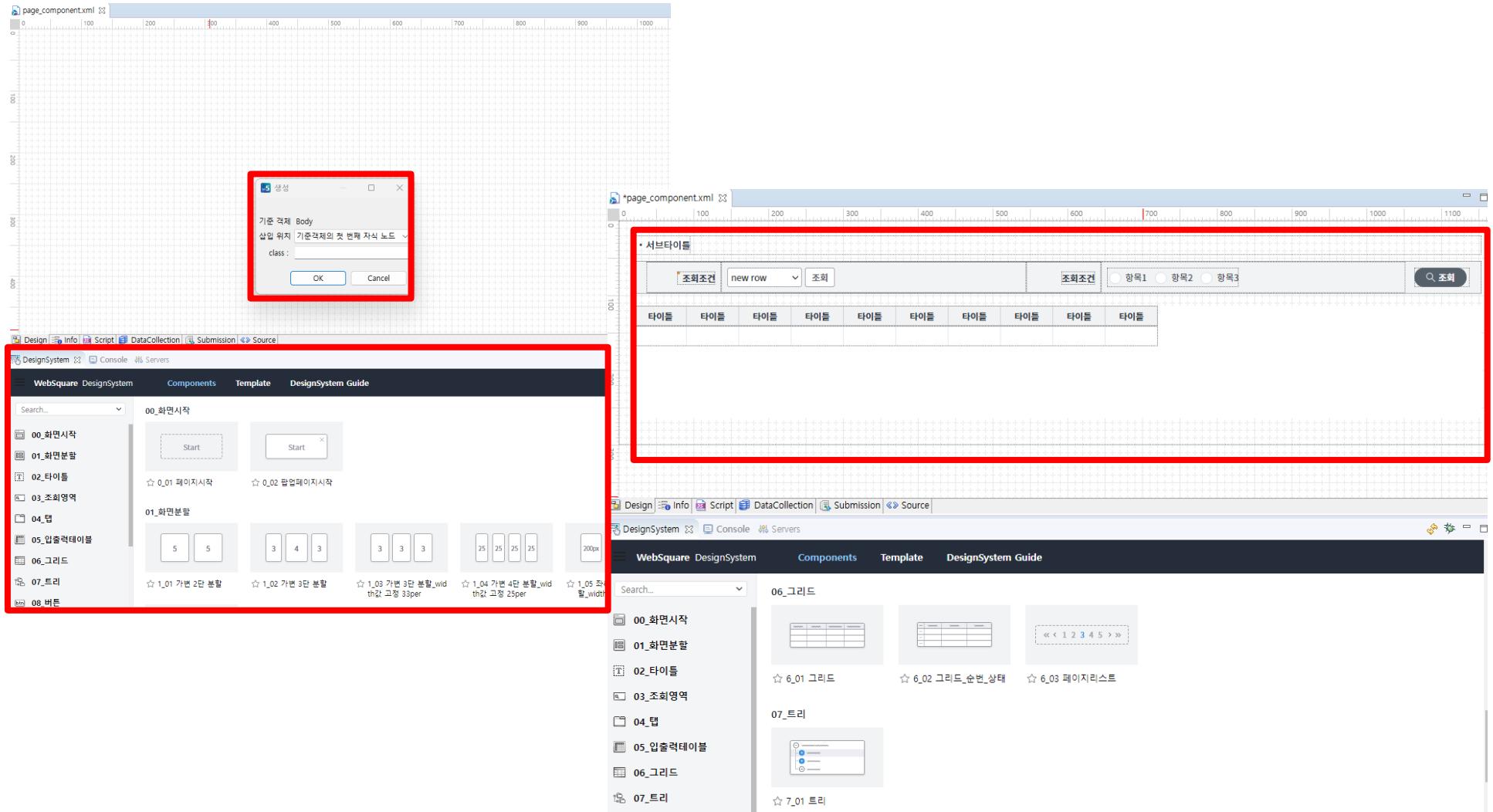
▪ External CSS

- Project Explorer에서 import 하고자 하는 css 파일 선택 후 Design Editor에 Drag & Drop 으로 적용 합니다.



▪ Design System

– 웹스퀘어5에서 기본으로 제공하는 CSS가 적용된 컴포넌트를 활용하여 화면을 구성합니다.



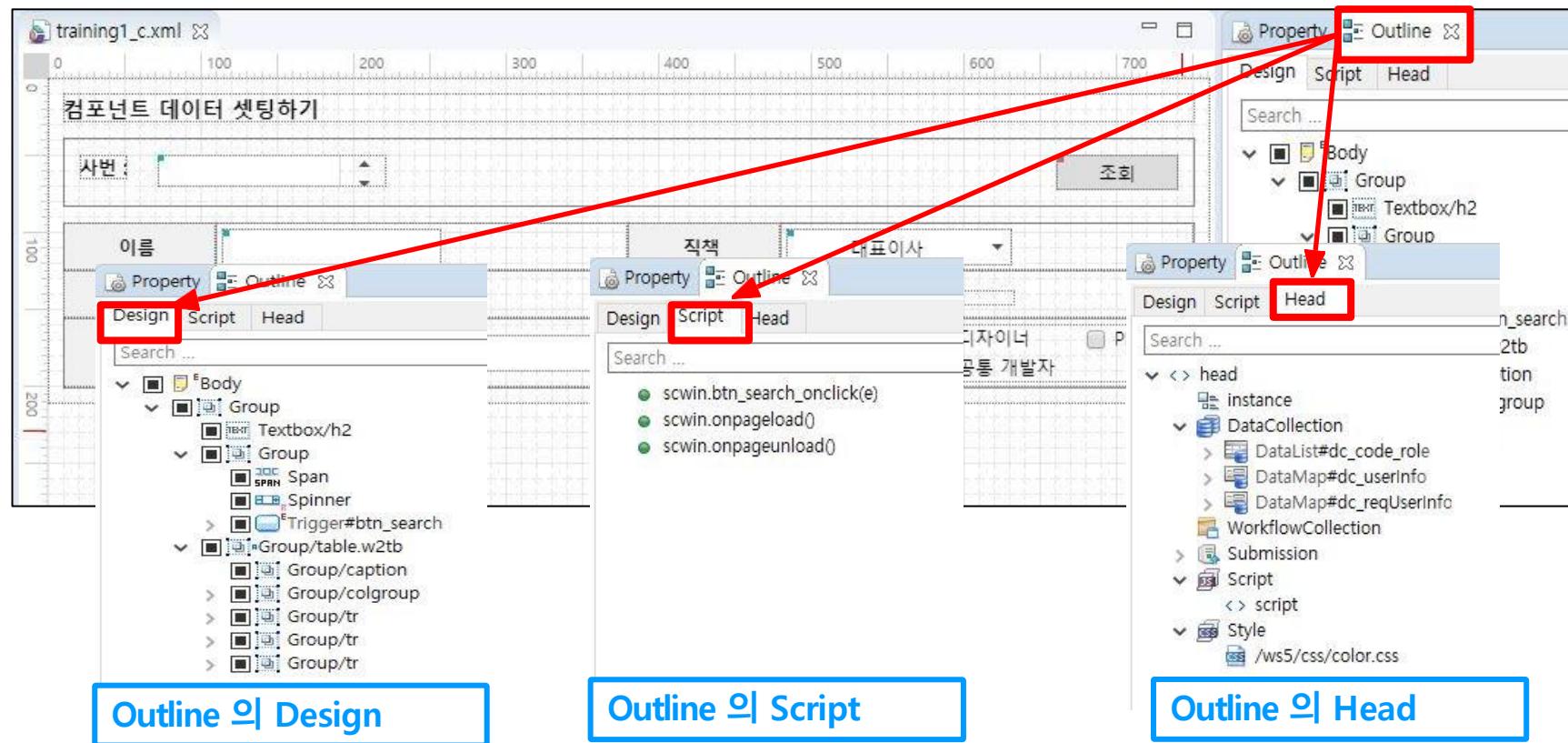
▪ Outline

- 전체 화면 구조를 한번에 확인할 수 있는 Viewer
- Design, Script, Head 부분으로 나눠서 확인할 수 있습니다.

Design에서는 전체 컴포넌트의 구조

Script 부분에서는 생성된 function 및 Event

Head 부분에서는 DataCollection 및 Submission, 외부 css 및 js 파일 확인



| 컴포넌트 설명

- 기본 컴포넌트
- 응용 컴포넌트
- OEM 컴포넌트
- 오픈 소스 사용 컴포넌트
- IFRAME
- WFRAME

1. 입력용 컴포넌트

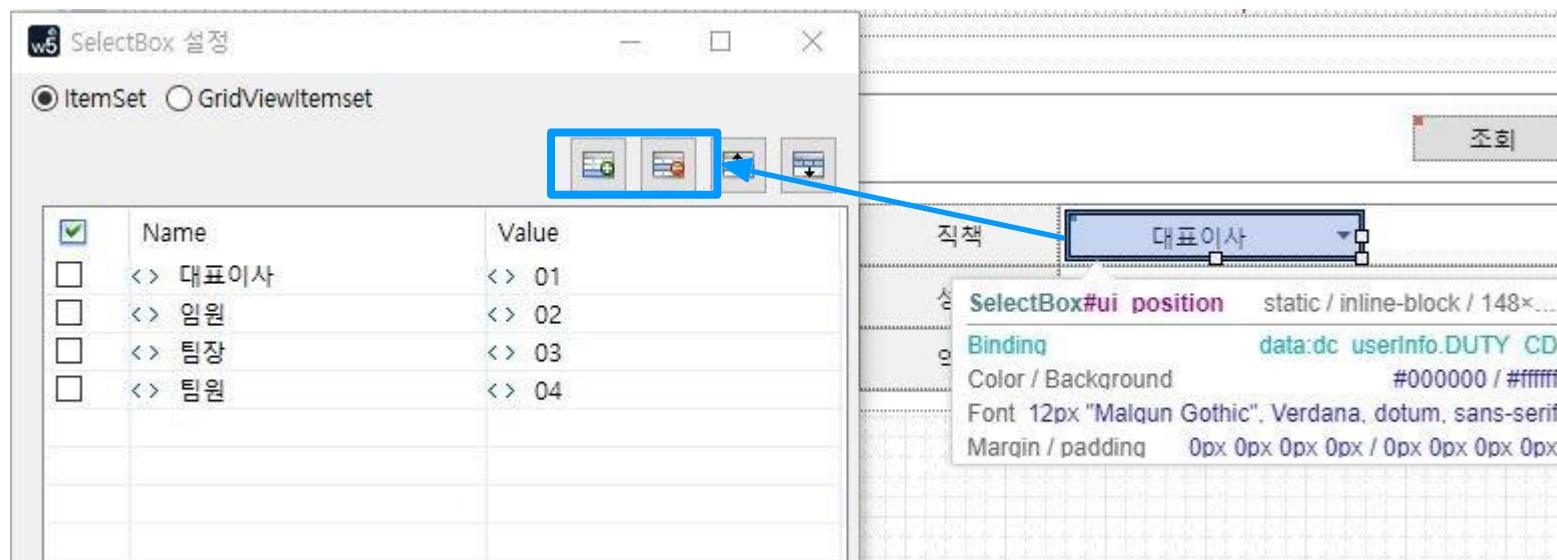
- inputBox, inputCalendar, spinner, searchBar, secret, textarea 등

2. 출력용 컴포넌트

- textbox, span, image, progressBar 등

3. List 컴포넌트

- selectBox, radio, checkbox, MultiSelect, checkComboBox, AutoComplete 등
- Hard Coding, Script Coding, DataList 바인딩 등과 같은 방법으로 Setting 할 수 있다.



Hard Coding

| | |
|----|---|
| 직책 | 대표이사 |
| 성별 | <input checked="" type="radio"/> 남성 <input type="radio"/> 여성 |
| 역할 | <input type="checkbox"/> 개발자 <input type="checkbox"/> 디자이너 <input type="checkbox"/> PM <input type="checkbox"/> PL <input type="checkbox"/> 공통 개발자 |

```
//성별 항목 script에서 추가하기
ui_gender.addItem( "M", "남성" );
ui_gender.addItem( "F", "여성" );
ui_gender.setSelectedIndex( 0 );
```

Script Coding

The screenshot shows the WebSquare IDE interface with the following components:

- Design View:** Shows a form with fields for Name, Birthdate, Gender, and Roles.
- Script View:** Displays the following JavaScript code:


```
//성별 항목 script에서 추가하기
ui_gender.addItem( "M", "남성" );
ui_gender.addItem( "F", "여성" );
ui_gender.setSelectedIndex( 0 );
```
- Component Properties:** Shows the component's properties under the **head** tab.
- DataCollection Editor Dialog:** Shows a table of roles:

| No | code | name |
|----|------|-------|
| 1 | 01 | 개발자 |
| 2 | 02 | 디자이너 |
| 3 | 03 | PM |
| 4 | 04 | PL |
| 5 | 05 | 공통개발자 |
- BindItemSet Dialog:** Shows configuration for the gender dropdown:

| |
|---|
| <input checked="" type="checkbox"/> BindItemSet |
| NodeSet : <input type="text" value="data:dc_role"/> |
| Label : <input type="text" value="name"/> |
| Value : <input type="text" value="code"/> |
| ref : <input type="text" value="data:dc.userInfo.ROLE_CD"/> |
| Span Direction : <input type="text" value="rows"/> |
| Span Count : <input type="text" value="1"/> |

Data Binding

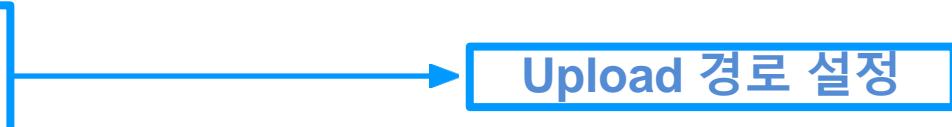
4. 전송 · 이동 컴포넌트

- trigger, anchor, upload, multiUpload 등

※ 파일 업로드 경로 설정

websquare_home/config/websquare.xml 파일에서 설정함.

```
<?xml version="1.0" encoding="UTF-8"?>  
<websquare>  
  <upload>  
    <baseDir value="~/~/~/upload">  
      <subImg value="~/~/~/upload/subImg"/>  
    </baseDir>  
    <encoding value="UTF-8" />  
    <maxUploadSize value="200000000" />  
    <uploadMode value="session" />  
    <folderName value="up"/>  
    <duplicatedFilePostFix value="yyyyMMddHHmmss"></duplicatedFilePostFix>  
    <allowedExtension>  
      gif, jpg, jpeg, doc, xls, ppt, pdf, txt, xlsx, png, pptx, xml, docx  
    </allowedExtension>  
    <deniedExtension>  
      exe  
    </deniedExtension>  
  </upload>  
</websquare>
```



5. 컨테이너 컴포넌트

- group
 - : tagName이라는 속성을 이용하여 지정된 html Tag로 브라우저 상에 Rendering
init이라는 API를 통해 하위 Component의 값을 초기화 시킬 수 있음(group1.init();)
- tabControl
 - : tab 형태로 제공됨
- windowContainder
 - : MDI 형태로 제공됨
- widgetContainer
 - : 구성된 Container의 (위치)배열 정보를 import/export 할 수 있음.
(접속된 사용자별로 화면구성을 개인화 하여 처리 가능)
- Accordion
 - : 모바일 웹 페이지에서 주로 사용하며, 화면을 접고, 펼치면서 구성 할 수 있음.

1. GridView

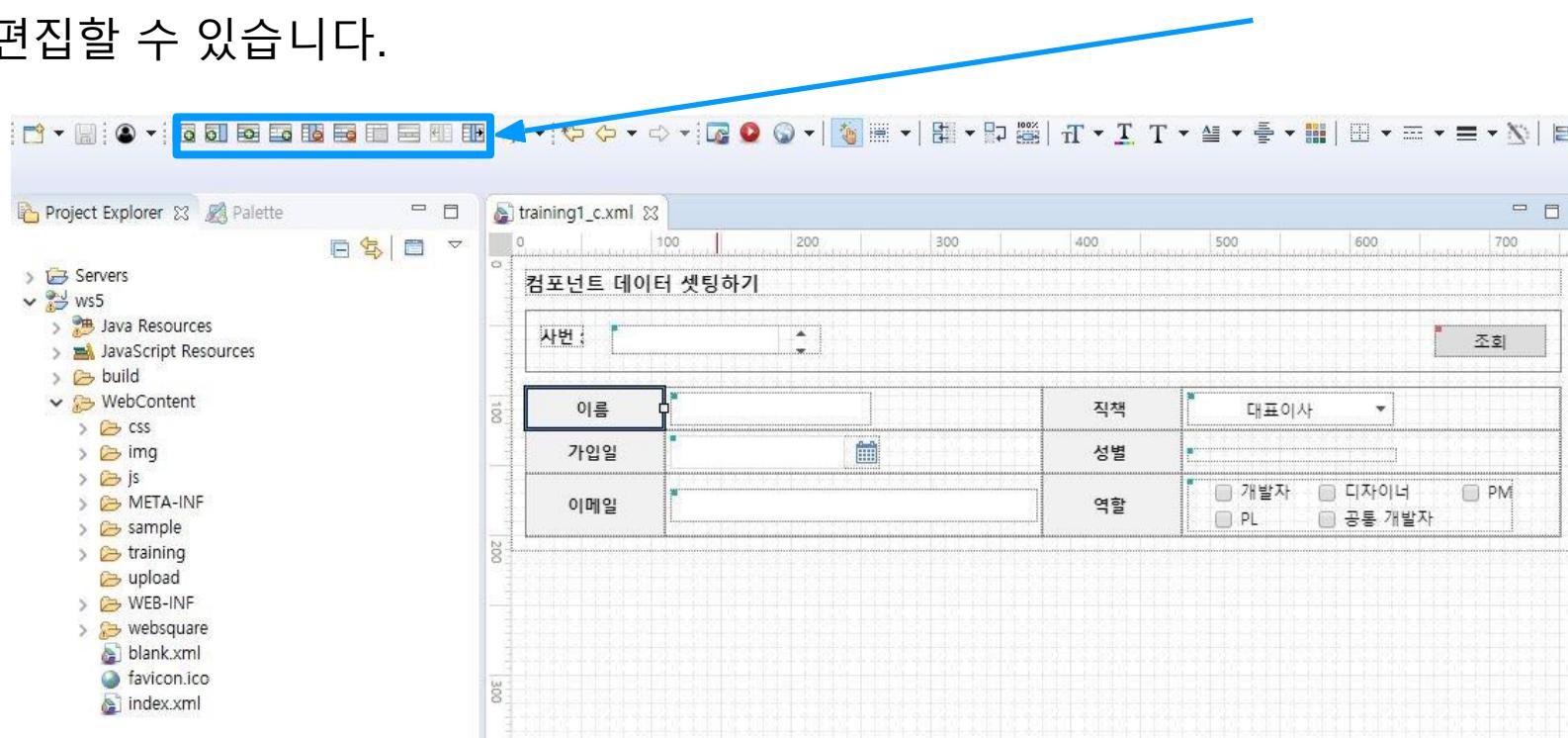
- Grid 형태로 나타내며, dataList와 연동하여 사용

2. Generator

- 반복된 형태를 표현, 제약은 비교적 없는 편이나 소스 코드량이 많습니다.

3. tableLayout

- 다수의 group 컴포넌트를 이용하여 Table 형태를 표현
- Pallet에서 tableLayout으로 검색하여 사용할 수 있으며, [Tool 상단의 버튼](#)을 이용하여 편집할 수 있습니다.



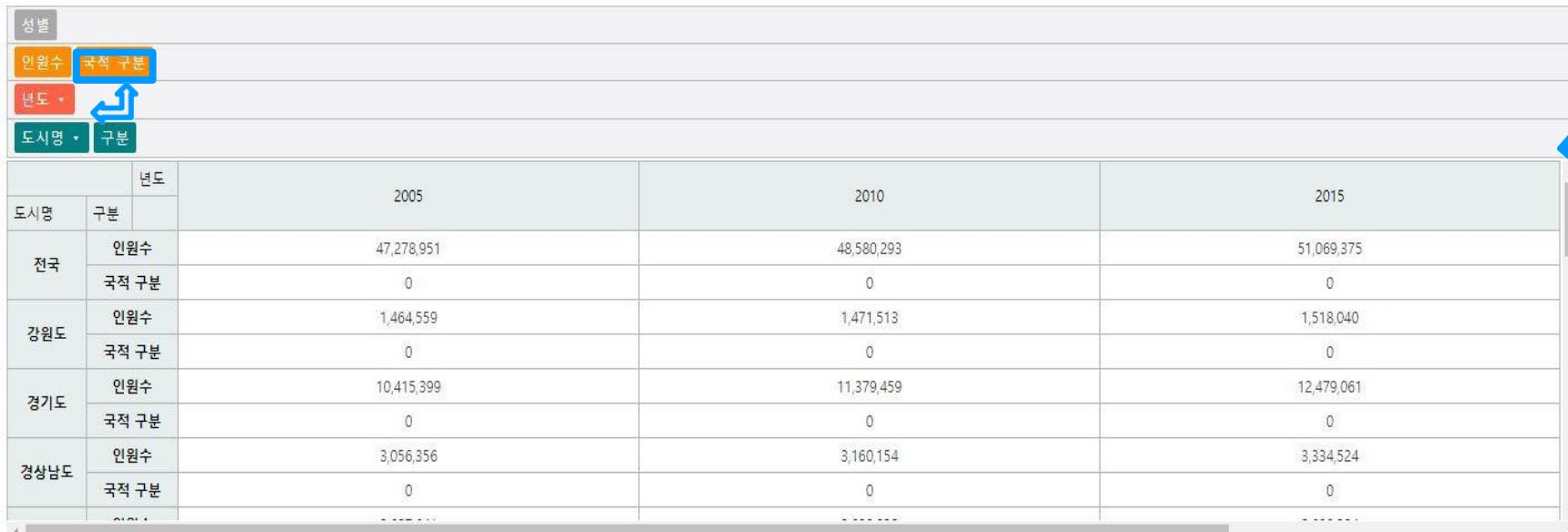
4. Pivot

- Excel의 pivot 기능이며, 행과 열을 변경하여 데이터를 표현할 때 사용



성별
인원수
년도 ▾ 국적 구분 ▾
도시명 ▾

| 년도 | | 2005 | | 2010 | | 2015 | |
|-------|-------|------------|---------|------------|---------|------------|-----------|
| 도시명 | 국적 구분 | 내국인 | 외국인 | 내국인 | 외국인 | 내국인 | 외국인 |
| 전국 | | 47,041,434 | 237,517 | 47,990,761 | 589,532 | 49,705,663 | 1,363,712 |
| 강원도 | | 1,460,770 | 3,789 | 1,463,650 | 7,863 | 1,499,734 | 18,306 |
| 경기도 | | 10,341,006 | 74,393 | 11,196,053 | 183,406 | 12,026,429 | 452,632 |
| 경상남도 | | 3,040,993 | 15,363 | 3,119,571 | 40,583 | 3,244,163 | 90,361 |
| 경상북도 | | 2,594,719 | 12,922 | 2,575,370 | 24,662 | 2,622,729 | 57,565 |
| 광주광역시 | | 1,413,644 | 4,072 | 1,466,143 | 9,602 | 1,481,289 | 21,592 |
| 대구광역시 | | 2,456,016 | 8,531 | 2,431,774 | 14,644 | 2,436,770 | 29,282 |

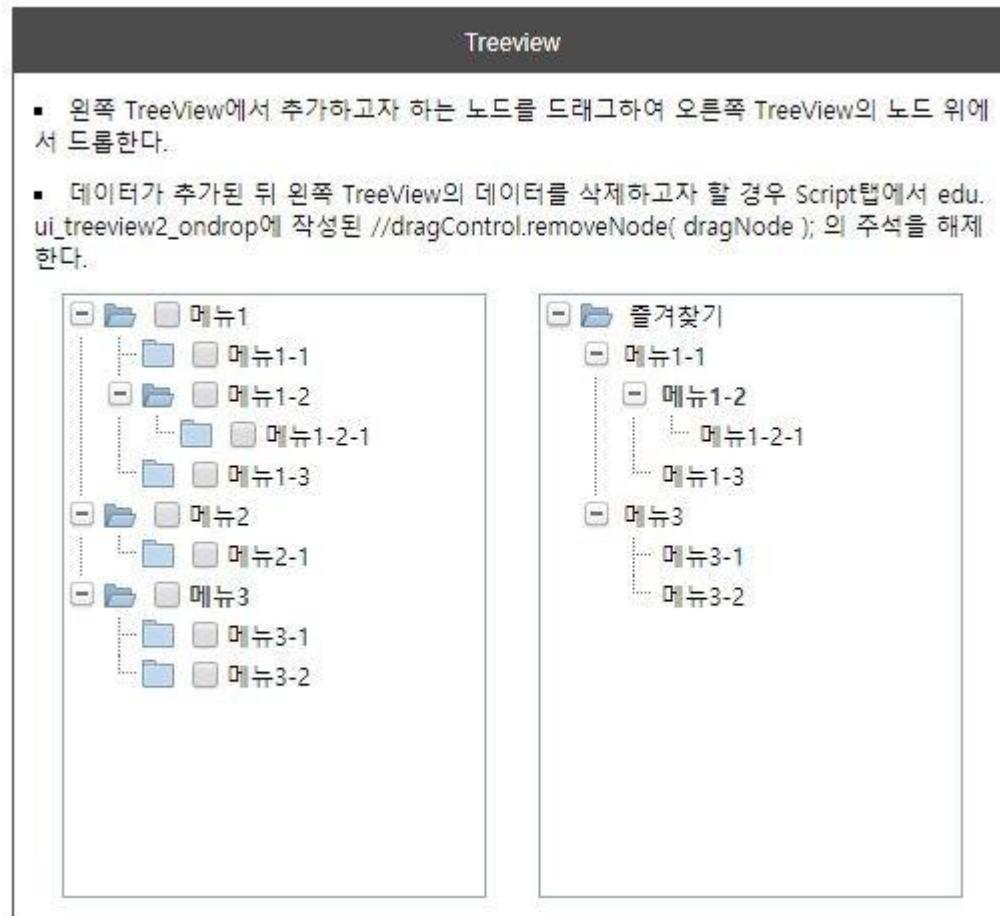


성별
인원수
국적 구분
년도 ▾
도시명 ▾ 구분

| | | 년도 | 2005 | | | 2010 | | | 2015 | | |
|------|-------|----|------------|-------|--|------------|-------|--|------------|-------|--|
| 도시명 | 구분 | | 인원수 | 국적 구분 | | 인원수 | 국적 구분 | | 인원수 | 국적 구분 | |
| 전국 | 인원수 | | 47,278,951 | | | 48,580,293 | | | 51,069,375 | | |
| | 국적 구분 | | 0 | | | 0 | | | 0 | | |
| 강원도 | 인원수 | | 1,464,559 | | | 1,471,513 | | | 1,518,040 | | |
| | 국적 구분 | | 0 | | | 0 | | | 0 | | |
| 경기도 | 인원수 | | 10,415,399 | | | 11,379,459 | | | 12,479,061 | | |
| | 국적 구분 | | 0 | | | 0 | | | 0 | | |
| 경상남도 | 인원수 | | 3,056,356 | | | 3,160,154 | | | 3,334,524 | | |
| | 국적 구분 | | 0 | | | 0 | | | 0 | | |

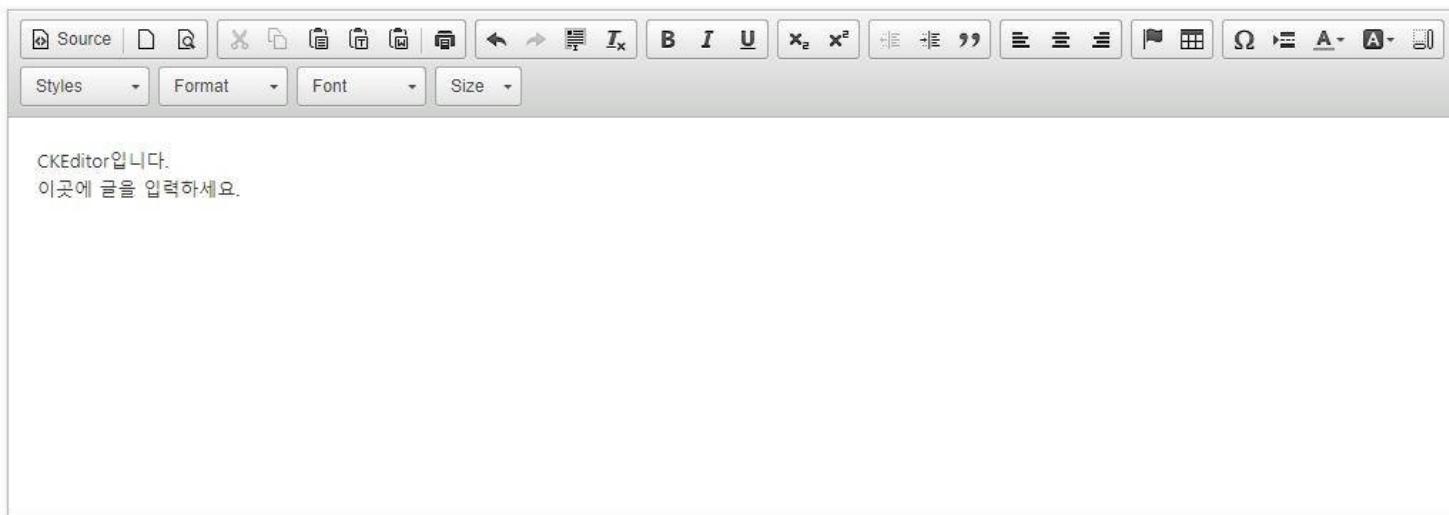
5. Treeview

- Tree구조의 노드를 표현하는 component



1. CK Editor

- Editor는 CK Editor를 사용
- (WebRoot)/websquare/externalJS 에 editor 관련 resource 폴더를 확인할 수 있으며, 하위 버전부터 최상위 버전 모두 확인할 수 있습니다.



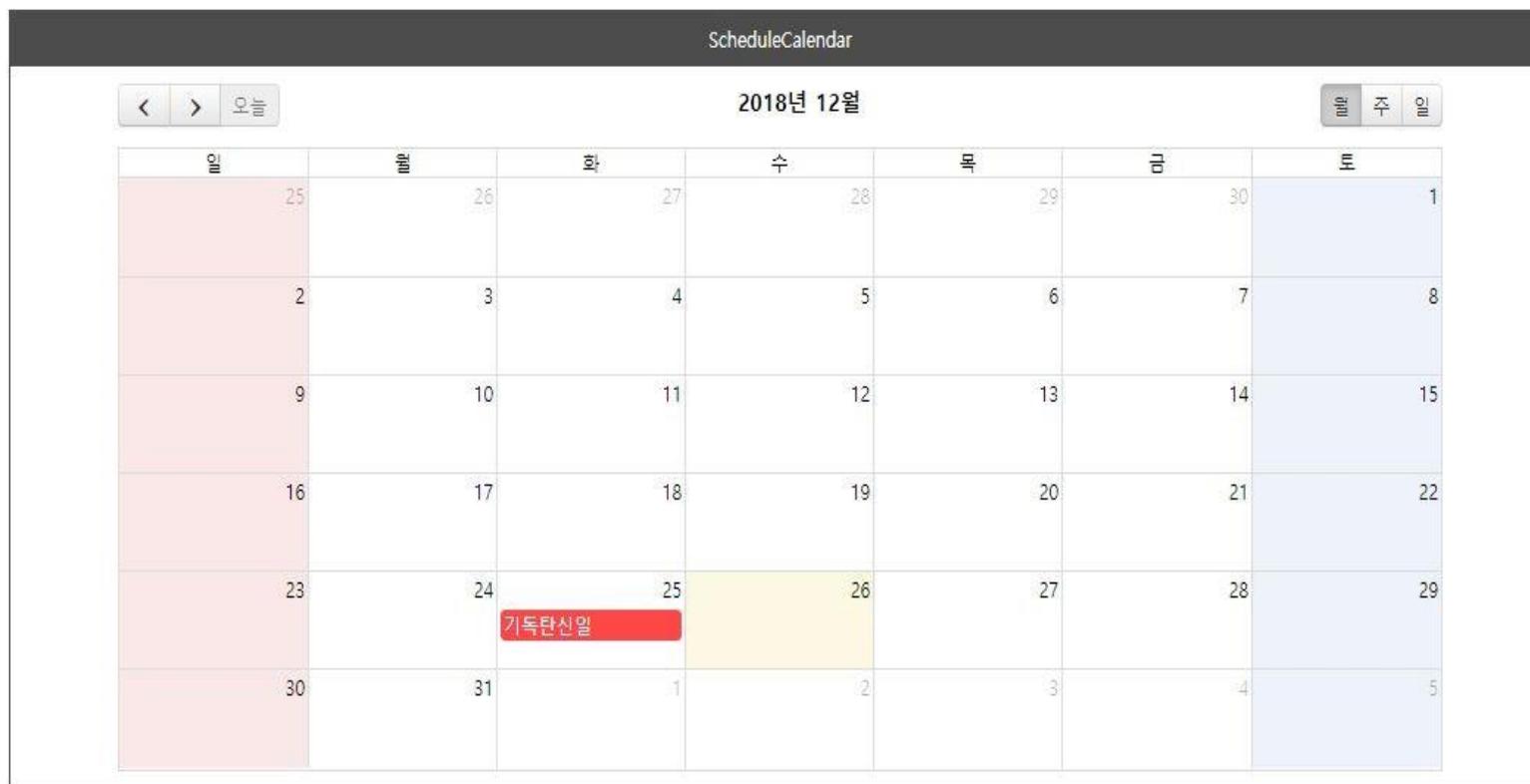
2. Fusion Chart

- Chart는 FusionChart 를 사용(Palette에서 f로 시작하는 chart 종류들)
- 사이트 주소: <http://www.fusioncharts.com>
- Document API 주소: <http://docs.fusioncharts.com/charts/>
- 현재 WebSquare5 내에 포함된 최상위 버전은 3.19이며, 관련 리소스는 (WebRoot)/websquare/externalJS 에 포함되어 있습니다.



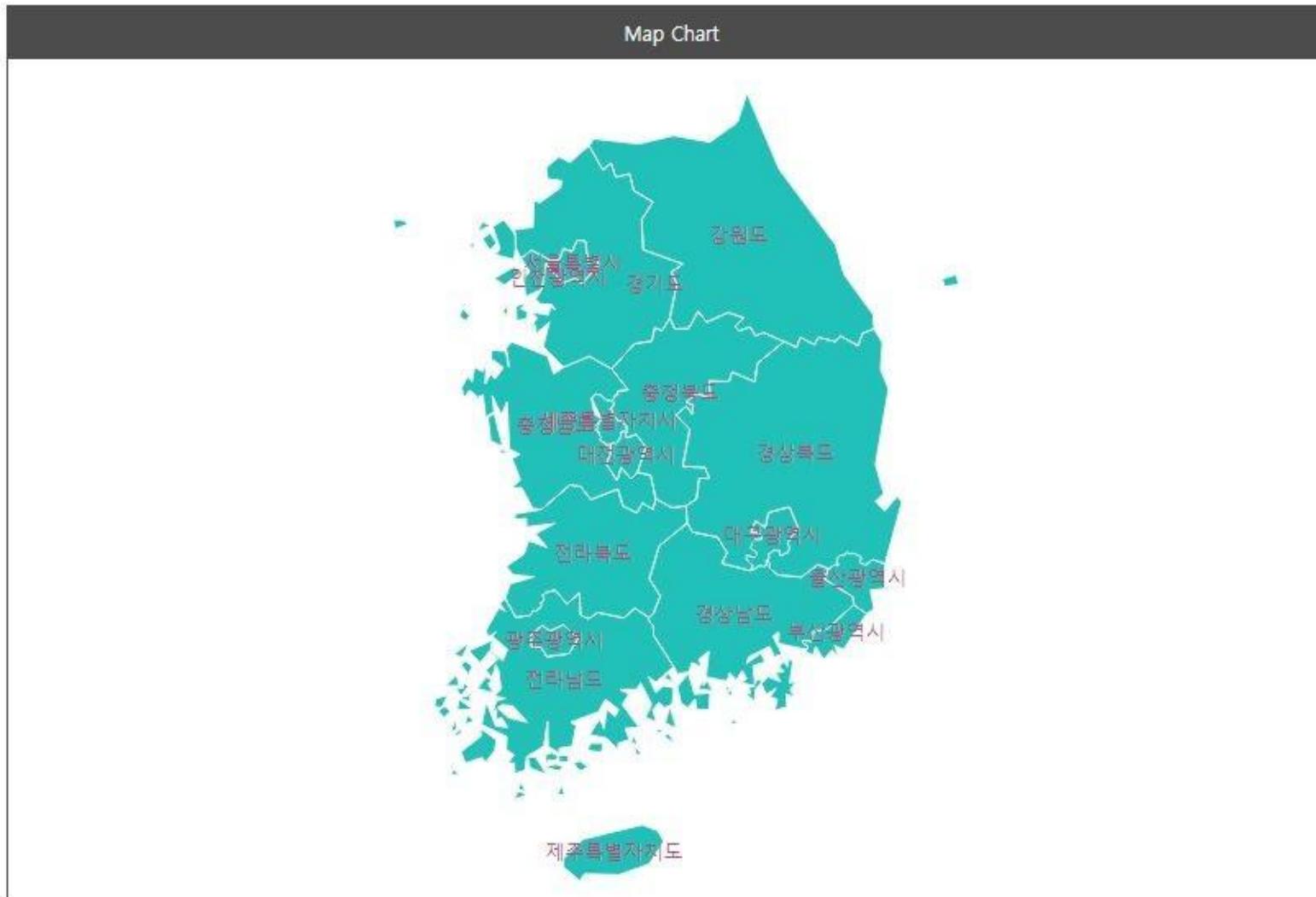
1. scheduleCalendar

- 구글 Calendar와 유사한 형태로 표현됨
- fullCalendar 를 이용하여 표현



2. mapChart

- 행안부 데이터를 이용하여 지도 형태의 차트를 나타냄
 - D3.js 를 이용하여 표현



1. 개념

- Html의 IFrame과 같은 개념
- 별도의 영역으로 제어되기 때문에 부모 페이지와 id가 중복된 컴포넌트가 있어도 사용 가능
- 별도의 브라우저 영역이므로 페이지 리소스를 IFrame 개수만큼 내려받음
(많을수록 화면 로딩이 느려짐)

2. TabControl에서 사용

- frameMode 속성을 'iframe' 으로 설정하여 사용
- API를 이용할 경우 wframe 옵션을 'false' 로 설정하여 사용

3. WindowCaontainer에서 사용

- frameMode 속성을 'iframe' 으로 설정하여 사용
- API를 이용할 경우 wframe 옵션을 'false' 로 설정하여 사용

4. SPA 적용

- 개념 : 웹페이지 코딩 기법인 SPA(화면 전환없이 로딩된 자원을 재활용 하는 기법)를 웹스퀘어5에 적용한 방법
- IFRAME, TabControl, WindowCaontainer 적용
- Url 호출방식 변경됨 기본) 쿼리 스트링 방식

ex) <http://domain.com/websquare/websquare.html?w2xPath=/MA/MA01M01.xml> SPA) Hash 방식

ex) <http://domain.com/websquare/websquare.html#w2xPath=/MA/MA01M01.xml>

다음은 SPA를 적용하기 위한 필수 설정입니다.

| 대상 | 기존 | SPA 적용 방식 |
|--------|---|--|
| 화면 전환 | <ul style="list-style-type: none"> 화면 전환 | <ul style="list-style-type: none"> 화면 전환 |
| API | <pre>ex1) location.href = ".....";</pre> <pre>ex2) \$p.url("/ws5/page/A01.xml");</pre> | <pre>\$p.url("/ws5/page/A01.xml", {spa:true});</pre> |
| Anchor | <ul style="list-style-type: none"> 화면 전환, 새 창으로 연결 <pre>ex1) <w2:anchor target="_self" href="/ws5/ws5.do?w2xPath=/ws5/page/A01.xml" /></pre> <pre>ex2) <w2:anchor target="_blank" href="/ws5/ws5.do?w2xPath/ws5/page/A0001" /></pre> | <ul style="list-style-type: none"> 화면 전환, 새 창으로 연결 <pre>ex1) <w2:anchor target="_self" href="#w2xPath=/ws5/page/A01.xml"/></pre> <pre>ex2) <w2:anchor target="_blank" href="/ws5/ws5.do#w2xPath=/ws5/page/A0001" /></pre> |
| IFrame | N/A | <ul style="list-style-type: none"> spa 속성 <p>spa로 화면을 전환 할지 여부</p> <pre>ex) <w2:iframe spa="true" src="/ws5/page/A01.xml"/></pre> |

다음은 SPA를 적용하기 위한 필수 설정입니다.

| 대상 | 기준 | SPA 적용 방식 |
|--------------------------------|-----|--|
| TabControl, WindowContainer | N/A | <ul style="list-style-type: none"> ▪ spaInitCount 속성 몇 개의 탭(IFrame)을 미리 로딩할지 개수를 설정. ▪ windowMaxNum 속성 최대 생성할 탭(IFrame)의 개수 ex1) <w2:tabControl spaInitCount="10" windowMaxNum="10"/> ex2) <w2:windowContainer spaInitCount="10" windowMaxNum="10"/> ▪ spaAuto 속성 spaInitCount와 같이 쓸 수 없으며, spaAuto 속성 지정하면, 다음 화면을 열 때 사용할 하나의 리소스 만을 더 받아 사용함, spaAutoDelay를 3000으로 설정하면 3초 후에 추가적인 리소스를 로드함. <w2:tabControl spaAuto="true" spaAutoDelay="3000"/> <w2:windowContainer spaAuto="10" spaAutoDelay="3000"/> |

1. 개념

- 파일 include 개념 (부모, 자식 관계가 없고 하나의 페이지로 인식)
- 부모 페이지와 동일한 영역으로 인식함(id 중복을 허용하지 않음)
- IFrame과는 달리 별도의 브라우저 영역이 아니므로, 최초 내려 받은 리소스를 그대로 이용함 (개수가 많다고 하여 화면이 느려 지지 않음)

2. TabControl에서 사용

- frameMode 속성을 'wframe' 으로 설정하여 사용 (default : wframe)
- API를 이용할 경우 wframe 옵션을 'true' 로 설정하여 사용 (default : wframe)

3. windowContainer에서 사용

- frameMode 속성을 'wframe' 으로 설정하여 사용
- API를 이용할 경우 wframe 옵션을 'true' 로 설정하여 사용

4. SCOPe 적용

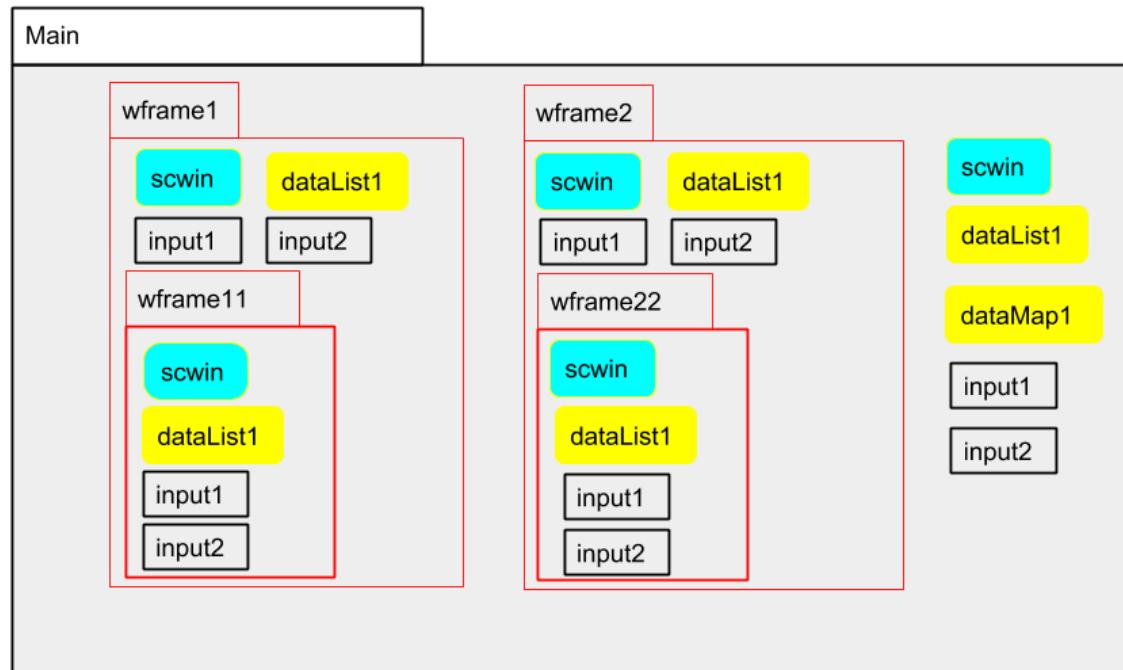
- 개념 : 기존 wframe과 달리 wframe 내의 영역을 별도의 scope 객체 내에 생성하여 id 중복의 문제를 피하며 사용할 수 있습니다.
(주의사항 : 별도의 scope 객체 내에 그리므로, 기존의 id가 내부적으로 변경되기에, scope 적용시 사용하는 코딩법을 준수 하여야 합니다.)
- WFrame ,TabControl, WindowCaontainer 적용
- **IFrame SPA 보다 권장함 (성능이 더 뛰어남)**

다음은 SCOPE 를 적용하기 위한 필수 설정입니다.

| 대상 | SPA+IFrame 적용 방식 | WFrame+Scope 적용 방식 |
|------------|--|---|
| 화면 전환 | <ul style="list-style-type: none"> 화면 전환 | <ul style="list-style-type: none"> 동일 |
| API | <pre>\$p.url("/ws5/page/A01.xml", {spa:true});</pre> | |
| Anchor | <ul style="list-style-type: none"> 화면 전환, 새 창으로 연결 <pre>ex1) <w2:anchor target="_self" href="#w2xPath=/ws5/page/A01.xml"/></pre> <pre>ex2) <w2:anchor target="_blank" href="/ws5/ws5.do#w2xPath=/ws5/page/A0001" /></pre> | <ul style="list-style-type: none"> 동일 |
| Frame | <ul style="list-style-type: none"> IFrame | <ul style="list-style-type: none"> WFrame의 Scope 속성 지정 <pre>ex1) <w2:wframe id="ui_contFrame" scope="true" ></w2:wframe></pre> |
| TabControl | <ul style="list-style-type: none"> spaInitCont 속성 몇 개의 탭(IFrame)을 미리 로딩할지 개수를 설정. windowMaxNum 속성 최대 생성할 탭(IFrame)의 개수 <pre>ex1) <w2:tabControl spaInitCount="10" windowMaxNum="10"/></pre> | <ul style="list-style-type: none"> addTab의 contentsOptions 속성 지정 <pre>ex1) contentsOptions = { src : tabUrl, wframe : true, scope : true }; [TabControl ID].addTab(tabKey, tabOptions, contentsOptions);</pre> |

| 대상 | SPA+IFrame 적용 방식 | WFrame+Scope 적용 방식 |
|---------------------|--|--|
| WindowContainer | <ul style="list-style-type: none"> ▪ spaInitCount 속성 몇 개의 탭(IFrame)을 미리 로딩할지 개수를 설정. ▪ windowMaxNum 속성 최대 생성할 탭(IFrame)의 개수 ex1) <w2:windowContainer spaInitCount="10" windowMaxNum="10"/> | <ul style="list-style-type: none"> ▪ frameMode 속성 ex1) <w2:windowContainer id="ui_windowContainer" frameMode="wframe"></w2:windowContainer> |
| 설정 파일 config.xml | <ul style="list-style-type: none"> ▪ spa/scriptCache 페이지 전환 시 JS를 다시 load하고 실행할지에 대한 여부. ▪ spa/variable 페이지 전환 시 자동으로 삭제할 객체명. ex) <pre><spa> <scriptCache value="true"/> <variable value="scwin"/> </spa></pre> | <ul style="list-style-type: none"> ▪ 동일 |

SCOPE Coding

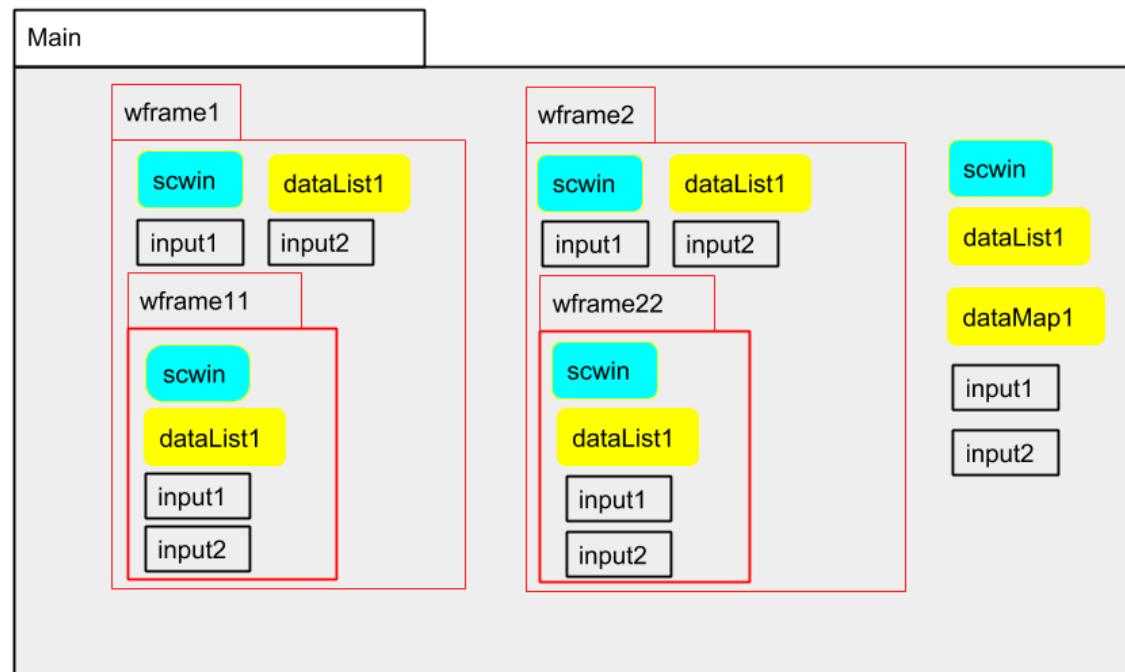


- 자식 페이지에서 부모 페이지로 접근할 때는 **\$p.parent()**를 사용합니다.
- 부모 페이지에서 자식 페이지를 접근할 때는 **wframe.getWindow()**로 scope 객체를 얻어온 다음 scope 객체를 통해 자식 페이지에 있는 객체를 접근 합니다.

예시)

- 1) Main에서 wframe11의 input1 객체의 값을 가져오는 코드 : `wframe1 getWindow().wframe11 getWindow().input1.getValue();`
- 2) wframe22에서 Main의 input2 객체에 값을 설정하는 코드 : `$p.parent().$p.parent().input2.setValue("test");`
- 3) wframe11에서 wframe22의 scwin.test(); 함수를 실행하는 방법 : `$p.parent().$p.parent().wframe2 getWindow().wframe22 getWindow().scwin.test();`

AliasDataCollection



- 자식 페이지에서 부모 페이지의 dataCollection 접근할 때 사용 합니다.
- 동일한 type의 dataCollection에만 접근이 가능 합니다. (dataMap => dataMap, dataList =>dataList)
- 부모 페이지에서 자식 페이지의 dataCollection으로는 접근할 수 없습니다.

예시)

1) wframe11에서 wframe1 화면에 있는 dataList1을 참조하는 aliasDataCollection 정의

```
<w2:aliasDataList id="aliasDataList1" scope="../../../dataList1"/> <!-- 상대경로 -->
<w2:aliasDataList id="aliasDataList1" scope="/wframe1/dataList1"/> <!-- 절대경로 -->
```

2) wframe22에서 Main화면에 있는 dataMap1을 참조하는 aliasDataCollection 정의

```
<w2:aliasDataMap id="aliasDataMap1" scope="../../..//dataMap1"/> <!-- 상대경로 -->
<w2:aliasDataMap id="aliasDataMap1" scope="/wframe2/wframe22/dataMap1"/> <!-- 절대경로 -->
```

| UI Design

- 개요
- 화면 구성 방법
- Static Poistion
- 주요 CSS

화면 디자인은 HTML에 CSS를 적용하는 방식을 그대로 적용했습니다.

UI 컴포넌트들은 모두 id, style, class속성을 가지고 있으며 CSS의 Selector 문법을 사용하여 컴포넌트의 스타일을 확장시킬 수 있습니다.

이 장은 화면 레이아웃을 구성하는 기본 CSS에 대한 내용을 담고 있으며, 개발자들에게는 다소 생소하거나 어렵게 느낄 수 있는 반응형 레이아웃에 대해 개발자 관점에서 전달하고자 합니다.(~~문서의 작성자가 개발자입니다.~~)

안내 사항

이 장은 CSS를 잘 모르는 개발자를 대상으로 CSS에 대한 최소한의 내용만 다뤘기 때문에 몇몇 내용에 대해서 의견이 있을 수 있음을 양해 드립니다.

여기서 말하는 반응형 레이아웃은 브라우저의 폭에 따라 컴포넌트가 배치가 하위로 떨어지는 구성 정도를 말하며, 그밖에 자세한 내용이나 세부적인 내용은 "반응형 웹"을 주제로 한 책 또는 블로그를 찾아보시기 바랍니다.

브라우저에 HTML UI Tag들이 그려지는 기준은 크게 고정 위치와 상태 위치로 나눌 수 있습니다.

어떤 방식으로 구성할지는 CSS 속성의 position으로 결정 됩니다.

아래는 position속성의 값 별 동작 방식 입니다.

- **Static Position**

position이 정의되지 않았거나 position:static;으로 정의된 경우입니다.

소스(코드)의 순서와 display값에 따라 배치되며 위치 속성(left,top,right,bottom)들은 무시 됩니다.

ex) style="width:150px; height:24px;"

- **Relative Postion**

Static Position의 확장으로 위치 속성들(left, top 등)을 사용할 수 있습니다.

- **Absolute Position**

파워포인트 또는 그림판의 도형 배치와 매우 흡사하다.

기준 점(left, top,right,bottom)을 잡고 크기(width, height)를 지정하여 배치하는 방법입니다.

ex) style="position:absolute; left:20px; top:10px; width:150px; height:24px;"

- **Fixed Position**

Absolute Position의 확장으로 스크롤이 되어도 해당 위치(left, top 등)에 고정 되어 있습니다.

- **Sticky Position**

Relative Position + Fixed Position와 같은 효과로 부모 컨테이너에 고정되어 fixed position 처럼 위치 됩니다.

CSS3 기능으로 CSS3 지원 브라우저에서만 적용 됩니다.

▪ HTML Elements의 display 성질

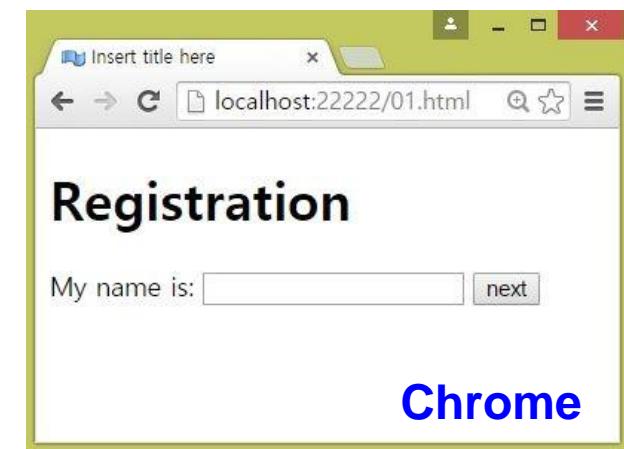
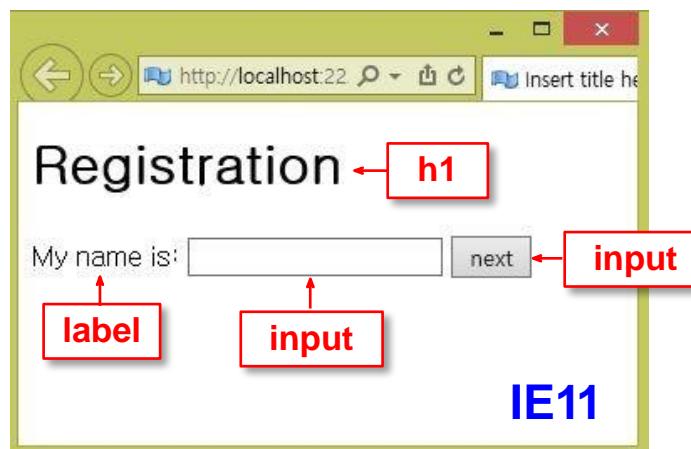
HTML 요소(Tag)들은 화면(브라우저)에 배치되기 위한 기본 성질을 가지고 있습니다.

즉, CSS가 적용되지 않았을 때도 사용자는 기본 골격을 갖춘 문서를 볼 수 있습니다.

▪ HTML 소스 예시

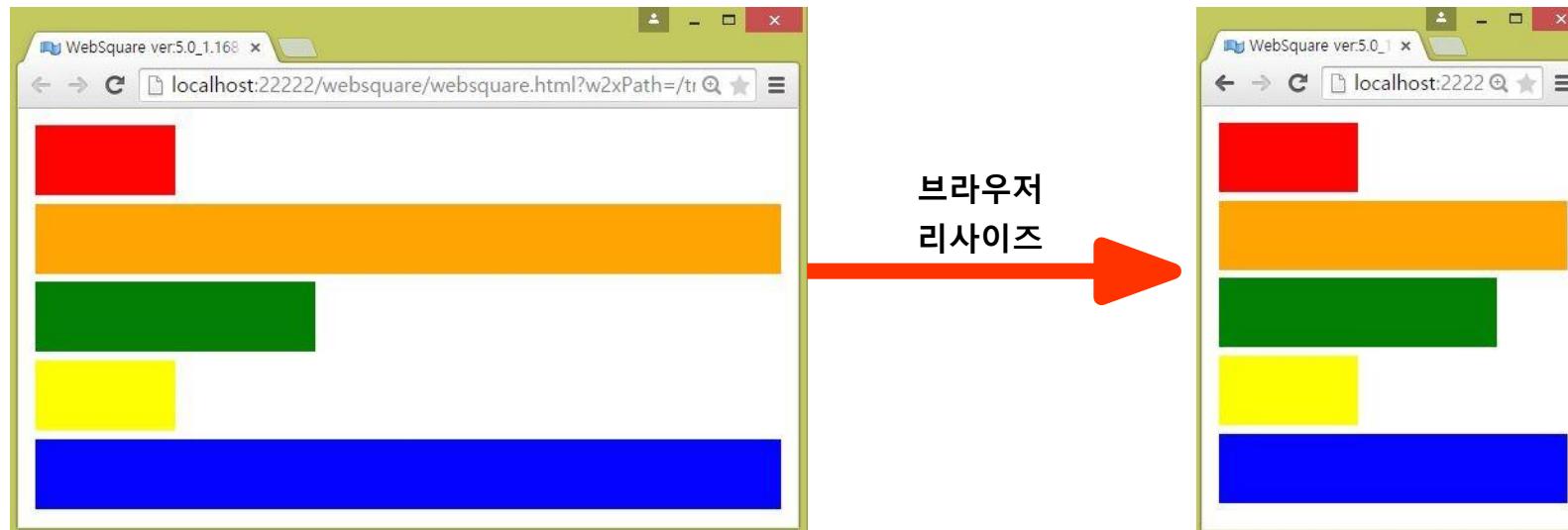
```
<body>  
  <h1>Registration</h1>  
  <label for="ui_name">My name is:</label>  
  <input type="text" id="ui_name">  
  <input type="button" value="next">  
</body>
```

▪ 브라우저에서 실행된 예시



▪ Block level element

- HTML : <div>, <h1>, <p>, 등
- WebSquare 컴포넌트 : Group, Textbox, GridView 등



▪ Inline level element

- HTML : , <a>, , <input> 등
- WebSquare 컴포넌트 : Span, InputBox, Trigger, Anchor 등



1. margin – 컴포넌트 간의 여백

- margin:10px; (모두 10px 적용)
- margin: 20px 0px 50px 100px; (top, right, bottom, left로 적용)
- margin-left:10px (left에만 적용)

2. padding – 컴포넌트 내부의 여백

- padding:10px;
- padding:20px 0px 50px 100px; (top, right, bottom, left로 적용)
- padding-left:10px;

3. text-align – 자식 컴포넌트/컨텐츠의 정렬

- text-align:center;

4. float – 컴포넌트의 정렬(흐름)

- float:right;

5. overflow – 자식 컴포넌트에 float을 적용했을 경우 부모 Group에 적용 합니다.

- overflow:hidden;

6. width – 컴포넌트의 폭.

Group, Textbox, GridView의 경우 width를 지정하지 않으면 부모 컴포넌트의 크기 만큼 화면에 가득 찹니다.

7. Height – 컴포넌트의 높이

Group, Textbox의 경우 height를 지정하지 않으면 컨텐츠의 크기 만큼 늘어 납니다.

단, 자식 컴포넌트에 float 속성이 있을 경우 해당 자식의 컴포넌트 크기를 계산하지 못하므로 overflow를 적용하여 하위 자식의 크기를 계산할 수 있게 해줍니다.

| 컴포넌트와 Data 연동

- 개요
- 기본 값(Value) 설정
- 기본 항목(Item) 설정
- Script로 값(Value) 설정
- Script로 항목(Item) 설정
- DataCollection과 컴포넌트의 값(value) 연동
- DataCollection과 컴포넌트의 항목(Item) 연동
- GridView와 DataList 연동

▪ 데이터 구분

– Value(값)

컴포넌트가 가지고 있는 고유한 값.

컴포넌트에 연결된 항목 중 선택된 값.

– Item(항목)

SelectBox, CheckBox등의 컴포넌트에 사용자가 선택할 수 있는 데이터 항목.

▪ 연동 방법

– Hard Coding

일반적으로 디자인 탭에 그려진 컴포넌트를 더블 클릭하면 기본값을 설정할 수 있는 창이 표현 됩니다.

– Script로 설정

컴포넌트의 id를 정의하고 Method를 이용하여 값(value) 또는 항목(item)을 추가 합니다.

ex1) input1.**setValue**("WebSquare");

ex2) selectbox1.**addItem**("Value01" , "Label-01");

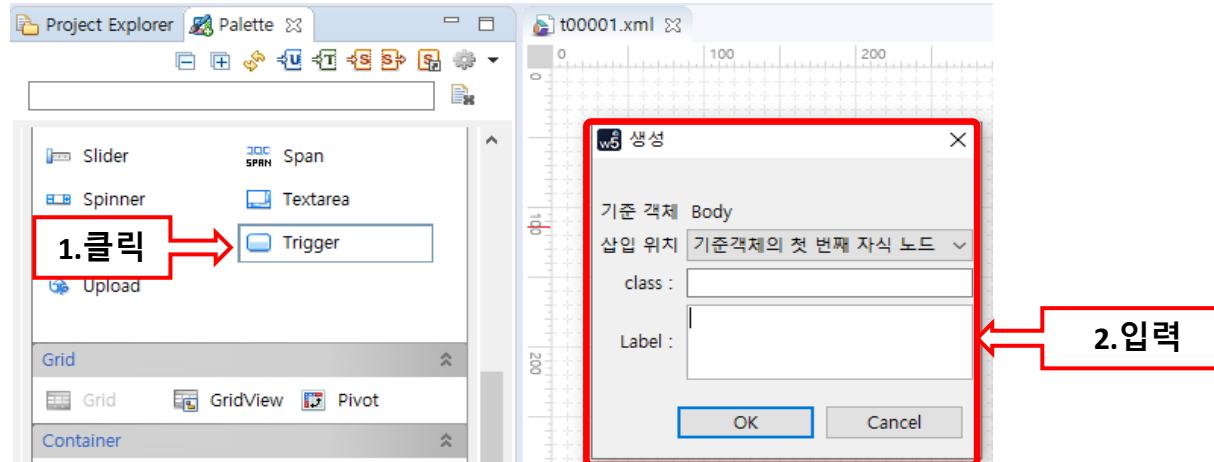
– DataCollection과의 연동

컴포넌트의 값(Value)는 ref 속성을 이용하여 연동 합니다.

컴포넌트의 항목(Item)은 itemSet을 이용하여 연동 합니다.

■ 방법1

Palette뷰에서 컴포넌트를 선택하면 설정 창에 Label(value)을 입력할 수 있습니다.

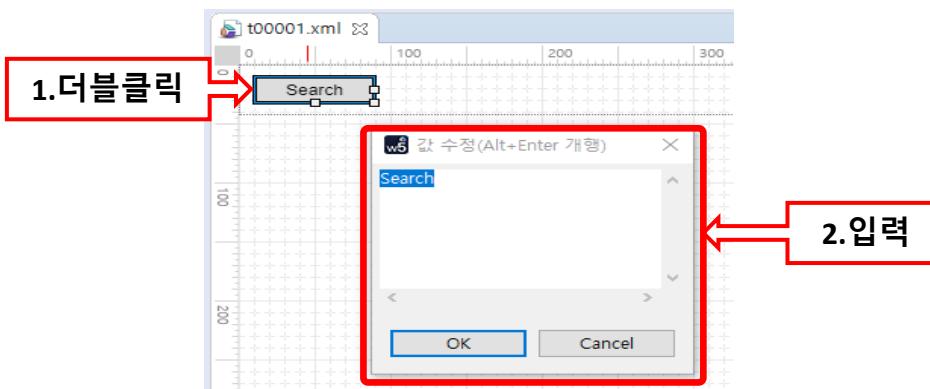


(단, 그리기 모드가 Static 일 경우에 사용 가능)



■ 방법2

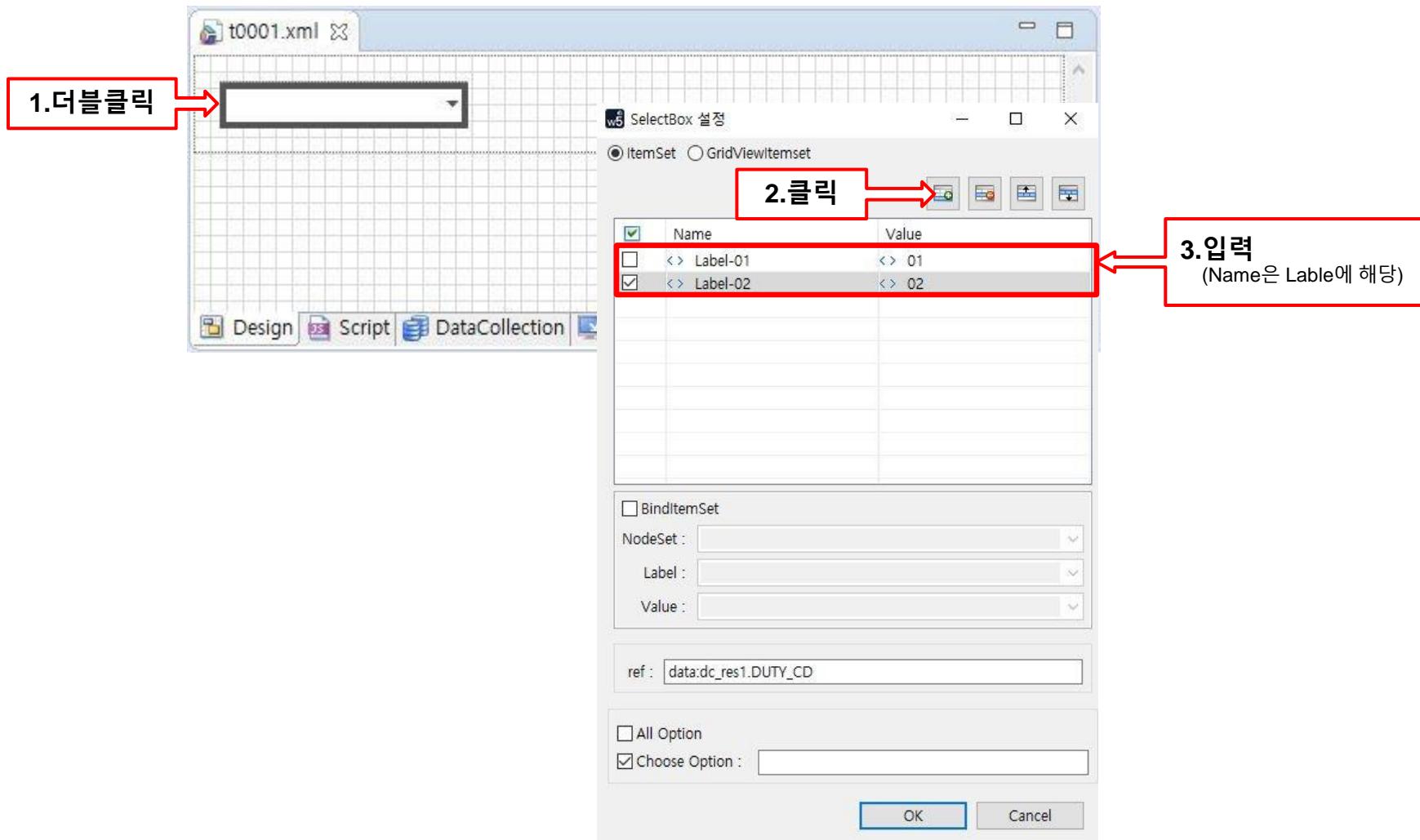
디자인 탭에서 컴포넌트를 double click 합니다.



▪ 방법

디자인 탭에서 component를 Double Click 합니다.

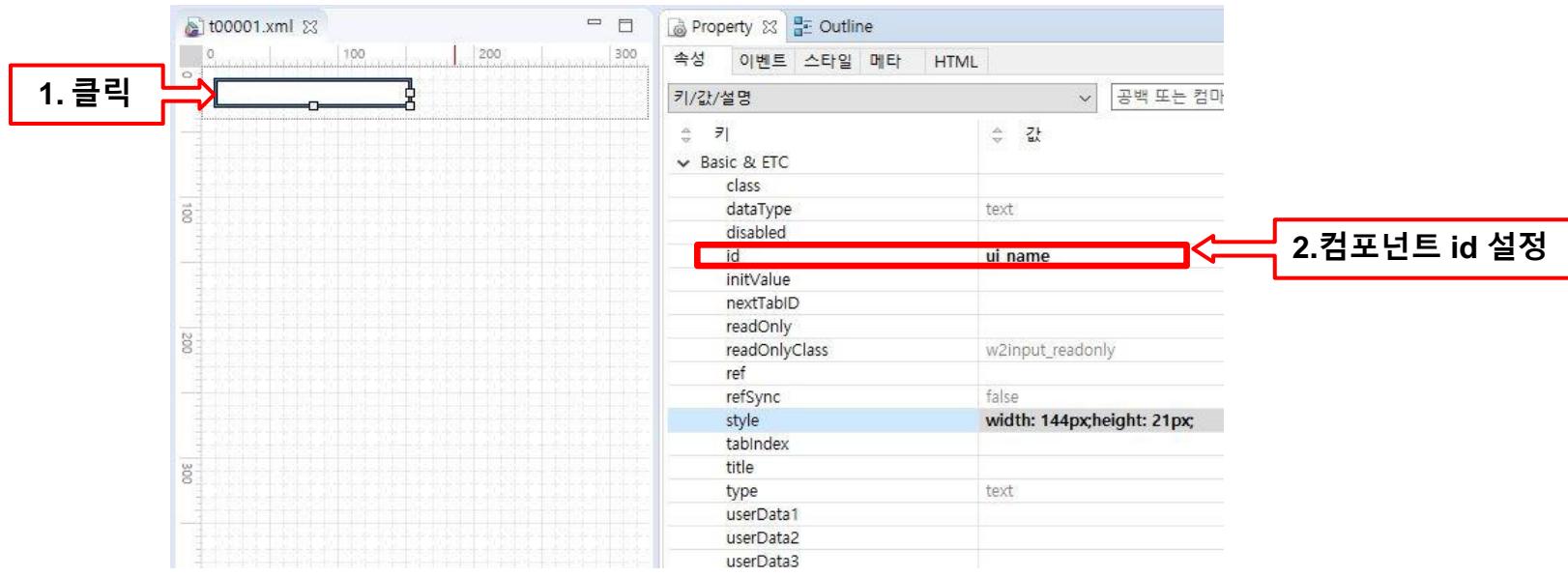
설정 창에서 [Add Row]버튼을 Click 하여 항목을 추가 합니다.



- 방법

디자인 탭에서 component를 Click 하고 id 속성을 정의 합니다.

Script블록에서 아래와 같이 component의 Method를 호출 합니다.



The screenshot shows the 't00001.xml' script block. It contains two lines of JavaScript code:

```
scwin.onpageload = function() {
    ui_name.setValue( "WebSquare5" );
};
```

A red box highlights the line 'ui_name.setValue("WebSquare5");' and is labeled '3. 컴포넌트의 id로 접근하여 Method 호출' (Access the component by its id and call the method).

▪ 방법

디자인 탭에서 component를 클릭하고 id 속성을 정의 합니다.

Script블록에서 아래와 같이 component의 Method를 호출 합니다.

The screenshot illustrates a three-step process for setting item properties:

- 1. 클릭**: A red box highlights a component on the design tab (t00001.xml). An arrow points from this box to the component.
- 2. 컴포넌트 id 설정**: A red box highlights the "id" field in the Property panel. The value "ui_gender" is entered. An arrow points from this box to the "id" field in the code editor.
- 3. 컴포넌트의 id로 접근하여 Method 호출**: A red box highlights the line of code `ui_gender.addItem("F", "여성");` in the script editor. An arrow points from this box to the "ui_gender" variable in the code.

Design Tab (t00001.xml):

Property Panel:

| 키 | 값 |
|----|-----------|
| id | ui_gender |

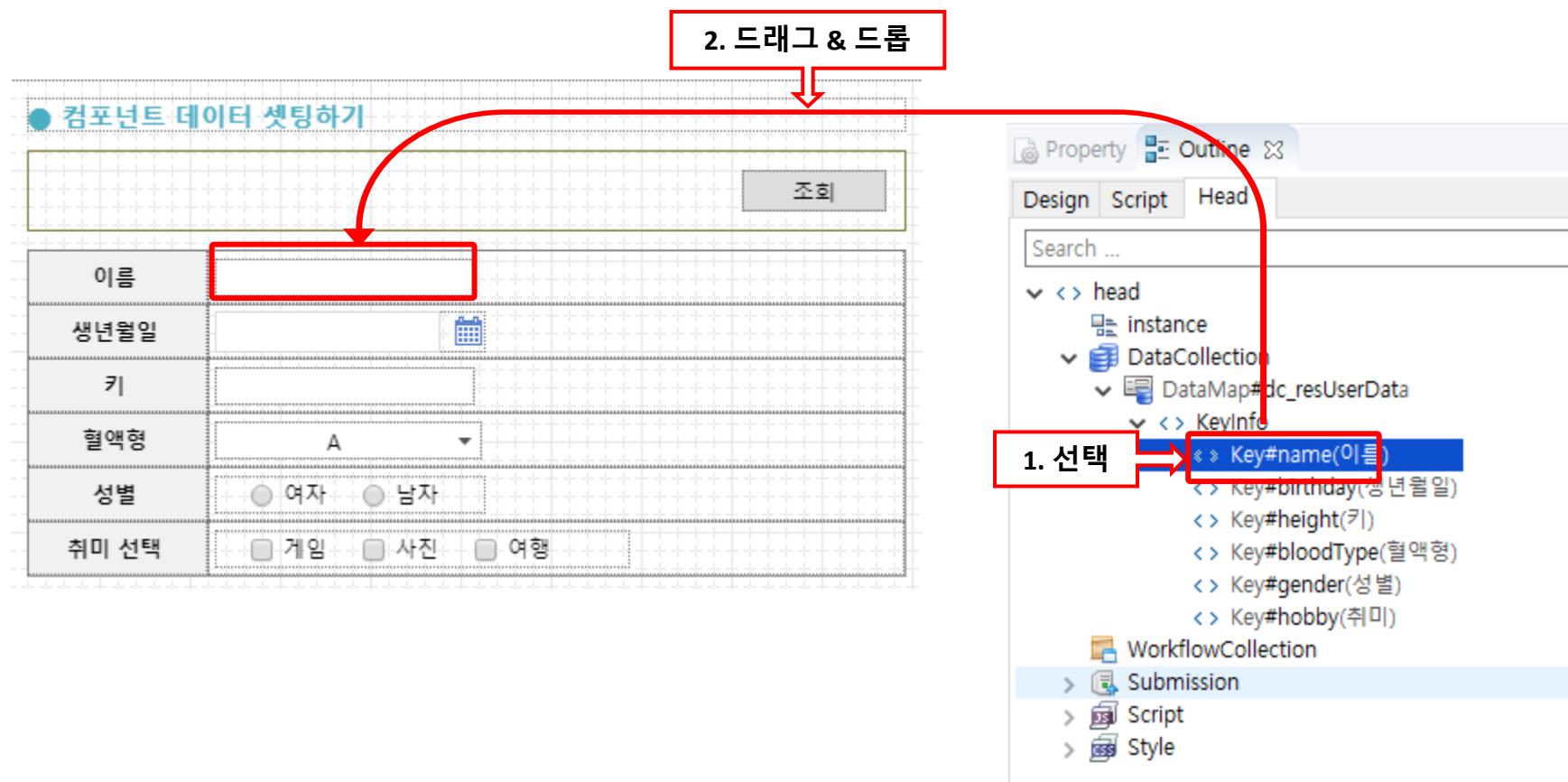
Script Editor (t00001.xml):

```
scwin.onload = function() {
    ui_gender.addItem("F", "여성");
    ui_gender.addItem("M", "남성");
};

scwin.onpageunload = function() {
};
```

■ 방법

우측의 Outline에서 연동할 DataMap의 Key를 마우스로 Drag하여 디자인 탭에 그려진 component 위에서 Drop 합니다.



▪ 설정 확인

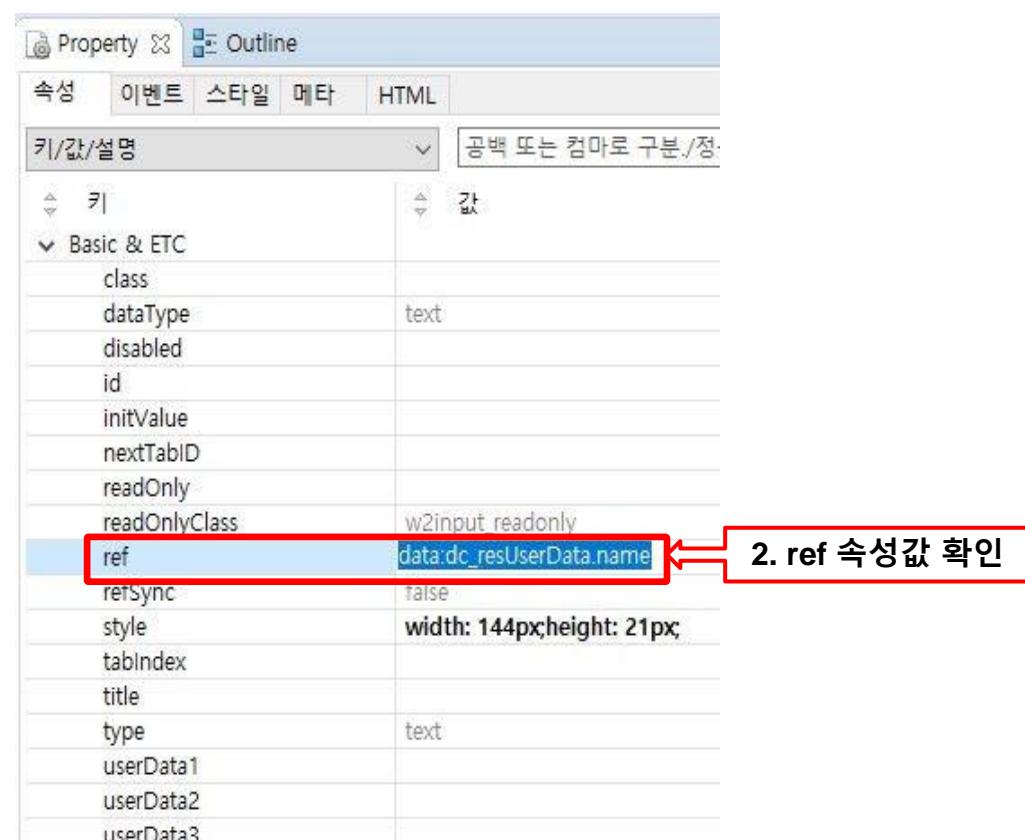
설정이 완료 되면 디자인 탭에서 컴포넌트의 좌측 상단에 초록색 마크가 생성되며 속성창에서 ref속성에 값을 확인할 수 있습니다.

● 컴포넌트 데이터 셋팅하기

1. 초록색 마크 확인

조회

| | |
|-------|---|
| 이름 | <input type="text"/> |
| 생년월일 | <input type="text"/> |
| 키 | <input type="text"/> |
| 혈액형 | A |
| 성별 | <input checked="" type="radio"/> 여자 <input type="radio"/> 남자 |
| 취미 선택 | <input type="checkbox"/> 게임 <input type="checkbox"/> 사진 <input type="checkbox"/> 여행 |



itemSet은 아래와 같이 총 3가지의 항목을 가지고 있습니다.

- **NodeSet**

항목으로 표현할 DataList의 id.

- **Label**

화면에 표현될 항목의 Label로 DataList의 컬럼 id.

- **Value**

각 항목의 Value로 설정할 DataList의 컬럼 id.

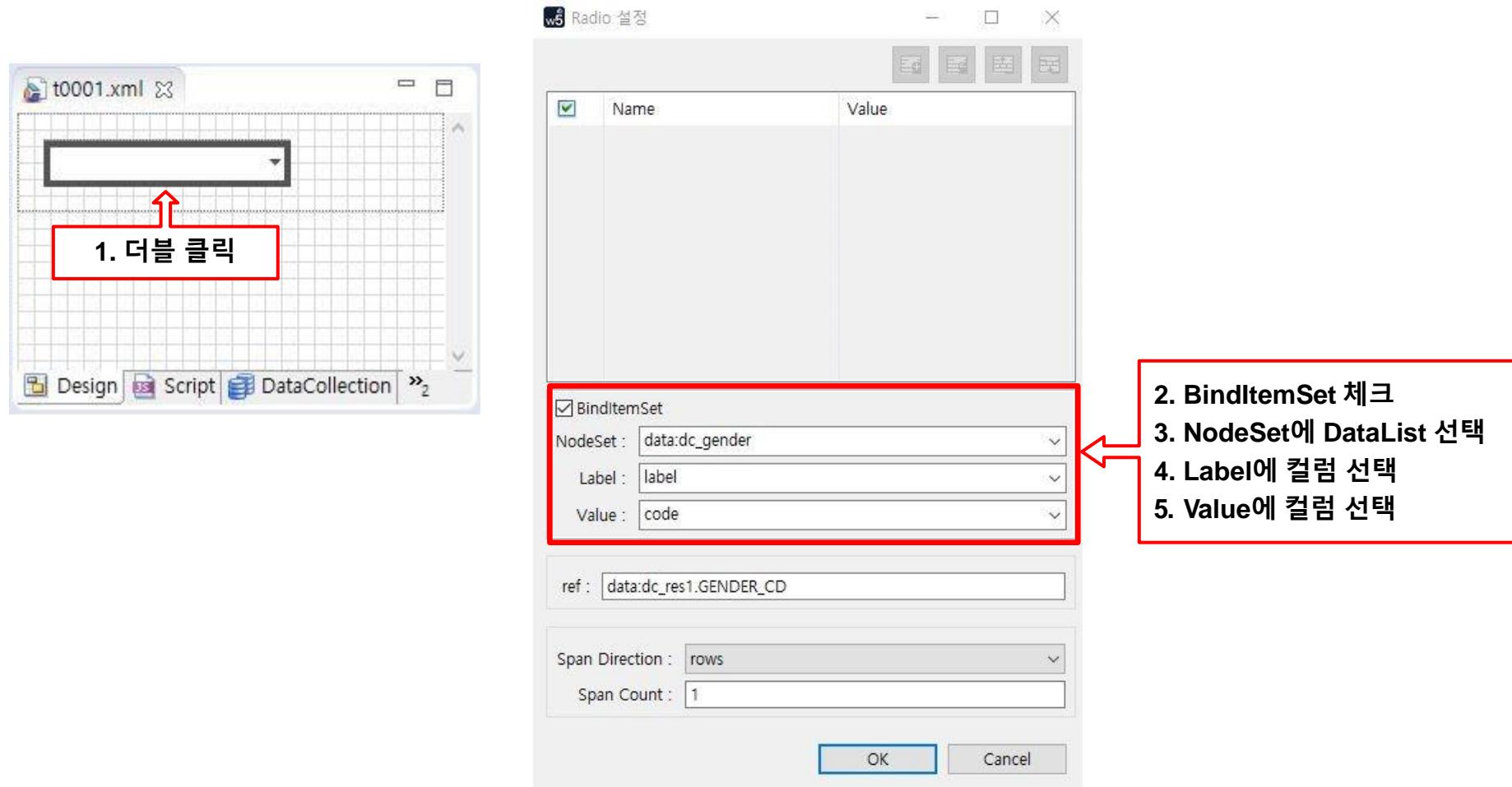
주의 사항

NodeSet은 DataList의 id의 앞에 "data:"를 필수로 기입해야 합니다.

ex) **data:dc_hobby**

▪ 방법

디자인 탭에서 컴포넌트를 더블클릭 후 항목 입력창이 뜨면 itemSet항목을 체크한 뒤 환경에 맞게 선택 합니다.



GridView 컴포넌트는 정의된 DataList를 기반으로 데이터를 출력하는 component 입니다.
그렇기 때문에 GridView컴포넌트를 사용하기 위해서는 먼저 DataList를 정의해야 합니다.
연동 방법은 DataList의 id를 GridView의 dataList속성에 정의하고,
컬럼에 표현할 DataList의 컬럼 id를 GridView의 Body컬럼의 id와 일치시켜서 사용합니다.

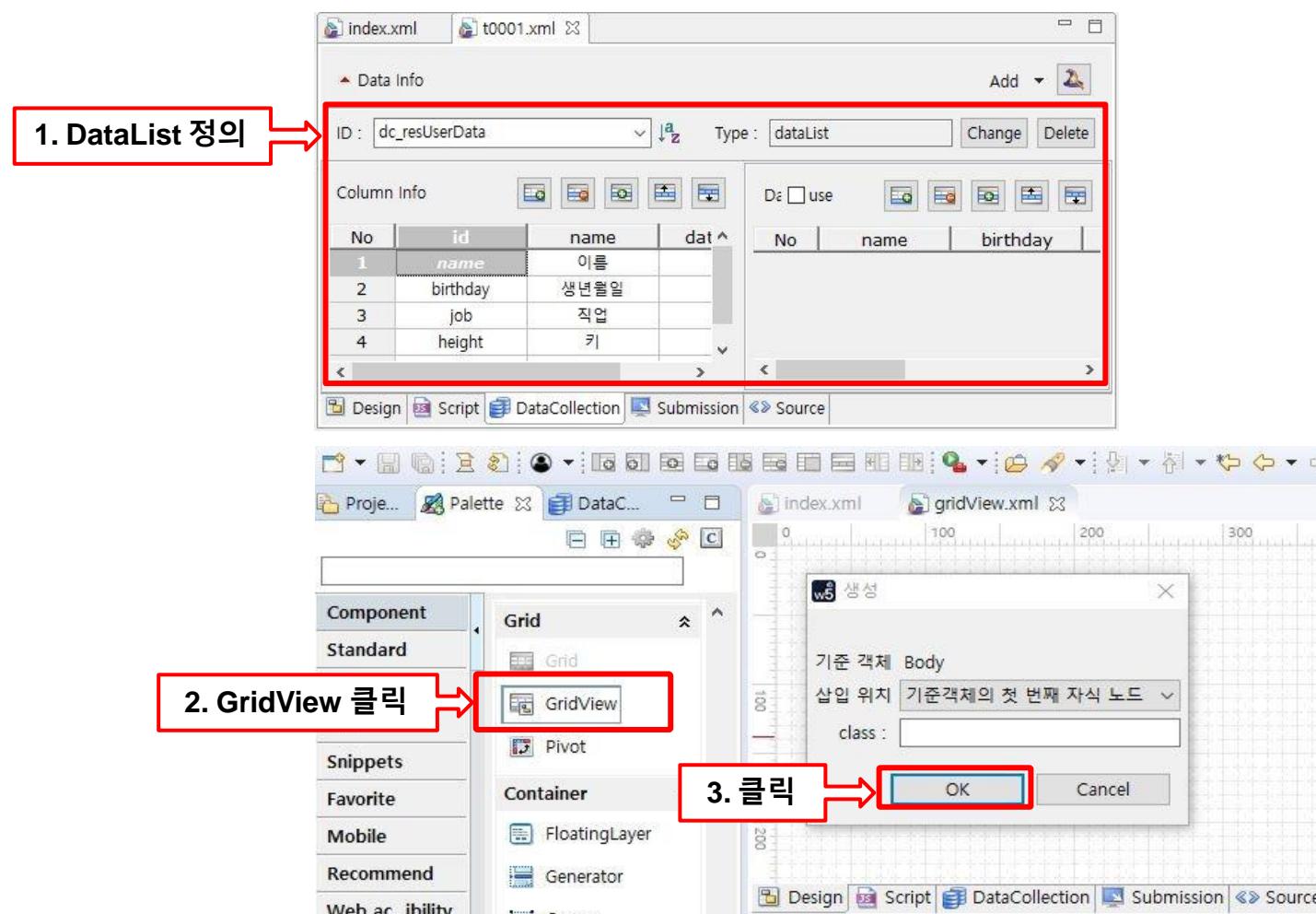
▪ 방법1

GridView와 연동할 dataList를 생성 합니다.(생성방법은 생략)

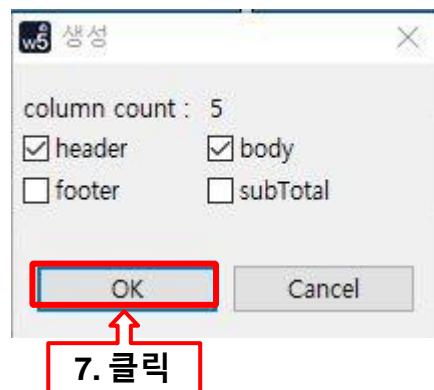
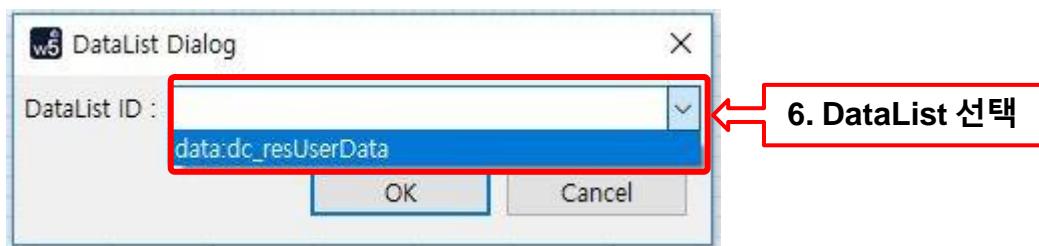
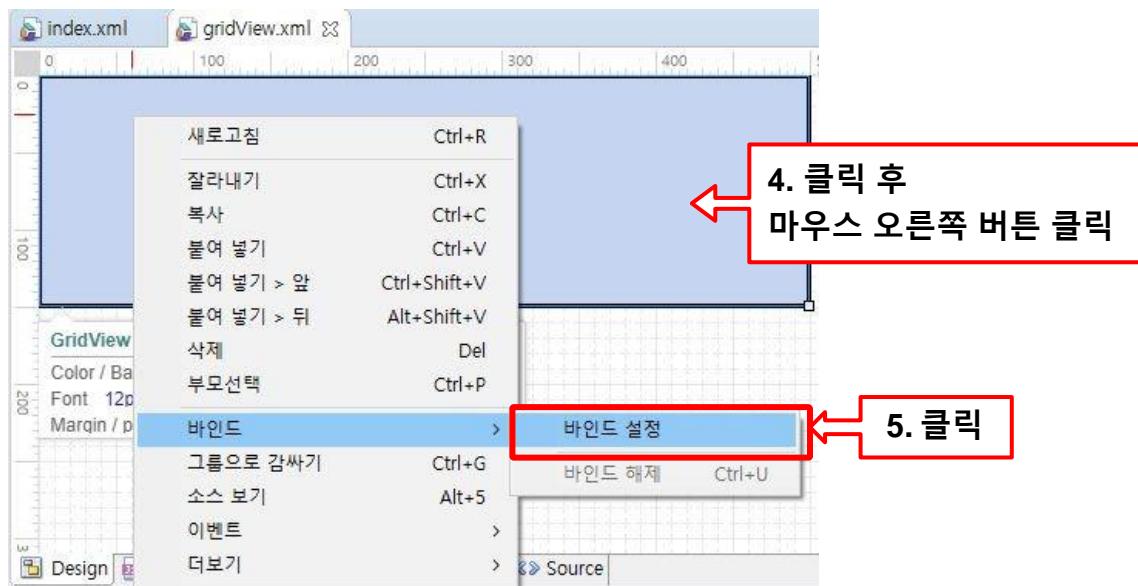
Palette뷰에서 GridView컴포넌트를 Click하고 생성 팝업 창에서 OK를 Click 합니다.

디자인 탭에서 GridView 컴포넌트 클릭하고 마우스 오른쪽 버튼을 Click 합니다.

메뉴의 항목 중 [DataList 바인딩]을 Click하고 연결할 dataList를 선택 합니다.



✓ DataCollection과 GridView 연동 1-2



▪ 확인

DataList에 정의된 컬럼 정보를 통해 아래와 같이 생성 됩니다.

DataList의 컬럼의 name항목에 정의 된 값이 헤더에 표현 됩니다.

헤더의 컬럼을 선택하면 value속성에 값이 입력된 것을 확인 할 수 있습니다.

The screenshot shows the WebSquare development environment. At the top, there is a toolbar with various icons. Below it is a main workspace containing a 'gridView.xml' editor window. Inside the editor, a table header row is visible with columns labeled '이름', '생년월일', '직업', '키', and '성별'. A red box highlights this row, and a red arrow points from the 'name' column of the 'Column Info' table below to the '이름' column in the header. The 'gridView.xml' window has scroll bars on the right and bottom. Below the editor is a navigation bar with tabs: Design, Script, DataCollection, Submission, and Source. The 'DataCollection' tab is selected. In the bottom left corner, there is a 'Data Info' panel. It contains fields for 'ID' (dc_resUserData), 'Base Node' (empty), 'JSON Path' (empty), and 'Id Attribute' (empty). Below these is a 'Column Info' section with a table:

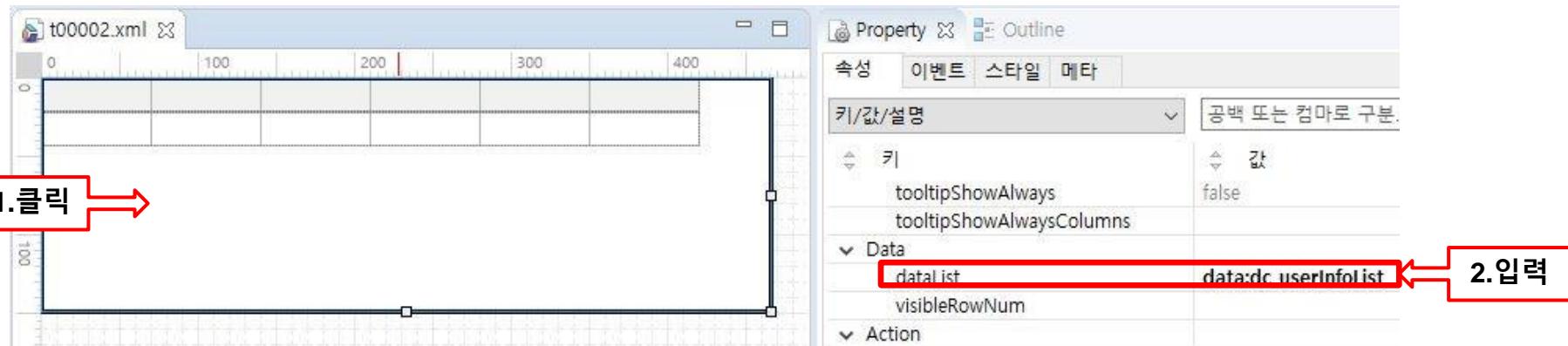
| No | id | name | datatype |
|----|----------|------|----------|
| 1 | name | 이름 | |
| 2 | birthday | 생년월일 | |
| 3 | job | 직업 | |
| 4 | height | 키 | |
| 5 | gender | 성별 | |

A red box highlights the entire 'Column Info' table, and a red arrow points from the 'name' column of the table to the 'name' column in the 'gridView.xml' editor's header.

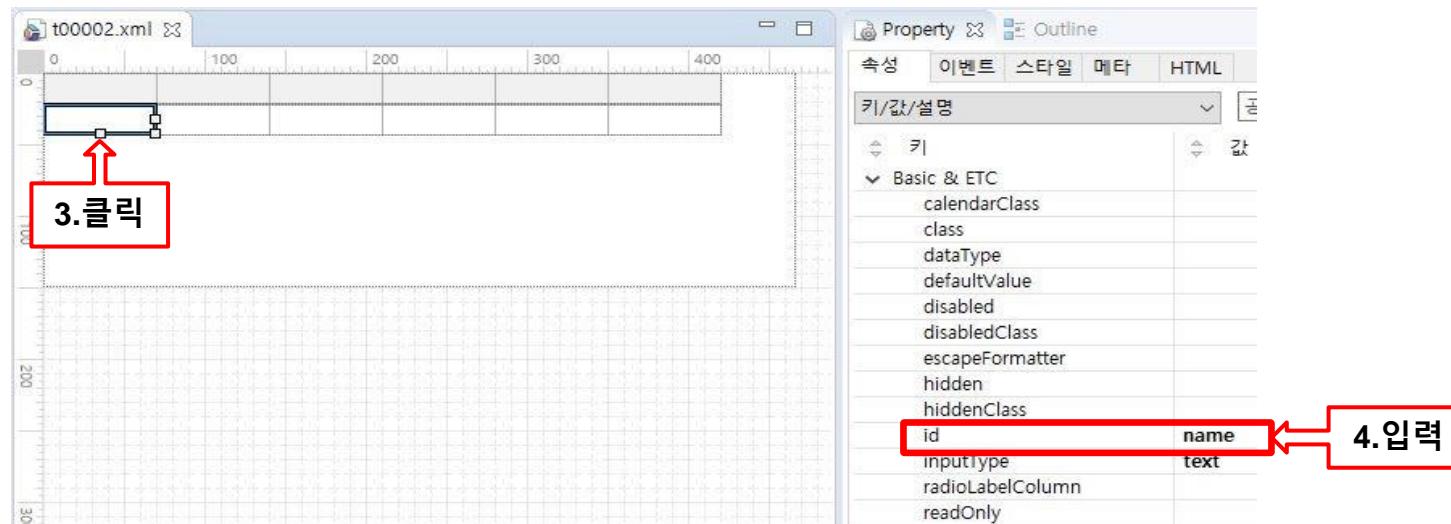
▪ 방법2

GridView컴포넌트 배치 합니다.(생성 방법 생략)

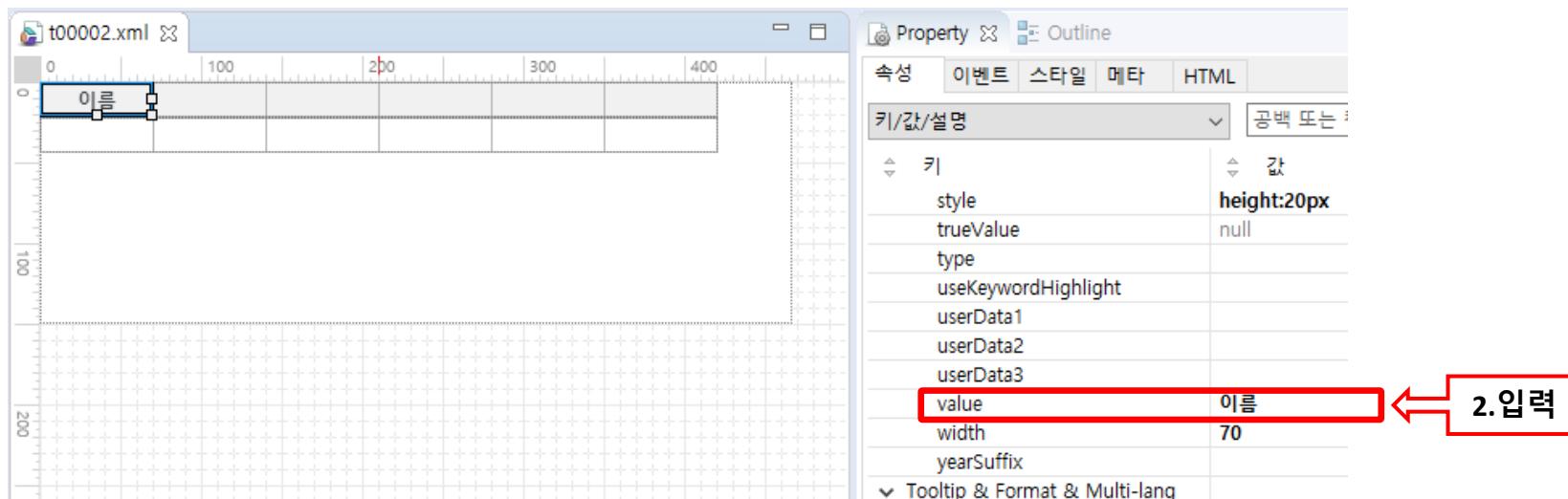
GridView를 클릭하고 Property의 dataList속성에 dataList의 id를 정의 합니다.



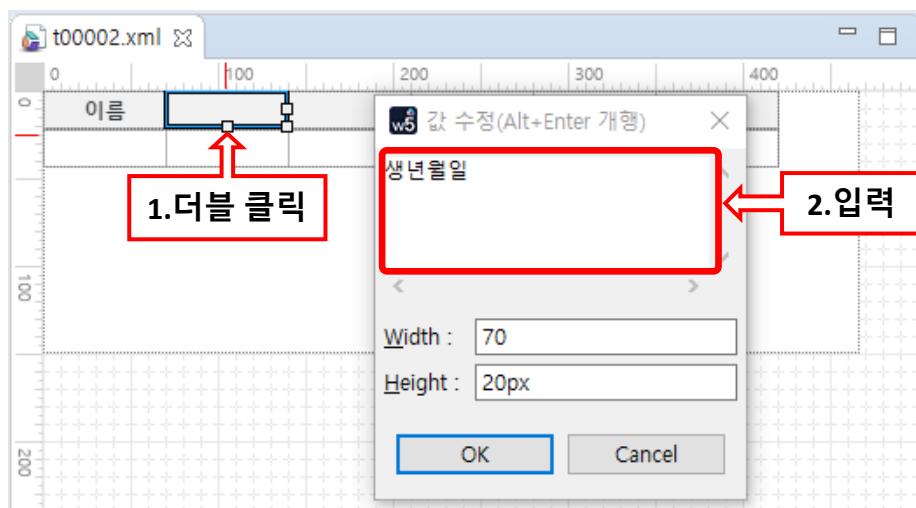
GridView의 body의 컬럼을 클릭하고 id속성을 dataList 컬럼의 id와 일치하게 정의 합니다.



GridView의 header의 컬럼을 클릭하고 value속성에 값을 입력 합니다.



또는 header의 컬럼을 더블 클릭하고 value속성에 값을 입력 합니다.



| 데이터 객체와 Submission

- Data 객체
- Submission 정의
- Submission 주요 구성 항목
- Submission 기타 구성 항목

■ 정의

브라우저의 메모리에 할당되는 JavaScript 데이터 객체.

■ 용도

서버와의 통신을 위한 request, response 데이터

화면에서 사용하고자 하는 임시 데이터

UI컴포넌트와 연동

■ 종류

DataMap : Key와 Value로 이루어진 단건 데이터 객체

DataList : Column과 Value로 이루어진 다건 데이터 객체

LinkedDataList : 정의된 dataList 객체를 참조하여 Filter, Sort를 적용한 다건 데이터 객체

AliasDataMap : Wframe의 Scope 사용한 경우, 자식 페이지에서 부모 페이지의 DataMap을 참조하는
Data 객체

AliasDataList : Wframe의 Scope 사용한 경우, 자식 페이지에서 부모 페이지의 dataList을 참조하는
Data 객체

■ 특징

각 데이터 객체의 id는 필수값입니다.

JSON, XML, JSON Array 포맷으로 설정(set)하거나 또는 반환(get) 받을 수 있습니다.

(LinkedDataList는 반환만 가능 합니다)

초기값을 정의할 수 있습니다. (LinkedDataList 제외)

동적(Script)으로 생성할 수 있습니다.

데이터의 상태(추가/수정/삭제)값을 관리 합니다.

- 서버와 데이터를 통신하는 모듈로 XHR로 구현되어 있습니다.

화면의 전환 없이 데이터만 주고 받습니다.

통신에 대한 동기/비동기 방식을 선택할 수 있습니다.

통신의 전,후 처리는 각 이벤트에 호출할 Function을 정의하여 실행 합니다.

- Request, Response데이터는 DataCollection에 정의한 데이터 객체와 연동 합니다.

Request 데이터는 reference에 정의 합니다. (소스에서는 ref 속성으로 표기 됩니다)

Response는 target에 정의 합니다.

- 용도별로 Submission을 등록하는 것을 권장 합니다. (유지보수 시 파악 용이)

- Script(동적)로 Submission을 생성할 수 있습니다.

화면의 코드성 데이터 통신은 공통 모듈 만들어 이용하는 것이 용이 합니다.

- 생성된 Submission은 ID를 제외한 속성을 동적으로 변경하여 사용할 수 있습니다.

안내 사항

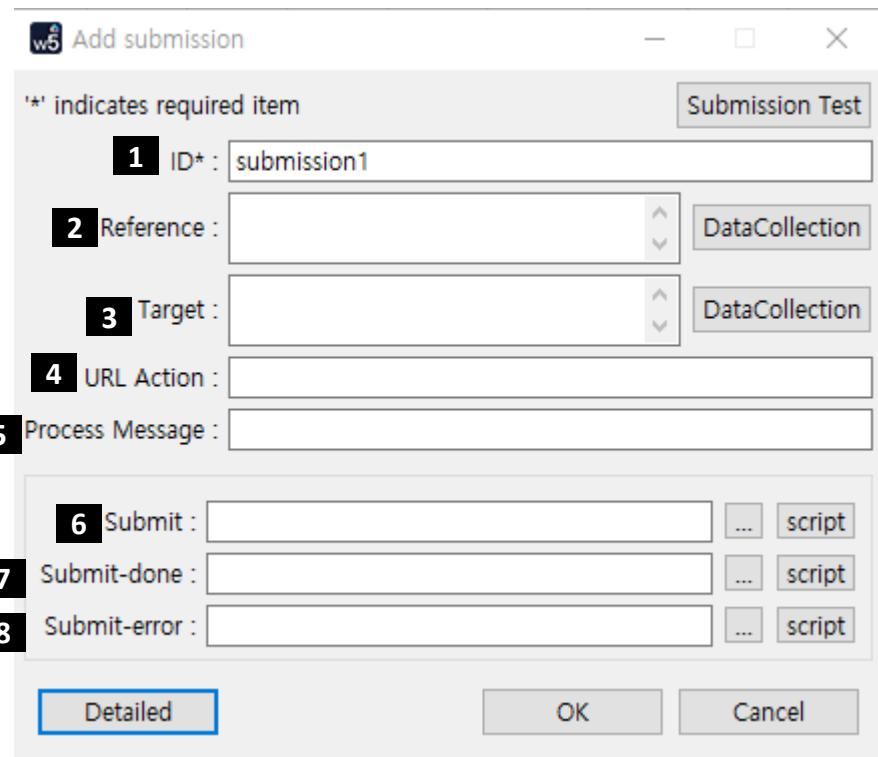
이번 장은 Submission에 대한 기본적인 내용을 담고 있습니다. 기본 구조를 이해하신 뒤 docs.inswave.com에서 세부 내용을 참조하시기 바랍니다.(Submission의 공통 처리, 통신 전후의 처리, 데이터 암복호화 등)

통신 데이터 포맷은 JSON을 기준으로 작성되어 있습니다.

본 장에서 언급되는 DataCollection이란 화면에 정의 된 DataMap, DataList, LinkedDataList, AliasDataMap, AliasDataList 객체를 포괄적으로 나타낸 단어임을 알려드립니다.

Submission 등록을 위한 주요 속성에 대한 설명입니다.

1. **ID** : [필수] submission을 구분하기 위한 ID.
2. **Reference** : 서버로 전송할 Request용 데이터 객체의 맵핑 정보.
3. **Target** : 서버에서 응답 받은 Response 데이터와 데이터 객체의 맵핑 정보.
4. **URL Action** : [필수] 통신을 위한 서버 URL.
5. **Process Message** : 사용자의 화면 제어를 방지하기 위해 서버 통신 중에 띄울 메시지 값.
6. **Submit** : 통신을 수행하기 전에 실행할 Function명. Function의 return 값(true/false)에 따라 실행/중지 시킬 수 있습니다.
7. **Submit-done** : 응답 상태 코드(ResponseStatusCode)가 2xx대의 성공(정상)일 경우 실행할 Function명.
8. **Submit-error** : 응답 상태 코드(ResponseStatusCode)가 200 미만 300 이상 대의 성공(정상)일 경우 실행할 Function명.



참고 사항

Reference (ref), Target 과 DataCollection을 연동하는 방법은 [DataCollection]버튼을 클릭하여 맵핑 합니다.
(다음 페이지 참조)

Reference와 Target에 연동할 DataCollection 설정 항목에 대한 설명입니다.

1. Type : [필수] 서버와 통신할 데이터의 포맷

2. ID : [필수] 정의된 데이터 객체의 ID

3. Action : DataList의 데이터를 상태(추가, 수정, 삭제 등)별로 전송

4. Key : 선택된 데이터 객체의 ID를 변경할 때

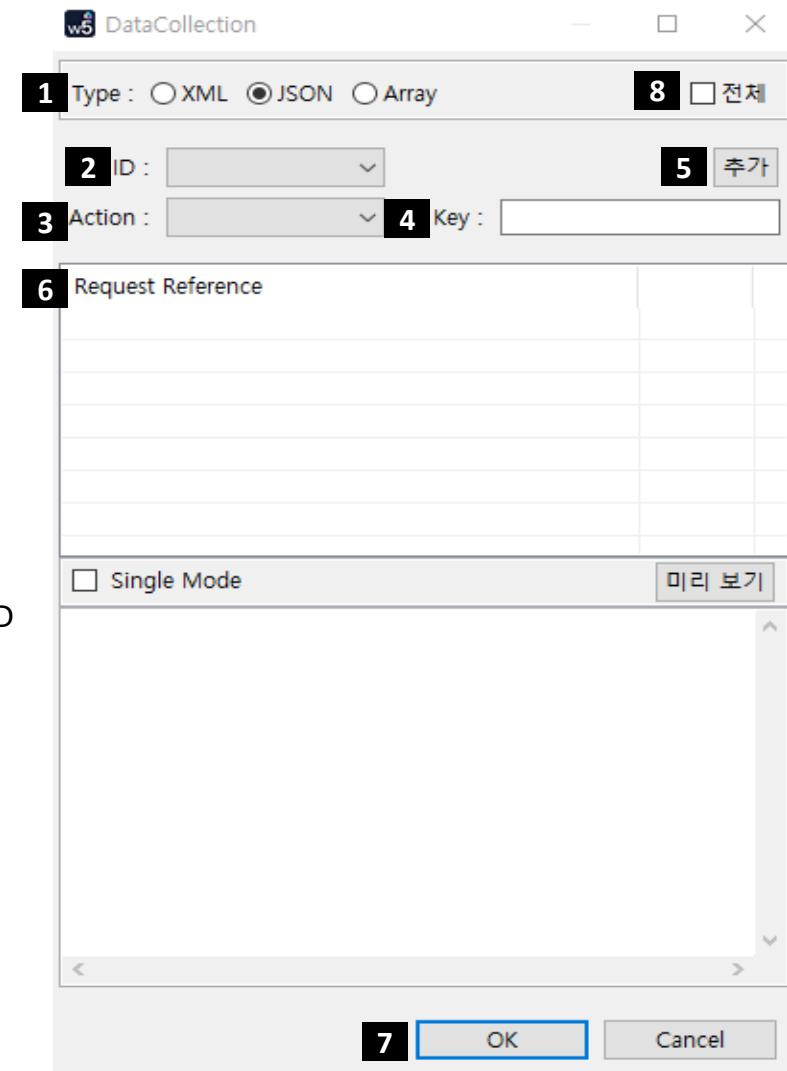
5. 추가 : 2~4번의 항목을 선택/입력한 뒤 적용

6. Request Reference : [추가]버튼을 클릭하여 적용한 데이터 객체 리스트

7. OK : 최종 반영

8. 전체

- reference창 : 전체 데이터를 서버로 보낼 때 사용 합니다.
- target창 : 설정할 경우는 서버에서 내려온 데이터 중 데이터 객체의 ID 와 동일한 KEY의 데이터를 자동으로 할당 합니다.



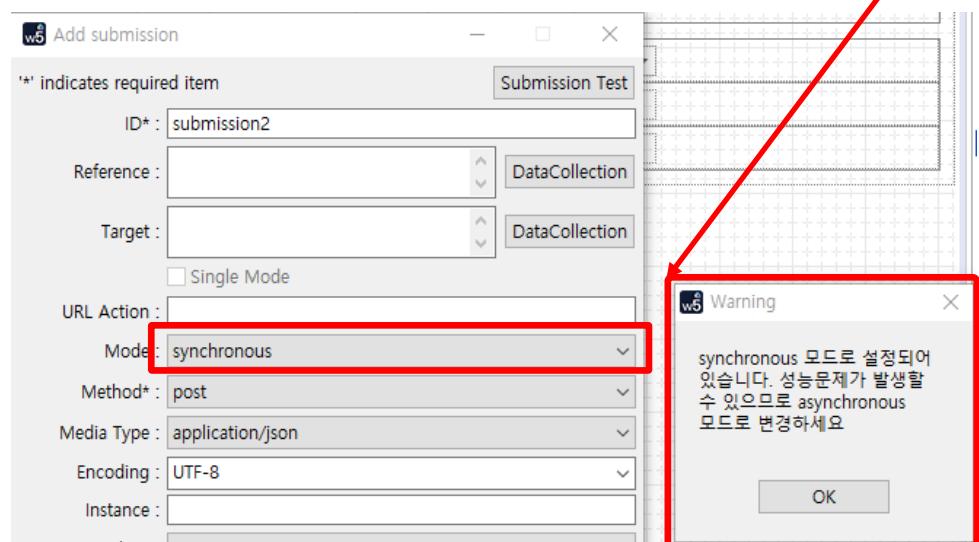
참고

2번 항목을 선택해야 3, 4번 항목을 적용할 수 있습니다.

입력된 데이터 객체는 5번 항목(추가 버튼)을 클릭해야 적용됩니다.

Submission의 기타 속성에 대한 설명입니다.

1. **Single Mode** : 최상위 {} 가 없이 하나의 Data 객체만 사용할 때 사용
2. **Mode** : 통신에 대한 실행 방식으로 asynchronous 사용합니다.
synchronous로 사용할 경우 'warning'이 발생합니다.
3. **Method** : Http Method (post를 가장 많이 사용)
4. **Media Type** : Request Header의 Content-Type과 Accept에 적용.
5. **Encoding** : Content-Type의 charset에 할당.
6. **Error Handler** : 통신 후 응답 데이터의 정상/에러 처리를 위한 Function명.
7. **Custom Handler** : request 정보를 server에 송신하기 직전에
수행처리하는 Function명



The screenshot shows the 'Add submission' dialog for 'submission1'. It includes fields for ID*, Reference, Target, URL Action, Mode, Method*, Media Type, Encoding, Instance, Replace, AbortTrigger, Error Handler, Custom Handler, Process Message, Submit, Submit-done, and Submit-error. The 'Mode' dropdown is highlighted with a red box and has a red arrow pointing to the 'Warning' dialog box.

| 씨버깅

- Debugging 방법
- 로그 보기
- 로그 출력
- Data Collection 보기
- 페이지 소스 보기
- 브라우저 별 개발자 도구 실행 방법

▪ WebSquare5 기능

브라우저에서 [Ctrl+마우스 오른쪽 버튼을 클릭]하면 컨텍스트 메뉴 팝업이 뜹니다.

해당 메뉴는 설정 파일(config.xml)을 통해 활성/비활성화 시킬 수 있습니다.

주요 디버깅 관련 메뉴는 아래와 같습니다.

로그보기

Script 오류를 확인하거나 Script에서 출력한 로그를 확인할 수 있습니다.

dataCollection보기

데이터 객체의 현재 값을 확인할 수 있습니다.

소스보기

WebSquare화면 소스를 확인할 수 있습니다.

현재 화면 디버그 실행

Submission을 이용한 통신의 Input, Output, Header 정보를 확인할 수 있습니다.

▪ 브라우저 개발자 도구(Chrome, IE 기준)

브라우저 실행 후 [F12]를 누르면 개발자 도구가 실행 됩니다.

또는 브라우저 메뉴를 통해 개발자 도구를 실행 시킬 수 있습니다.(개발자 도구 실행 관련 페이지 참조)

주요 디버깅 관련 메뉴는 아래와 같습니다.

Elements, DOM탐색기

브라우저에 그려진 HTML과 적용된 CSS 확인.

Console (콘솔)

WebSquare로그 또는 console 객체를 이용한 로그 확인.

Network (네트워크)

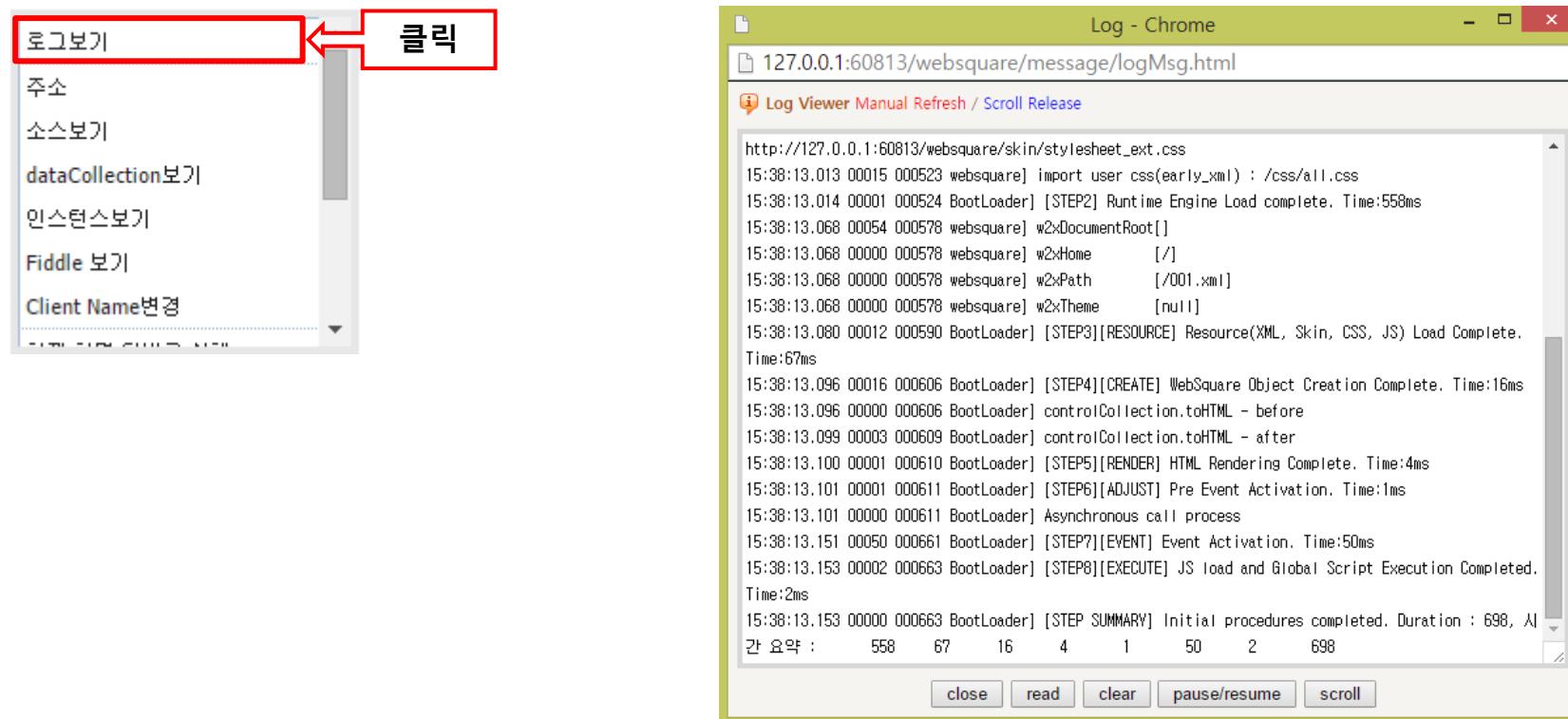
서버와 주고 받은 통신 확인.

- 웹스퀘어에서는 화면의 스크립트가 웹스퀘어 엔진 위에서 실행되기 때문에 화면의 로그가 (오류 메세지 및 사용자 출력) 별도로 관리 됩니다.

그래서 Script 오류 발생 시 오류 메시지가 브라우저 경고 창으로 표현되지 않고 로그 창에 출력됩니다.

- **로그 창 보는 방법 (기본)**

웹스퀘어 화면이 실행된 브라우저에서 Ctrl+마우스 오른쪽 버튼을 클릭한 뒤 "로그보기" 항목을 선택 합니다.

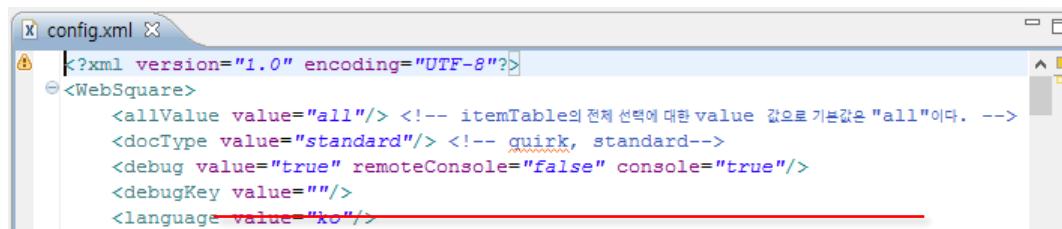


▪ 브라우저의 Console로 로그 보기

config.xml 설정

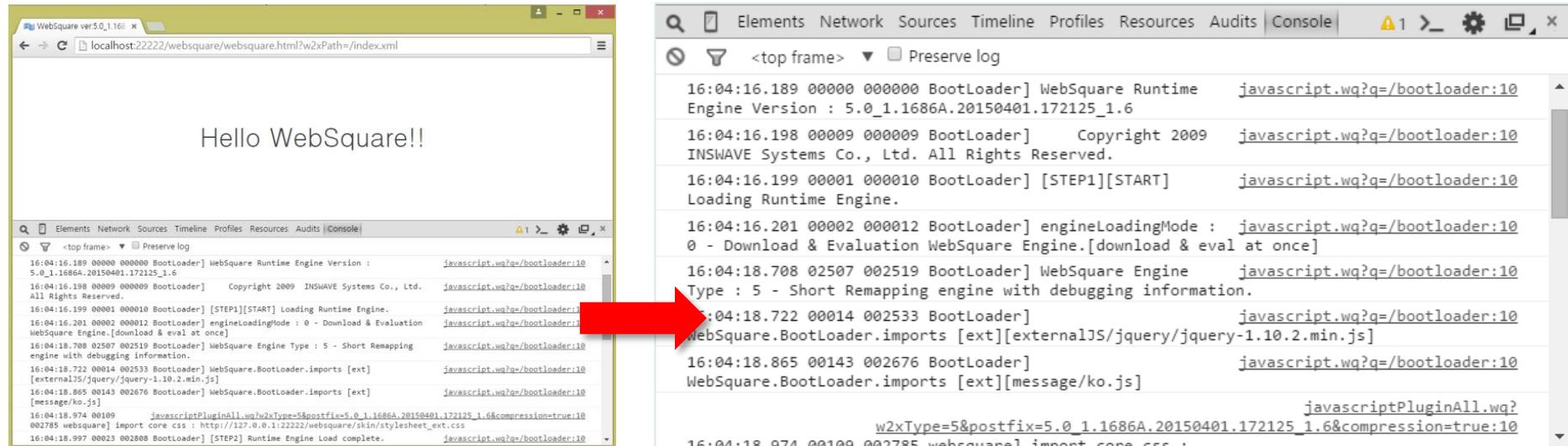
- 파일 경로 : /WebContent/websquare/config.xml
 - 설정 항목

```
<debug value="true" remoteConsole="false" console="true"/>
```



■ 실행하기

브라우저의 개발자 도구를 F12 단축키를 이용하여 실행한 뒤 Console 탭을 선택 합니다.



■ 기본 로그 출력하기

\$p.log 메소드를 이용 합니다.

ex) \$p.log(":::: WebSquare ::::");

■ 기본 로그 출력 화면 예시

```
16:22:58.547 00002 000548 websquare] :::: WebSquare :::: javascriptPluginAll.wq:10
```

■ 브라우저의 console 객체를 이용하여 로그 출력하기

브라우저에서 console 객체를 지원해야 하며 IE의 경우 IE8이상 부터 지원 됩니다.

console.log 메소드를 이용 합니다.

ex) console.log(":::: Console Log ::::");

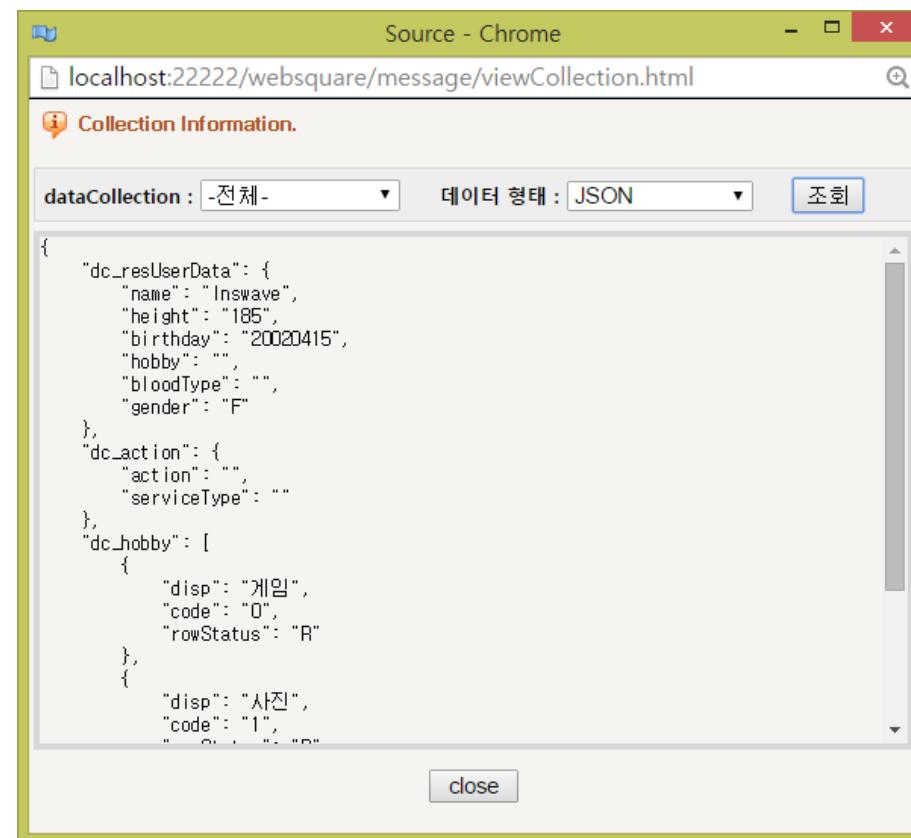
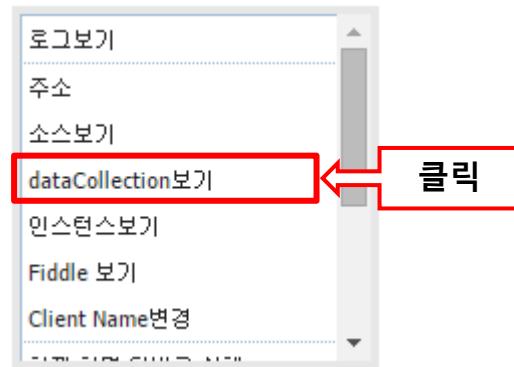
■ console 로그 출력 화면 예시

```
:::: Console Log :::: VM52:3
```

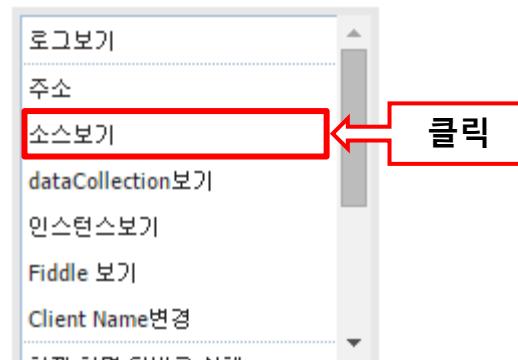
- 동적으로 변경되는 DataCollection의 경우 브라우저에서 실제 메모리에 할당된 데이터를 확인할 수 있습니다.

- **dataCollection 보기**

웹스퀘어 화면이 실행된 브라우저에서 **Ctrl+마우스 오른쪽 버튼**을 클릭한 뒤 "dataCollection 보기" 항목을 선택 합니다.



- 웹스퀘어 화면은 동적으로 생성되기 때문에 브라우저에서의 "페이지 소스보기"를 통해 볼 수가 없습니다.
- 화면 소스 보기
웹스퀘어 화면이 실행된 브라우저에서 Ctrl+마우스 오른쪽 버튼을 클릭한 뒤 "소스보기" 항목을 선택합니다.



Source - Chrome

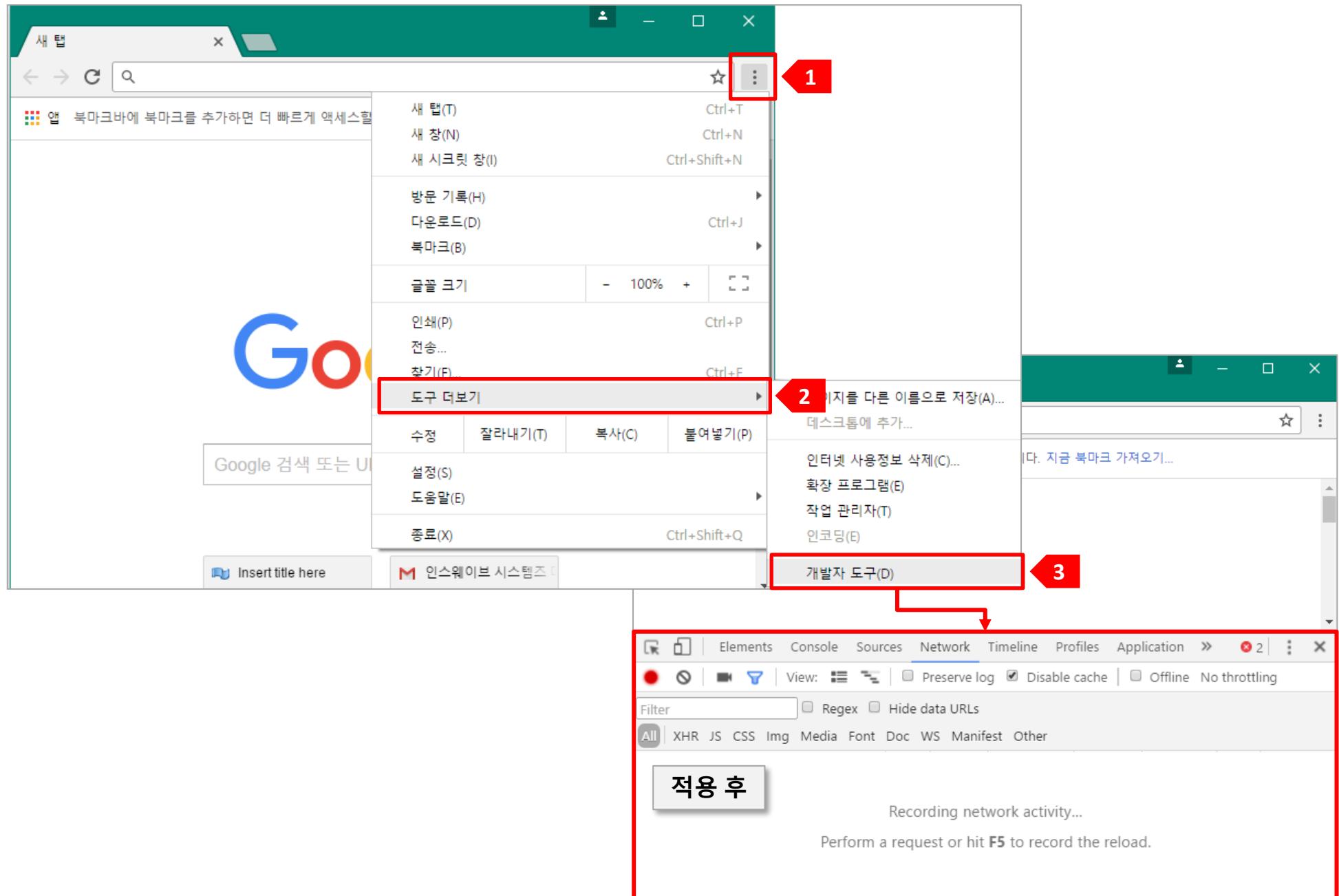
localhost:2222/websquare/message/viewSource.html

Source Information. SourceHighlight

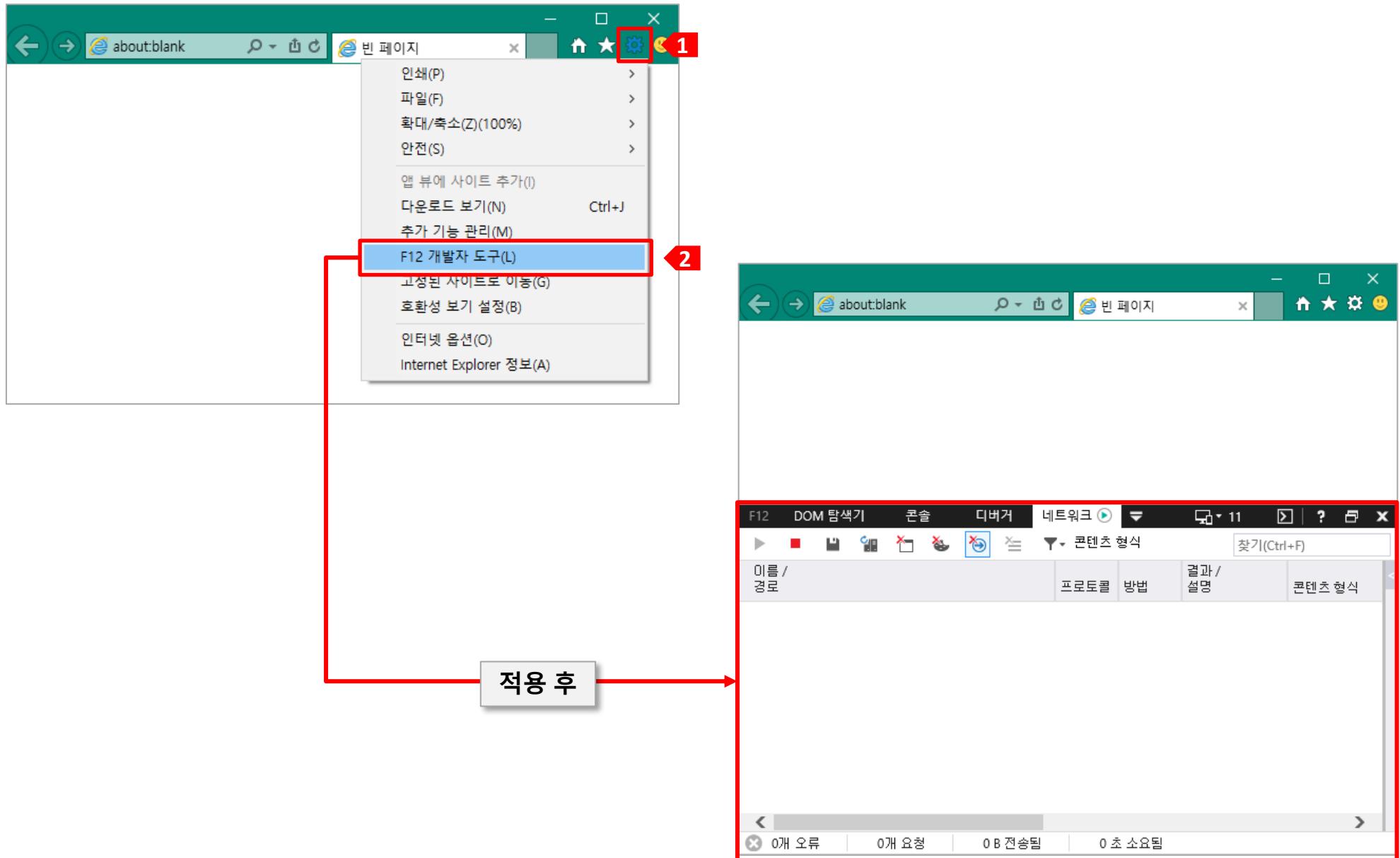
```
<?xml-stylesheet href="/css(btn.css" type="text/css"?><?xml-stylesheet href="/css/temp.css" type="text/css"?><html xmlns="http://www.w3.org/1999/xhtml" xmlns:ev="http://www.w3.org/2001/xml-events" xmlns:w="http://www.inswave.com/websquare" xmlns:xsi="http://www.w3.org/2002/xsi;xsi">
<head>
  <w:buildDate/>
  <f:mode>
    <f:instance>
      <data value="" />
    </f:instance>
    <w:dataCollection>
      <w:dataProvider id="dc_resUserdata" style="" valueAttribute="">
        <w:keyInfo>
          <w:key id="dc_resUserdata??dataList_name" name="name" dataType="text" col_id="name" />
          <w:key id="dc_resUserdata??dataList_height" name="height" dataType="text" col_id="height" />
          <w:key id="dc_resUserdata??dataList_birthday" name="birthday" dataType="text" col_id="birthday" />
          <w:key id="dc_resUserdata??dataList_hobby" name="hobby" dataType="text" col_id="hobby" />
          <w:key id="dc_resUserdata??dataList_bloodType" name="bloodType" dataType="text" col_id="bloodType" />
          <w:key id="dc_resUserdata??dataList_gender" name="gender" />
        </w:keyInfo>
      </w:dataProvider>
    </w:dataCollection>
  </f:mode>
</head>
<body>
</body>
</html>
```

close

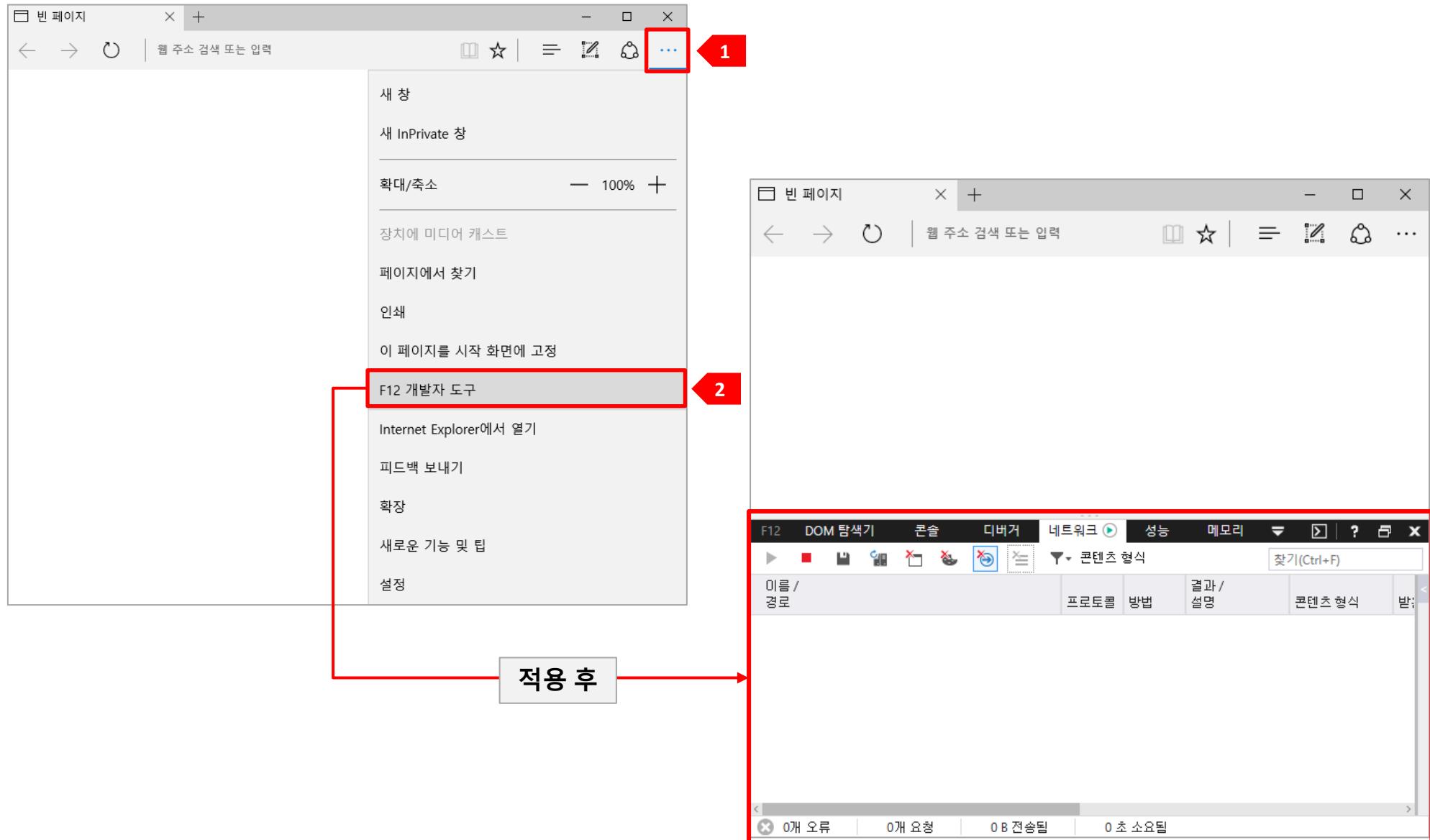
■ Chrome



■ IE11



■ Edge



감사합니다.