Atitit.atiDataStoreService v2 新特性

[1.1. V1 基础实现 1](#_Toc13032)

[1.2. V2 增加了对 $uuid $cur\_uid参数的支持 1](#_Toc28220)

[1.3. 增加了fld fun的支持 1](#_Toc23926)

[2. fld fun 1](#_Toc7765)

[2.1. Invoke 简化版全局函数txt2html 1](#_Toc6533)

[2.2. ---------atiDataStoreService .js 1](#_Toc29244)

[2.3. dslUtil 2](#_Toc15919)

[2.4. jAva 3](#_Toc31043)

[3. code 4](#_Toc18353)

## V1 基础实现

## V2 增加了对 $uuid $cur\_uid参数的支持

## 增加了fld fun的支持

# fld fun

## Invoke 简化版全局函数txt2html

cmsPhone4jobusImp/edit.html

<textarea id="copy\_content" name="copy\_content" style="width:100%; height: 300px;" fun="txt2html">工作日期

## ---------atiDataStoreService .js

function merge(metaData,merge\_callback\_fun)

{

var mp=$("form").serialize();

var fldAppFun=dslUtil.getFldAppFun( mp);

mp=mp+"&$method=com.attilax.urldsl.UrlDsl2SqlStoreService.exe&$callback=page\_load\_callback&$mod=userMod&$view\_store\_path=com/attilax/order&$op=insert&"+metaData+"&$fldAppFun="+encodeURIComponent(fldAppFun);

// alert("get post mp:"+mp);

HRE.method="post";

try{

if( arguments.length==1)

HRE.exe(mp,merge\_callback);

else

HRE.exe(mp,merge\_callback\_fun);

}catch(e)

{

showErr(e);

}

}

## dslUtil

var dslUtil={};

dslUtil.getFldAppFun=function(url)

{

var json={};

var keys=urlUtil.getKeys(url);

for(e in keys )

{

var key=keys[e];

key=decodeURIComponent(key);

var key\_for\_jq=escapeJquery(key);

//alert(key\_for\_jq);

var fun=$("#"+key\_for\_jq).attr("fun");

if(fun)

json[key]=fun;

}

return JSON.stringify(json);

};

## jAva

\*/

**public** **class** UrlDsl2SqlStoreService **extends** Dsl2sqlService {

**public** **static** Map<String, Function> *scriptMapper* = Maps.*newLinkedHashMap*();

{

*scriptMapper*.put("txt2html", (txt) -> {

**return** htmlx.*txt2html*(txt.toString());

});

}

DslUtil.*appFldFun*(m);

**public** **class** DslUtil {

**public** **static** **void** appFldFun(Map m) {

**if**(m.get("$fldAppFun" )==**null** || m.get("$fldAppFun" ).toString().trim().length()==0)

**return**;

String json\_str=(String) m.get("$fldAppFun");

Map fldFunMap=AtiJson.*fromJson*(json\_str);

Set<String> st = fldFunMap.keySet();

**for** (String k : st) {

String script\_tag=(String) fldFunMap.get(k);

Function fun=UrlDsl2SqlStoreService.*scriptMapper*.get(script\_tag);

Object v=fun.apply(m.get(k));

m.put(k, v);

}

}

}

# code

**package** com.attilax.urldsl;

**import** java.util.HashMap;

**import** java.util.Map;

**import** java.util.Set;

**import** java.util.function.Function;

**import** javax.servlet.http.HttpServletRequest;

**import** org.apache.commons.lang3.StringUtils;

**import** com.attilax.anno.Inj;

**import** com.attilax.html.htmlx;

**import** com.attilax.io.filex;

**import** com.attilax.ioc.IocXq214;

**import** com.attilax.lang.Global;

**import** com.attilax.lang.MapX;

**import** com.attilax.net.requestImp;

**import** com.attilax.sql.Dsl2sqlService;

**import** com.attilax.sql.SqlService;

**import** com.attilax.store.OrmStoreService;

**import** com.attilax.token.TokenService;

**import** com.attilax.user.NotLoginEx;

**import** com.attilax.web.ReqX;

**import** com.google.common.collect.Maps;

**import** com.google.inject.Inject;

/\*\*

\* q317 com.attilax.urldsl.UrlDsl2SqlStoreService

\*

\* **@author** Administrator

\*

\*/

**public** **class** UrlDsl2SqlStoreService **extends** Dsl2sqlService {

**public** **static** Map<String, Function> *scriptMapper* = Maps.*newLinkedHashMap*();

{

*scriptMapper*.put("txt2html", (txt) -> {

**return** htmlx.*txt2html*(txt.toString());

});

}

@Inject

SqlService sqlSrv;

@Inject

Dsl2sqlService d2sSrv;

@Inject

TokenService tkSrv;

**public** **static** **void** main(String[] args) {

requestImp m = **new** requestImp();

m.put("$tb", "wxb\_good\_copy");

// m.put("$tabletype", "view");

// m.put("$view\_store\_path","com/attilax/order");

m.put("$op", "i");

m.put("copy\_title", "copy\_title\_2005");

Global.*req*.set(m);

System.*setProperty*("apptype", "jobus");

// OrmStoreService ormSvr = IocXq214.getBean(OrmStoreService.class);

UrlDsl2SqlStoreService srv = IocXq214

.*getBean*(UrlDsl2SqlStoreService.**class**);

// Map m=new HashMap();

System.***out***.println(srv.exe());

System.***out***.println("--f");

}

**public** Object exe() {

HttpServletRequest req = Global.*req*.get();

tkSrv.setModule(req.getParameter("$utype") + "Mod");

Map m = ReqX.*toMap*(req);

**if** (m.get("$table") == **null**)

m.put("$table", m.get("$tb"));

**if** (m.get("$op").equals("i"))

m.put("$op", "insert");

Set<String> st = m.keySet();

**for** (String k : st) {

**if** (m.get(k).equals("$cur\_uid")) {

String getuid = tkSrv.getuid(req);

**if** (StringUtils.*isEmpty*(getuid))

**throw** **new** NotLoginEx("NotLoginEx");

m.put(k, getuid);

}

**if**(m.get(k).toString().equals("$uuid"))

{

m.put(k,filex.*getUUidName*());

}

}

DslUtil.*appFldFun*(m);

String sql = d2sSrv.dsl2sql(m);

**if** (StringUtils.*isEmpty*(sql))

**throw** **new** RuntimeException(

" cant convert sql str rzt is null or empty");

**return** sqlSrv.exe(sql);

}

}