Оглавление

избавил страницу от фиксированного имени ДС

xPage

```
<?xml version="1.0" encoding="UTF-8"?>
<xp:view xmlns:xp="http://www.ibm.com/xsp/core"
    xmlns:xe="http://www.ibm.com/xsp/coreex"
    xmlns:xc="http://www.ibm.com/xsp/custom">
    <xc:globalvars></xc:globalvars>
    <xp:this.resources>
<xp:headTag tagName="script" rendered="true" loaded="true">
  <xp:this.attributes>
   <!--<xp:parameter name="clientSide" value="true"></xp:parameter>-->
   <xp:parameter name="src">
    <xp:this.value>
     <![CDATA[#{javascript:return docserver+"web-apps/apps/api/documents/api.js"}]]>
    </xp:this.value>
   </xp:parameter>
  </xp:this.attributes>
</xp:headTag>

<xp:script src="/DocEditor.js" clientSide="true"></xp:script>
<xp:script src="/SSJSUtils.jss" clientSide="false"></xp:script>
</xp:this.resources>
    <xp:this.beforeRenderResponse><![CDATA[#{javascript:try{
  var url = context.getUrl();
  var serverURI=url.getScheme() + '://' + url.getHost();
  //var httpURL = database.getHttpURL();//globalvars
  var unid=context.getUrlParameter('documentId');
  var sesID=facesContext.getExternalContext().getRequest().getSession().getId();

  var last=unid;
  unid=getConvUNID_S(unid);
  print("<<"+view.getPageName()+">>conversion unid:\""+unid+"\"");
  if (unid==="") unid=last;
  //first attach
  var att=session.evaluate('@AttachmentNames', database.getDocumentByUNID(unid))[0];
  var doc:NotesDocument=database.getDocumentByUNID(unid);
  var modtimeS=doc.getLastModified().getLocalTime();
  //var modtime=doc.getLastModified().toJavaDate();
  //print("<<"+view.getPageName()+">>doc Modification:"+ modtime);
  doc.recycle();

  var token=LtpaGenerator(@UserName());
  var LtpaToken=cookie.get("LtpaToken").getValue();
  var LtpaTokenB64=base64_encode(LtpaToken);
  var sid=base64_encode(modtimeS);

  var user=@UserName().toString();

  var jwtSigned=jwtSign(user);
  print("<<"+view.getPageName()+">>JWT signed:"+jwtSigned);
  //print("<<"+view.getPageName()+">>encrypt/decrypt:"+user+";result:"+decrypt(encrypt(user)));
}catch(e){
  _dump(e);
}}]]></xp:this.beforeRenderResponse>
    <xp:panel id="placeholder"></xp:panel>
```

```
    <xp:scriptBlock id="scriptBlock1">
        <xp:this.value><![CDATA[var host="#{javascript:return url.getHost();}";
var dbURL = "#{javascript:return httpURL}".replace("http://", "https://").split("?", 1)[0];
//unid as key
var unid="#{javascript:return unid}";
var att = "#{javascript:return att}";
var placeholder="#{id:placeholder}";
var user="#{javascript:return @UserName()}";
var sid="#{javascript:return sid}";
if (user==="Anonymous"){alert("unexpected user:"+user);exit;}
var sesID="#{javascript:sesID}";
var token="#{javascript:return LtpaTokenB64}";
console.log("sessionID:" + sesID)
console.log("Current LtpaToken:"+"#{javascript:cookie.get('LtpaToken').getValue();}");
var jwtSigned="#{javascript:return jwtSigned}";
readDoc(dbURL,token,unid,att, placeholder);
]]></xp:this.value>
    </xp:scriptBlock>
</xp:view>
```

SSJSUtils

```
import Utils;
//https://stackoverflow.com/questions/2793150/how-to-use-java-net-urlconnection-to-fire-and-handle-http-requests
var errAttr="error!->";
var pdfUpload="PDFupload.xsp";
var pdfUpload_path="/"+pdfUpload;
var redirect="redirect_att.xsp";
var oo_read="oo_read_batch.xsp";
var oo_read_path="/"+oo_read;
var id_param="documentId";
var id_query="?"+id_param+"=";
var OPTIONS="Options";
var OPT_SECRET="optSecret";
var SECRETJWT_FLD="secretJWT";
var SECRETEXT_FLD="secretEXT";
var DOCSERVER_FLD="docserver";
var CONVERTSERVICE="ConvertService.ashx";
var CONV_VIEW="converted";
/*
 *
 */
function LtpaGenerator(userDN){
  importPackage(lmike.org);
  try{
    var ltpa:LtpaGenerator1=new LtpaGenerator1();
    if (arguments.length > 1){
      //print("::LtpaGenerator::call Init with:"+arguments[1])
      ltpa.initByConfiguration(sessionAsSigner,arguments[1]);
    }else{
      //print("::LtpaGenerator::call Init");
      ltpa.initByConfiguration(sessionAsSigner);
    }
    //print("::LtpaGenerator::userDN"+userDN);
    var token=ltpa.generateLtpaToken(userDN);
    return token;
  }catch(e){
    return errAttr+e.toString();
  }
}
/*
 *
 */
```

```
function jwtSign(subject){
  importPackage(lmike.org);
  try{
    var jwt:JWT=new JWT();
    return jwt.sign(subject, secretJWT());
  }catch(e){
    _dump(e);
  }
}

function requestURL(urlStr, cookies){
  //importPackage(lmike.org);
  try{
  print("installing TrustAllCertificates...");
  lmike.org.TrustAllCertificates.install();
  print("TrustAllCertificates have been installed");
  var url = new java.net.URL(urlStr);
  var conn:java.net.HttpURLConnection = url.openConnection();
  conn.setRequestProperty("Cookie", cookies);
  //conn.setRequestProperty("Accept", "application/json");
  print("url:"+urlStr);
  print("geting response...");
  if (conn.getResponseCode() == "200") {
    print("status is OK");
    return conn.getHeaderFields();
    // Get the response
    var reader = new java.io.BufferedReader(new java.io.InputStreamReader(conn.getInputStream()));
    var buffer = new java.lang.StringBuffer();
    var line = "";
    while ((line = reader.readLine()) != null) {
      buffer.append(line);
    }
    reader.close();
    return buffer;
    // Create array from response
    var jsonarray = eval('(' + buffer + ')');

    // Get filenames and titles from Domino database collection resource
    // On XPage, requestScope.status is bound to a multi-line text control
    for (var i = 0; i < jsonarray.length; i++) {
      requestScope.status += jsonarray[i].@filepath + " - " + jsonarray[i].@title + "\n";
    }

  } else { // if connection fails
    requestScope.status = conn.getResponseCode() + " " + conn.getResponseMessage();
    print("request status:"+requestScope.status);
    return {"status":requestScope.status,
      "error":conn.getResponseCode()};
  }
  }catch(e){
    print(e.toString());
    return {"status":errAttr,
       "error":e.toString()
    }
  }
}
function getDbUrl(){
     var httpURL = database.getHttpURL();
     var dbUrl= httpURL.replace("http://", "https://").match(/(.+)\?/)[1];
}
function getConnect(url):java.net.HttpURLConnection{
    lmike.org.TrustAllCertificates.install();
    print("TrustAllCertificates have been installed");
    var url = new java.net.URL(url);
```

```
    var conn:java.net.HttpURLConnection = url.openConnection();
    return conn;
}

function getVars(){
  var gVars={};
  gVars.url = context.getUrl();
  gVars.host = gVars.url.getHost();
  gVars.domain = gVars.host.match(/\..*/)[0];
  gVars.httpURL = database.getHttpURL();
  gVars.dbUrl= gVars.httpURL.replace("http://", "https://").match(/(.+)\?/)[1];
  return gVars
}
/*
 * callback for ONLYOFFCE document server (DS) for reading option (see redirect_att.xsp)
 * called only if document is new for DS (key=unid+sid for notes doc)
 */
function url2Output(token, unid, att){
    print("--Name--"+Function.name);
  var gVars=getVars()
  print("::url2Output::url:"+ gVars.dbUrl);
  var target=gVars.dbUrl + "/0/"+unid+"/$FILE/"+att;
  var cookies="";
  if (token.contains(".")){
    var s=token.split(".")
    token=s[0];
    print("::url2Output::SessionID:"+s[1]);
    cookies="SessionID=" + s[1] + "; ";
  }
  print("::url2Output::token in base64:"+token);
  cookies+="LtpaToken=" + base64_decode(token) + "; domain=" + gVars.domain + "; path=/";
  print("::url2Output::Cookies would be set to:" + cookies);
  try{
    print("::url2Output::installing TrustAllCertificates...");
    lmike.org.TrustAllCertificates.install();
    print("::url2Output::TrustAllCertificates have been installed");
    var url = new java.net.URL(target);
    var conn:java.net.HttpURLConnection = url.openConnection();
    conn.setRequestProperty("Cookie", cookies);
    //conn.setRequestProperty("Accept", "application/json");
    print("::url2Output::url:"+target);
    print("::url2Output::geting response...");
    if (conn.getResponseCode() == "200") {
      print("::url2Output::status is OK");
      //return conn.getHeaderFields();
      copyAttach2Output(conn, token, att);
    }
  }catch(e){
    print("::url2Output::->");
    _dump(e);
    return {"status":errAttr,
        "error":e.toString()
    }
  }
}

function getDocserver(){
  var db:NotesDatabase=sessionAsSigner.getDatabase(session.getServerName(),
session.getCurrentDatabase().getFilePath());
  var NV:NotesView=db.getView(OPTIONS);
  var doc:NotesDocument=NV.getDocumentByKey(OPT_SECRET);
  var docserver=doc.getItemValueString(DOCSERVER_FLD);
  doc.recycle();NV.recycle();db.recycle();
  return docserver;
```

```
}

/**
 * conver file to PDF
 * call to OO docserver for conversion
 * send json
 * create new doc for pdf attach
 * receive and analize response in request2DBXml
 *
 * unid - document unid with attach for convertion
 */
function convert(unid){
  var result={};
  result.status="start";
  try{
    var externalContext:javax.faces.context.ExternalContext = facesContext.getExternalContext();
    var response:javax.servlet.http.HttpServletResponse = externalContext.getResponse();
    response.setContentType("application/json");
    response.setHeader("Cache-Control", "no-cache");
    var writer:java.io.PrintWriter = response.getWriter();

    var att=session.evaluate('@AttachmentNames', database.getDocumentByUNID(unid))[0];
    var dbURL:String=database.getHttpURL().replace("http://", "https://");
    dbURL=@Left(dbURL,"?");
    var ext=@RightBack(att,".");
    var doc:NotesDocument=database.getDocumentByUNID(unid);
    var modtimeS=doc.getLastModified().getLocalTime();
    doc.recycle();
    var user=@UserName().toString();
    //var jwtSigned=jwtSign(user);

    //var LtpaToken=cookie.get("LtpaToken").getValue();
    //token for domino session
    var token=base64_encode(LtpaGenerator(@UserName()));
    var LtpaToken=cookie.get("LtpaToken").getValue();
    var LtpaTokenB64=base64_encode(LtpaToken);

    var sid=base64_encode(modtimeS);
    var key=unid+sid;
    var url=dbURL+"/" + redirect + "?token=" + LtpaTokenB64 + "&unid=" + unid + "&att=" + att;
    var toPDF={
        "async": false,
        "filetype": ext,
        "key": key,
        "outputtype": "pdf",
        "title": att,
        "url": url
    }
    var payload={"payload":toPDF};
    result.payload=toJson(toPDF);

    var jwt:lmike.or.JWT=new lmike.org.JWT();
    var jwtSigned=jwt.signPayload(toJson(payload), secretJWT());
    //toPDF.token=jwtSigned;
    result.jwt=jwtSigned;
    result.requestData=toJson(toPDF);

    result.urlSourceDoc=url;
    var urlconvert=getDocserver()+CONVERTSERVICE;
    //https://stackoverflow.com/questions/3324717/sending-http-post-request-in-java
    result.urlConvert=urlconvert;
    result.status=+"connecting...";

    lmike.org.TrustAllCertificates.install();
```

```
    var url:java.net.URL = new java.net.URL(urlconvert);
    var http:java.net.HttpURLConnection=url.openConnection(); //getConnect(urlconvert);
    http.setRequestMethod("POST");
    http.setDoOutput(true);
    http.setRequestProperty("Content-Type", "application/json; charset=UTF-8")
    http.setRequestProperty("Authorization", "Bearer "+result.jwt);
    http.connect();
    var bw:java.io.BufferedWriter=new java.io.BufferedWriter(new
java.io.OutputStreamWriter(http.getOutputStream()));
    bw.write(result.requestData);
    bw.flush();
    bw.close();
    att=@LeftBack(att,".")+".pdf";
    result.convertStatus=request2DBXml(http, unid, att);
    result.status="end connection";

  }catch(e){
    print("::convert::exception")
    _dump(e);
    result.status="!!!exception:"+e.toString();
  }
  if (writer!=null){
    writer.write(toJson(result));
    writer.endDocument();
  }
}
function copyAttach2Output(conn, token, att){
  try{
  var externalContext:javax.faces.context.ExternalContext = facesContext.getExternalContext();
  var response:javax.servlet.http.HttpServletResponse = externalContext.getResponse();
  response.setHeader("Cache-Control", "no-cache");
  response.setDateHeader("Expires", -1);
  var gVars=getVars();
  response.setHeader("Set-Cookie", "LtpaToken=" + token + "; domain="+ gVars.domain + "path=/");
  var type=conn.getContentType();
  if (type.startsWith("text/html;")){
    print("::copyAttach2Output::text content!");
  }
  response.setContentType(type);
  print("::copyAttach2Output::ContentType:"+type);
  response.setHeader("Content-Disposition", "attachment; filename=" + att);
  var outStream:java.io.OutputStream = response.getOutputStream();
  var inStream:java.io.InputStream = conn.getInputStream();
  org.apache.commons.io.IOUtils.copy(inStream,outStream);
  inStream.close();
  outStream.close();
  facesContext.responseComplete();
  }catch(e){
    print("::copyAttach2Output::->");
    _dump(e);
      return {"status":errAttr,
        "error":e.toString()
    }
  }
}

/**
 * request to action from Domino side
 * action is defined by status field in JSON request body
 * callback for ONLYOFFCE document server (DS) for editing option (see redirect_att_edit.xsp)
 */
function request2DB(token, unid, att){
  try{
    //print("::request2DB::User:"+userDN);
```

```
    print("::request2DB::Receiving request...");
    var externalContext:javax.faces.context.ExternalContext = facesContext.getExternalContext();
    var request:javax.servlet.http.HttpServletRequest = externalContext.getRequest();
    print("::request2DB::Content type:"+request.getContentType());
    print("::request2DB::Data size:"+request.getContentLength());
    var scanner:java.util.Scanner=new java.util.Scanner(request.getInputStream()).useDelimiter("\\A");
    var body:String = scanner.hasNext() ? scanner.next() : "";
    var params=JSON.parse(body);
/*    var reader = new java.io.BufferedReader(new java.io.InputStreamReader(request.getInputStream()));
    var buffer = new java.lang.StringBuffer();
    var line = "";
    while ((line = reader.readLine()) != null) {
      print(line);
    }
    reader.close();
*/
    print("::request2DB::request Status:"+params.status);
    if (params.status==2){
      print("::request2DB::URL to save:"+params.url);
      var conn:java.net.HttpURLConnection=getConnect(params.url);
      if (conn.getResponseCode() == "200"){
        print("::request2DB::Saving...");
        save2DB(conn, unid, att);
      }
    }
    print("::request2DB::Request has been processed");
    var response:javax.servlet.http.HttpServletResponse = externalContext.getResponse();
    var writer:java.io.PrintWriter = response.getWriter();
    writer.write("{\"error\":0}");
  }catch(e){
    print("::request2DB::->")
    _dump(e);
  }
}
function getConvUNID(parent:NotesDocument){
  var unid="";
  var NV:NotesView=database.getView(CONV_VIEW);
  var doc:NotesDocument=NV.getDocumentByKey(parent.getUniversalID());
  NV.recycle();
  if (doc!=null){
    unid=doc.getUniversalID();
    doc.recycle();
  }
  return unid;
}
function getConvUNID_S(parent:String){
  var doc:NotesDocument=database.getDocumentByUNID(parent);
  var unid=getConvUNID(doc);
  doc.recycle();
  return unid;
}
/*
 * http - connection for analize response
```

```
    var match=result.response.match(/<error>(.+)<\/error>/i);
    if (match!=null){result.error=match[1];return result}
    match=result.response.match(/<fileurl>(.+)<\/fileurl>/i);
    if (match!=null)result.fileurl=org.apache.commons.lang.StringEscapeUtils.unescapeHtml(match[1]);
    //return result;
    print("::request2DBXml::URL to save:"+result.fileurl);
    var conn:java.net.HttpURLConnection=getConnect(result.fileurl);
    if (conn.getResponseCode() == "200"){
      var NV:NotesView=database.getView(CONV_VIEW);
      var doc:NotesDocument=NV.getDocumentByKey(parent);
      NV.recycle();
      if (doc==null){
        doc=database.createDocument();
        doc.replaceItemValue("form", "converted");
        doc.replaceItemValue("parentUNID",parent);
        doc.computeWithForm(false,false);
      }
      //com.setralubs.Access.setDefAccess(doc, @UserName());
      doc.save();
      var unid=doc.getUniversalID();
      doc.recycle();

      print("::request2DBXml::Saving...");
      result.saveRequest="Saving...";
      result.saveRequest=save2DB(conn, unid, att);
    }else{
      result.fileUrlStatus="Code:"+conn.getResponseCode()+ "->"+conn.getResponseMessage();
    }
    print("::request2DBXml::Request has been processed");
    return result;
  }else{
    result.connStatus="Code:"+http.getResponseCode()+ "->"+http.getResponseMessage();
    return result;
  }
 }catch(e){
  print("::request2DBXml::!!!exception")
  _dump(e);
  result.error="!!!exception:"+e.toString();
  return result;
 }
}

//https://xpagesandme.wordpress.com/2015/02/13/sessionassigner-oddities-part-1/
function save2DB(conn, unid, att){
 var result={};
 var ses:NotesSession=null;
 var db:NotesDatabase=null;
 var doc:NotesDocument=null;

 try{
 //importPackage(com.setralubs);
 var ses:NotesSession=sessionAsSigner;
 var db:NotesDatabase=ses.getDatabase(session.getServerName(), session.getCurrentDatabase().getFilePath());
 var doc:NotesDocument=db.getDocumentByUNID(unid);
 doc.removeItem("Body");
 print("::save2DB::creating RichTextOutputStream...");
 var rtos:com.setralubs.RichTextOutputStream = new com.setralubs.RichTextOutputStream(sessionAsSigner,
"Body", doc);
 rtos.setFileName(att);
 print("::save2DB::getting stream from docserver...");
 var inStream:java.io.InputStream = conn.getInputStream();
 org.apache.commons.io.IOUtils.copy(inStream, rtos);
 inStream.close(); rtos.close();
 print("::save2DB::session User:"+ses.getUserName());
```

```
  result.status="sucess"
  }catch(e){
    print("::save2DB::->")
    result.status="!!!exception:"+e.toString();
    _dump(e);
  }
  if (doc!=null){ doc.save();doc.recycle();}
  if (db!=null) db.recycle();
  return result;
}

function secretJWT(){
  var db:NotesDatabase=sessionAsSigner.getDatabase(session.getServerName(),
session.getCurrentDatabase().getFilePath());
  var NV:NotesView=db.getView(OPTIONS);
  var doc:NotesDocument=NV.getDocumentByKey(OPT_SECRET);
//  print("optSecret UNID:"+doc.getUniversalID())
  secret=doc.getItemValueString(SECRETJWT_FLD);
  doc.recycle();NV.recycle();db.recycle();
  //print("::secretJWT::secret:\""+secret+"\"");
  return secret;
}

function secretEXT(){
  var db:NotesDatabase=sessionAsSigner.getDatabase(session.getServerName(),
session.getCurrentDatabase().getFilePath());
  var NV:NotesView=db.getView(OPTIONS);
  var doc:NotesDocument=NV.getDocumentByKey(OPT_SECRET);
//  print("optSecret UNID:"+doc.getUniversalID())
  secret=doc.getItemValueString(SECRETEXT_FLD);
  doc.recycle();NV.recycle();db.recycle();
  //print("::secretEXT::secret:\""+secret+"\"");
  return secret;
}

function encrypt(text){
  importPackage(lmike.org);
  try{
    var jwt:JWT=new JWT();
    return jwt.encrypt(text, secretEXT());
  }catch(e){
    _dump(e);
  }

}
function decrypt(text){
  importPackage(lmike.org);
  try{
    var jwt:JWT=new JWT();
    return jwt.decrypt(text, secretEXT());
  }catch(e){
    _dump(e);
  }
}


Utils
//http://lpar.ath0.com/2013/05/03/cookies-with-xpages/
var Cookies = (function () {
 /**
  * Get a cookie value as a Cookie object.
  */
 var get:javax.servlet.http.Cookie = function (cookiename:string) {
   return cookie.get(cookiename);
```

```
},

/**
 * Convenience method to get a cookie value as a string.
 */
getString:string = function (cookiename:string) {
  return cookie.get(cookiename).getValue();
},

/**
 * Set a cookie value. Optional max age is in seconds.
 * Must be called during at least a partial update.
 */
set = function (cookiename:string, value, maxagesecs:double) {
  var xcon = facesContext.getExternalContext();
  resp = xcon.getResponse(), cmap = xcon.getRequestCookieMap(), ck;
  if (cmap.containsKey(cookiename)) {
    ck = cmap.get(cookiename);
    ck.setValue(value);
  } else {
    ck = new javax.servlet.http.Cookie(cookiename, value);
  }
  if (typeof maxagesecs !== 'undefined') {
    ck.setMaxAge(maxagesecs);
  }
  resp.addCookie(ck);
};

return {'get': get, 'set': set, 'getString': getString};
}());

/**
* Convert From/To Binary/Decimal/Hexadecimal in JavaScript
* https://gist.github.com/faisalman
*
* Copyright 2012-2015, Faisalman <fyzlman@gmail.com>
* Licensed under The MIT License
* http://www.opensource.org/licenses/mit-license
*/

    var ConvertBase = function (num) {
        return {
            from : function (baseFrom) {
                return {
                    to : function (baseTo) {
                        return parseInt(num, baseFrom).toString(baseTo);
                    }
                };
            }
        };
    };

    // binary to decimal
    ConvertBase.bin2dec = function (num) {
        return ConvertBase(num).from(2).to(10);
    };

    // binary to hexadecimal
    ConvertBase.bin2hex = function (num) {
        return ConvertBase(num).from(2).to(16);
    };

    // decimal to binary
    ConvertBase.dec2bin = function (num) {
```

```
    return ConvertBase(num).from(10).to(2);
  };

  // decimal to hexadecimal
  ConvertBase.dec2hex = function (num) {
    return ConvertBase(num).from(10).to(16);
  };

  // hexadecimal to binary
  ConvertBase.hex2bin = function (num) {
    return ConvertBase(num).from(16).to(2);
  };

  // hexadecimal to decimal
  ConvertBase.hex2dec = function (num) {
    return ConvertBase(num).from(16).to(10);
  };
/*
* Usage example:
* ConvertBase.bin2dec('111'); // '7'
* ConvertBase.dec2hex('42'); // '2a'
* ConvertBase.hex2bin('f8'); // '11111000'
* ConvertBase.dec2bin('22'); // '10110'
*/
  String.prototype.hexEncode = function(){
    var hex, i;

    var result = "";
    for (i=0; i<this.length; i++) {
      hex = this.charCodeAt(i).toString(16);
      result += ("000"+hex).slice(-4);
    }

    return result
  }
  String.prototype.hexDecode = function(){
    var j;
    var hexes = this.match(/.{1,4}/g) || [];
    var back = "";
    for(j = 0; j<hexes.length; j++) {
      back += String.fromCharCode(parseInt(hexes[j], 16));
    }

    return back;
  }
  String.prototype.hashCode = function() {
    var hash = 0, i, chr;
    if (this.length === 0) return hash;
    for (i = 0; i < this.length; i++) {
      chr   = this.charCodeAt(i);
      hash  = ((hash << 5) - hash) + chr;
      hash |= 0; // Convert to 32bit integer
    }
    return hash;
  };
//https://stackoverflow.com/questions/18729405/how-to-convert-utf8-string-to-byte-array
  function toUTF8Array(str) {
    var utf8 = [];
    for (var i=0; i < str.length; i++) {
      var charcode = str.charCodeAt(i);
      if (charcode < 0x80) utf8.push(charcode);
      else if (charcode < 0x800) {
        utf8.push(0xc0 | (charcode >> 6),
                  0x80 | (charcode & 0x3f));
```

```
            }
            else if (charcode < 0xd800 || charcode >= 0xe000) {
               utf8.push(0xe0 | (charcode >> 12),
                        0x80 | ((charcode>>6) & 0x3f),
                        0x80 | (charcode & 0x3f));
            }
            // surrogate pair
            else {
               i++;
               // UTF-16 encodes 0x10000-0x10FFFF by
               // subtracting 0x10000 and splitting the
               // 20 bits of 0x0-0xFFFFF into two halves
               charcode = 0x10000 + (((charcode & 0x3ff)<<10)
                        | (str.charCodeAt(i) & 0x3ff));
               utf8.push(0xf0 | (charcode >>18),
                        0x80 | ((charcode>>12) & 0x3f),
                        0x80 | ((charcode>>6) & 0x3f),
                        0x80 | (charcode & 0x3f));
            }
         }
         return utf8;
      }
//https://xomino.com/2013/12/18/safe-json-parsing-in-xpages-ssjs/
//  json2.js
//  2016-10-28
//  Public Domain.
//  NO WARRANTY EXPRESSED OR IMPLIED. USE AT YOUR OWN RISK.
//  See http://www.JSON.org/js.html
//  This code should be minified before deployment.
//  See http://javascript.crockford.com/jsmin.html

//  USE YOUR OWN COPY. IT IS EXTREMELY UNWISE TO LOAD CODE FROM SERVERS YOU DO
//  NOT CONTROL.

//  This file creates a global JSON object containing two methods: stringify
//  and parse. This file provides the ES5 JSON capability to ES3 systems.
//  If a project might run on IE8 or earlier, then this file should be included.
//  This file does nothing on ES5 systems.

//     JSON.stringify(value, replacer, space)
//        value       any JavaScript value, usually an object or array.
//        replacer    an optional parameter that determines how object
//                    values are stringified for objects. It can be a
//                    function or an array of strings.
//        space       an optional parameter that specifies the indentation
//                    of nested structures. If it is omitted, the text will
//                    be packed without extra whitespace. If it is a number,
//                    it will specify the number of spaces to indent at each
//                    level. If it is a string (such as "\t" or " "),
//                    it contains the characters used to indent at each level.
//        This method produces a JSON text from a JavaScript value.
//        When an object value is found, if the object contains a toJSON
//        method, its toJSON method will be called and the result will be
//        stringified. A toJSON method does not serialize: it returns the
//        value represented by the name/value pair that should be serialized,
//        or undefined if nothing should be serialized. The toJSON method
//        will be passed the key associated with the value, and this will be
//        bound to the value.

//        For example, this would serialize Dates as ISO strings.

//            Date.prototype.toJSON = function (key) {
//               function f(n) {
//                  // Format integers to have at least two digits.
```

```
//                 return (n < 10)
//                     ? "0" + n
//                     : n;
//             }
//             return this.getUTCFullYear()   + "-" +
//                 f(this.getUTCMonth() + 1) + "-" +
//                 f(this.getUTCDate())      + "T" +
//                 f(this.getUTCHours())     + ":" +
//                 f(this.getUTCMinutes())   + ":" +
//                 f(this.getUTCSeconds())   + "Z";
//         };

//         You can provide an optional replacer method. It will be passed the
//         key and value of each member, with this bound to the containing
//         object. The value that is returned from your method will be
//         serialized. If your method returns undefined, then the member will
//         be excluded from the serialization.

//         If the replacer parameter is an array of strings, then it will be
//         used to select the members to be serialized. It filters the results
//         such that only members with keys listed in the replacer array are
//         stringified.

//         Values that do not have JSON representations, such as undefined or
//         functions, will not be serialized. Such values in objects will be
//         dropped; in arrays they will be replaced with null. You can use
//         a replacer function to replace those with JSON values.

//         JSON.stringify(undefined) returns undefined.

//         The optional space parameter produces a stringification of the
//         value that is filled with line breaks and indentation to make it
//         easier to read.

//         If the space parameter is a non-empty string, then that string will
//         be used for indentation. If the space parameter is a number, then
//         the indentation will be that many spaces.

//         Example:

//         text = JSON.stringify(["e", {pluribus: "unum"}]);
//         // text is '["e",{"pluribus":"unum"}]'

//         text = JSON.stringify(["e", {pluribus: "unum"}], null, "\t");
//         // text is '[\n\t"e",\n\t{\n\t\t"pluribus": "unum"\n\t}\n]'

//         text = JSON.stringify([new Date()], function (key, value) {
//             return this[key] instanceof Date
//                 ? "Date(" + this[key] + ")"
//                 : value;
//         });
//         // text is '["Date(---current time---)"]'

//     JSON.parse(text, reviver)
//         This method parses a JSON text to produce an object or array.
//         It can throw a SyntaxError exception.

//         The optional reviver parameter is a function that can filter and
//         transform the results. It receives each of the keys and values,
//         and its return value is used instead of the original value.
//         If it returns what it received, then the structure is not modified.
//         If it returns undefined then the member is deleted.

//         Example:
```

```
//          // Parse the text. Values that look like ISO date strings will
//          // be converted to Date objects.

//          myData = JSON.parse(text, function (key, value) {
//              var a;
//              if (typeof value === "string") {
//                  a =
//   /^(\d{4})-(\d{2})-(\d{2})T(\d{2}):(\d{2}):(\d{2}(?:\.\d*)?)Z$/.exec(value);
//                  if (a) {
//                      return new Date(Date.UTC(+a[1], +a[2] - 1, +a[3], +a[4],
//                          +a[5], +a[6]));
//                  }
//              }
//              return value;
//          });

//          myData = JSON.parse('["Date(09/09/2001)"]', function (key, value) {
//              var d;
//              if (typeof value === "string" &&
//                      value.slice(0, 5) === "Date(" &&
//                      value.slice(-1) === ")") {
//                  d = new Date(value.slice(5, -1));
//                  if (d) {
//                      return d;
//                  }
//              }
//              return value;
//          });

//  This is a reference implementation. You are free to copy, modify, or
//  redistribute.

/*jslint
    eval, for, this
*/

/*property
    JSON, apply, call, charCodeAt, getUTCDate, getUTCFullYear, getUTCHours,
    getUTCMinutes, getUTCMonth, getUTCSeconds, hasOwnProperty, join,
    lastIndex, length, parse, prototype, push, replace, slice, stringify,
    test, toJSON, toString, valueOf
*/

// Create a JSON object only if one does not already exist. We create the
// methods in a closure to avoid creating global variables.

if (typeof JSON !== "object") {
    JSON = {};
}
```

```
    }

    function this_value() {
        return this.valueOf();
    }

    if (typeof Date.prototype.toJSON !== "function") {

        Date.prototype.toJSON = function () {

            return isFinite(this.valueOf())
                ? this.getUTCFullYear() + "-" +
                        f(this.getUTCMonth() + 1) + "-" +
                        f(this.getUTCDate()) + "T" +
                        f(this.getUTCHours()) + ":" +
                        f(this.getUTCMinutes()) + ":" +
                        f(this.getUTCSeconds()) + "Z"
                : null;
        };

        Boolean.prototype.toJSON = this_value;
        Number.prototype.toJSON = this_value;
        String.prototype.toJSON = this_value;
    }

    var gap;
    var indent;
    var meta;
    var rep;

    function quote(string) {

// If the string contains no control characters, no quote characters, and no
// backslash characters, then we can safely slap some quotes around it.
// Otherwise we must also replace the offending characters with safe escape
// sequences.

        rx_escapable.lastIndex = 0;
        return rx_escapable.test(string)
            ? "\"" + string.replace(rx_escapable, function (a) {
                var c = meta[a];
                return typeof c === "string"
                    ? c
                    : "\\u" + ("0000" + a.charCodeAt(0).toString(16)).slice(-4);
            }) + "\""
            : "\"" + string + "\"";
    }

    function str(key, holder) {

// Produce a string from holder[key].

        var i;          // The loop counter.
        var k;          // The member key.
        var v;          // The member value.
        var length;
        var mind = gap;
        var partial;
        var value = holder[key];

// If the value has a toJSON method, call it to obtain a replacement value.

        if (value && typeof value === "object" &&
                typeof value.toJSON === "function") {
```

```
            value = value.toJSON(key);
        }

// If we were called with a replacer function, then call the replacer to
// obtain a replacement value.

        if (typeof rep === "function") {
            value = rep.call(holder, key, value);
        }

// What happens next depends on the value's type.

        switch (typeof value) {
        case "string":
            return quote(value);

        case "number":

// JSON numbers must be finite. Encode non-finite numbers as null.

            return isFinite(value)
                ? String(value)
                : "null";

        case "boolean":
        case "null":

// If the value is a boolean or null, convert it to a string. Note:
// typeof null does not produce "null". The case is included here in
// the remote chance that this gets fixed someday.

            return String(value);

// If the type is "object", we might be dealing with an object or an array or
// null.

        case "object":

// Due to a specification blunder in ECMAScript, typeof null is "object",
// so watch out for that case.

            if (!value) {
                return "null";
            }

// Make an array to hold the partial results of stringifying this object value.

            gap += indent;
            partial = [];

// Is the value an array?

            if (Object.prototype.toString.apply(value) === "[object Array]") {

// The value is an array. Stringify every element. Use null as a placeholder
// for non-JSON values.

                length = value.length;
                for (i = 0; i < length; i += 1) {
                    partial[i] = str(i, value) || "null";
                }

// Join all of the elements together, separated with commas, and wrap them in
// brackets.
```

```
            v = partial.length === 0
                ? "[]"
                : gap
                    ? "[\n" + gap + partial.join(",\n" + gap) + "\n" + mind + "]"
                    : "[" + partial.join(",") + "]";
            gap = mind;
            return v;
        }

// If the replacer is an array, use it to select the members to be stringified.

        if (rep && typeof rep === "object") {
            length = rep.length;
            for (i = 0; i < length; i += 1) {
                if (typeof rep[i] === "string") {
                    k = rep[i];
                    v = str(k, value);
                    if (v) {
                        partial.push(quote(k) + (
                            gap
                                ? ": "
                                : ":"
                        ) + v);
                    }
                }
            }
        } else {

// Otherwise, iterate through all of the keys in the object.

            for (k in value) {
                if (Object.prototype.hasOwnProperty.call(value, k)) {
                    v = str(k, value);
                    if (v) {
                        partial.push(quote(k) + (
                            gap
                                ? ": "
                                : ":"
                        ) + v);
                    }
                }
            }
        }

// Join all of the member texts together, separated with commas,
// and wrap them in braces.

        v = partial.length === 0
            ? "{}"
            : gap
                ? "{\n" + gap + partial.join(",\n" + gap) + "\n" + mind + "}"
                : "{" + partial.join(",") + "}";
        gap = mind;
        return v;
    }
}

// If the JSON object does not yet have a stringify method, give it one.

if (typeof JSON.stringify !== "function") {
    meta = {    // table of character substitutions
        "\b": "\\b",
        "\t": "\\t",
```

```
            "\n": "\\n",
            "\f": "\\f",
            "\r": "\\r",
            "\"": "\\\"",
            "\\": "\\\\"
        };
        JSON.stringify = function (value, replacer, space) {

// The stringify method takes a value and an optional replacer, and an optional
// space parameter, and returns a JSON text. The replacer can be a function
// that can replace values, or an array of strings that will select the keys.
// A default replacer method can be provided. Use of the space parameter can
// produce text that is more easily readable.

            var i;
            gap = "";
            indent = "";

// If the space parameter is a number, make an indent string containing that
// many spaces.

            if (typeof space === "number") {
                for (i = 0; i < space; i += 1) {
                    indent += " ";
                }

// If the space parameter is a string, it will be used as the indent string.

            } else if (typeof space === "string") {
                indent = space;
            }

// If there is a replacer, it must be a function or an array.
// Otherwise, throw an error.

            rep = replacer;
            if (replacer && typeof replacer !== "function" &&
                    (typeof replacer !== "object" ||
                    typeof replacer.length !== "number")) {
                throw new Error("JSON.stringify");
            }

// Make a fake root object containing our value under the key of "".
// Return the result of stringifying the value.

            return str("", {"": value});
        };
    }

// If the JSON object does not yet have a parse method, give it one.

    if (typeof JSON.parse !== "function") {
        JSON.parse = function (text, reviver) {

// The parse method takes a text and an optional reviver function, and returns
// a JavaScript value if the text is a valid JSON text.

            var j;

            function walk(holder, key) {

// The walk method is used to recursively walk the resulting structure so
// that modifications can be made.
```

```
            var k;
            var v;
            var value = holder[key];
            if (value && typeof value === "object") {
                for (k in value) {
                    if (Object.prototype.hasOwnProperty.call(value, k)) {
                        v = walk(value, k);
                        if (v !== undefined) {
                            value[k] = v;
                        } else {
                            delete value[k];
                        }
                    }
                }
            }
            return reviver.call(holder, key, value);
        }

// Parsing happens in four stages. In the first stage, we replace certain
// Unicode characters with escape sequences. JavaScript handles many characters
// incorrectly, either silently deleting them, or treating them as line endings.

        text = String(text);
        rx_dangerous.lastIndex = 0;
        if (rx_dangerous.test(text)) {
            text = text.replace(rx_dangerous, function (a) {
                return "\\u" +
                        ("0000" + a.charCodeAt(0).toString(16)).slice(-4);
            });
        }

// In the second stage, we run the text against regular expressions that look
// for non-JSON patterns. We are especially concerned with "()" and "new"
// because they can cause invocation, and "=" because it can cause mutation.
// But just to be safe, we want to reject all unexpected forms.

// We split the second stage into 4 regexp operations in order to work around
// crippling inefficiencies in IE's and Safari's regexp engines. First we
// replace the JSON backslash pairs with "@" (a non-JSON character). Second, we
// replace all simple value tokens with "]" characters. Third, we delete all
// open brackets that follow a colon or comma or that begin the text. Finally,
// we look to see that the remaining characters are only whitespace or "]" or
// "," or ":" or "{" or "}". If that is so, then the text is safe for eval.

        if (
            rx_one.test(
                text
                    .replace(rx_two, "@")
                    .replace(rx_three, "]")
                    .replace(rx_four, "")
            )
        ) {

// In the third stage we use the eval function to compile the text into a
// JavaScript structure. The "{" operator is subject to a syntactic ambiguity
// in JavaScript: it can begin a block or an object literal. We wrap the text
// in parens to eliminate the ambiguity.

            j = eval("(" + text + ")");

// In the optional fourth stage, we recursively walk the new structure, passing
// each name/value pair to a reviver function for possible transformation.

            return (typeof reviver === "function")
```

```
                ? walk({"": j}, "")
                : j;
        }

// If the text is not JSON parseable, then a SyntaxError is thrown.

        throw new SyntaxError("JSON.parse");
      };
  }
}());
//https://openntf.org/XSnippets.nsf/snippet.xsp?id=encode-decode-base-64
function base64_encode (data) {
  // http://kevin.vanzonneveld.net
  // +   original by: Tyler Akins (http://rumkin.com)
  // +   improved by: Bayron Guevara
  // +   improved by: Thunder.m
  // +   improved by: Kevin van Zonneveld (http://kevin.vanzonneveld.net)
  // +   bugfixed by: Pellentesque Malesuada
  // +   improved by: Kevin van Zonneveld (http://kevin.vanzonneveld.net)
  // +   improved by: Rafał Kukawski (http://kukawski.pl)
  // *     example 1: base64_encode('Kevin van Zonneveld');
  // *     returns 1: 'S2V2aW4gdmFuIFpvbm5ldmVsZA=='
  // mozilla has this native
  // - but breaks in 2.0.0.12!
  //if (typeof this.window['btoa'] == 'function') {
  //    return btoa(data);
  //}
  var b64 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
  var o1, o2, o3, h1, h2, h3, h4, bits, i = 0,
    ac = 0,
    enc = "",
    tmp_arr = [];

  if (!data) {
    return data;
  }

  do { // pack three octets into four hexets
    o1 = data.charCodeAt(i++);
    o2 = data.charCodeAt(i++);
    o3 = data.charCodeAt(i++);

    bits = o1 << 16 | o2 << 8 | o3;

    h1 = bits >> 18 & 0x3f;
    h2 = bits >> 12 & 0x3f;
    h3 = bits >> 6 & 0x3f;
    h4 = bits & 0x3f;

    // use hexets to index into b64, and append result to encoded string
    tmp_arr[ac++] = b64.charAt(h1) + b64.charAt(h2) + b64.charAt(h3) + b64.charAt(h4);
  } while (i < data.length);

  enc = tmp_arr.join('');

  var r = data.length % 3;

  return (r ? enc.slice(0, r - 3) : enc) + '==='.slice(r || 3);

}


function base64_decode (data) {
  // http://kevin.vanzonneveld.net
```

```
    // +   original by: Tyler Akins (http://rumkin.com)
    // +   improved by: Thunder.m
    // +      input by: Aman Gupta
    // +   improved by: Kevin van Zonneveld (http://kevin.vanzonneveld.net)
    // +   bugfixed by: Onno Marsman
    // +   bugfixed by: Pellentesque Malesuada
    // +   improved by: Kevin van Zonneveld (http://kevin.vanzonneveld.net)
    // +      input by: Brett Zamir (http://brett-zamir.me)
    // +   bugfixed by: Kevin van Zonneveld (http://kevin.vanzonneveld.net)
    // *     example 1: base64_decode('S2V2aW4gdmFuIFpvbm5ldmVsZA==');
    // *     returns 1: 'Kevin van Zonneveld'
    // mozilla has this native
    // - but breaks in 2.0.0.12!
    //if (typeof this.window['atob'] == 'function') {
    //    return atob(data);
    //}
    var b64 = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/=";
    var o1, o2, o3, h1, h2, h3, h4, bits, i = 0,
      ac = 0,
      dec = "",
      tmp_arr = [];

    if (!data) {
      return data;
    }

    data += '';

    do { // unpack four hexets into three octets using index points in b64
      h1 = b64.indexOf(data.charAt(i++));
      h2 = b64.indexOf(data.charAt(i++));
      h3 = b64.indexOf(data.charAt(i++));
      h4 = b64.indexOf(data.charAt(i++));

      bits = h1 << 18 | h2 << 12 | h3 << 6 | h4;

      o1 = bits >> 16 & 0xff;
      o2 = bits >> 8 & 0xff;
      o3 = bits & 0xff;

      if (h3 == 64) {
        tmp_arr[ac++] = String.fromCharCode(o1);
      } else if (h4 == 64) {
        tmp_arr[ac++] = String.fromCharCode(o1, o2);
      } else {
        tmp_arr[ac++] = String.fromCharCode(o1, o2, o3);
      }
    } while (i < data.length);

    dec = tmp_arr.join('');

    return dec;
}
```

T.o. все настройки вынесены в БД, в конфигурационный док, там находит ся урл для ДС, ключ для JWT, ключ для доступа внешнему коду (шифрует имя пользователя, на стороне сервера расшифровывает, см. https://codeby.net/forum/threads/unifikacija-razrabotki.63162/)

globalvars

кастомконтрол, где устанавливаются переменные для серверной и клиентской части

```
<?xml version="1.0" encoding="UTF-8"?>
<xp:view xmlns:xp="http://www.ibm.com/xsp/core">
    <xp:this.beforeRenderResponse><![CDATA[#{javascript:
try{
```

```
  var httpURL = database.getHttpURL();
  var userDN=@UserName();
  var userName=@Name("[CN]",userDN);
  var userID=userDN.hashCode().toString();
  var docserver=getDocserver();
  print("<<"+view.getPageName()+">>User name hash:"+userID);
} catch(e) {
   _dump(e);
}

}]]></xp:this.beforeRenderResponse>
   <xp:this.resources>
      <xp:script src="/SSJSUtils.jss" clientSide="false"></xp:script>
   </xp:this.resources>
   <xp:scriptBlock>
      <xp:this.value><![CDATA[
var userName="#{javascript:userName}";
var userDN="#{javascript:userDN}";
var userID="#{javascript:userID}";
]]></xp:this.value>
   </xp:scriptBlock>
</xp:view>
```