기존 소스코드들을 쭉 리버스 해보다 보니, 현재 아키텍처를 좀더 추상적으로 만들 필요가 있음을 너무 느꼈습니다.

이유는

첫째 자바스크립트 -> 서버 -> 데이터베이스에 이르는 메타데이터를 2중 3중으로 관리하는 비용이 높다.

둘째, 서버객체를 '표현'하는데 있어서의 로직을 클라이언트에서 html화 하고 있는데, 이 html 생성 방법이 너무 어렵다.

세째, 그리고 생성을 하는 로직 자체가 가능한 객체에 근접하게 응집도 높게 자리해야 하는데, 저기 자바스크립트 단 어딘가에 쿡 박혀있다.

결국 하고 싶은건, DWR 같이 메타데이터의 관리를 서버객체 자체로 두어 ajax/JSON기반의 Remoting 아키텍처로 쓰고 싶은것이다. 이미 GWT 등에서는 그렇게 하고 있으나, 초기 비용이 너무 높고, lock-in이 많다. 해서 우리의 현재 아키텍처에 어울리는 것을 만들고자 합니다.

만들고 싶은것:

[메타웍스2]

java의 "toString"을 모르는 분 없으십니다. 이 메시지는 자바세상에서 최소한도로 객체는 지 자신을 사람이 읽을 수 있도록 자신을 대변할 수 있어야 한다는 기본적 의미입니다. 때로, 자신을 표현할때, 객체는 라이프사이클을 경험하는데, 자신이 GUI 로 표현되고 싶을때는 고전적으로 Swing 객체등을 다음과 같이 리턴할 수 있어야 했습니다:

e.g.

```
Component comp = SomeObject.createDesigner();
```

그런데, 우리는 지금 웹기반의 아키텍처 상에서 객체들을 다루려고 하며, 특히나 서버상에 있는 객체를 바로 얻어내어서 이를 자바스크립트상에서 렌더링 하기를 바라며, 그 사이의 데이터는 최적화된 JSON 으로만 왔다갔다하길 바랍니다. 그러면서도 추상화되고 우아한 프로그래밍 모형을 포기하지 않기를 바랍니다. 해서, 다음과 같이 하고 싶다는 거죠….

```
<script type='text/javascript' src='../engine.js'></script>

<script type='text/javascript' src='../util.js'></script>

<script type='text/javascript' src='../interface/AddressBook.js'></script>
<script type='text/javascript' src='../interface/Person.js'></script>


<script>

    Person person = AddressBook.getPersonByName("장진영");
    $("user").html(person.toHtml());

</script>

<div id="user"/>
```

이렇게 하면 person.toHtml()은 이쁘게 사용자의 사진, 사진이하에 이름, 사진을 클릭하면 나올 세부정보까지를 이쁘게 포장하여 html로 뱉어주기를 원하는 겁니다. 꼭 페이스북에서 어디에 사용자 이름의 링크가 존재하던, 이를 클릭하면 세부정보가 오버레이되어 뿌려지듯이요… 물론, 상황별로 출력되는 html 은 다음과 같이 약간의 표준화된 옵션 값에 의하여 적절히 대응해주면 좋습니다:

```
person.toHtml({"in-table", "view"}) --- returns --> <tr><td> Jinyoung Jang </td> <td> 35 </td> <td> jyjang@xxx.com</td> </tr>
person.toHtml({"edit"}) --- returns --> name: <input value="Jinyoung Jang" name="Person.hash001.name"> age: <input value="35" name="Person.hash001.age"> emailaddress: <input value="jyjang@xxx.com" name="Person.hash001.emailAddress">
```

그리고 이러한 것은 재정의 되기 전에 기본적으로 자동 생성되는 로직을 생성할 수 있습니다.

이것은 메타데이터 중심의 어플리케이션 개발 전략과 일맥상통합니다 (참조: metaworks.sf.net)

좀 더 길게 봐서 다양한 자바스크립트 컨트롤들을 이렇게 생성된 DWR객체들과 호환되게할 수 있습니다. 예를 들어:

```
someGridControl.addData (person);
….
```

이러한 로직은 알아서 person객체의 메타데이터에 접근하여 적절히 이를 입력받고, 수정하고, 저장할 수 있습니다. 물론, 특별한 코딩이 없이도 person 객체가 저장될 서버로직과도 호환되어 동작할 수 있어야 합니다. 이를 위해서는 객체 자체에 자신의 라이프사이클상에서 자신이 어떻게 생성되어서 소멸될때까지의 로직을 갖추어야 하는가를 자신이 갖고 다녀야 합니다.

그러면 다음과 같은 표현까지 가능해집니다:

<자바스크립트에서>

```
persons = AddressBook.findPersons("jinyoung%");
someGridControl.addData (persons);
```

여기서 someGridControl은 persons가 서버상에 저장되어야 할 시점에 person에게 "save"를 요청하고, person은 자신을 어떻게 저장할 수 있는지를 서버객체에게 요청합니다. 서버객체는 자신이 어떻게 저장되어야 할지에 대한 로직을 자신이 갖고 있어야 하며, 그것은 다음의 표현을 명시적으로 someGridControl에 리스너등으로 사용하지 않아도 되는 상황을 열어준다는 것입니다:

```
AddressBook.savePerson(person);
```

정리하자면, 객체는 자기 자신으로서 자신을 표현, 입력, 저장할 수 있어야 하고, 해당 객체가 서버에 있든, 클라이언트에 있든, 심지어 자바스크립트로 존재하더라도 이것이 가능해야 한다는 것입니다.

이를 구현하기 위해서
먼저 DWR (Direct Web Remoting)과 스프링을 혼합하고, jQuery의 몇가지 컨트롤들을 사용하려고 합니다.

〔DWR 이해하기〕

다음과 같은 서버객체를 생성된 매핑 자바스크립트 객체를 이용하여 접근할 수 있는 일종의 Stub/Skeleton을 생성하여 준다:

Java Server:

```
public interface ACLManager{

    Hashtable<String> getPermission(String objectId);

}
```

JavaScript Client:

```
<script type="text/javascript"
    src="[WEBAPP]/dwr/interface/ACLManager.js"> </script>
<script type="text/javascript"
    src="[WEBAPP]/dwr/engine.js"> </script>
...
permissions = ACLManager.getPermission("42");
```

여기서 DWR이 하는일은 ACLManager.js 를 리얼타임 생성해주는 일을 하고 추측하시다시피 ACLManager.js의 인터페이스는 서버단의 ACLManager.java의 javascript mapping 버전을 생성하고, 구현체는 서버의 ACLManager를 호출하는 ajax call 및 리턴타입 unmarshalling 구현체를 생성해주는 일을 한다.

여기서 필요한건 Spring서버객체와 연결해서 현재 프로세스 코디를 그대로 이용하면서 기존의 Spring MVC를 사용하지 않고 Spring AOP만을 사용코자 함이다. (물론, 기존 접근도 좋다. 나쁘다는게 아니다.. DWR은 스텁을 매번 생성하기 때문에 성능이 않좋다는 이야기가 있다. 일단 스텁(맞나?)을 추후에는 캐싱시켜 성능을 높힐 작정이다)

〔구현〕

일단, DWR사이트에 가서 최신 버전인 DWR3.0을 다운받고 설정하고 할려고 봤더니, 이미 프로세스 코디 web.xml에 이렇게 이쁘게 세팅이 되어있다:

#web.xml

```
    <!--
    servlet
    Configuration for a dwr servlet
    -->
    <servlet xmlns="">
        <servlet-name>dwr-invoker</servlet-name>
        <!-- display-name>DWR Servlet</display-name -->
        <!--<servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>-->
               <servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>

        <init-param>
            <param-name>debug</param-name>
            <param-value>true</param-value>
        </init-param>
        <!-- Remove this unless you want to use active reverse ajax -->
                <init-param>
                  <param-name>activeReverseAjaxEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- By default DWR creates application scope objects when they are first
                used. This creates them when the app-server is started -->
                <init-param>
                  <param-name>initApplicationScopeCreatorsAtStartup</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- WARNING: allowing JSON-RPC connections bypasses much of the security
                protection that DWR gives you. Take this out if security is important -->
                <init-param>
                  <param-name>jsonRpcEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- WARNING: allowing JSONP connections bypasses much of the security
                protection that DWR gives you. Take this out if security is important -->
                <init-param>
                  <param-name>jsonpEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- data: URLs are good for small images, but are slower, and could OOM for
                larger images. Leave this out (or keep 'false') for anything but small images -->
                <init-param>
                  <param-name>preferDataUrlSchema</param-name>
                  <param-value>false</param-value>
                </init-param>

                <!-- This enables full streaming mode. It's probably better to leave this
                out if you are running across the Internet -->
```

```
            <init-param>
              <param-name>maxWaitAfterWrite</param-name>
              <param-value>-1</param-value>
            </init-param>

            <!--
            For more information on these parameters, see:
            - http://getahead.org/dwr/server/servlet
            - http://getahead.org/dwr/reverse-ajax/configuration
            -->
            <load-on-startup>1</load-on-startup>
    </servlet>


    <!--
    servlet-mapping
    Specifies the mapping between a servlet and URL pattern
    -->
    <servlet-mapping xmlns="">
        <servlet-name>dwr-invoker</servlet-name>
        <url-pattern>/dwr/*</url-pattern>
    </servlet-mapping>
```

숫가락만 얹으면 되는군… (기존에 설정되어 있는 이유는 okmindmap 서비스가 롱-폴링 구현을 위하여 DWR의 callback remoting을 일부사용하기 때문이다)

이제, spring과 연결하기 전에 기존에는 어떻게 dwr을 okmindmap이 사용하고 있는지를 살펴보았다:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dwr PUBLIC "-//GetAhead Limited//DTD Direct Web Remoting 3.0//EN" "http://getahead.org/dwr/dwr30.dtd">


<dwr>

  <allow>

    <!--
    <filter class="com.jinotech.dwr.monitor.MonitoringAjaxFilter"/>
    <filter class="org.directwebremoting.filter.ExtraLatencyAjaxFilter">
      <param name="delay" value="200"/>
    </filter>
    -->
<!-- chat -->
    <create creator="new" javascript="JavascriptChat" scope="page">
      <param name="class" value="com.okmindmap.collaboration.JavascriptChat"/>
    </create>

  </allow>

</dwr>
```

추측대로, 채팅서비스를 위해서만 사용하고 있음을 알 수 있다.

이, dwr.xml이 세팅된 것을 DWR은 읽어서 JavascriptChat.js를 생성해줄것이다. (최신 버전은 서버 클래스에 annotation만 주면 알아서 만들어 준다하니 많은 발전을 한 모양이다)

http://localhost:8080/uengine-web/dwr/index.html 을 접속해보니 등록된 서버객체들을 확인할 수 있네..

Classes known to DWR:

- JavascriptChat (com.okmindmap.collaboration.JavascriptChat)

으로 뜬다. 오호.. 링크를 클릭해보니… 다음과 같이 뜬다:

Methods For: JavascriptChat (com.okmindmap.collaboration.JavascriptChat)

To use this class in your javascript you will need the following script includes:

```
  <script type='text/javascript' src='/uengine-web/dwr/interface/JavascriptChat.js'></script>
  <script type='text/javascript' src='/uengine-web/dwr/engine.js'></script>
```

In addition there is an optional utility script:

```
  <script type='text/javascript' src='/uengine-web/dwr/util.js'></script>
```

Replies from DWR are shown with a yellow background if they are simple or in an alert box otherwise.
The inputs are evaluated as Javascript so strings must be quoted before execution.

```
emptyQueue( "" );
getUserName( );
```

(Warning: No Converter for org.directwebremoting.WebContext. See below)

으흠… 이거 좋네.. 알아서 어떻게 스크립트에서 호출할 수 있을지까지..

.js를 클릭해보니.. 다음과 같이 생성된

```
// Provide a default path to dwr.engine
if (typeof this['dwr'] == 'undefined') this.dwr = {};
if (typeof dwr['engine'] == 'undefined') dwr.engine = {};
if (typeof dwr.engine['_mappedClasses'] == 'undefined') dwr.engine._mappedClasses = {};

if (window['dojo']) dojo.provide('dwr.interface.JavascriptChat');

if (typeof this['JavascriptChat'] == 'undefined') JavascriptChat = {};

JavascriptChat._path = '/uengine-web/dwr';

/**
 * @param {class java.lang.String} p0 a param
 * @param {function|Object} callback callback function or options object
 */
JavascriptChat.emptyQueue = function(p0, callback) {
  return dwr.engine._execute(JavascriptChat._path, 'JavascriptChat', 'emptyQueue', arguments);
```

```
};

/**
 * @param {class java.lang.String} p0 a param
 * @param {class java.lang.String} p1 a param
 * @param {class java.lang.String} p2 a param
 * @param {class java.lang.String} p3 a param
 * @param {function|Object} callback callback function or options object
 */
JavascriptChat.insertNode = function(p0, p1, p2, p3, callback) {
  return dwr.engine._execute(JavascriptChat._path, 'JavascriptChat', 'insertNode', arguments);
};
```

…

예측해던대로 되어있다.. 보통 원격객체서버를 만들때 접근하는 방식으로, 클라이언트 인터페이스 스텁이 스텁 엔진을 제네릭한 방법으로 호출하도록 매핑하는 방식으로 코딩한다. DWR이 전형적인 방식으로 구현되었음을 알 수 있다. 보나마나, `dwr.engine._execute`은 ajax호출을 하는 모듈이 들어있을 것이다:

```
  dwr.engine._execute = function(path, scriptName, methodName, args) {
    var singleShot = false;
    if (dwr.engine._batch == null) {
      dwr.engine.beginBatch();
      singleShot = true;
    }

    var batch = dwr.engine._batch;
    // All the paths MUST be to the same servlet
    if (batch.path == null) {
      batch.path = path;
    }
    else {
      if (batch.path != path) {
        dwr.engine._handleError(batch, { name:"dwr.engine.multipleServlets", message:"Can't batch requests to multiple DWR Servlets." });
        return;
      }
    }

    dwr.engine.batch.addCall(batch, scriptName, methodName, args);

    // Now we have finished remembering the call, we increment the call count
    batch.map.callCount++;
    if (singleShot) {
      return dwr.engine.endBatch();
    }
  };


  /**
   * Finished grouping a set of remote calls together. Go and execute them all.
   * @param {Object} options A options object to customize processing
   * @see getahead.org/dwr/browser/engine/batch
   */
  dwr.engine.endBatch = function(options) {
    var batch = dwr.engine._batch;
    if (batch == null) {
      dwr.engine._handleError(null, { name:"dwr.engine.batchNotBegun", message:"No batch in progress" });
      return;
    }
    dwr.engine._batch = null;
    if (batch.map.callCount == 0) {
      return;
    }

    // The hooks need to be merged carefully to preserve ordering
    if (options) {
      dwr.engine.batch.merge(batch, options);
    }

    // In ordered mode, we don't send unless the list of sent items is empty
    if (dwr.engine._ordered && dwr.engine._batchesLength != 0) {
      dwr.engine._batchQueue[dwr.engine._batchQueue.length] = batch;
    }
    else {
      return dwr.engine.transport.send(batch);
    }
  };


  dwr.engine.transport = {
    /**
     * Actually send the block of data in the batch object.
     * @private
     * @param {Object} batch
     */
    send:function(batch) {
      dwr.engine.batch.prepareToSend(batch);

      // Work out if we are going cross domain
      var isCrossDomain = false;
      if (batch.path == null) {
        batch.path = dwr.engine._pathToDwrServlet;
      }
      if (batch.path.indexOf("http://") == 0 || batch.path.indexOf("https://") == 0) {
        var dwrShortPath = dwr.engine._pathToDwrServlet.split("/", 3).join("/");
        var hrefShortPath = window.location.href.split("/", 3).join("/");
        isCrossDomain = (dwrShortPath != hrefShortPath);
      }

      if (batch.fileUpload) {
        if (isCrossDomain) {
          throw new Error("Cross domain file uploads are not possible with this release of DWR");
        }
        batch.transport = dwr.engine.transport.iframe;
      }
      else if (isCrossDomain && !dwr.engine.isJaxerServer) {
        batch.transport = dwr.engine.transport.scriptTag;
      }
      // else if (batch.isPoll && dwr.engine.isIE) {
```

```
    //   batch.transport = dwr.engine.transport.htmlfile;
    // }
    else {
      batch.transport = dwr.engine.transport.xhr;
    }

    return batch.transport.send(batch);
  }
```

음… 호출 또한 여러 상황 대비하여 `fileupload` 상황, `Jaxer`서버상황, `xhr`상황 등에 따라서 동작하도록 되어있다: 그러면 기본 상황인 xhr을 볼까..

```
  /**
   * Remoting through XHR
   */
  xhr:{
    /**
     * The default HTTP method to use
     */
    httpMethod:"POST",

    /**
     * The ActiveX objects to use when we want to do an XMLHttpRequest call.
     * TODO: We arrived at this by trial and error. Other toolkits use
     * different strings, maybe there is an officially correct version?
     */
    XMLHTTP:["Msxml2.XMLHTTP.6.0", "Msxml2.XMLHTTP.5.0", "Msxml2.XMLHTTP.4.0", "MSXML2.XMLHTTP.3.0", "MSXML2.XMLHTTP", "Microsoft.XMLHTTP"],

    /**
     * Setup a batch for transfer through XHR
     * @param {Object} batch The batch to alter for XHR transmit
     */
    send:function(batch) {
      if (batch.isPoll) {
        batch.map.partialResponse = dwr.engine._partialResponseYes;
      }

      // Do proxies or IE force us to use early closing mode?
      if (batch.isPoll && dwr.engine._pollWithXhr == "true") {
        batch.map.partialResponse = dwr.engine._partialResponseNo;
      }
      if (batch.isPoll && dwr.engine.isIE) {
        batch.map.partialResponse = dwr.engine._partialResponseNo;
      }

      if (window.XMLHttpRequest) {
        batch.req = new XMLHttpRequest();
      }
      else if (window.ActiveXObject) {
        batch.req = dwr.engine.util.newActiveXObject(dwr.engine.transport.xhr.XMLHTTP);
      }

      // Proceed using XMLHttpRequest
      if (batch.async == true) {
        batch.req.onreadystatechange = function() {
          if (typeof dwr != 'undefined') {
            dwr.engine.transport.xhr.stateChange(batch);
          }
        };
      }

      // If we're polling, record this for monitoring
      if (batch.isPoll) {
        dwr.engine._pollReq = batch.req;
        // In IE XHR is an ActiveX control so you can't augment it like this
        if (!dwr.engine.isIE) batch.req.batch = batch;
      }

      httpMethod = dwr.engine.transport.xhr.httpMethod;

      // Workaround for Safari 1.x POST bug
      var indexSafari = navigator.userAgent.indexOf("Safari/");
      if (indexSafari >= 0) {
        var version = navigator.userAgent.substring(indexSafari + 7);
        if (parseInt(version, 10) < 400) {
          if (dwr.engine._allowGetForSafariButMakeForgeryEasier == "true") {
            httpMethod = "GET";
          }
          else {
            dwr.engine._handleWarning(batch, {
              name: "dwr.engine.oldSafari",
              message: "Safari GET support disabled. See getahead.org/dwr/server/servlet and allowGetForSafariButMakeForgeryEasier."
            });
          }
        }
      }

      batch.mode = batch.isPoll ? dwr.engine._ModePlainPoll : dwr.engine._ModePlainCall;
      var request = dwr.engine.batch.constructRequest(batch, httpMethod);

      try {
        batch.req.open(httpMethod, request.url, batch.async);
        try {
          for (var prop in batch.headers) {
            var value = batch.headers[prop];
            if (typeof value == "string") {
              batch.req.setRequestHeader(prop, value);
            }
          }
          if (!batch.headers["Content-Type"]) {
            batch.req.setRequestHeader("Content-Type", "text/plain");
          }
        }
        catch (ex) {
          dwr.engine._handleWarning(batch, ex);
        }
        batch.req.send(request.body);
        if (batch.async == false) {
          dwr.engine.transport.xhr.stateChange(batch);
```

```
        }
      }
      catch (ex) {
        dwr.engine._handleError(batch, ex);
      }

      if (batch.isPoll && batch.map.partialResponse == dwr.engine._partialResponseYes) {
        dwr.engine.transport.xhr.checkCometPoll();
      }

      // This is only of any use in sync mode to return the reply data
      return batch.reply;
    },

    /**
     * Called by XMLHttpRequest to indicate that something has happened
     * @private
     * @param {Object} batch The current remote operation
     */
    stateChange:function(batch) {
      var toEval;

      if (batch.completed) {
        dwr.engine._debug("Error: _stateChange() with batch.completed");
        return;
      }

      var req = batch.req;
      try {
        var readyState = req.readyState;
        var notReady = (req.readyState != 4);
        if (notReady) {
          return;
        }
      }
      catch (ex) {
        dwr.engine._handleWarning(batch, ex);
        // It's broken - clear up and forget this call
        dwr.engine.batch.remove(batch);
        return;
      }

      if (dwr.engine._unloading && !dwr.engine.isJaxerServer) {
        dwr.engine._debug("Ignoring reply from server as page is unloading.");
        return;
      }

      try {
        var reply = req.responseText;
        reply = dwr.engine._replyRewriteHandler(reply);
        var status = req.status; // causes Mozilla to except on page moves

        if (reply == null || reply == "") {
          dwr.engine._handleWarning(batch, { name:"dwr.engine.missingData", message:"No data received from server" });
        }
        else if (status != 200) {
          dwr.engine._handleError(batch, { name:"dwr.engine.http." + status, message:req.statusText });
        }
        else {
          var contentType = req.getResponseHeader("Content-Type");
          if (dwr.engine.isJaxerServer) {
            // HACK! Jaxer does something b0rken with Content-Type
            contentType = "text/javascript";
          }
          if (!contentType.match(/^text\/plain/) && !contentType.match(/^text\/javascript/)) {
            if (contentType.match(/^text\/html/) && typeof batch.textHtmlHandler == "function") {
              batch.textHtmlHandler({ status:status, responseText:reply, contentType:contentType });
            }
            else {
              dwr.engine._handleWarning(batch, { name:"dwr.engine.invalidMimeType", message:"Invalid content type: '" + contentType +
"'" });
            }
          }
          else {
            // Comet replies might have already partially executed
           if (batch.isPoll && batch.map.partialResponse == dwr.engine._partialResponseYes) {
              dwr.engine.transport.xhr.processCometResponse(reply, batch);
            }
            else {
              if (reply.search("//#DWR") == -1) {
                dwr.engine._handleWarning(batch, { name:"dwr.engine.invalidReply", message:"Invalid reply from server" });
              }
              else {
                toEval = reply;
              }
            }
          }
        }
      }
      catch (ex) {
        dwr.engine._handleWarning(batch, ex);
      }

      dwr.engine._callPostHooks(batch);

      // Outside of the try/catch so errors propagate normally:
      dwr.engine._receivedBatch = batch;
      if (toEval != null) toEval = toEval.replace(dwr.engine._scriptTagProtection, "");
      dwr.engine._eval(toEval);
      dwr.engine._receivedBatch = null;
      dwr.engine.batch.validate(batch);
      if (!batch.completed) dwr.engine.batch.remove(batch);
    },

    /**
     * Check for reverse Ajax activity
     * @private
     */
    checkCometPoll:function() {
      if (dwr.engine._pollReq) {
        var req = dwr.engine._pollReq;
```

```
          var text = req.responseText;
          if (text != null) {
            dwr.engine.transport.xhr.processCometResponse(text, req.batch);
          }
        }

        // If the poll resources are still there, come back again
        if (dwr.engine._pollReq) {
          setTimeout(dwr.engine.transport.xhr.checkCometPoll, dwr.engine._pollCometInterval);
        }
      },

      /**
       * Some more text might have come in, test and execute the new stuff.
       * This method could also be called by the iframe transport
       * @private
       * @param {Object} response from xhr.responseText
       * @param {Object} batch The batch that the XHR object pertains to
       */
      processCometResponse:function(response, batch) {
        if (batch.charsProcessed == response.length) return;
        if (response.length == 0) {
          batch.charsProcessed = 0;
          return;
        }

        var firstStartTag = response.indexOf("//#DWR-START#", batch.charsProcessed);
        if (firstStartTag == -1) {
          // dwr.engine._debug("No start tag (search from " + batch.charsProcessed + "). skipping '" +
response.substring(batch.charsProcessed) + "'");
          batch.charsProcessed = response.length;
          return;
        }
        // if (firstStartTag > 0) {
        //   dwr.engine._debug("Start tag not at start (search from " + batch.charsProcessed + "). skipping '" +
response.substring(batch.charsProcessed, firstStartTag) + "'");
        // }

        var lastEndTag = response.lastIndexOf("//#DWR-END#");
        if (lastEndTag == -1) {
          // dwr.engine._debug("No end tag. unchanged charsProcessed=" + batch.charsProcessed);
          return;
        }

        // Skip the end tag too for next time, remembering CR and LF
        if (response.charCodeAt(lastEndTag + 11) == 13 && response.charCodeAt(lastEndTag + 12) == 10) {
         batch.charsProcessed = lastEndTag + 13;
        }
        else {
          batch.charsProcessed = lastEndTag + 11;
        }

        var exec = response.substring(firstStartTag + 13, lastEndTag);

        try {
          dwr.engine._receivedBatch = batch;
          dwr.engine._eval(exec);
          dwr.engine._receivedBatch = null;
        }
        catch (ex) {
          // This is one of these annoying points where we might be executing
          // while the window is being destroyed. If dwr == null, bail out.
          if (dwr != null) {
            dwr.engine._handleError(batch, ex);
          }
        }
      },

      /**
       * Tidy-up when an XHR call is done
       * @param {Object} batch
       */
      remove:function(batch) {
        // XHR tidyup: avoid IE handles increase
        if (batch.req) {
          // If this is a poll frame then stop comet polling
          if (batch.req == dwr.engine._pollReq) dwr.engine._pollReq = null;
          delete batch.req;
        }
      }
    },
```

음결국 특별한 라이브러리 없이 직접적으로

```
if (window.XMLHttpRequest) {
          batch.req = new XMLHttpRequest();
        }
        else if (window.ActiveXObject) {
          batch.req = dwr.engine.util.newActiveXObject(dwr.engine.transport.xhr.XMLHTTP);
        }
```

을 통해 AJAX호출을 하고 있다.

스텁 부분에 대한 분석은 간단히 끝났다. 이제 서버단 설정을 어떻게 연결할지를 살펴보자:

〔Spring 과 통합〕

참조: http://directwebremoting.org/dwr/documentation/server/integration/spring.html

먼저 스프링과 통합을 위해서 다음과 같이 스프링 MVC dispatcher가 dwr영역에도 영향을 줄 수 있도록 해준다:

#web.xml

```
    <servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/applicationContext.xml</param-value>
        </init-param>
        <load-on-startup>1</load-on-startup>
    </servlet>
    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
    </servlet-mapping>

-------  추가 ----------

    <servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/dwr/*</url-pattern>
    </servlet-mapping>
```

그런 후에 applicationContext 설정에서 DWR로 리모팅 되길 원하는 빈이 있다면 다음과 같이 해준다:

#applicatoinContext.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:aop="http://www.springframework.org/schema/aop"
    xmlns:dwr="http://www.directwebremoting.org/schema/spring-dwr"   <-------------- 추가 a
….
>

    <dwr:controller id="dwrController" debug="true" />    <---------------- 추가 b
    ….


    <bean id="codiProcessManagerBean" class="org.uengine.codi.processmanager.CodiProcessManagerBean"
        scope="request">
        <property name="managedTransaction" value="true" />
        <property name="autoCloseConnection" value="false" />
        <property name="connectionFactory" ref="springConnectionFactory" />
        <aop:scoped-proxy />

        <dwr:remote javascript="AjaxTimeConvert">     <------------------- 추가 c
           <dwr:include method="*" />
        </dwr:remote>
    </bean>
```

…삽질 & 삽질

삽질1: 네임스페이스에 xsi:schemaLocation 를 주지 않아서 XML validation 오류로 한참 삽질… (추가 a)

```
    xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd
        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-aop.xsd
        http://www.springframework.org/schema/context http://www.springframework.org/schema/context/spring-context.xsd
        http://www.springframework.org/schema/jee http://www.springframework.org/schema/jee/spring-jee.xsd
        http://www.springframework.org/schema/lang http://www.springframework.org/schema/lang/spring-lang.xsd
        http://www.springframework.org/schema/tx http://www.springframework.org/schema/tx/spring-tx.xsd
        http://www.springframework.org/schema/util http://www.springframework.org/schema/util/spring-util.xsd


        http://www.directwebremoting.org/schema/spring-dwr
        http://www.directwebremoting.org/schema/spring-dwr-3.0.xsd">
```

삽질2:

```
<bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
  <property name="alwaysUseFullPath" value="true"/>
  <property name="mappings">
  <props>
    <prop key="/dwr/**/*">dwrController</prop>
  </props>
  </property>
</bean>
```

를 주지 않아서 그냥 Spring MVC dispatcher를 타버리지 말아야 할 회피 대상을 그대로 통과시켜서 한참 삽질…

삽질 3:

그런 삽질을 해줬는데도 불구하고 결국 다음 오류.. 스프링 어노테이션에 의하여 등록되는 DWR 빈의 이름이 중복되었다는 황당한 오류로 삽질…

Error creating bean with name 'dwrController'

구글링을 열심히 했더니 DWR을 업그래이드 해보라고 해서 해봐도 오류… ㅜㅜ

그래서 찾은 http://whiteship.me/?p=10060

오.. 기선씨 오랜만..

요대로 하던중… web.xml의 설정에서

```
<servlet-name>dwr-invoker</servlet-name>
        <!-- display-name>DWR Servlet</display-name -->
        <!--<servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>-->
              <servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>
```

이 아니라,

DwrSpringServlet을 줘야 한다고 쓰여져서… 그걸로 바꿔봐도 같은 오류… (뭐, 오류의 근원은 bean명 중복인데.. 당연히 안되겠지.. 순진하긴..)

결국, 설정 방법중 "고전적인" 방법을 택하기로 함. 고전적인 방법은 오히려 쉽고 간단하고, 오류가 적었음… ㅜㅜ

방법은 그냥 다음과 같이 dwr.xml의 수정만으로 간단히 해결하고 있네:

```
<dwr>

  <allow>

    <!--
    <filter class="com.jinotech.dwr.monitor.MonitoringAjaxFilter"/>
    <filter class="org.directwebremoting.filter.ExtraLatencyAjaxFilter">
      <param name="delay" value="200"/>
    </filter>
    -->
<!-- chat -->
    <create creator="new" javascript="JavascriptChat" scope="page">
      <param name="class" value="com.okmindmap.collaboration.JavascriptChat"/>
    </create>

  </allow>

--------------  추가 --------------------

    <allow>
        <create creator="spring" javascript="codiProcessManagerBean">
            <param name="beanName" value="codiProcessManagerBean" />
        </create>
    </allow>

</dwr>
```

음… creator 로 new대신 spring에서 서버객체를 갖고오삼… 이라고 바꿔주고, 파라미터로 spring에 선언된 빈만 갖고 오니 끝나넹.. ㅜㅜ

이런

어려운게 좋은게 아닌데.. 어렵게시리..

(사실 여기서도 org.xml.sax.SAXParseException: Open quote is expected for attribute "creator" associated with an element type "create". 이런 오류가 났었는데, 이건 DashCode라는 편집기를 쓰면서 특수문자가 저장되어 (맥에서는 "(Quotation)이 두가지 버전이 있다.. ㅜㅜ) 난 오류였음… 내 인생이 이런 잡오류로 갉아먹히고 있다니.. 내 인생을 위해서라도 빨랑 ACM을 개발해야겠다는 생각이 투철해짐..)

…

설랜다..

Modules known to DWR:

- JavascriptChat (NewCreator for com.okmindmap.collaboration.JavascriptChat)
- codiProcessManagerBean (SpringCreator for org.uengine.codi.processmanager.CodiProcessManagerBean$$EnhancerByCGLIB$$c37fe969)
-

와우… 이제 자바스크립트로 서버상의, 그것도 스프링으로 ProcessManager를 호출하게 된것이다.. ㅎㅎ

그럼 한번 해볼까??

http://localhost:8080/uengine-web/dwr/test/codiProcessManagerBean

에 접속한후 "listProcessVariableValues"를 검색하고.. 아규먼트 입력 부분에

1을 입력하고 우측에 Execute를 클릭하니 JSON 객체의 결과가 나온다..

훌륭하네..

호출코드 자바스크립트를 우측클릭해서 소스코드를 얻어서 바로 쓸 수 있겠다:

```
<head>
<script type='text/javascript' src='../engine.js'></
script>
                                                    <script type='text/javascript' src='../util.js'></script>
                                                    <script type='text/javascript' src='../interface/codiProcessManagerBean.js'></
                                                    script>


<style>
    input.itext { font-size: smaller; background:
    #E4E4E4; border: 0; }
    input.ibutton { font-size: xx-small; border: 1px
    outset; margin: 0px; padding: 0px; }
    span.reply { background: #ffffdd; white-space:
    pre; }
    span.warning { font-size: smaller; color: red; }
    </style>

</head>
```

```
getProcessVariable( 〈input class='itext'
type='text' size='10' value=''''
id='p520' title='Will be converted to:
java.lang.String'/〉,
                                    〈input class='itext' type='text' size='10' value=''''' id='p521' title='Will be converted to:
                                    java.lang.String'/〉,
                                    〈input class='itext' type='text' size='10' value=''''' id='p522' title='Will be converted to:
                                    java.lang.String'/〉 );
                                    〈input class='ibutton' type='button'
                                    onclick='codiProcessManagerBean.getProcessVariable(objectEval($("p520").value),
                                    objectEval($("p521").value), objectEval($("p522").value), reply52);' value='Execute'
                                    title='Calls codiProcessManagerBean.getProcessVariable(). View source for details.'/〉
                                    〈script type='text/javascript'〉
                                    var reply52 = function(data)
                                    {
                                    if (data != null && typeof data == 'object') alert(dwr.util.toDescriptiveString(data, 2));
                                    else dwr.util.setValue('d52', dwr.util.toDescriptiveString(data, 1));
                                    }
                                    〈/script〉
                                    〈span id='d52' class='reply'〉〈/span〉
```

대략 이렇다.. DWR … 맘에 들게 만들어놨네… 이제 우리는 다음에서 해방된거라 할 수 있다:

1. 서버의 서비스 객체의 인터페이스 및 리턴 객체의 변경에 대하여 클라이언트의 영향을 최소화
2. 호출을 위한 jQuery ajax call 아규먼트와 서버 매핑, JSON mapping, 그리고 오브젝트 매핑의 코딩 및 수정시의 오류와 디버깅 시간에서 해방됐다.

〔자, 이제 더 재밌는것을 해보자… 서버에서 객체의 표현 방법을 정의해주고, 클라이언트는 'toHtml'만 부르면 해당 오브젝트가 알아서 자신을 표현한다〕

앞서 목표를 다시 상기시켜 보자….

```
〈script type='text/javascript' src='../engine.js'〉〈/script〉

〈script type='text/javascript' src='../util.js'〉〈/script〉

〈script type='text/javascript' src='../interface/AddressBook.js'〉〈/script〉
〈script type='text/javascript' src='../interface/Person.js'〉〈/script〉
```

```
〈script〉

   Person person = AddressBook.getPersonByName("장진영");
   $("user").html(person.toHtml());

〈/script〉

〈div id="user"/〉
```

이를 가능하게 하기위해서 개발자에게 가능한 표준적인 접근을 사용하게 하기 위하여 jQuery.View 플러그인과의 조합을 구성하려고 합니다. 즉, 옵션값을 넣어주고, 서버상에서 toHtml에 대한 템플릿의 표현을 가능하게 하면 어떨까 합니다:

먼저, jQuery.view라는 놈이 뭐하는 놈이냐면…. 다음과 같이 JSP와 같은 표현식으로 객체를 html화 하는 로직을 자바스크립트로 표현할 수 있도록 만든 라이브러립니다. 내부적으로는 아마도 eval을 무쟈게 쓰고 있으리라 생각됩니다:

(참고: http://jupiterjs.com/news/jquery-view-client-side-templates-for-jquery)

1. 템플릿 파일을 'mytemplate.ejs'. It might look like:
```
<h2><%= message %></h2>
```
2. 사용은…

```
$("#foo").html('mytemplate.ejs',{message: 'hello world'})
```

혹은 템플릿 표현으로 인라인으로:

```
<script type='text/ejs' id='recipes'>
<% for(var i=0; i < recipes.length; i++){ %>
  <li><%=recipes[i].name %></li>
<%} %>
</script>
```

Render with this template like:

```
$("#foo").html('recipes',recipeData)
```

먼저, jQuery.view라는 놈이 뭐하는 놈이냐면…. 다음과 같이 JSP와 같은 표현식

저는 간단히, 저 JSP문장을 객체의 명과 동일하게 주려고 하고, 저것을 다음과 같은 네이밍 규칙으로 주고 싶습니다:

$("foo").html(recipeData, options)로 주면 … 알아서 foo에 recipeData가 담긴다. 즉, "html" 평션을 재정의 하고 싶다. 해당 html평션의 내부는 다음과 같이 기존 코드를 돌려 호출만 해주면 된다:

```
$("#foo").html(recipeData.getType().getClassName() + ".ejs",{value: recipeData})
```

눈치채셨다시피, 위의 재정의가 가능하려면, DWR로 생성되는 객체는 자신의 클래스명, 즉, 메타데이터를 갖고 있어야 하며, 언제든지 getType()에 의하여 이를 리턴할 준비가 되어있어야 한다.
그리고 더 재미난 것은, ejs 가 해당위치에 없다면, 다음과 같은 ejs 템플릿을 DWR은 동적으로 생성해놓든, 동적으로 생성해서 리턴해야 한다는 것이다 :

```
<% foreach(var fieldDescriptor in value.getType().fieldDescriptors()){ %>
  <li><%fieldDescriptor.getFieldName()%>: <%=eval("value." + fieldList[i].getName())%></li>
<%} %>
```

〔메타웍스2와의 연결〕

기존의 메타웍스2는 서버객체의 메타데이터를 Java Reflection API 를 통하여 얻게되는 Class 객체에서 좀더 확장적으로 매핑한 Type이라는 자바객체를 일반 객체에게서 생성시켜주고, 이에 대한 다양한 추가 정보를 가질 수 있게 해준다.

이를 자바스크립트와 리모팅의 기능을 접목한 메타웍스3에서는 다음과 같은 기능을 추가하길 바라는 것이다:

1. annotation을 기반한 서버객체 필드데스크립터의 자동생성
2. DWR과 연동되면서 DWR의 클래스 메타데이터 요청을 받아 interface생성시 기본으로 자바스크립트용 메타데이터를 생성하기. (때로는 동적으로 메타데이터가 바뀔 경우도 있을까?? 그런경우는 없어야 하는데, inputter에 대해서는 모르겠다)

예를 들어 메타웍스2에서는 다음과 같이 객체에 대한 추가 메타데이터를 주입할 수 있게 설계되었다:

```
class Person{

  String name;
  int age;

  //@inputter = new SelectInput({"CEO", "CTO", "CXO", "Director"});
  String title;

}
```

위와 같이 선언해놓고… 다음과 같이 필드속성을 참조할 수 있다..

```
(new ObjectType(Person.class)).getFieldDesriptor();
```

뭐, 이걸로는 가치를 못느낀다 보통..

메타웍스의 가치는 메타웍스화 된 오브젝트를 감싸는 주변 서비스에 있다:

〈메타웍스 서비스〉
1. UI 자동생성
2. 적절한 자동 Persistence
3. 자동 밸리데이션

이는 객체가 생성되어 소멸되기까지… 객체가 존재하는 공간 -〉 화면, 데이터베이스, XML파일, 입력창, 심지어 웹화면상, 모바일 화면상… 의 이동이 아무리 심하더라도 자신을 당당하고 손실없이 표현할 수 있도록 한다.

아래의 예제를 보면,

```
new GridApplication(new ObjectType(Person.class))).run();
```

이 단순한 표현은 Person객체를 읽고 쓰는 기본적인 Swing 어플리케이션을 생성하여 유저와 다이얼로그 하게 한다.

다음과 같은 상황을 보자:

```
class Person{
  String name;
  int age;
  Address address[];
}

class Address{
  String state;
  String addressDetail;
}
```

이런 오브젝트는 알아서 Person객체이 UI와 Adress객체의 UI를 조합하여 person객체를 여러개 그리드로 입력받아 저장하는 복잡한 UI를 메타웍스는 자동생성할 수 있다. 심지어 이렇게 생성된 객체의 저장로직을 다음과 같이 지정해주면, 메타웍스는 그것이 변경되는 시점에 해당 로직을 알아서 호출해준다:

```
class Person{
  String name;
  int age;
  Address address[];

  //@persistLogic
  void save(){
      Database.insert(this); //whatever..
```

  }
}

메타웍스는 심지어

@autoPersist
class Person{

...

}

으로 되어있는경우.. 알아서 테이블명 생성하고 db에 객체를 관리해주는 ORM작업까지도 수행한다.

〔무엇이 우리를 이토록 복잡하게 하는가?〕

프로그래밍에서 개발자 시간의 대부분을 차지하는 프로그래밍 작업과 이에 따른 디버깅 작업이 존재위치에 따른 메타데이터 손실에서 오기 때문이다… (이는 좀더 검증하겠다) 이를 회피하기 위한 다양한 접근법 DDD, Recflective Programming, Declarative Programming, Generic Programming, Adaptive Object Model, Type-Square Pattern 등이 존재하였는데, 메타웍스는 이러한 모든 접근들을 적절히 수용한다.

〔다시 구현으로 돌아가자〕

본 프레임워크를 개발하는데 있어서 다음과 같은 요즘 대세인 원칙들을 따르기로 한다:

1. 가능한 개발자들이 기존에 익숙한 코딩 컨벤션을 유지할 수 있도록 한다.
2. 가능한 프레임워크의 전체와 일부를 모두 사용할 수 있도록 양파껍질을 표준적으로 만든다.

해서 몇가지 사용 레벨을 나누어서 적용해보기로 한다

1. 레벨1: 객체에 대한 얼굴(*.ejs)을 클래스명과 매핑하여 서버에서 온 객체를 showObject(objName, 'targetDiv') .. 같은 표현으로 표시할 수 있다.
2. 레벨2: 객체내에 연결된 객체를 참조하여 재귀적으로 레벨1에서 그리던 것들을 호출해주면서 오브젝트를 표현한다.
3. 레벨3: 객체의 메서드에 접근하여 리모트 객체의 저장/읽기 등의 기능을 수행한다.

자, 이제 구현에 들어가보자..

〔〔레벨1의 구현〕〕

레벨 1은 객체에 대한 얼굴(jQuery.view 템플릿)과 서버 객체명을 매핑하는 다음과 같은 원칙을 두고 간다:

#Person.java

```
package org.metaworks.example;

public class Person{

	String name;
		public String getName() {
			return name;
		}
		public void setName(String name) {
			this.name = name;
		}

	int age;
		public int getAge() {
			return age;
		}
		public void setAge(int age) {
			this.age = age;
		}

}
```

#AddressBook.java

```
package org.metaworks.example;

import java.util.Hashtable;

public class AddressBook{

	static Hashtable<String, Person> book = new Hashtable<String, Person>();

	public Person findPerson(String name){
		return book.get(name);
	}

	public void putPerson(Person person){
		book.put(person.getName(), person);
	}

}
```

#dwr.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dwr PUBLIC "-//GetAhead Limited//DTD Direct Web Remoting 3.0//EN" "http://getahead.org/dwr/dwr30.dtd">


<dwr>

  <allow>

     <create creator="new" javascript="Metaworks" scope="page">
      <param name="class" value="org.metaworks.dwr.MetaworksRemoteService"/>
    </create>

     <create creator="new" javascript="AddressBook" scope="page">
      <param name="class" value="org.metaworks.example.AddressBook"/>
    </create>

     <convert converter="bean" match="org.metaworks.*"/>
     <convert converter="bean" match="org.metaworks.inputter.*"/>
     <convert converter="bean" match="org.metaworks.example.*"/>
     <convert converter="bean" match="java.lang.Class"/>
  </allow>

</dwr>
```

#metaworks3Level1.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
     <head>
          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
          <title>JQuery View Test</title>
          <script type="text/javascript" src="scripts/jquery.view/jquery-1.6.1.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquerymx-1.0.custom.js"></script>

          <script type="text/javascript" src="scripts/jquery.view/jquery.view.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.ejs.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.jaml.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.micro.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.tmpl.js"></script>

          <script type='text/javascript' src='dwr/engine.js'></script>
          <script type='text/javascript' src='dwr/util.js'></script>
          <script type='text/javascript' src='dwr/interface/Metaworks.js'></script>
          <script type='text/javascript' src='dwr/interface/AddressBook.js'></script>


          <script type="text/javascript">


               function showObject(object, objectTypeName, targetDiv){
                    //alert(objectTypeName.replaceAll('.','/'));
                    $("#" + targetDiv).html('faces/'+objectTypeName + '.ejs',{value: object});
               }

               var person = {
                           name:"Fred Bloggs",
                           age:42,
                          //  appointments:[ new Date(), new Date("1 Jan 2008") ]
                          };

               $(document).ready(function() {
                    AddressBook.putPerson(person, function(data){
                         AddressBook.findPerson('Fred Bloggs',
                                   function(data1){
                                        showObject(data1, 'org.metaworks.example.Person', 'foo');
                                   }
                         )

                    });
               });

          </script>
     </head>
     <body>
          <div id="foo"></div>
     </body>
</html>
```

#faces/org.metaworks.example.Person.ejs

```
<li>Name: <%=value.name%>
<li>Age: <%=value.age%>
```

이렇게 설정해놓고, 실행하면…

Name: Fred Bloggs
Age: 42

이 잘 출력됨을 볼 수 있다. 뭐, 당연한거 아닌가??? 다만, 네이밍 컨벤션으로 인하여 서버측 객체에 대한 화면에 대한 메타정보를 네이밍으로 연결하여 객체 메타데이터에 대한 응집도를 조금 높혔다는 것이 평가다.

그런데, 이걸로 만족할 수 있는가? 다음의 예제를 보자:

#Contact.java

```
package org.metaworks.example;

public class Contact {

	String type;
		public String getType() {
			return type;
		}
		public void setType(String type) {
			this.type = type;
		}

	String address;
		public String getAddress() {
			return address;
		}
		public void setAddress(String address) {
			this.address = address;
		}

}
```

#Person.java

```
package org.metaworks.example;

public class Person{

	String name;
		public String getName() {
			return name;
		}
		public void setName(String name) {
			this.name = name;
		}

	int age;
		public int getAge() {
			return age;
		}
		public void setAge(int age) {
			this.age = age;
		}

	Contact contact;
		public Contact getContact() {
			return contact;
		}
		public void setContact(Contact contact) {
			this.contact = contact;
		}

}
```

#Contact.ejs

```
<li>Type: <%=value.type%>
<li>Address: <%=value.address%>
```

#Person.ejs

```
<li>Name: <%=value.name%>
<li>Age: <%=value.age%>
<li>Age: <%=value.contact%>
```

이렇게 한후 실행하면? 추측하시다시피,

Name: Fred Bloggs
Age: 42
Contact: [object Object]

이렇게 나온다.. 어떻게해야 Contact 부분도 얼굴을 쐬울까? 답은 간단하다:

```
<li>Name: <%=value.name%>
<li>Age: <%=value.age%>
<li>Contact: <div id='foo2'></div>

<script>
   //아래 주석 처리된 부분 모두 실
   //$('#foo2').html('faces/org.metaworks.example.Contact.ejs',{value: value.contact}); //value 가 undefined -- 템플릿에서 교체된 후 들어가기 때문에 여기서 인식 안됨.

  //됨
  //showObject(person.contact, 'org.metaworks.example.Contact', 'foo2'); //성공 / 대신 메인에 들어있는 'person'이라는 변수명을 기억해야함.
  //됨
  showObject(<%=JSON.stringify(value.contact)%>, 'org.metaworks.example.Contact', 'foo2');  //성공. 하지만 객체를 괜히 문자열했다가 다시 파싱하는 바보같은 이슈 발생.
</script>
```

하지만, 그렇게 간단하지도 않다. 좀, 지저분하다…, 사용자가 많은것을 알아야 하고..말이징..

요약하면, 다음과 같이 표현하여 객체에 상호 포함된 구조에 대한 템플릿을 재귀호출 할 수 있다:

```
<li>Name: <%=value.name%>
```

```
<li>Age: <%=value.age%>
<li>Contact: <ul><div id='foo2'></div></ul>

<script>
  showObject(<%=JSON.stringify(value.contact)%>, 'org.metaworks.example.Contact', 'foo2');
</script>
```

문제점:

1. 일단 현재단계로 toJSON을 하지않고 (성능안좋음) jQuery.view와 함께 사용할 방법을 찾는것은 jQuery.view 내부를 조금 수정하던지 파악하던지 해서 실제 넘겨지는 값들 중 'value'에 접근하는 온전한 객체에 접근하는 방법을 찾은 후에나 가능할 것 같다. 지금은 skip
2. 'foo2'의 값-div id 값 - 은 동적으로 생성되던지 해야할 판이다. 아니면 분명 화면 내 하나 이상의 동일한 유형의 객체만 있으면 오류가 난다. 이는 〈div〉 출력에서부터 〈script〉생성 부분까지를 hiding해야할 필요와 함께 응집도있는 하나의 묶음임을 암시한다. 지금은 마찬가지 skip...

〔〔레벨2의 구현〕〕

앞서 레벨1에서는 '알고있는 객체'의 경우에 통용가능하다. 하지만 다음과 같은 상황은 어떠한가?

#ExtendedContact.java

이렇게 자바스크립트단으로 넘어온 서버객체는 자신의 객체정보 (메타데이터) 를 참조하기 위한 적어도 키값 정도는 자신이 뱉어낼 수 있어야 한다. 그래야 세부 메타데이터를 자바스크립트에 캐싱해놓고 클라이언트 어플리케이션들이 이의 생성과 소멸에 이르는 다양한 행위를 도와줄 수 있다:

var clazzName = person.__className;

이렇게 줄 수 있다면, 앞서 'showObject' 메서드는 사전정보 (클래스명) 없이도 동적으로 호출될 수 있다:

```
                function showObject(object, objectTypeName, targetDiv){
                     //alert(objectTypeName.replaceAll('.','/'));
                     $("#" + targetDiv).html('faces/'+objectTypeName + '.ejs',{value: object});
                }
===〉
                function showObject(object, targetDiv){
                     //alert(objectTypeName.replaceAll('.','/'));
                     $("#" + targetDiv).html('faces/'+object.__className + '.ejs',{value: object});
                }
```

그런데, 그러고보니, 만약 object = null 인 경우는 어쩌려고 그러지? 그런 경우는 어차피 클래스 정보가 없으니 objectTypeName 사전정보를 어차피 제공해야 하는 것 아닌가 하는 생각도 든다…

〔어쨌든 만약에 위와 같이 dwr이 뱉어내는 클래스 정보에 __className을 리턴시키고자 한다면 다음과 같이 할 수 있을거 같다:〕

DWR은 converter 라는 컴포넌트를 통하여 자바객체를 JSON으로 바꿔주는데, 인터페이스가 다음과 같다:

```
/*
 * Copyright 2005 Joe Walker
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */
package org.directwebremoting.extend;

import org.directwebremoting.ConversionException;

/**
 * An interface for converting types from a string to some other type.
 * @author Joe Walker [joe at getahead dot ltd dot uk]
 */
public interface Converter
{
    /**
     * If we are a compound converter that farms out part of the conversion
     * to other converters then you farm the conversion out via a configuration.
     * @param converterManager The configuration object
     */
    void setConverterManager(ConverterManager converterManager);

    /**
     * Attempt to coerce the data from a string to an Object.
     * If anything goes wrong with inbound conversion then we generally throw
     * an exception because we are converting data from the untrusted Internet
     * so we take the assumption that anything wrong is someone hacking.
     * @param paramType The type to convert to
     * @param data The data to convert
     * @return The converted data, or null if the conversion was not possible
     * @throws ConversionException If the conversion failed for some reason
     */
    Object convertInbound(Class<?> paramType, InboundVariable data) throws ConversionException;

    /**
     * Return a javascript string that defines the variable named varName to
     * have the contents of the converted object data.
     * <p>In contrast to <code>convertInbound()</code> failures in converting
```

```
     * data on the way out should not stop processing, and we should carry on
     * if we can. Failures are probably down to some misconfiguration so as much
     * information about the error as can be safely generated to console logs is
     * good. In other words if you need to loop in outbound conversion then it
     * might be a good idea to catch issues inside the loop, log, and carry on.
     * @param data The data to convert
     * @param outctx A collection of objects already converted and the results
     * @return The OutboundVariable that represents the data to convert
     * @throws ConversionException If the conversion failed for some reason
     */
    OutboundVariable convertOutbound(Object data, OutboundContext outctx) throws ConversionException;
}
```

전략은 기존 "BeanConverter"의 `convertOutbound(Object data, OutboundContext outctx)` 를 다음과 같이 살짝 꼬아 (오버라이드 해) 주는것이다:

```
package org.metaworks.dwr;

import java.util.Map;
import java.util.TreeMap;

import org.directwebremoting.ConversionException;
import org.directwebremoting.convert.BeanConverter;
import org.directwebremoting.extend.OutboundContext;
import org.directwebremoting.extend.OutboundVariable;

public class MetaworksBeanConverter extends BeanConverter{

     @Override
     public OutboundVariable convertOutbound(Object arg0, OutboundContext arg1)
               throws ConversionException {
          // TODO Auto-generated method stub
          OutboundVariable ov =  super.convertOutbound(arg0, arg1);

        Map<String, OutboundVariable> ovs = new TreeMap<String, OutboundVariable>();

        ov.

          //ovs.put("__className", arg0.getClass().getName());


          return ov;
     }

}
```

뭐, 이렇게 만든다음에 dwr.xml에서 "bean" 대신에 "metaworks-bean" 머 이런식으로 넣어주믄 딱일 거 같은데… 이상하게 오류가 자꾸 뜬다..

당장 저것이 불필요하기에 스킵~

레벨 2는…… 변형된 객체 - 상속, 배열, 컴포지션 - 등에 따른 적절한 face의 자동 매핑..

만약, 해당 class명에 매핑되는 ejs가 없는 경우에 대한 자동화된 대응… 으로 바꾸는 게 어떨까?

예를 들어..

class SuperPerson extends Person{….} 객체에 매핑된 적절한 ejs는 분명 존재하지 않지만, 아다시피 org.metaworks.example.Person.ejs를 대체사용 가능할 것이다.

그리고,

Contact[] (배열) 에 대한 매핑된 ejs는 분명 존재하지 않지만, 적절히 org.metaworks.example.Contact.ejs를 반복호출하여 사용가능할 것이다.

그리고,

컴포지션 객체를 만났을때 예를 들어,

```
class PersonAndContact{
  Person person;
  Contact contact;
}
```

는 적절히, …Person.ejs와 Contact.ejs를 둘다 호출하는 Composition이 가능할 것이다.

가장먼저, 가장 간단한 ArrayFace.js를 만들어보자:

#index.html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
     <head>
          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
          <title>JQuery View Test</title>
          <script type="text/javascript" src="scripts/jquery.view/jquery-1.6.1.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquerymx-1.0.custom.js"></script>

          <script type="text/javascript" src="scripts/jquery.view/jquery.view.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.ejs.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.jaml.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.micro.js"></script>
          <script type="text/javascript" src="scripts/jquery.view/jquery.view.tmpl.js"></script>
```

```
            <script type='text/javascript' src='dwr/engine.js'></script>
            <script type='text/javascript' src='dwr/util.js'></script>
            <script type='text/javascript' src='dwr/interface/Metaworks.js'></script>
            <script type='text/javascript' src='dwr/interface/AddressBook.js'></script>


          <script type="text/javascript">

               //will be automatically generated by server
               var actualFaceClassMappings={
                    'org.metaworks.example.Contact'                    : 'org.metaworks.example.Contact',
                    'org.metaworks.example.Person'               : 'org.metaworks.example.Person',
                    'org.metaworks.example.SuperPerson'          : 'org.metaworks.example.Person',
                    'org.metaworks.example.PersonAndContact' : 'org.metaworks.faces.ObjectFace',
                    //'org.metaworks.example.Contact[]'          : 'org.metaworks.faces.ArrayFace'  //array face doesn't need to manually
map
               }


               function showObject(object, objectTypeName, targetDiv){

                    //choosing strategy for actual Face file.
                    var actualFace;
                    if(objectTypeName.substr(-2) == '[]'){                                        //if array of some object type, use ArrayFace
with mapped class mapping for the object type.
                         actualFace = 'org.metaworks.faces.ArrayFace';
                         objectTypeName = objectTypeName.substr(0, objectTypeName.length - 2);
                         //alert("subtR" + objectTypeName)
                         var substitution = actualFaceClassMappings[objectTypeName];
                         if(substitution) objectTypeName = substitution;
                    }else{
                         actualFace = actualFaceClassMappings[objectTypeName];         //use mapped one
                    }

                    if(!actualFace) actualFace = "org.metaworks.faces.ObjectFace";     //eventhough there's no mapping, use ObjectFace

                    //alert("requested: " + objectTypeName + " - actualFace: " + actualFace);

                    $("#" + targetDiv).html('faces/' + actualFace + '.ejs',{value: object, objectTypeName: objectTypeName, targetDiv:
targetDiv});

               }

               var person = {
                           name:"Fred Bloggs",
                           age:42,
                           contact:{
                                type: "e-mail",
                                address: "fbloggs@uegine.org"
                           }
                         };

               var person2 = {
                           name:"jinyoung Jang",
                           age:42,
                           contact:{
                                type: "e-mail",
                                address: "jyj@uegine.org"
                           }
                         };

               $(document).ready(function() {
                    AddressBook.putPerson(person, function(data){
                         AddressBook.putPerson(person2, function(data){
                         AddressBook.listPersonsInSameAge('42',
                                   function(data1){
                                        //alert(dwr.util.toDescriptiveString(data1, 2))

                                        showObject(data1, 'org.metaworks.example.Person[]', 'foo');
                                   }
                         )

                         });
                    });
               });



          </script>
     </head>
     <body>
          <div id="foo"></div>

     </body>
</html>
```

#faces/org.metaworks.faces.ArrayFace.ejs

```
<%

alert('in ArrayFace');
  for (var i=0; i<value.length; i++){

 %>

<div id='<%=targetDiv%>_array_item_<%=i%>'>V</div>

<script>
  showObject(<%=JSON.stringify(value[i])%>, '<%=objectTypeName%>', '<%=targetDiv%>_array_item_<%=i%>');
```

```
</script>

<%
  }
%>
```

#faces/org.metaworks.example.Person.ejs

```
<li>Name: <%=value.name%>
<li>Age: <%=value.age%>
<li>Contact: <ul><div id='<%=targetDiv%>_contact_'>XXXXXX</div></ul>

<script>
alert('in Person ejs');
  showObject(<%=JSON.stringify(value.contact)%>, 'org.metaworks.example.Contact', '<%=targetDiv%>_contact_');
</script>
```

음… 멋진 결과가 다음과 같이 나온다:

Name: jinyoung Jang
Age: 42
Contact:
- Type: e-mail
- Address: jyj@uegine.org

Name: Fred Bloggs
Age: 42
Contact:
- Type: e-mail
- Address: fbloggs@uegine.org

솔루션 개발 및 SI의 공통모듈 개발의 입장에서 ArrayFace.ejs 는 모든 객체에 대한 배열을 알아서 뿌려주는 공통 로직이 된다. 즉, ArrayFace.js만 바꿔주면 어떤 대상이던 일관성 있게 배치하는 방식을 관리할 수 있게 된다. 예를 들어 테이블에 이쁘게 넣거나, 배열객체만 만나면 알아서 그리드에 뿌린다던가 하는 일관성을 부여시키기 용이해진다.

자, 이번엔 ObjectFace를 만들어볼까?

ObjectFace.ejs의 특징은 이제 동적으로 서버에서 객체의 메타데이터를 얻어와야만 한다는 것이다. 위에서처럼 매핑을 갖고는 한계가 있다:

해서 꿍짝꿍짝… 서버에 메타데이터 갖고 올때 ejs 가 존재하는지 찾기해서 없으면 해당 도메인 객체를 쭉 타고 올라가면서 가장 적절한 옷 (face)을 만나면 그걸 매핑 (WebObjectType.setFaceComponentPath(…))해주는 등의 작업을 해서 다음과 같이 라이브러리를 수정하였다:

```
          <script type="text/javascript">

               var metaworksMetadata = new Array();

               function getMetadata(objectTypeName, onLoadDone){

                    if(!metaworksMetadata[objectTypeName]){
                         Metaworks.getMetaworksType(objectTypeName,
                                   {
                                   async: false,
                                   callback: function( webObjectType ){
                                             metaworksMetadata[objectTypeName] = webObjectType;
                                   }
                              }
                         )
                    }

                    var objectMetadata = metaworksMetadata[objectTypeName];

                    return objectMetadata;
               }

               function showObject(object, objectTypeName, targetDiv){
                    //choosing strategy for actual Face file.
                    var actualFace;
                    if(objectTypeName.substr(-2) == '[]'){                                        //if array of some object type, use ArrayFace
with mapped class mapping for the object type.
                         actualFace = 'faces/genericfaces/ArrayFace.ejs';
                         objectTypeName = objectTypeName.substr(0, objectTypeName.length - 2);
                    }else{

                         var metadata = getMetadata(objectTypeName);

                         actualFace = metadata.faceComponentPath;

                    if(!actualFace)
                         actualFace = 'faces/genericfaces/ObjectFace.ejs';//eventhough there's no mapping, use ObjectFace
                    }

/*                  $("#" + targetDiv).html(
                         'metaworks/' + faceLocation,
                         {value: object, objectTypeName: objectTypeName, targetDiv: targetDiv, objectMetadata: getMetadata(objectTypeName)}
                    );
 */


                    try {
```

```
                        var html = new EJS({url: 'metaworks/' + actualFace})
                        .render({value: object, objectTypeName: objectTypeName, targetDiv: targetDiv, objectMetadata:
getMetadata(objectTypeName)})


                        alert(html);
                        $("#" + targetDiv).html(html);

                    } catch(e) {
                        template_error(e)
                        return
                    }

                }
```

호출단은 이렇게 바꾼다:

```
                var person = {
                            name:"Fred Bloggs",
                            age:42,
                            contact:{
                                type: "e-mail",
                                address: "fbloggs@uegine.org"
                            }
                        };

    $(document).ready(function() {

        AddressBook.putPerson(person2, function(data){
            AddressBook.findPersonAndContact('jinyoung Jang',
                        function(data1){
                            showObject(data1, 'org.metaworks.example.PersonAndContact', 'foo');
                        }
            );
        });


    });

        </script>
    </head>
    <body>
        <div id="foo">LOADING...</div>

    </body>
</html>
```

〔노트〕 ejs파일들의 위치를 metaworks/faces/〈package폴더〉/〈클래스명〉.ejs 로 바꿨다.. 이클립스에서 파일명으로 바로 찾기 용이하게 만들기 위해서

참고로, 여기서 PersonAndContact 는 ejs는 없기 때문에 기존 얼굴인 ObjectFace가 사용될 수 있도록 유도 했다….

많은 삽질작업 끝내 만들어낸 ObjectFace.ejs :

#metaworks/faces/genericfaces/ObjectFace.ejs

```
<%

  for (var i=0; i<objectMetadata.fieldDescriptors.length; i++){

    var fd = objectMetadata.fieldDescriptors[i];
    var fieldValue = eval("value." + fd.name);
 %>

<li> <%=fd.name%>: <ul><div id='<%=targetDiv%>_<%=fd.name%>'>... LOADING PROPERTY ...</div></ul>

<script>
  showObject(<%=JSON.stringify(fieldValue)%>, '<%=fd.classType.name%>', '<%=targetDiv%>_<%=fd.name%>');
</script>

<%
  }
%>
```

#PersonAndContact.java

```
package org.metaworks.example;

public class PersonAndContact {

    Person person;
        public Person getPerson() {
            return person;
        }
        public void setPerson(Person person) {
            this.person = person;
        }

    Contact contact;
        public Contact getContact() {
            return contact;
        }
        public void setContact(Contact contact) {
            this.contact = contact;
```

```
        }

}
```

실행결과는 … 짜자잔~:

contact:
- Type: e-mail
- Address: jyj@uegine.org

person:
- Name: jinyoung Jang
- Age: 42
- Contact:
  - Type: e-mail
  - Address: jyj@uegine.org

멋있네… (뭐가 멋있는지 참..)

자, 이젠 좀더 일반적인 프로퍼티들을 PersonAndContact 에 추가하여 테스트 해보자:

```
package org.metaworks.example;

public class PersonAndContact {

     Person person;
          public Person getPerson() {
               return person;
          }
          public void setPerson(Person person) {
               this.person = person;
          }

     Contact contact;
          public Contact getContact() {
               return contact;
          }
          public void setContact(Contact contact) {
               this.contact = contact;
          }

     String id;
          public String getId() {
               return id;
          }
          public void setId(String id) {
               this.id = id;
          }

     int order;
          public int getOrder() {
               return order;
          }
          public void setOrder(int order) {
               this.order = order;
          }


}
```

보시다시피, id 와 order 라는 Primitive 타입의 필드 두개를 추가했다. 과연 이것을 제대로 그려낼 수 있을까?

실행해봤더니…

"Cannot read property 'style' of null" 이러한 오류가 난다..

이 오류는 매핑되는 ejs 가 없을때 전형적으로 뿌려졌다.. (왜 오류 메세지가 이렇게 뜨는지 모르겠는데, 이유를 좀 있다 파악해야겠다.. EJS에서 그냥 나오는 오류인지 ㅡㅡ;)

무슨 ejs가 없다는걸까?

바로… 프리미티브 형 클래스들에 대한 Face가 없다는 거지…. 즉,

faces/java/lang/String.ejs
faces/java/lang/Number.ejs

를 만들러갈 차례라는 것이다. (Integer.ejs 나 Long.ejs 보다는 Number.ejs가 낫다.. 이유는 숫자들의 최상위에 Number가 있으니까 한방에 다 걸린다)

소스는 간단히 다음과 같이 만들 수 있다:

#faces/java/lang/String.ejs

```
<% if(value){%>
  "<%=value%>"
<%} else {%>
  Empty String
<%}%>
```

#faces/java/lang/Number.ejs

```
<% if(value){%>
  [<%=value%>]
<%} else {%>
  Empty Number
<%}%>
```

음… id까지는 잘 들어온다.. 그런데 마지막 order에서 템플릿(얼굴) 없다는 오류가… 음.. 봤더니.. primitive 형 int (Integer말고 int)는 클래스명이 "int"네…
그냥 단순히 Number.ejs를 faces/int.ejs 로 파일명 바꿔서 저장해준다.

[노트:TODO #001] 여기서 생각나는 것이 int.ejs와 Number.ejs는 같은 로직을 태우고 변경시도 같이 바꿀 공산이 높다. . 이런경우 자바에서는 상속을 해서 상호 반영을 시켜줄텐데.. ejs는 텍스트 파일이니.. 다음과 같은 표현이 되면 좋을거 같은데…

#int.ejs

```
〈%
 include("java/lang/Number.ejs");
%〉
```

음… 저걸 하려면 EJS.js 를 확장 쬐금 하면 될거이 같은데…. 어쨌거나 Skip.

음.. 그럼에도 불구하고 오류난다… 삽질좀 하다 알아낸 사실은 이미 그 사이에 getMetadata.js가 불려져서, "int" 라는 클래스에 대한 메터정보를 얻으려고 시도 하게된다는 것이다. 음.. 바보네.. 이를 해결한 방법은 간단하다:

#WebObjectType.java

```
	public WebObjectType(Class actCls) throws Exception {
		super(actCls);


		for(FieldDescriptor fd : getFieldDescriptors()){
			fd.setName(fd.getName().substring(0, 1).toLowerCase() + fd.getName().substring(1));

			if(fd.getClassType().isPrimitive()){
				fd.setClassType(fd.getClassType().newInstance().getClass());
			}
		}

	}
```

그런데.. 또 이상한 에러가.. 음… 자세히 봤더니.. 나는 저..fd.getClassType().newInstance().getClass()이 표현이 프리미티비 -〉 적절한 객체형 매핑 클래스로 세탁을 해줄 줄 알았더니.. (순진한넘) 에러가 뜬다. 결국 귀찮은 짓을 하는 수 밖에 없구나.. .ㅜㅜ

```
	public WebObjectType(Class actCls) throws Exception {
		super(actCls);


		for(FieldDescriptor fd : getFieldDescriptors()){
			fd.setName(fd.getName().substring(0, 1).toLowerCase() + fd.getName().substring(1));

			if(fd.getClassType().isPrimitive()){

				String clsName = fd.getClassType().getName();

				if("int".equals(clsName)){
					fd.setClassType(Integer.class);
				}//TODO: consider other primitive class types as well
			}
		}

	}
```

저.. TODO는 누가 언제하지??? long, double.. 두개나 더 되나?? 또 뭐가 있드라?? 난 절대 못하겠다… 〉〈

쫙쟈자~쟈~잔~~~~~

id:
- "#ID001"

contact:
- Type: e-mail
- Address: jyj@uegine.org

person:
- Name: jinyoung Jang
- Age: 42
- Contact:
  - Type: e-mail
  - Address: jyj@uegine.org
-

order:
- [5]

자, 이제 더 재미있는 실험을 해보자.. 기냥 기존에 만들어놨던 Person.ejs와 Contact.ejs도 없애버리는 것이다:

그러면, 어떤일이 벌어질꼬?

결과는 다음과 같았다:

id:
- "#ID001"

contact:
- address:
  - "jyj@uegine.org"
- type:
  - "e-mail"

person:
- name:
  - "jinyoung Jang"
- age:
  - [42]

- contact:
  - address:
    - "jyj@uegine.org"
  - type:
    - "e-mail"
-

order:
- [5]

----- 왜 이렇게 되는지 설명을 하면 여러분을 무시하는 것 같아서 안하겠사옵니다 .^^

[이쯤에서… 쉬어가는 중요한 이야기?]

메타웍스2는 현존의 유엔진 프로세스 디자이너 (스윙기반) UI의 특히 다이얼로그 부분의 대부분을 생성하는 역할을 하고 있다. 이의 웹버전은 페이스북에서 보는것과 같이 웹을 보다 오브젝트적인 수준의 응집도 높은 표현을 가능하게 할 것이다.

여기서의 객체를 기반한 UI 컨트롤들의 특징은 매우 응집도가 높다는 것이다.

예를 들어, 페이스북에서 사용자의 사진을 클릭하면 사용자의 얼굴 위에 표현되는 풍선 다이얼로그는 그것이 어느 위치에 존재하던간에 동일한 UI로 뿌려질것이 User에게로 하여금 기대되어진다.

또 다른 예제로, 윈도우즈에서는 사용자가 특정 어플리케이션에서 '열기' 및 '다른이름으로 저장'을 눌러 열려진 파일찾기 다이얼로그 상에서도, 탐색기 내에서 파일 관리를 수행하듯, 우측클릭을 하여 잘라내기, 붙여넣기 편집을 할 수 있는데, 이것은 탐색기와 파일찾기 다이얼로그가 동일한 상속구조를 갖는 객체이기 때문에 기능의 상속이 이루어진 것이고, 그것은 유저에게는 자연스런 연상작용을 하게 하여 좋은 경험을 이루게 한다. 이것은 가끔 동일한 UI를 보여주는 자바기반 어플리케이션에서는 통용되지 않는다. 그건 두 파일찾기 다이얼로그가 동일하게 생겼지만 실제로는 다른 객체이기 때문이다. 이것은 때로 사용자를 당황하게 한다.

뭐, 우리가 달나라에 가도 젓소와 비슷하게 생긴 동물이 보이면 그 동물이 음메~하고 울거같고, 젖도짜서 먹을 수 있을거라 연상할 상황과 유사하다 보면 되겠다. 철두철미한 객체 기반의 설계는 재사용의 문제만을 담지 않는다. 요즘 미치도록 우리를 괴롭히는 UX와 관계된 것이다. 다시말해 단순한 개발 생산성 측면이 아니라 UX의 일관성을 높혀 사용자로 하여금 integrity가 높은 서비스와 솔루션으로 느껴지게 하는 기초이다.

자, 위에서 이야기 한 것과 같은 "응집도 높은" 설계를 함 해볼라문, 우째해야 함미껴??

다음의 경우를 보자.

Person객체에 사용자의 사진을 담는 'portraitURL'이라고 하는 필드를 서버에 추가하자:

그러면, 누가봐도 Person을 화면에 뿌릴때 이제 사진을 같이 보여줄 수 있다는 것을 느낀다. 기존에 Person이라는 JSON으로 처리해오던 수많은 html code들을 다 고치는게 아니다. 이제 우리가 할건, Person 객체에 추가한 portrait을 보여줄 '얼굴' 컴포넌트만 고치면 모든 자리의 Person 종? 들은 알아서 진화된다:

#Person.java

```
package org.metaworks.example;

public class Person{

	String name;
		public String getName() {
			return name;
		}
		public void setName(String name) {
			this.name = name;
		}

	int age;
		public int getAge() {
			return age;
		}
		public void setAge(int age) {
			this.age = age;
		}

	Contact contact;
		public Contact getContact() {
			return contact;
		}
		public void setContact(Contact contact) {
			this.contact = contact;
		}

	String portraitURL;
		public String getPortraitURL() {
			return portraitURL;
		}
		public void setPortraitURL(String portraitURL) {
			this.portraitURL = portraitURL;
		}


}
```

#Person.ejs

```
<img src='<%=value.portraitURL%>' alter='Age is <%=value.age%>'>

<li>Name: <%=value.name%>
<li>Contact: <ul><div id='<%=targetDiv%>_contact_'>LOADING...</div></ul>

<script>

<% if (value.contact!=null) {%>
 showObject(<%=JSON.stringify(value.contact)%>, 'org.metaworks.example.Contact', '<%=targetDiv%>_contact_');

<%} else{%>
 $("#<%=targetDiv%>_contact_").html("There's no Contact Info.");
```

```
<%}%>

</script>
```

이렇게 해놓고,

```
                var person2 = {
                            name:"jinyoung Jang",
                            portraitURL: "http://www.bloter.net/files/2009/02/uengineceo090209.jpg",
                            age:42,
                            contact:{
                                 type: "e-mail",
                                 address: "jyj@uegine.org"
                            }
                       };
```

를 이렇게 바꿔주니….

화면이 이렇게 바뀐다:

http://a2.sphotos.ak.fbcdn.net/hphotos-ak-ash4/315050_2188670881927_1401720840_32260893_1243701094_n.jpg

쫌… 깜찍하게 하기 위해서 마우스가 올려져서 몇 초 지나면 나이를 알려주는 획기적인 엔터테인먼트 기능까지 추가했다. 헉.. 42개??

물론, 나머지 애들 (Contact, PersonAndContact) 은 기본 얼굴 (튜닝전 얼굴)을 하고 있을 것이다.

〔이제… 메타데이터의 파워를 실감하자〕

메타데이터란… 별것 아니다.. 그냥 많이 쓰는 어노테이션이 메타데이터 기반 개발의 가장 현존하는 성공한 이쪽 집안의 장남 격이다.

간단하게 어노테이션으로 아까 변경한 Person 객체의 portrait성격을 부여하기 위하여 Person.ejs를 변경하였는데, 아까 아까 했던것 처럼 Person.ejs 는 이제 날려버리고 다음과 같이 더 게을러 질 수 있을까?

```
     @org.metaworks.Metadata(viewer="image")
     String portraitURL;
          public String getPortraitURL() {
               return portraitURL;
          }
          public void setPortraitURL(String portraitURL) {
               this.portraitURL = portraitURL;
          }
```

#Metadata.java

```
package org.metaworks;

public @interface Metadata {
     String viewer();
}
```

#WebViewer.java

```
package org.metaworks.dwr;

import java.util.Hashtable;

import org.metaworks.viewer.Viewer;

public class WebViewer implements Viewer{

     @Override
     public void initialize(Hashtable arg0) {
          // TODO Auto-generated method stub

     }

     String viewerFace;

          public String getViewerFace() {
               return viewerFace;
          }

          public void setViewerFace(String viewerFace) {
               this.viewerFace = viewerFace;
          }

}
```

#WebObjectType.java

```
     public WebObjectType(Class actCls) throws Exception {
```

```
        super(actCls);


        for(FieldDescriptor fd : getFieldDescriptors()){
             fd.setName(fd.getName().substring(0, 1).toLowerCase() + fd.getName().substring(1));

             if(fd.getClassType().isPrimitive()){

                  String clsName = fd.getClassType().getName();

                  if("int".equals(clsName)){
                       fd.setClassType(Integer.class);
                  }//TODO: consider other primitive class types as well
             }

             Class clazz = getClassType();
             org.metaworks.Metadata metadata =
                       clazz.getField(fd.getName()).getAnnotation(org.metaworks.Metadata.class);

             WebViewer viewer = new WebViewer();

             viewer.setViewerFace(metadata.viewer());

             fd.setViewer(viewer);
        }


    }
```

이런…..

```
    <convert converter="bean" match="org.metaworks.dwr.*"/>
```

나를 날밤새게 했던 오류: s3 is not defined…

이건뭐, 구글링을 해도 답이 없다.. 결국 다른 사람도 이런 문제가 있고, 해결책이 딱히 없는 것을 확인하게 된다: 좌절 ~ㅠㅠ

Byte

Did you run into any problems yet?

For us, sometimes the browser doesn't block, which results in undefined being returned, since the variable r isn't set yet.

This only seems to happen in certain cases, when you come back to the page using the back button or via a bookmark.

음… 호출단이 문제라면 DWR이 생성해주는 기본 클라이언트에서도 호출이 안될까?

http://localhost:8080/metaworks3/dwr에 접속하여 MetaworksService를 선택해서 해봤다..

호출을 받아사 "Loading"이라는 메세지가 상단 우측에 뜨고 결과가 나오지 않는다.

일만인 즉슨, 서버가 잘못이다.

분명 리턴할때까지는 문제가 없었는데, 오류가 있다는 것은… 음…

음… 그렇다면 서버에서 순간적으로 다른 스크립트를 또 만들어내나?? 아니면, DWR의 호출 부를 읽어봐야겠다:

오류의 스택을 출력해보면 아래와 같다:

```
Daemon Thread [http-bio-8080-exec-5] (Suspended (breakpoint at line 14 in MetaworksRemoteService))
    MetaworksRemoteService.getMetaworksType(String) line: 14
    NativeMethodAccessorImpl.invoke0(Method, Object, Object[]) line: not available [native method]
    NativeMethodAccessorImpl.invoke(Object, Object[]) line: 39
    DelegatingMethodAccessorImpl.invoke(Object, Object[]) line: 25
    Method.invoke(Object, Object...) line: 597
    CreatorModule$1.doFilter(Object, Method, Object[]) line: 229
    CreatorModule.executeMethod(MethodDeclaration, Object[]) line: 241
    DefaultRemoter.execute(Call) line: 379
    DefaultRemoter.execute(Calls) line: 332
    PlainCallHandler(BaseCallHandler).handle(HttpServletRequest, HttpServletResponse) line: 104
    UrlProcessor.handle(HttpServletRequest, HttpServletResponse) line: 120
    DwrServlet.doPost(HttpServletRequest, HttpServletResponse) line: 141
    DwrServlet(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 641
    DwrServlet(HttpServlet).service(ServletRequest, ServletResponse) line: 722
    ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 304
    ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 210
    StandardWrapperValve.invoke(Request, Response) line: 224
    StandardContextValve.invoke(Request, Response) line: 185
    NonLoginAuthenticator(AuthenticatorBase).invoke(Request, Response) line: 472
    StandardHostValve.invoke(Request, Response) line: 151
    ErrorReportValve.invoke(Request, Response) line: 100
    AccessLogValve.invoke(Request, Response) line: 929
    StandardEngineValve.invoke(Request, Response) line: 118
    CoyoteAdapter.service(Request, Response) line: 405
    Http11Processor.process(SocketWrapper<Socket>) line: 269
    Http11Protocol$Http11ConnectionHandler(AbstractProtocol$AbstractConnectionHandler<S,P>).process(SocketWrapper<S>, SocketStatus)
line: 515
```

```
JIoEndpoint$SocketProcessor.run() line: 302
ThreadPoolExecutor$Worker.runTask(Runnable) line: 886
ThreadPoolExecutor$Worker.run() line: 908
TaskThread(Thread).run() line: 637
```

〔호출단 검증〕

engine.js: 1505 라인을 디버그 포인트 잡아서 본 호출 부분:

```
. arguments: Arguments[1]
. batch: Object
. indexSafari: 105
. prop: undefined
. request: Object
    . body: "callCount=1↵windowName=c0-e4↵c0-scriptName=Metaworks↵c0-methodName=getMetaworksType↵c0-id=0↵c0-
      param0=string:org.metaworks.example.Person↵batchId=3↵instanceId=0↵page=
      %2Fmw3%2FjqueryViewTest.html↵scriptSessionId=MqM2$DLwbLvuJ5Ye6aZUnm$0Vaj/daxjWaj-EjeBA7FPx↵"
    . url: "/mw3/dwr/call/plaincall/Metaworks.getMetaworksType.dwr"
    . __proto__: Object
. this: Object
. value: undefined
. version: "535.1"
```

오홋.. 중요한 라인이 저깃다: engine.js : 1587
reply만 보믄 되는구나..

```
        try {
          var reply = req.responseText;
          reply = dwr.engine._replyRewriteHandler(reply);

          if (status != 200) {
            dwr.engine._handleError(batch, { name:"dwr.engine.http." + status, message:req.statusText });
          }
          else if (reply == null || reply == "") {
            dwr.engine._handleError(batch, { name:"dwr.engine.missingData", message:"No data received from server" });
          }
```

```
"throw 'allowScriptTagRemoting is false.';
(function(){
var r=window.dwr._[0];
//#DWR-INSERT
//#DWR-REPLY
var s0=[];var s1={};var s7={};var s2={};
s1.annotation=false;s1.annotations=s0;s1.anonymousClass=false;s1.array=false;s1.canonicalName="java.lang.Object";s1.classLoader=null;s1.classe
s=[];s1.componentType=null;s1.constructors=[null /* No converter found for 'java.lang.reflect.Constructor'
*/];s1.declaredAnnotations=s0;s1.declaredClasses=[];s1.declaredConstructors=[null /* No converter found for 'java.lang.reflect.Constructor'
*/];s1.declaredFields=[];s1.declaredMethods=[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method'
*/];s1.declaringClass=null;s1.enclosingClass=null;s1.enclosingConstructor=null;s1.enclosingMethod=null;s1['enum']=false;s1.enumConstants=null;
s1.fields=[];s1.genericInterfaces=[];s1.genericSuperclass=null;s1['interface']=false;s1.interfaces=[];s1.localClass=false;s1.memberClass=false
;s1.methods=[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /*
No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found
for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */];s1.modifiers=1;s1.name="java.lang.Object";s1['package']=null /* No converter found for 'java.lang.Package'
*/;s1.primitive=false;s1.protectionDomain=null /* No converter found for 'java.security.ProtectionDomain'
*/;s1.signers=null;s1.simpleName="Object";s1.superclass=null;s1.synthetic=false;s1.typeParameters=[];
s7.Name=s3;s7.PortraitURL=s4;s7.Contact=s5;s7.Age=s6;
s2.actualObject=s2;s2.classType={annotation:false,annotations:s0,anonymousClass:false,array:false,canonicalName:"org.metaworks.example.Person"
,classLoader:null /* No converter found for 'org.apache.catalina.loader.WebappClassLoader' */,classes:[],componentType:null,constructors:
[null /* No converter found for 'java.lang.reflect.Constructor' */],declaredAnnotations:s0,declaredClasses:[],declaredConstructors:[null /* No
converter found for 'java.lang.reflect.Constructor' */],declaredFields:[null /* No converter found for 'java.lang.reflect.Field' */,null /* No
converter found for 'java.lang.reflect.Field' */,null /* No converter found for 'java.lang.reflect.Field' */,null /* No converter found for
'java.lang.reflect.Field' */],declaredMethods:[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method'
*/],declaringClass:null,enclosingClass:null,enclosingConstructor:null,enclosingMethod:null,"enum":false,enumConstants:null,fields:
[],genericInterfaces:[],genericSuperclass:s1,"interface":false,interfaces:[],localClass:false,memberClass:false,methods:[null /* No converter
found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for
'java.lang.reflect.Method' */],modifiers:1,name:"org.metaworks.example.Person","package":null /* No converter found for 'java.lang.Package'
*/,primitive:false,protectionDomain:null /* No converter found for 'java.security.ProtectionDomain'
*/,signers:null,simpleName:"Person",superclass:s1,synthetic:false,typeParameters:
[]};s2.componentName=null;s2.connection=null;s2.faceComponentPath=null;s2.fieldDescriptorTable=s7;s2.fieldDescriptors=[s3,s6,s5,s4];s2.fieldNa
mes=["name","age","contact","portraitURL"];s2.fullKeyMode=false;s2.keyFieldDescriptor=s3;s2.keyPropertyDescriptor=s3;s2.name="org.metaworks.ex
ample.Person";s2.primitive=false;s2.propertyDescriptorTable=s7;s2.propertyDescriptors=[null /* Conversion error for
[Lorg.metaworks.AbstractPropertyDescriptor;. */,null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */,null /*
Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */,null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;.
*/];s2.title="org.metaworks.example.Person";
r.handleCallback("3","0",s2);
})();
"
```

이게 오류의 원천이었구만… 뎅장..

음.. 그런데 reply객체를 보아하는 뭔가 디버그 정보를 주려고 애쓴 부분이 처절하게 느껴지는데… 내 클라이언트에서는 절대해석못할 s3 is not defined ?? 흠….

결국 매핑된 컨버터가 없다는 메세지를 전달하려 기특하게 노력은 했으나 뿌려진 메시지는 s3 is not defined 였구나… . "~~ 대화가 필요해~~~"

그리고 DWR-REPLY 가 순수 JSON이 아니라는 것도 알았다… 이렇게 스크립트가 포함된 날 데이터로 보내오다니..헉… 아무리 눈에 안보이지만… 이건 좀.. 실망이다.

특히 아래 부분에서 오류가 난 것으로 추측된다: s3이 등장한다:

```
s7.Name=s3;s7.PortraitURL=s4;s7.Contact=s5;s7.Age=s6;
```

그렇지만, s3은 선언된 적이 없다.

흠..

결국 engine.js : 583 라인의 eval(script) 를 통하여 서버에서 날라온 스크립트를 걍 실행시키는 구마… 우왕…

다른 호출 때도 이런지 보자.

```
"
(function(){
var r=window.dwr._[0];
//#DWR-INSERT
//#DWR-REPLY
r.handleCallback("0","0",null);
})();
"
```

딴때는 딸랑 위와 같다… 움… 데이터는 어디에 실렸징??

또 다른 예:

```
"throw 'allowScriptTagRemoting is false.';
(function(){
var r=window.dwr._[0];
//#DWR-INSERT
//#DWR-REPLY
var s0={};s0.address="jyj@uegine.org";s0.type="e-mail";
r.handleCallback("2","0",{contact:s0,id:"#ID001",order:5,person:{age:42,contact:s0,name:"jinyoung Jang",portraitURL:"http://www.bloter.net/
files/2009/02/uengineceo090209.jpg"}});
})();
"
```

이렇다…

직접 데이터를 구성할 스크립트를 전달하네… 왜 이랬을까? 물론 동적 속성이 필요했겠고, DWR 3.0에 여러가지 분산객체의 동적 속성을 고려하는 작업이 진행중이지만.. 이 덕분에 그러한 찾기 힘든 오류가 낫던 거시었다…

〔그렇다면 분명 JSON으로만 Transport 하는 옵션이 있을거다!〕

찾아보자.. 옵션을…

#web.xml내의 DWR 부위:

```
    <servlet>
        <servlet-name>dwr-invoker</servlet-name>
        <!-- display-name>DWR Servlet</display-name -->
        <!--<servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>-->
                <servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>

        <init-param>
            <param-name>debug</param-name>
            <param-value>true</param-value>
        </init-param>
        <!-- Remove this unless you want to use active reverse ajax -->
                <init-param>
                  <param-name>activeReverseAjaxEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- By default DWR creates application scope objects when they are first
                used. This creates them when the app-server is started -->
                <init-param>
                  <param-name>initApplicationScopeCreatorsAtStartup</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- WARNING: allowing JSON-RPC connections bypasses much of the security
                protection that DWR gives you. Take this out if security is important -->
                <init-param>
                  <param-name>jsonRpcEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- WARNING: allowing JSONP connections bypasses much of the security
                protection that DWR gives you. Take this out if security is important -->
                <init-param>
                  <param-name>jsonpEnabled</param-name>
                  <param-value>true</param-value>
                </init-param>

                <!-- data: URLs are good for small images, but are slower, and could OOM for
                larger images. Leave this out (or keep 'false') for anything but small images -->
                <init-param>
                  <param-name>preferDataUrlSchema</param-name>
                  <param-value>false</param-value>
                </init-param>

                <!-- This enables full streaming mode. It's probably better to leave this
                out if you are running across the Internet -->
                <init-param>
                  <param-name>maxWaitAfterWrite</param-name>
                  <param-value>-1</param-value>
                </init-param>

                <!--
```

```
For more information on these parameters, see:
- http://getahead.org/dwr/server/servlet
- http://getahead.org/dwr/reverse-ajax/configuration
-->
<load-on-startup>1</load-on-startup>
```

음.. 눈에 띄는건…jsonRpcEnabled… 정도… true로 되어있다고 JSON 으로 주고받지는 않는구나.

2) **JSON/JSONP** - In DWR 3 a JSON/JSONP RESTian API has been added that allows users to access Java services exposed to DWR via an URL:`http://example.com/dwr/jsonp/ClassName/methodName/param1/param2` DWR will reply with a JSON structure containing the result of calling ClassName.methodName("param1", "param2");

고친후 데이터:

(function(){↵var r=window.dwr._[0];↵//#DWR-INSERT↵//#DWR-REPLY↵var s0=[];var s1={};var s7={};var s2={};
↵s1.annotation=false;s1.annotations=s0;s1.anonymousClass=false;s1.array=false;s1.canonicalName="java.lang.Object";s1.classLoader=null;s1.classes=[];s1.componentType=null;s1.constructors=[null /* No converter found for 'java.lang.reflect.Constructor' */];s1.declaredAnnotations=s0;s1.declaredClasses=[];s1.declaredConstructors=[null /* No converter found for 'java.lang.reflect.Constructor' */];s1.declaredFields=[];s1.declaredMethods=[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */];s1.declaringClass=null;s1.enclosingClass=null;s1.enclosingConstructor=null;s1.enclosingMethod=null;s1['enum']=false;s1.enumConstants=null;s1.fields=[];s1.genericInterfaces=[];s1.genericSuperclass=null;s1['interface']=false;s1.interfaces=[];s1.localClass=false;s1.memberClass=false;s1.methods=[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */];s1.modifiers=1;s1.name="java.lang.Object";s1['package']=null /* No converter found for 'java.lang.Package' */;s1.primitive=false;s1.protectionDomain=null /* No converter found for 'java.security.ProtectionDomain' */;s1.signers=null;s1.simpleName="Object";s1.superclass=null;s1.synthetic=false;s1.typeParameters=[];
↵s7.Name=s3;s7.PortraitURL=s4;s7.Contact=s5;s7.Age=s6;
↵s2.actualObject=s2;s2.classType={annotation:false,annotations:s0,anonymousClass:false,array:false,canonicalName:"org.metaworks.example.Person",classLoader:null /* No converter found for 'org.apache.catalina.loader.WebappClassLoader' */,classes:[],componentType:null,constructors:[null /* No converter found for 'java.lang.reflect.Constructor' */],declaredAnnotations:s0,declaredClasses:[],declaredConstructors:[null /* No converter found for 'java.lang.reflect.Constructor' */],declaredFields:[null /* No converter found for 'java.lang.reflect.Field' */,null /* No converter found for 'java.lang.reflect.Field' */,null /* No converter found for 'java.lang.reflect.Field' */,null /* No converter found for 'java.lang.reflect.Field' */],declaredMethods:[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */],declaringClass:null,enclosingClass:null,enclosingConstructor:null,enclosingMethod:null,"enum":false,enumConstants:null,fields:[],genericInterfaces:[],genericSuperclass:s1,"interface":false,interfaces:[],localClass:false,memberClass:false,methods:[null /* No converter found for 'java.lang.reflect.Method' */,null /* No converter found for 'java.lang.reflect.Method' */],modifiers:1,name:"org.metaworks.example.Person","package":null /* No converter found for 'java.lang.Package' */,primitive:false,protectionDomain:null /* No converter found for 'java.security.ProtectionDomain' */,signers:null,simpleName:"Person",superclass:s1,synthetic:false,typeParameters:[]};s2.componentName=null;s2.connection=null;s2.faceComponentPath=null;s2.fieldDescriptorTable=s7;s2.fieldDescriptors=[s3,s6,s5,s4];s2.fieldNames=["name","age","contact","portraitURL"];s2.fullKeyMode=false;s2.keyFieldDescriptor=s3;s2.keyPropertyDescriptor=s3;s2.name="org.metaworks.example.Person";s2.primitive=false;s2.propertyDescriptorTable=s7;s2.propertyDescriptors=[null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */,null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */,null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */,null /* Conversion error for [Lorg.metaworks.AbstractPropertyDescriptor;. */];s2.title="org.metaworks.example.Person";
↵r.handleCallback("3","0",s2);↵})();↵


```
org.metaworks.example.Person={
  actualObject:{
    actualObject:{
      actualObject:{
        actualObject:{actualObject:Object, classType:Object, componentName:null, connection:null, faceComponentPath:null, ...},
        classType:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        componentName:null,
        connection:null,
        faceComponentPath:null,
        fieldDescriptorTable:{Name:null, PortraitURL:null, Contact:null, Age:null},
        fieldDescriptors:[null, null, null, null],
        fieldNames:["name", "age", "contact", "portraitURL"],
        fullKeyMode:false,
        keyFieldDescriptor:null,
        keyPropertyDescriptor:null,
        name:"org.metaworks.example.Person",
        primitive:false,
        propertyDescriptorTable:{Name:null, PortraitURL:null, Contact:null, Age:null},
        propertyDescriptors:[null, null, null, null],
        title:"org.metaworks.example.Person"
      },
      classType:{
        annotation:false,
        annotations:[],
        anonymousClass:false,
        array:false,
        canonicalName:"org.metaworks.example.Person",
        classLoader:null,
        classes:[],
        componentType:null,
        constructors:[null],
        declaredAnnotations:[],
        declaredClasses:[],
        declaredConstructors:[null],
        declaredFields:[null, null, null, null],
        declaredMethods:[null, null, null, null, null, ...],
  ...s:null,
      enclosingClass:null,
      enclosingConstructor:null,
      enclosingMethod:null,
      enum:false,
      enumConstants:null,
      fields:[
      ],
      genericInterfaces:[
      ],
      genericSuperclass:null,
      interface:false,
      interfaces:[
      ],
      localClass:false,
      memberClass:false,
      methods:[
        null,
```

```
        null,
        null,
        null,
        null,
        null,
        null,
        null,
        null
      ],
      modifiers:1,
      name:"java.lang.Object",
      package:null,
      primitive:false,
      protectionDomain:null,
      signers:null,
      simpleName:"Object",
      superclass:null,
      synthetic:false,
      typeParameters:[
      ]
    },
    synthetic:false,
    typeParameters:[
    ]
  },
  componentName:null,
  connection:null,
  faceComponentPath:null,
  fieldDescriptorTable:{
    Name:null,
    PortraitURL:null,
    Contact:null,
    Age:null
  },
  fieldDescriptors:[
    null,
    null,
    null,
    null
  ],
  fieldNames:[
    "name",
    "age",
    "contact",
    "portraitURL"
  ],
  fullKeyMode:false,
  keyFieldDescriptor:null,
  keyPropertyDescriptor:null,
  name:"org.metaworks.example.Person",
  primitive:false,
  propertyDescriptorTable:{
    Name:null,
    PortraitURL:null,
    Contact:null,
    Age:null
  },
  propertyDescriptors:[
    null,
    null,
    null,
    null
  ],
  title:"org.metaworks.example.Person"
}



undefined={
  faceComponentPath:"genericfaces/ObjectFace.ejs",
  fieldDescriptors:[
    {
      classType:{
        annotation:false,
        annotations:[],
        anonymousClass:false,
        array:false,
        canonicalName:"org.metaworks.FieldDescriptor",
        classLoader:null,
        classes:[],
        componentType:null,
        constructors:[null, null, null, null, null, ...],
        declaredAnnotations:[],
        declaredClasses:[],
        declaredConstructors:[null, null, null, null, null, ...],
        declaredFields:[null, null, null, null, null, ...],
        declaredMethods:[null, null, null, null, null, ...],
        declaringClass:null,
        enclosingClass:null,
        enclosingConstructor:null,
        enclosingMethod:null,
        enum:false,
        enumConstants:null,
        fields:[null, null, null, null, null, ...],
        genericInterfaces:[],
        genericSuperclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        interface:false,
        interfaces:[],
        localClass:false,
        memberClass:false,
        methods:[null, null, null, null, null, ...],
        modifiers:1,
        name:"org.metaworks.FieldDescriptor",
        package:null,
        primitive:false,
        protectionDomain:null,
        signers:null,
        simpleName:"FieldDescriptor",
        superclass:{annotation:false, annotations:[], anonymousClass:false,...    classLoader:null,
        classes:[],
        componentType:null,
        constructors:[null, null, null, null, null, ...],
        declaredAnnotations:[],
        declaredClasses:[],
        declaredConstructors:[null, null, null, null, null, ...],
```

```
        declaredFields:[null, null, null, null, null, ...],
        declaredMethods:[null, null, null, null, null, ...],
        declaringClass:null,
        enclosingClass:null,
        enclosingConstructor:null,
        enclosingMethod:null,
        enum:false,
        enumConstants:null,
        fields:[null, null, null, null, null, ...],
        genericInterfaces:[],
        genericSuperclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        interface:false,
        interfaces:[],
        localClass:false,
        memberClass:false,
        methods:[null, null, null, null, null, ...],
        modifiers:1,
        name:"org.metaworks.FieldDescriptor",
        package:null,
        primitive:false,
        protectionDomain:null,
        signers:null,
        simpleName:"FieldDescriptor",
        superclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        synthetic:false,
        typeParameters:[]
      },
      displayName:"PortraitURL",
      inputFace:"org.metaworks.inputter.java_lang_StringInput",
      name:"portraitURL",
      viewFace:"image"
    }
  ]
}



undefined={
  faceComponentPath:"genericfaces/ObjectFace.ejs",
  fieldDescriptors:[
    {
      classType:{
        annotation:false,
        annotations:[],
        anonymousClass:false,
        array:false,
        canonicalName:"org.metaworks.FieldDescriptor",
        classLoader:null,
        classes:[],
        componentType:null,
        constructors:[null, null, null, null, null, ...],
        declaredAnnotations:[],
        declaredClasses:[],
        declaredConstructors:[null, null, null, null, null, ...],
        declaredFields:[null, null, null, null, null, ...],
        declaredMethods:[null, null, null, null, null, ...],
        declaringClass:null,
        enclosingClass:null,
        enclosingConstructor:null,
        enclosingMethod:null,
        enum:false,
        enumConstants:null,
        fields:[null, null, null, null, null, ...],
        genericInterfaces:[],
        genericSuperclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        interface:false,
        interfaces:[],
        localClass:false,
        memberClass:false,
        methods:[null, null, null, null, null, ...],
        modifiers:1,
        name:"org.metaworks.FieldDescriptor",
        package:null,
        primitive:false,
        protectionDomain:null,
        signers:null,
        simpleName:"FieldDescriptor",
        superclass:{annotation:false, annotations:[], anonymousClass:false,...        classLoader:null,
        classes:[],
        componentType:null,
        constructors:[null, null, null, null, null, ...],
        declaredAnnotations:[],
        declaredClasses:[],
        declaredConstructors:[null, null, null, null, null, ...],
        declaredFields:[null, null, null, null, null, ...],
        declaredMethods:[null, null, null, null, null, ...],
        declaringClass:null,
        enclosingClass:null,
        enclosingConstructor:null,
        enclosingMethod:null,
        enum:false,
        enumConstants:null,
        fields:[null, null, null, null, null, ...],
        genericInterfaces:[],
        genericSuperclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        interface:false,
        interfaces:[],
        localClass:false,
        memberClass:false,
        methods:[null, null, null, null, null, ...],
        modifiers:1,
        name:"org.metaworks.FieldDescriptor",
        package:null,
        primitive:false,
        protectionDomain:null,
        signers:null,
        simpleName:"FieldDescriptor",
        superclass:{annotation:false, annotations:[], anonymousClass:false, array:false, canonicalName:"org.metawo...", ...},
        synthetic:false,
        typeParameters:[]
```

```
      },
      displayName:"Order",
      inputFace:"org.metaworks.inputter.intInput",
      name:"order",
      viewFace:"faces/java/lang/Number.ejs"
    }
  ]
}
```

///// testing 하면서 찍었던 breakpoint들 //////

BasicObjectConverter (line: 284) - convertOutbound(Object, OutboundContext)
ChildNodesAction (line: 31) - handleRequest(HttpServletRequest, HttpServletResponse)
CreatorModule (line: 111) - executeMethod(MethodDeclaration, Object[])
MapAction (line: 28) - handleRequest(HttpServletRequest, HttpServletResponse)
MetaworksRemoteService (line: 14) - getMetaworksType(String)
SpringMindmapDAO (line: 1995) - getMapNode(int, boolean, String, String)
SpringMindmapDAO (line: 2017) - getMapNode(int, boolean, String, String)
SpringMindmapDAO (line: 2061) - getCurrentMapRevision(int)

//////////////// 주의 할 점 /////////////////

1. 서버가 잘 안내려가는 경우가 있으니 반영이 안되는 느낌이 있을때는 프로세스를 점검해서 확실히 죽여주라.
2. 서버 실행시 뜨는 다이얼로그를 꼭 보라. 반영하는데 실패했다는 오류가 있다 (보통 참조하는 라이브러리가 있을경우 디플로이 디펜던시를 외부에 갖고 있는 경우 발생. 이는 jar등으로 엮어서 내부에 넣어주니 ok)

```
"<div id='objDiv_0'>...  LOADING PROPERTY ...</div><script>   mw3.showObject("#ID001":faces/java/lang/String.ejs', 'objDiv_0');</script>"
```

////////////// 에러 유형들 //////////////

There is an error in your template (genericfaces/ObjectFace.ejs): Cannot read property 'fieldDescriptors' of undefined ==> 이건 해당 객체의 정보가 제공되지 않았을 경우 ObjectFace에서 뱉어내는 오류

팝업이 뜨면 ==> DWR오류이고,

팝업내에…. Object [object DOMWindow] … 가 언급되면, callback 메서드 내부에서 발생한 것이고, 앞의 표현은 리턴된 객체를 의미한다.

그 이후에는 해당 객체를 callback 메서드 내에서 참조하면서 나는 오류이다.

--- 백업 -------

메타웍스-어노테이티트 오브젝트는 DWR에 의하여 자신의 메타데이터를 얻어내는 getter인 .getType()을 멋있게 제공할 수 도 있고, 서버상에서 사용하는 패턴처럼 그냥 자바스크립트단에서 "ObjectType" 이라는 메타웍스2의 래퍼 클래스의 자바스크립트 버전을 제공할 수 도 있다. 당장은 심플한 후자를 만들자.

var objectType = new ObjectType('Person');

```
<script>
  showObject(<%=JSON.stringify(value[i])%>, '<%=objectTypeName%>', 'array_item_<%=i%>');
</script>
```

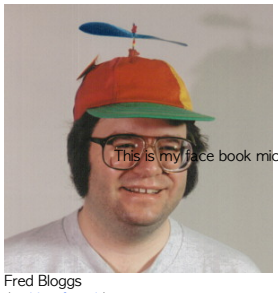
///////////////

#FaceBook.java 서비스

```
package org.metaworks.example;

import java.util.ArrayList;


public class FaceBook {

     static ArrayList<Posting> postings;

     public void post(Posting posting){
          postings.add(posting);
     }

     public ArrayList<Posting> getNewsFeed(){
          return postings;
     }
}
```

#dwr.xml 에 추가

```
     <create creator="new" javascript="FaceBook" scope="page">
       <param name="class" value="org.metaworks.example.FaceBook"/>
     </create>
```



Fred Bloggs
{ add to friend }

jinyoung Jang
{ add to friend }

〔이제 다음을 해볼 생각이다.. 〕

1. 리얼타임 채팅의 느낌을 위한 Long-Polling 도 이제 가볍게!
2. 스프링을 등장시켜서 서버 객체들을 보다 더 POJO답게 갖고 놀아보자
3. 데이터베이스 ORM 에 대해서 Hibernate와 iBatis 만이 답일까? 더 POJO답게 쓰는 방법은 없는가 - GenericDAO

```
    <servlet>
        <servlet-name>dwr-invoker</servlet-name>
        <!-- display-name>DWR Servlet</display-name -->
        <!--<servlet-class>org.directwebremoting.servlet.DwrServlet</servlet-class>-->
                <servlet-class>org.metaworks.dwr.TransactionalDwrServlet</servlet-class>

        <init-param>
            <param-name>defaultDS</param-name>
            <param-value>java:/uEngineDS</param-value>
        </init-param>
```

```
create table Person(
    name varchar(100),
    age int,
    portraitURL varchar(100)
);

create table Posting(
    writer varchar(100),
    document varchar(300)
);
```

---- 주의할 점 -----
IDAO는 Proxy object 이기 때문에 BeanConverter (id=bean)으로 그냥 interface를 넘기게 되면 beanconverter는 실패한다. DWR이 못난 것은 이오류를 그냥 'Error' 라고만 뱉는다는 것이다.

관련하여 DWR를 좀더 파해쳐봤다:

내장된 dwr.xml에 기존의 converter 들이 정의되어있는데, 여기에 이미 쓸만한 놈이 있을지 봤다: 그중에 제목이 'ProxyInterfaceConverter' 라는 놈이 있다.. 흥미로워 진다.

```
    DefaultJavascriptObject object = new DefaultJavascriptObject(session, id);

     return Proxy.newProxyInstance(paramType.getClassLoader(), new Class[] { paramType }, object);
```

이런 부분이 보이고, DefaultJavascriptObject는 InvocationHandler일것이라는 추측이 된다. 해당 invoke(…) 부분을 보면 이제 어찌 만들어졌는지 알 수 있겠지:

```
/* (non-Javadoc)
 * @see java.lang.reflect.InvocationHandler#invoke(java.lang.Object, java.lang.reflect.Method, java.lang.Object[])
 */
public Object invoke(Object proxy, Method method, Object[] args) throws Throwable
{
    if (method.getName().equals("equals") && args.length == 1)
    {
        return equals(args[0]);
    }

    if (method.getName().equals("hashCode") && (args == null || args.length == 0))
    {
        return hashCode();
    }

    if (method.getName().equals("toString") && (args == null || args.length == 0))
    {
        return toString();
    }
```

```
        ScriptBuffer script = EnginePrivate.getRemoteExecuteObjectScript(id, method.getName(), args);
        session.addScript(script);
        return null;
    }
```

음… 일종의 채팅에서 봤던 session을 가지고 자바스크립트단에 getter를 eval 한후에 그 값을 받아오거나, setting한다는 걸 알 수 있다.. 좀더 자세히 들여다 볼까?

#EnginePrivate.java

```
    public static ScriptBuffer getRemoteExecuteObjectScript(String id, String methodName, Object[] params)
    {
        ScriptBuffer script = new ScriptBuffer();
        script.appendCall("r.handleObjectCall", id, methodName, params);
        return script;
    }
```

그래… 'r'은 'remote' 를 암시하는 것 같고… 스크립트를 주입할 수 있도록 소켓을 열어두고, 값을 하나씩 취득하거나 명령을 주어 getter/setter의 행위를 가능하도록 유지하는구나..

이 이야기는 원격의 객체에 대한 일종의 완벽한 분산객체 시나리오를 DWR이 구현하려고 시도하고 있음을 알 수 있다. 하지만 long-polling으로 소켓을 열어야 하기 때문에 일반적인 용도로까지는 실행시 비용이 높을것 같다.

〔참고〕 원래 기대했던 목적인 인터페이스를 주고 받는 용도로 봤던 위 클래스는 허무한 결과로 파악되었다. 다음을 읽어보라.

```
    public OutboundVariable convertOutbound(Object data, OutboundContext outctx) throws ConversionException
    {
        throw new ConversionException(data.getClass(), "Interfaces can not be passed to a browser");
    }
```

응집도를 다시 높혀야 한다… javascript에서는 별도로 해당 모델에 대하여 프로토타입을 만들어주질 않았구나… 다음의 표현은 불가능하였다:

```
                Posting.prototype.post = function (){

                //   var document = mw3.getObjectValue(objId);

                     FaceBook.post(
                          this,
                          {
                               async: false,
                               callback: function(postId){

                               },
                          errorHandler:function(errorString, exception) {
                              alert('Errror in fb call ' + errorString);
                                    //document.getElementById(this.dwrErrorDiv).innerHTML = errorString;
                          }

                          }
                     );

                }
```

물론, 포함된 JavascriptProxy…라는 걸 써보면 뭔가 희한한게 될 것도 같은데… 그건 찬찬히 다음에 써보자… 어쨌든 ejs에 분리하여 응집도 높게 객체별로 사용될 스크립트를 나눠 담을 수 는 없을까… 혹은 지금은 그렇게 중요하지 않은것인가? 일단 나중에 다시 고민하기로 하자.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dwr PUBLIC "-//GetAhead Limited//DTD Direct Web Remoting 3.0//EN" "http://getahead.org/dwr/dwr30.dtd">


<dwr>

     <init>
         <converter id="metaworks" class="org.metaworks.dwr.MetaworksConverter"/>
     </init>

  <allow>



    <create creator="new" javascript="Metaworks" scope="page">
      <param name="class" value="org.metaworks.dwr.MetaworksRemoteService"/>
    </create>
    <create creator="new" javascript="FaceBook" scope="page">
      <param name="class" value="org.metaworks.example.FaceBook"/>
    </create>


     <convert converter="metaworks" match="org.metaworks.dwr.*"/>
     <convert converter="metaworks" match="org.metaworks.example.*"/>
     <convert converter="metaworks" match="java.lang.Class"/>
  </allow>

</dwr>
```

----- 이런 에러는 데이터 타입이 안맞는 경우에 특이하게 또 날 수 있다 -----------

function () { // If the string looks like an identifier, then we can return it as is. // If the string contains no control characters, no quote characters, and no // backslash characters, then we can simply slap some quotes around it. // Otherwise we must also replace the offending characters with safe // sequences. if (ix.test(this)) { return this; } if (/[&<"\\\x00-\x1f]/.test(this)) { return '"' + this.replace(/[&<"\\\x00-\x1f]/g, function (a) { var c = escapes[a]; if (c) { return c; } c = a.charCodeAt(); return '\\u00' + Math.floor(c / 16).toString(16) + (c % 16).toString(16); }) +

===〉 interface Domain extends IDAO 라도 해주던가..
아님 …

〔클리티컬〕

음… 오류를 먹는다. DWRServlet에서는.. 오류를 먹어버린다. CallHandler 특히, BasicPollHandler에서 reply 를 해당 서비스 빈의 로직호출 후에 얻어낸 후 그것을 stream으로 뱉고 끝이난다.
…. 지금은 이를 수정하려면 제법 많은 DWR자체의 리팩토링 작업이 소요될 것으로 직감이 된다. 혹은 Creator나 Handler등을 주입할 수 있으면 되는데, 그러지 않고는 DWR브랜치를 만드는 수 밖에 없는 상황에 봉착했다. 당장은 그냥 Spring + DWR 세팅을 사용하여 TX demarcation 이벤트를 받고 다음에 시간이 날때 이를 해결하자. 시간이 없다.

Daemon Thread〔http-bio-8080-exec-4〕(Suspended)
DefaultRemoter.execute(Calls) line: 332
PlainCallHandler(BaseCallHandler).handle(HttpServletRequest, HttpServletResponse) line: 104
UrlProcessor.handle(HttpServletRequest, HttpServletResponse) line: 120
TransactionalDwrServlet(DwrServlet).doPost(HttpServletRequest, HttpServletResponse) line: 141
TransactionalDwrServlet.doPost(HttpServletRequest, HttpServletResponse) line: 40
TransactionalDwrServlet(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 641
TransactionalDwrServlet(HttpServlet).service(ServletRequest, ServletResponse) line: 722
ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 304
ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 210
StandardWrapperValve.invoke(Request, Response) line: 224
StandardContextValve.invoke(Request, Response) line: 185
NonLoginAuthenticator(AuthenticatorBase).invoke(Request, Response) line: 472
StandardHostValve.invoke(Request, Response) line: 151
ErrorReportValve.invoke(Request, Response) line: 100
AccessLogValve.invoke(Request, Response) line: 929
StandardEngineValve.invoke(Request, Response) line: 118
CoyoteAdapter.service(Request, Response) line: 405
Http11Processor.process(SocketWrapper〈Socket〉) line: 269
Http11Protocol$Http11ConnectionHandler(AbstractProtocol$AbstractConnectionHandler〈S,P〉).process(SocketWrapper〈S〉, SocketStatus) line: 515
JIoEndpoint$SocketProcessor.run() line: 302
ThreadPoolExecutor$Worker.runTask(Runnable) line: 886
ThreadPoolExecutor$Worker.run() line: 908
TaskThread(Thread).run() line: 637

그리고 참고로 오류는 별도의 로그파일에 쌓이고 있다. 이유없이 'Error'만 alert으로 화면에 뜨는경우, 이 경우가 서버측 오류가 있는 경우고, 오류는 서버로그파일을 봐야한다.

#Posting.java에 프로퍼티 추가

```
boolean likeIt;
	public boolean isLikeIt() {
		return likeIt;
	}
	public void setLikeIt(boolean likeIt) {
		this.likeIt = likeIt;
	}
```

#index.html 에 like method 추가

```
		function like(objId){
			var likePosting = mw3.getObject(objId);

			FaceBook.like(
					likePosting,
					{
						async: false,
						callback: function(posting){
							mw3.setObject(objId, posting, "org.metaworks.example.Posting");
						}
					}
			 );
		}


<%
	if(mw3.when == mw3.WHEN_EDIT){
		value.document = 'Share your status...';
	}
%>

<table>

	<tr>
		<td><%=mw3.locateObject(value.writer, 'org.metaworks.example.Person')%></td>
		<td><%=mw3.locateObject(value.document, 'java.lang.String')%>
```

```
<%
	if(mw3.when == mw3.WHEN_EDIT){
%>
	<input type="button" value="SHARE" onclick="post('<%=value.writer.name%>', '<%=mw3.objectId%>')">
<%
	}
%>
		<script>
			$("#mapped_input_for_<%=mw3.objectId%>").focus();
		</script>

<%
	if(value.likeIt){
%>
You like this
<% }else{%>

<a onclick="mw3.call(<%=objectId%>, 'FaceBook', 'like')">like</a>

<%}%>


		</td>
<%
	if(mw3.when == mw3.WHEN_EDIT){
%>
		<td><a onclick="updatePosting('<%=objectId%>')">save</a></td>
<%
	}else{
%>
		<td><a onclick="<%=editFunction%>">edit</a></td>

<%
	}
%>

	</tr>

</table>
```

〔TODO〕

1. primitive type 에 대하여 convertInbound나, convertOutbound상에서 null 값인 경우 setter method를 제네릭하게 호출 못하여 오류가 나는 문제 =〉 근본적으로는 convertOutbound에서 해당 클래스의 기본값을 넣어서 넣어주어야 할 것임 - 논리적으로 맞음. 혹은 받을때 적절히 하는 것도 잠재적 오류를 원천 차단하는 측면에서 맞음. 둘다 처리하는게 맞겠음.

```
String upperPropertyName = propertyName.substring(0, 1).toLowerCase() + propertyName.substring(1);
Class propertyClass = webObjectType.metaworks2Type.getFieldDescriptor(upperPropertyName).getClassType();
if(value==null && propertyClass.isPrimitive()){
	value = propertyClass.newInstance();
}
```

2. DAO 밸류 오브젝트에 대하여 어떠한 key descriptor 도 설정되지 않은 경우와 하나 이상이 설정된 경우의 처리. ==〉 하나도 없으면 적절한 워닝혹은 오류를, 두개이상이면 다중키 처리를

```
FaceBook.like(object,
    {async: false,
     callback: function(obj){
        mw3.setObject(objId, obj)
     }
    }
);

$('#loading').hide();
```

〔Posting객체의 응집도를 높히자. 객체란 속성과 행위를 동시에 가지고 있어야 한다〕

```
    <create creator="new" javascript="PostingService" scope="page">
      <param name="class" value="org.metaworks.example.PostingService"/>
    </create>
```

다음과 같은 PostingService 클래스를 만들어주면, 메타웍스는 알아서 Posting 객체에 대한 행위들이 여기에 존재한다고 인식하자:

```
package org.metaworks.example;

public class PostingService{

	public IPosting post(IPosting posting) throws Exception{
		return posting;
	}

	public IPosting like(Posting posting) throws Exception{
		return posting;
	}

}
```

패턴은, 자신을 받아서 행위를 수행한 후, 자신을 리턴해야 한다. 물론, POJO 스럽진 않다.. 아숩게도.. 다음과 같이 POJO스러운 객체로만 움직이게 만드려면, 당장 2가지가 해결되어야 하기 때문이다:

첫째, DWR이 데이터 객체에 대해서도 서비스로 만들 수 있는 길을 열어주어야 한다. 물론, 밸류오브젝트 자체를 서비스 객체로 등록해서 쓸 수도 있다. 그렇지만, 결국 위에서처럼, 자기자신을 메서드로 받고, 자기자신을 리턴해야만 한다. 자신의 멤버변수를 바로 참고 하여 서비스하는 모형이 어차피 아니다.

둘째, IDAO와 같은 방식으로 가상객체를 만들어 dirty field세팅등을 주입하기가 어렵다. 메타웍스 GenericDAO는 Proxy 객체를 뒷단에 만들어놓고, setter에 의하여 key 매핑된 정보를 dirty flag에 표시하여 추후 변경된 필드만 update 하여 성능을 최적화 한다. 이 과정을 메타웍스는 후킹하지 않고서는 그러한 행위를 수행하기 어렵다. 물론, class 서비스객체 implements 밸류인터페이스… 이러한 방식으로 구현체와 인터페이스를 공존 시킬 수 있다. 그러나 위에서 이야기한대로, 자신의 프로퍼티 값을 참조하지 못하는 마당에 행위가 자신의 객체와 같이 있어서 얻어지는 응집도상의 장점은 별로 없다.

음.. 정말그런가?

```
@org.metaworks.Metadata(tableName="Posting")
public interface IPosting extends IDAO{

  @org.metaworks.Metadata(isKey=true)
  public String getPostId();
  public void setPostId(String postId);

  public boolean isLikeIt();
  public void setLikeIt(boolean likeIt);
}

@org.metaworks.MetaworksObject
public class Posting implements IPosting{
  String postId;
  public void post(){ Database.create(this); }
  public void like() { setLikeIt(true); Database.put(this); }
  public IPosting findAll(){ return (IPosting)Database.sql("select * from Posting"); }

/// just one cute dummy ////

  public AbstractGenericDAO dao(){return null;}
}
```

혹은,

```
 public class Posting extends MetaworksObject implements IPosting{
  String postId;

  public void like() { setLikeIt(true); save(); }

}
```

끝.

MetaworksObject에는 기본적으로, create(), save(), load() 등이 준비되어 있음. 물론, dummy인 dao()도 구현되어 있공..

===〉 이렇게 한후에, 스크립트에서:

mw3('value.post()') 〈= 전역메서드로 그냥 만들면 뭐.. 안될것 없지.

혹은 EJS를 뜯어다가.. 〈%value.post()%〉를 만나면, 즉, value로 시작하고, 끝이 ()로 끝나는데, 해당 메서드가 스크립트단에 없다면, 알아서 호출 구문으로 바꿔줄 수 도 있다.

==〉이렇게 해놓고, 걍 expose 시키면..

자바스크립트로…
"posting.post()" 이런 표현은 물론 못쓰지만..

mw3.call(posting, 'post'); 는 가능하고… 때로는 mw3.call('posting.post()') 가 가능해진다.. 혹은 ejs내에서는 그냥 mw3.call('objectId', 'post') 이거 유효하다. Service 객체로 만들지말고…

MetaworksService.call(Object object, String methodName)… 을 주고, Posting이 대신 메타웍스 어노테이션으로 "org.metaworks.MetaworksObject" 만 떡하니 정의됐다면, 이를 리플랙션으로 호출해주자:

그러면 이 'object'에서 알아서 서비스 클래스를 얻어낼 수 있을까? 그리고 상속구조는? object는 __className을 힌트로 갖고 있다. 그걸 이용하면 서비스객체의 매핑은 얻어낼 수 있다. (대신 클래스들을 뒤져야 한다)

====〉 오브젝트 모델은 좋은데, 대량 처리에 있어서는 좀 한계가 분명하게 보이긴 하다. 음.. 그래도 나쁘지 않은 접근이긴 해… 대량의 경우도 다음과 같이..

〈그래, 한번 해보자!〉

```
    String value = data.getValue();
       value = value.substring(1, value.length() - 1);

       Map<String, String> tokens = extractInboundTokens(paramType, value);
```

```
        String refName = tokens.get("__className");
        refName = refName.split(":")[1];

          String className = data.getContext().getInboundVariable(refName).getFormField().getString();
```

AbstractGenericDAO (line: 847) - invoke(Object, Method, Object[])
BasicObjectConverter (line: 58) - convertInbound(Class<?>, InboundVariable)
BasicObjectConverter (line: 284) - convertOutbound(Object, OutboundContext)
BasicObjectConverter (line: 313) - convertOutbound(Object, OutboundContext)
BeanConverter (line: 52) - getPropertyMapFromClass(Class<?>, boolean, boolean)
ChildNodesAction (line: 31) - handleRequest(HttpServletRequest, HttpServletResponse)
ConversionException (line: 30) - ConversionException(Class<?>)
ConversionException (line: 60) - ConversionException(Class<?>, String, Throwable)
CreatorModule (line: 111) - executeMethod(MethodDeclaration, Object[])
DefaultRemoter (line: 348) - execute(Call)
DefaultRemoter (line: 399) - execute(Call)
DefaultRemoter (line: 406) - execute(Call)
DwrServlet (line: 66) - init(ServletConfig)
MapAction (line: 28) - handleRequest(HttpServletRequest, HttpServletResponse)
MetaworksConverter (line: 54) - convertInbound(Class<?>, InboundVariable)
MetaworksConverter (line: 95) - convertOutbound(Object, OutboundContext)
MetaworksRemoteService (line: 43) - callMetaworksService(String, Object, String)
ScriptBufferUtil (line: 74) - createOutput(ScriptBuffer, ConverterManager, boolean)
SpringCreator (line: 126) - getInstance()
SpringCreator (line: 128) - getInstance()
SpringCreator (line: 163) - getBeanFactory()
SpringMindmapDAO (line: 1995) - getMapNode(int, boolean, String, String)
SpringMindmapDAO (line: 2017) - getMapNode(int, boolean, String, String)
SpringMindmapDAO (line: 2061) - getCurrentMapRevision(int)
TransactionContext (line: 214) - beforeCommit()
TransactionalDwrServlet (line: 45) - doPost(HttpServletRequest, HttpServletResponse)
TransactionalDwrServlet (line: 60) - doPost(HttpServletRequest, HttpServletResponse)

Daemon Thread [http-bio-8080-exec-12] (Suspended (breakpoint at line 399 in DefaultRemoter))
    DefaultRemoter.execute(Call) line: 399
    DefaultRemoter.execute(Calls) line: 332
    PlainCallHandler(BaseCallHandler).handle(HttpServletRequest, HttpServletResponse) line: 104
    UrlProcessor.handle(HttpServletRequest, HttpServletResponse) line: 120
    TransactionalDwrServlet(DwrServlet).doPost(HttpServletRequest, HttpServletResponse) line: 141
    TransactionalDwrServlet.doPost(HttpServletRequest, HttpServletResponse) line: 43
    TransactionalDwrServlet(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 641
    TransactionalDwrServlet(HttpServlet).service(ServletRequest, ServletResponse) line: 722
    ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 304
    ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 210
    StandardWrapperValve.invoke(Request, Response) line: 224
    StandardContextValve.invoke(Request, Response) line: 185
    NonLoginAuthenticator(AuthenticatorBase).invoke(Request, Response) line: 472
    StandardHostValve.invoke(Request, Response) line: 151
    ErrorReportValve.invoke(Request, Response) line: 100
    AccessLogValve.invoke(Request, Response) line: 929
    StandardEngineValve.invoke(Request, Response) line: 118
    CoyoteAdapter.service(Request, Response) line: 405
    Http11Processor.process(SocketWrapper<Socket>) line: 269
    Http11Protocol$Http11ConnectionHandler(AbstractProtocol$AbstractConnectionHandler<S,P>).process(SocketWrapper<S>, SocketStatus)
line: 515
    JIoEndpoint$SocketProcessor.run() line: 302
    ThreadPoolExecutor$Worker.runTask(Runnable) line: 886
    ThreadPoolExecutor$Worker.run() line: 908
    TaskThread(Thread).run() line: 637

```
function ( result ){ // alert(webObjectType.name + "=" + dwr.util.toDescriptiveString(webObjectType, 5)) this.setObject(objId, result); $
('#loading').hide(); }
```

```
public class Posting implements IPosting{


    Person writer;
        public Person getWriter() {
            return writer;
        }
        public void setWriter(Person writer) {
            this.writer = writer;
        }

    String document;
        @org.metaworks.Metadata(isKey = true)
```

```
	public String getDocument() {
		return document;
	}
	public void setDocument(String document) {
		this.document = document;
	}

boolean likeIt;
	public boolean isLikeIt() {
		return likeIt;
	}
	public void setLikeIt(boolean likeIt) {
		this.likeIt = likeIt;
	}



public void like() throws Exception{

	setLikeIt(true);

	IPosting postingInDatabase;
	postingInDatabase = (IPosting) Database.get(IPosting.class, getDocument());
	postingInDatabase.setLikeIt(true);

}



package org.metaworks.example;

import org.metaworks.dwr.MetaworksObject;

public class Posting extends MetaworksObject<IPosting> implements IPosting{


Person writer;
	public Person getWriter() {
		return writer;
	}
	public void setWriter(Person writer) {
		this.writer = writer;
	}

String document;
	@org.metaworks.Metadata(isKey = true)
	public String getDocument() {
		return document;
	}
	public void setDocument(String document) {
		this.document = document;
	}

boolean likeIt;
	public boolean isLikeIt() {
		return likeIt;
	}
	public void setLikeIt(boolean likeIt) {
		this.likeIt = likeIt;
	}



public void like() throws Exception{

	setLikeIt(true);
	getDatabaseMe().setLikeIt(isLikeIt());

}


}



#AbstractGenericDAO.java

		if(propertyName.length()==0 && args.length==2){
			propertyName = args[0].toString().toUpperCase();
			cache.put(propertyName, args[1]);
		}else{//TODO: type check for typed DAO

			if(getSynchronizedObject()!=null){
				WebObjectType wot = MetaworksRemoteService.getMetaworksType(getSynchronizedObject().getClass().getName());
				ObjectInstance instance = (ObjectInstance) wot.metaworks2Type().createInstance();
				instance.setObject(getSynchronizedObject());
				instance.setFieldValue(propertyName, args[0]);

			}

			cache.put(propertyName.toUpperCase(), args[0]);
		}
```

〈DWR의 수정〉

```
public abstract class BaseCallHandler extends BaseDwrpHandler
{
    /* (non-Javadoc)
     * @see org.directwebremoting.Handler#handle(javax.servlet.http.HttpServletRequest, javax.servlet.http.HttpServletResponse)
     */
    public void handle(HttpServletRequest request, HttpServletResponse response) throws IOException
    {
```

```
        try
        {
            CallBatch batch = new CallBatch(request);

            // Security checks first, once we've parsed the input
            checkGetAllowed(batch);
            boolean checkCsrf = false;
            for (int i = 0; i < batch.getCalls().getCallCount(); i++)
            {
                Call call = batch.getCalls().getCall(i);
                if (!ExportUtil.isUnprotectedSystemMethod(call.getScriptName(), call.getMethodName()))
                {
                    checkCsrf = true;
                    break;
                }
            }
            if (checkCsrf) {
                checkNotCsrfAttack(request, batch);
            }

            // Save the batch so marshallException can get at a batch id
            request.setAttribute(ATTRIBUTE_BATCH, batch);

            String normalizedPage = pageNormalizer.normalizePage(batch.getPage());
            RealWebContext webContext = (RealWebContext) WebContextFactory.get();
            webContext.checkPageInformation(normalizedPage, batch.getScriptSessionId(), batch.getWindowName());

            // Various bits of the CallBatch need to be stashed away places
            storeParsedRequest(request, webContext, batch);

            Calls calls = marshallInbound(batch);

            Replies replies = remoter.execute(calls);

          ////// 추가 ////////
            try{
                org.metaworks.dao.TransactionContext.getThreadLocalInstance().setSharedContext("replies", replies);
            }catch(Exception e){}

          ////// 끝 ///////////

            marshallOutbound(replies, response);
        }
        catch (Exception ex)
```

잘 설명하는 그게 문제입니다!.... 한번해보자면...

클라이언트 어플리케이션에서는 객체들의 배치와 이동에 대해서 모델 중심으로 사고하여 개발할 수 있고, 각자 객체별 UI에 대한 고민은 ejs 파일에 화려한 자바스크립트 신공을 쏟아붇게 할 수 있고, 서버상에서 벌어질 작업은 서버객체내에서 하고픈 대로 하고 나면 알아서 DB에 보존되는, '역할의 분리 & 구현의 복잡성을 낮추는'가 체계화된 프레임워크입니다. 역할의 분리가 잘되면 분업화도 잘되고, 혼자 일을 하더라도 적어도 로직이 반복되거나 메타데이터가 반복관리되는 것을 막아주는 경계를 잘 형성해줍니다. 요술은 아니고요... ^^

현장 까지는 아니고, 지식 산업 집약적인 SW개발 -〉 광고, 특히 오픈이노베이션을 생각하는 광고/홍보 팀.. 팀별로 확산시키는 전략. UX에 대한 신경. 오픈서비스..

SNS는 창의적/자발적인 지식활동 동기부여 / 기업 내부… =〉 깊이 있는 지식 활동을 보호하고/타임라인을 넘어가 버리는 상황들… 기업들이 당면한 상황… ADHD?? 주의력이 결핍되는 문제들.

내부에 신중한 실험들 많이 하고 있음. 문화에 녹일 수 있을 방법은 무엇인가?

1. 좀더 학습을 돋보이게 해야 한다. ==〉 학습 LMS와 연결성을 높여야
2. 인적 개발에 포커스된 것 ==〉 업무 프로세스에 포커스 됨. ==〉 직무분석 기능을 실질적으로 연결적으로 보여줬어야.
3. 복합지식에 특화된 것이 2세부에서 무엇이 연결된 것인지.. ==〉 온톨로지와 마맵 연결 부분은 보여줬어야 함.
4. 2단계의 연결성을 잘 모르겠다 ==〉 UX 분석 및 개선 작업이 추가되고 회의관리 같은 지엽적인 투-두작업은 배제하는 것이 좋을 것.

고객은 제품을 보기전에 그것이 필요한 것인지를 모른다 〈-〉 고객이 돈을 지불하지 않을 것은 절대 만들지 마라
===〉 결국은 고객의 페인 포인트 분석에 집중해야 한다.

```
    //<%=mw3.objectId%>post('<%=value.writer.name%>', '<%=mw3.objectId%>')">
```

```
{
  IDAOClass:{
    __className:"java.lang.Class",
    annotation:false,
    annotations:{
      __className:"[Ljava.lang.annotation.Annotation;"
    },
    anonymousClass:false,
    array:false,
    canonicalName:"org.metaworks.example.IPosting",
    classLoader:{
      URLs:{
        __className:"[Ljava.net.URL;"
      },
      __className:"org.apache.catalina.loader.WebappClassLoader",
      antiJARLocking:false,
      clearReferencesHttpClientKe...:true,
      clearReferencesLogFactoryRe...:true,
      clearReferencesStatic:false,
      clearReferencesStopThreads:false,
      clearReferencesStopTimerThr...:false,
      contextName:"/mw3",
```

```
      delegate:false,
      jarPath:"/WEB-INF/lib",
      parent:{
        URLs:{__className:"[Ljava.net..."},
        __className:"org.apache.catalina.loader.StandardClassLoader",
        parent:{URLs:Object, __className:"sun.misc.L...", parent:Object}
      },
      resources:{
        __className:"org.apache.naming.resources.ProxyDirContext",
        cache:{__className:"org.apache...", accessCount:671, cacheMaxSize:10240, cacheSize:452, desiredEntryAccessRatio:3, ...},
        contextName:"/mw3",
        contextPath:"/mw3",
        dirContext:{__className:"org.apache...", aliases:, allowLinking:false, cacheMaxSize:10240, cacheObjectMaxSize:512, ...},
        docBase:"/Users/jyjang/Documents/workspace/.metadata/.plugins/org.eclipse.wst.server.core/tmp0/wtpwebapps/mw3",
        environment:{},
        hostNam...s:Object, __className:"org.apache...", parent:Object},
        resources:{__className:"org.apache...", cache:Object, contextName:"/mw3", contextPath:"/mw3", dirContext:Object, ...},
        searchExternalFirst:false,
        started:true,
        state:{__className:"org.apache...", available:false, declaringClass:Object, lifecycleEvent:null},
        stateName:"NEW"
      },
      codeSource:{
        __className:"java.security.CodeSource",
        certificates:null,
        codeSigners:null,
        location:"file:/Users/jyjang/Documents/workspace/.metadata/.plugins/org.eclipse.wst.server.core/tmp0/wtpwebapps/mw3/WEB-INF/classes/
org/metaworks/example/IPosting.class"
      },
      permissions:{
        __className:"java.security.Permissions",
        readOnly:true
      },
      principals:{
        __className:"[Ljava.security.Principal;"
      }
    },
    signers:null,
    simpleName:"IPosting",
    superclass:null,
    synthetic:false,
    typeParameters:{
      __className:"[Ljava.lang.reflect.TypeVariable;"
    }
  },
  __className:"org.metaworks.WebObjectType",
  faceComponentPath:"genericfaces/ObjectFace.ejs",
  fieldDescriptors:{
    __className:"[Lorg.metaworks.WebFieldDescriptor;"
  },
  keyFieldDescriptor:{
    __className:"org.metaworks.WebFieldDescriptor",
    className:"java.lang.String",
    displayName:"Document",
    inputFace:"org.metaworks.inputter.TextInput",
    name:"document",
    viewFace:"faces/java/lang/String.ejs"
  }
}
```

---- Converter 찾기 로직 ------

```
Daemon Thread [http-bio-8080-exec-2] (Suspended)
     DefaultConverterManager.getConverterAssignableFrom(Class<?>) line: 657
     DefaultConverterManager.getConverter(Class<?>) line: 529
     DefaultConverterManager.getConverter(Object) line: 498
     DefaultConverterManager.convertOutbound(Object, OutboundContext) line: 433
     MetaworksConverter.convertOutbound(Object, OutboundContext) line: 149
     DefaultConverterManager.convertOutbound(Object, OutboundContext) line: 441
     ScriptBufferUtil.createOutput(ScriptBuffer, ConverterManager, boolean) line: 68
     BaseCallHandler$CallScriptConduit.addScript(ScriptBuffer) line: 485
     PlainCallHandler(BaseCallHandler).marshallOutbound(Replies, HttpServletResponse) line: 345
     PlainCallHandler(BaseCallHandler).handle(HttpServletRequest, HttpServletResponse) line: 105
     UrlProcessor.handle(HttpServletRequest, HttpServletResponse) line: 120
     TransactionalDwrServlet(DwrServlet).doPost(HttpServletRequest, HttpServletResponse) line: 141
     TransactionalDwrServlet.doPost(HttpServletRequest, HttpServletResponse) line: 43
     TransactionalDwrServlet(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 641
     TransactionalDwrServlet(HttpServlet).service(ServletRequest, ServletResponse) line: 722
     ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 304
     ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 210
     StandardWrapperValve.invoke(Request, Response) line: 224
     StandardContextValve.invoke(Request, Response) line: 185
     NonLoginAuthenticator(AuthenticatorBase).invoke(Request, Response) line: 472
     StandardHostValve.invoke(Request, Response) line: 151
     ErrorReportValve.invoke(Request, Response) line: 100
     AccessLogValve.invoke(Request, Response) line: 929
     StandardEngineValve.invoke(Request, Response) line: 118
     CoyoteAdapter.service(Request, Response) line: 405
     Http11Processor.process(SocketWrapper<Socket>) line: 269
     Http11Protocol$Http11ConnectionHandler(AbstractProtocol$AbstractConnectionHandler<S,P>).process(SocketWrapper<S>, SocketStatus) line: 515

     JIoEndpoint$SocketProcessor.run() line: 302
     ThreadPoolExecutor$Worker.runTask(Runnable) line: 886
     ThreadPoolExecutor$Worker.run() line: 908
     TaskThread(Thread).run() line: 637
```

```
beanExpression[beanExpression.length] =
{
     parentObjectId  : objectId,
     fieldName       : fd.name
     valueObjectId   : mw3.objectId
}
```

MetaworksConverter (extends BeanConverter) 가 Marshalling을 할때 여러가지 잠재적 오류가 나는 이유중 하나가.. primitive 형들의 변환이다: MetaworksConverter에는 다음의 처리가 이루어져있다:

```
if(propertyClass == int.class)
                                                  value = new Integer(0);
                                             else if(propertyClass == boolean.class)
                                                  value = new Boolean(false);
                                             else {
                                                  new Exception("please add more primitive type's default value mapping for " +
propertyClass).printStackTrace();
                                             }
```

이 처리를 inbound (JSON -〉 Java)에서도 처리를 해줘야 한다. 아니면, 아래와 같은 호출을 할때에는 모든 primitive 형들의 값을 채워서 서버로 날려야 한다:

〈Metaworks 3 기본 Breakpoints 설정:〉

1. inbound/outbound 마샬링 관련 오류 검출:
여기에 break point:
     ConversionException [line: 60] - ConversionException(Class<?>, String, Throwable)

Expressions 에 다음을 추가:
     ex.printStackTrace()

2.

<Metaworks Object 개발 주요 가이드라인〉

1. Database 동기화가 벌어지는 모든 객체는
1.1. interface I객체명 extends IDAO 를 따른다.
1.2. 구현객체는 class 객체명 extends MetaworksObject〈I객체명〉 implements I객체명 을 따른다.
1.3. 모든 annotation은 interface 객체에 두며, 각 필드에 대한 정보는 getter에만 준다. (메타웍스는 setter를 참조하지 않음)

2. Control 객체는 1을 따르지 않는 순수한 POJO객체를 사용할 수 있지만 다음과 같은 규칙을 따라야 한다.
2.1. 서비스 객체는 빈 constructor를 가져야 한다. 값을 실어나르기 위하여 리플렉션을 사용하기 때문이다.
2.1. 서비스메서드 (ServiceMethod)는 아규먼트가 없어야 한다. 아규먼트를 넘기고 싶다면 해당 클래스에 프로퍼티를 추가한다.
2.2. 페이지 네비게이션은 서비스 메서드의 리턴으로 오브젝트를 전환시킴으로서 깜빡임이 없이 div를 이동하게 한다.
2.3. 리턴 오브젝트는 key field가 반드시 존재해야 리턴된 후 자신의 위치를 찾아서 변환된다
2.4. 서비스 메서드는 기본적으로 isCallByContent = false이다. 이말은, request가 이루어질때는 키값(그리고 context 정보)만 포함된 오브젝트로 요청이 서버에 전달되며, 리턴시에는 전체값이 다운로드된다. 내용 수정, 내용 초기 submit 등이 이루어질때는 isCallByContent=true로 준다.
2.5. 2.4.에서 키값이외의 단순 아규먼트 몇가지를 넘기기 위한 용도라면 모든 객체에 포함되어 전달되는 context 객체에 아규먼트를 심어서 보내는 것이 방법중 하나이다 (객체.metaworksContext).

```
org.metaworks.example.Login={
  __className:"org.metaworks.WebObjectType",
  displayName:"Login",
  faceComponentPath:"genericfaces/ObjectFace.ejs",
  fieldDescriptors:[
    {
      __className:"org.metaworks.WebFieldDescriptor",
      boolOptions:null,
      className:"java.lang.String",
      displayName:"Message",
      inputFace:"org.metaworks.inputter.TextInput",
      name:"message",
      options:null,
      values:null,
      viewFace:null
    },
    {
      __className:"org.metaworks.WebFieldDescriptor",
      boolOptions:null,
      className:"java.lang.String",
      displayName:"Password",
      inputFace:"org.metaworks.inputter.TextInput",
      name:"password",
      options:null,
      values:null,
      viewFace:null
    },
    {
      __className:"org.metaworks.WebFieldDescriptor",
```

```
      boolOptions:null,
      className:"java.lang.String",
      displayName:"UserId",
      inputFace:"org.metaworks.inputter.TextInput",
      name:"userId",
      options:null,
      values:null,
      viewFace:null
    },
    {
      __className:"org.metaworks.WebFieldDescriptor",
      boolOptions:null,
      className:"org.metaworks.MetaworksContext",
      displayName:"MetaworksContext",
      inputFace:"org.metaworks.inputter.ObjectInput",
      name:"metaworksContext",
      options:null,
      values:null,
      viewFace:null
    }
  ],
  keyFieldDescriptor:{
    __className:"org.metaworks.WebFieldDescriptor",
    boolOptions:null,
    className:"java.lang.String",
    displayName:"UserId",
    inputFace:"org.metaworks.inputter.TextInput",
    name:"userId",
    options:null,
    values:null,
    viewFace:null
  },
  name:"org.metaworks.example.Login",
  serviceMethodContexts:{
    subscribe:{
      __className:"org.metaworks.ServiceMethodContext",
      callByContent:true,
      methodName:"subscribe",
      target:null,
      when:"new",
      where:"wherever"
    },
    login:{
      __className:"org.metaworks.ServiceMethodContext",
      callByContent:true,
      methodName:"login",
      target:null,
      when:"whenever",
      where:"wherever"
    }
  }
}


call.argument={
  __className:"org.metaworks.example.MM_ACL",
  aclId:22,
  all:[
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:0,
      metaworksContext:null,
      nodeId:"1",
      permission:"xxx"
    },
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:5,
      metaworksContext:null,
      nodeId:"1",
      permission:"xxx"
    },
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:6,
      metaworksContext:null,
      nodeId:"1",
      permission:"xxx"
    },
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:7,
      metaworksContext:null,
      nodeId:"1",
      permission:null
    },
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:8,
      metaworksContext:null,
      nodeId:"1",
```

```
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:9,
    metaworksContext:null,
    nodeId:"1",
    permission:"M"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:10,
    metaworksContext:null,
    nodeId:"1",
    permission:"M"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:11,
    metaworksContext:null,
    nodeId:"1",
    permission:"M"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:12,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metawork...     permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:14,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:15,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:16,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:17,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:18,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:19,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
```

디버

```
    __className:"org.metaworks.example.IMM_ACL",
    aclId:20,
    metaworksContext:null,
    nodeId:"1",
    permission:"R"
  },
  {
    __className:"org.metaworks.example.IMM_ACL",
    aclId:21,
    metaworksContext:null,
    nodeId:"1",
```

```
      permission:"R"
    },
    {
      __className:"org.metaworks.example.IMM_ACL",
      aclId:22,
      metaworksContext:null,
      nodeId:"1",
      permission:"R"
    }
  ],
  implementationObject:null,
  metaworksContext:null,
  nodeId:"1",
  permission:"R"
}
```

http://icant.co.uk/csstablegallery/tables/51.php

디버깅 포인트

```
                for(var i=beanPaths.length -1; i>-1; i--){
                    var beanPath = beanPaths[i];
                    eval("this.objects[objectId]" + beanPath.fieldName + "=this.getObject('" + beanPath.valueObjectId + "')");
                }
```

여기서 beanPaths를 잘 보라. 2동일한 필드명에 2개 이상이 있지 않은가? 값이 안바뀌는 원인이 될 수 있다. 그리고 그 원인은 "newBeanProperty()"메서드가 제대로 혹은, 제 시간이 불려지지 않았기 때문이다.

========= 메타웍스 3 설치 ============

#Web.xml

```
  <servlet>
    <servlet-name>dwr-invoker</servlet-name>
    <servlet-class>org.metaworks.dwr.TransactionalDwrServlet</servlet-class>
    <init-param>
      <param-name>defaultDS</param-name>
      <param-value>java:/uEngineDS</param-value>
    </init-param>
    <init-param>
      <param-name>accessLogLevel</param-name>
      <param-value>CALL</param-value>
    </init-param>
    <init-param>
      <param-name>debug</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>activeReverseAjaxEnabled</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>initApplicationScopeCreatorsAtStartup</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>jsonRpcEnabled</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>jsonpEnabled</param-name>
      <param-value>true</param-value>
    </init-param>
    <init-param>
      <param-name>preferDataUrlSchema</param-name>
      <param-value>false</param-value>
    </init-param>
    <init-param>
      <param-name>maxWaitAfterWrite</param-name>
      <param-value>-1</param-value>
    </init-param>
    <load-on-startup>1</load-on-startup>
  </servlet>
  <servlet-mapping>
    <servlet-name>dwr-invoker</servlet-name>
    <url-pattern>/dwr/*</url-pattern>
  </servlet-mapping>
```

#WEB-INF/dwr.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE dwr PUBLIC "-//GetAhead Limited//DTD Direct Web Remoting 3.0//EN" "http://getahead.org/dwr/dwr30.dtd">
```

```
<dwr>

    <init>
        <converter id="metaworks" class="org.metaworks.dwr.MetaworksConverter"/>
 </init>

  <allow>



    <create creator="new" javascript="Metaworks" scope="page">
      <param name="class" value="org.metaworks.dwr.MetaworksRemoteService"/>
    </create>
    <create creator="new" javascript="FaceBook" scope="page">
      <param name="class" value="org.metaworks.example.FaceBook"/>
    </create>


    <convert converter="bean" match="org.metaworks.dwr.*"/>
    <convert converter="metaworks" match="org.metaworks.example.*"/>
    <convert converter="metaworks" match="java.lang.Class"/>
    <convert converter="metaworks" match="java.lang.Object"/>

    <convert match="java.lang.Exception" converter="exception"/>


  </allow>

</dwr>
```

그리고 dwr.jar, metaworks3.jar를 web-inf/lib 아래에 넣어줌

〈DWR서비스 확인〉

http://localhost:8080/uengine-web/dwr/index.html 로 접속하여 DWR서비스가 제대로 다음과 같이 기동되었는가를 확인:

- Classes known to DWR:

- Metaworks (org.metaworks.dwr.MetaworksRemoteService)
- JavascriptChat (com.okmindmap.collaboration.JavascriptChat)
- FaceBook (org.metaworks.example.FaceBook)

〔Tip1〕 아무에러로그도 출력되지 않는 경우라면:
실제 WAS에 포함된 정확한 web.xml에 TransactionalDwrServlet이 설정되었는지 확인하라. 기존 DwrServlet이 설정되었다면 오류는 DWR로그에만 남는다. Std out으로 출력되는 오류가 있다면 TransactionalDwrServlet이 설정된 경우다.

------ 뭔가 이상하다 ------

```
jdbc/uEngineDS
```

이걸로 넣어야 하는 경우가 있다.. server.xml에서 java:/uEngineDS였다라고 가정하면 말이다.

** here() of undefined ==〉 몇가지 경우가 있다
1. @ServiceMethod가 Interface에 넣어주지 않고 상속 클래스에만 정의한 경우 -〉 인터페이스에도 넣어주고, 인터페이스가 우선이 되니, 거기에 @ServiceMethod를 준다
2. 정말 이름이 틀린경우. 대소문자를 구분하여 확인한다.

```
    <bean class="org.metaworks.spring.TransactionContextAdvice" />  ==>이거이 꼭 필요한지는… 지금 확인해 볼것임.


    <bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
        destroy-method="close">

        <property name="driverClassName" value="com.mysql.jdbc.Driver" />
        <property name="url" value="jdbc:mysql://localhost:3306/uengine?
autoReconnect=true&amp;useUnicode=true&amp;characterEncoding=UTF8" />
        <property name="username" value="root" />
        <property name="password" value="" />
        <property name="validationQuery" value="SELECT 1" />

    </bean>



    <bean id="springConnectionFactoryForMetaworks"
```

```
        class="org.metaworks.spring.SpringConnectionFactory">
        <property name="datasource" ref="datasource"/>
    </bean>
    <bean id="MetaworksRemoteService" class="org.metaworks.dwr.MetaworksRemoteService">
        <property name="connectionFactory" ref="springConnectionFactoryForMetaworks"/>
    </bean>
```

DWR 오류 : 무한루프에 의하여 죽는 경우

==〉 리턴타입에 대한 컨버터가 존재하지 않는 경우 발생. 예를 들어 java.io.File 리턴 객체..

```
Caused by: java.lang.UnsupportedOperationException: Can't getInputStream() from a string FormField
    at org.directwebremoting.extend.FormField.getInputStream(FormField.java:99)
    at org.directwebremoting.convert.FileConverter.convertInbound(FileConverter.java:89)
    at org.directwebremoting.impl.DefaultConverterManager.convertInbound(DefaultConverterManager.java:397)
    at org.directwebremoting.convert.BasicObjectConverter.convert(BasicObjectConverter.java:263)
    at org.directwebremoting.convert.BasicObjectConverter.createUsingSetterInjection(BasicObjectConverter.java:230)
    at org.directwebremoting.convert.BasicObjectConverter.convertInbound(BasicObjectConverter.java:90)
```

이런류의 오류가 나는이유:

결국 메타웍스는 dwr에 넘기기전에 모든 input tag객체의 값을 object (문자열 중심)로 만든다음 아규먼트로 전달해주는데, 그 순간 손실이 발생. 즉, mw3.getObject(objId) 내에서 getElementById…여기서 손실이 발생하는듯. 그렇담.. dwr.util.getValue를 써서 바꿔보자…

NPE 가 발생하는 이유중 하나가 이게 될 수 있다: spring id로 연결한 적 있는가?

==========

cannot read property 'serviceMethodContexts' of undefined ==〉 오류나면 dwr.jar 버전중 옛날 버전이 라이브러리에 같이 포함되었지 않은지 확인 필요함.

java.lang.NoSuchMethodError: org.metaworks.FieldDescriptor.getViewer() ==〉 옛날 버전 metaworks.jar(메타웍스2)가 설치 된 것인지 확인 필요함.

Caused by: java.lang.NullPointerException
at org.uengine.codi.mw3.model.WorkItemHandler.〈init〉(WorkItemHandler.java:24) 오류가 나면서, 해당 객체에서 @Autowired 로 스프링 빈을 가져다 쓰는 경우라면, AOP주입이 제대로 아뇌는 경우가 있다. 이 경우는 ApplicationContext.xml에
다음과 같이 빈을 명시적으로 추가해주어야 한다:

```
    <bean id="WorkItemHandler" class="org.uengine.codi.mw3.model.WorkItemHandler"/>
```

그리고 여기서 중요한것은, 이 빈이 호출되는 초기 서비스메서드가 포함된 오브젝트가 빈 등록 대상이다. 즉, 서비스 메서드를 가진 오브젝트내에서 언젠가 한번이라도 Autowired로 주입된
스프링 빈을 사용할 것이 존재하면 반드시 해당 서비스 메서드가 포함된 오브젝트는 빈 등록이 되어야 한다.

메타웍스 객체는 결국은 Stateless 한 객체라고 봐야 한다. 서버상에 캐싱되지 않음을 가정하여서 작성해야 함을 잊어서는 안된다.

org.directwebremoting.ConversionException: Data conversion error. See logs for more details. /// 이하 아님 --〉 이런 오류는 빈에 리턴타입이 혹시 abstract class가 아닌지 살펴봐야 한다: 예) Number --〉 Long등으로 바꾼다..

==〉이것은 아래에 브레이크 포인트를 잡은후,

```
Daemon Thread [http-8080-1] (Suspended (breakpoint at line 78 in BasicObjectConverter))
    MetaworksConverter(BasicObjectConverter).convertInbound(Class<?>, InboundVariable) line: 78
    MetaworksConverter.convertInbound(Class<?>, InboundVariable) line: 74
    DefaultConverterManager.convertInbound(Class<T>, InboundVariable, Property) line: 397
    MetaworksConverter(BasicObjectConverter).convert(String, Class<?>, InboundContext, Property) line: 263
    MetaworksConverter(BasicObjectConverter).createUsingSetterInjection(Class<?>, InboundVariable, Map<String,String>) line: 230
    MetaworksConverter(BasicObjectConverter).convertInbound(Class<?>, InboundVariable) line: 90
    MetaworksConverter.convertInbound(Class<?>, InboundVariable) line: 74
    DefaultConverterManager.convertInbound(Class<T>, InboundVariable, Property) line: 397
    PlainCallHandler(BaseCallHandler).marshallInbound(CallBatch) line: 206
    PlainCallHandler(BaseCallHandler).handle(HttpServletRequest, HttpServletResponse) line: 102
    UrlProcessor.handle(HttpServletRequest, HttpServletResponse) line: 120
    TransactionalDwrServlet(DwrServlet).doPost(HttpServletRequest, HttpServletResponse) line: 141
    TransactionalDwrServlet.doPost(HttpServletRequest, HttpServletResponse) line: 57
    TransactionalDwrServlet(HttpServlet).service(HttpServletRequest, HttpServletResponse) line: 637
    TransactionalDwrServlet(HttpServlet).service(ServletRequest, ServletResponse) line: 717
    ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 290
    ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 206
    CharacterEncodingFilter.doFilterInternal(HttpServletRequest, HttpServletResponse, FilterChain) line: 88
    CharacterEncodingFilter(OncePerRequestFilter).doFilter(ServletRequest, ServletResponse, FilterChain) line: 76
    ApplicationFilterChain.internalDoFilter(ServletRequest, ServletResponse) line: 235
    ApplicationFilterChain.doFilter(ServletRequest, ServletResponse) line: 206
    StandardWrapperValve.invoke(Request, Response) line: 233
    StandardContextValve.invoke(Request, Response) line: 191
    StandardHostValve.invoke(Request, Response) line: 128
    ErrorReportValve.invoke(Request, Response) line: 102
    StandardEngineValve.invoke(Request, Response) line: 109
    CoyoteAdapter.service(Request, Response) line: 293
    Http11Processor.process(Socket) line: 849
    Http11Protocol$Http11ConnectionHandler.process(Socket) line: 583
    JIoEndpoint$Worker.run() line: 454
    Thread.run() line: 680
```

```
            log.warn("Expected object while converting data for " + paramType.getName() + " in " +
data.getContext().getCurrentProperty() + ". Passed: " + value);
            throw new ConversionException(paramType, "Data conversion error. See logs for more details.");
```

위의 warning message 영역을 evaluation 해보면 아래와 비슷한 오류가 나온다: 오류를 확인하는데 좀더 도움이 될 것이다:

```
Expected object while converting data for [Lorg.uengine.codi.mw3.model.ParameterValue; in
PropertyDescriptorProperty[parameters=class [Lorg.uengine.codi.mw3.model.ParameterValue;]. Passed: [reference:c0-
e4,reference:c0-e14]
```

===〉 이건 배열형 컨버터가 제대로 선택되지 못한 경우다.

컨버터를 찾는 과정을 알고 싶은 경우는 다음을 본다:

DefaultConverterManager (line: 529) - getConverter(Class〈?〉)

```
org.directwebremoting.ConversionException: No converter found inbound for 'org.uengine.codi.mw3.model.ILogin'
        at org.directwebremoting.impl.DefaultConverterManager.convertInbound(DefaultConverterManager.java:393)
        at org.directwebremoting.convert.BasicObjectConverter.convert(BasicObjectConverter.java:263)
        at org.directwebremoting.convert.BasicObjectConverter.createUsingSetterInjection(BasicObjectConverter.java:230)
```

==〉 추후에 DefualtConverterManager.java를 오버라이드 하는 클래스를 개발하여 기본적으로 BeanConverter대신 MetaworksConverter를 사용토록 유도할 수 있다. 지금은 그냥 dwr.xml 에 값을 넣어서 처리하자.. `<convert converter="metaworks" match="org.uengine.codi.mw3.model.*"/>`

DWR의 호출 중에 원치않는 엉뚱한 정보에 대하여 converter를 못찾는다는 오류가 발견되는 경우도 있다. 사실 기본적으로 위에서 이야기한것 처럼, BeanConverter와 MetaworksConverter를 교체해버린다면 문제는 없을것 같으나 지금은 오류가 계속난다. 혹은 IDAO의 경우에 대한 기본적인 매핑을 추가하는 작업이 의미가 있을 수 도 있다. 하지만 다음과 같이 계속 커버터 못찾겠다는 오류에 대해서는 다음의 MetaworksServiceCall의 메서드에 대한 아규먼트 리스트를 만드는 DWR내부를 점검할 필요가 있다:

```
                    try
                    {
                        arguments[j] = converterManager.convertInbound(paramType, param, property);
                    }
                    catch (Exception ex)
                    {
                        log.debug("Problem converting param=" + param + ", property=" + property + ", into paramType=" +
paramType.getName() + ": " + ex);
                        throw ex;
                    }
```

로그를 안찍어서 보니, 뭐, 저걸 매번 breakpoint찍는다는 자체가 사실 문제군..

==========

이제 할 수 있는 것은?

1. 로그인 정보를 전달할 방법으로 Main 객체를 참조시킬 방법 ==〉 이것 보다는, mw3에 컨텍스트를 싣고 보내는 방법이 낫겠음… 'ContextAware' 인터페이스를 통하여 전달해주겠삼.

개발자는 서비스 메서드에 다음과 같은 클라이언트에서 값을 autowired로 참조할 수 있는 길이 있다. 스프링에서의 접근과 유사하다. 아래와 같이 해놓으면, 어디에서든 클라이언트 객체인 Login객체를 metaworks는 클라이언트와 서버가 협동으로 이의 값을 서비스 클래스에 전달해준다:

```
    @AutowiredFromClient
    public Login loginInfo;
```

물론 여기서 현재 버전이 Login객체가 2중으로 선언된 경우라면 최근에 발생한 인스턴스가 기준이 되기 때문에 이 값을 잘 전달해주는 것을 확실히 해야한다. 그리고 스프링과 다른점은 메타웍스는 기본 자바 리플랙션만 사용하기 때문에 필드 보안설정이 public 이어야만 injection이 가능하다.

그리고 현재의 메타웍스는 화면상에 뿌리는 객체 이외에는 특별히 리턴 값의 bean property 들 내부를 파고들어서 object로 모두 등록하지는 못한다. 해서 hidden으로 처리되었거나 혹은 커스텀 face로 된 객체에 대해서는 명시적으로 이 필드값은 클라이언트로 보낼때 autowired꼭 되어야 합니다… 를 다음과 같이 'AutowiredToClient'를 지정해주어야 한다:

```
        ILogin login;
        @Hidden
        @Id
        @AutowiredToClient
        public ILogin getLogin() {
            return login;
        }
        public void setLogin(ILogin login) {
            this.login = login;
        }
```

2. 가비지 컬렉션
3. 레이지 로딩을 위한 처리
4. join query 자동 생성 로직의 완성

우리가 해야할 일은?

다음 몇가지

4.1. 오브젝트에 세팅된 오브젝트 필드 정보에서 dao객체에 필드값중 오브젝트 (릴레이션) 값이 있는 경우, 이 오브젝트의 내부필드 각자의 필드명으로 dao에 매핑시켜 where clause를 완성시켜주는 것

ex) select userid, name from addressbook where user = ?user

AddressBook{ User user, String name }
User { userid }

==〉 이런 경우 알아서 User.userid로 쿼리의 where절을 만드는가?

=====〉 #Database.java

```
    for(FieldDescriptor fd : webObjectType.metaworks2Type().getFieldDescriptors()){
            Object fieldValue = objInst.getFieldValue(fd.getName());

            if(webObjectType.iDAOClass()!=null){
                //if IDAO interface doesn't have this field, it is not a database synchronizable field

if(MetaworksRemoteService.getMetaworksType(webObjectType.iDAOClass().getName()).metaworks2Type().getFieldDescriptor(fd.getNa
me()) == null)
                    continue;
            }

            if(fd.isSavable() || fd.isKey()){
                if(!dbPrimitiveTypes.containsKey(fd.getClassType())){
                    ObjectType referenceTableType = (ObjectType)
MetaworksRemoteService.getMetaworksType(fd.getClassType().getName()).metaworks2Type();
                    referenceTableType.getKeyFieldDescriptor();
                    ObjectInstance instance = (ObjectInstance) referenceTableType.createInstance();
                    instance.setObject(fieldValue);

                    fieldValue = instance.getKeyFieldValue();
                }

                dao.set(fd.getName(), fieldValue);
            }

        }
```

이 부분이 그 작업을 해준다.

4.2. 그렇다면 이제 값을 select 를 한 후에 오브젝트로 매핑시킬때는 어떠한가?

select initEp as initiator@endpoint, initRsNm as initiator___name, currEp as currentUser___endpoint, currRsNm as currentUser___name, taskId from bpm_worklist 에서 얻어지는 2개의 객체 - RoleMapping : initiator, currentUser 가 있다고 하자. 다음과 같은 객체모델을 가졌다고 보자:

WorkList{ RoleMapping initiator, RoleMapping currentUser, String taskId }
RoleMapping { String endpoint, String name }

일단 사람이 쿼리를 만들때 잘 만들어줘야 한다는 것, 그리고 객체 매핑에 대한 정보를 잘 알고 있어야 한다는것이 규칙이다.

그리고 alias로 넘겨주는것이 어색하다면, 다음과 같이 할 수 도 있다:

```
    @ORMapping(
            objectFields = { "endpoint", "name"},
            databaseFields = { "initEp", "initRsNm" }
    )
    public IUser getInitiator();
    public void setInitiator(IUser user);

    @ORMapping(
            objectFields = { "endpoint", "name"},
            databaseFields = { "currEp", "currRsNm" }
    )
    public IUser getCurrentUser();
    public void setCurrentUser(IUser user);
```

입력해야 문자열 길이는 좀 더 길다. 대신 객체에 찰싹 달라 붙어, 어떤 db필드가 오브젝트에 값으로 연결될지를 잘 읽을 수 있게 해준다.

4.3.

5. listing (ArrayFace.ejs) 된 정보의 quick sort
6. 남은 face작업들을 하과장님께 요청
7. 〔오류〕 리스트에서 자기 위치를 제대로 못잡아 변경되는 문제
8. 〔오류〕 때로 로딩된 결과가 화면에 보이지 않는 - 아마 key object가 없어졌는데 그놈에게 헛발질 하는 듯한 - 문제

9. 홈페이지 자체를 mw3로 만들기
10. okjsp.or.kr에 알리기
11. 누구나 사용할만한 많은 ejs들이 양산될 수 있도록 하기 - ex) 소스코드 뷰 등…
12. 프로세스 코디용 폼? --〉 Simple Form Editor의 재개발

13. 프로세스 정의 편집

- 아줌마 개발자, 외부 개발자들이 자신이 참여하도록 유도…. 빅 마우스들의 활용
- 멀티태넌시 이슈를 담아내기
- 페이스북 스킨의 개발