自定义任务开发指南

- 1 任务脚本编写规范
- 2 引入Java对象
- 3 常用开发技巧
- 4 系统内置对象
- 5 SDK对象调用
  - 5.1 常用接口说明
  - 5.2 SDK调用示例
- 6 任务组件说明
  - 6.1 组件接口说明
  - 6.2 组件调用示例

本文档说明的是任务脚本定制开发指南，一般在通过系统任务配置界面配不出来的任务，需要编写任务脚本，譬如：使用计划任务定时备份知识库，通过任务把电子表格报表内容以邮件正文发送，通过计划任务自动同步用户等等。

任务脚本是在应用服务器上运行，它依赖了 Rhino 工具包，Rhino 是一种使用 Java 语言编写的 JavaScript 的开源实现，语法遵循Javascript 语法规范，能够引用 Java 类并创建 Java 对象来使用，但是并不代表可以使用 Java 语法。

阅读本文的前提是了解计划任务的基本概念及使用方法，并且可以在任务配置界面点击【查看运行脚本】初识任务脚本。

任务脚本编写规范

- 脚本必须满足Javascript语法规范。
- 脚本中可以引入Java对象并使用。
- 脚本中还可调用如下对象：
  - 系统内置对象：为系统内置的对象，如connector、logger对象、execute函数等。
  - SDK接口：Smartbi服务器端SDK，共提供七项服务接口，可按需调用实现任务自定义。常用接口如打开报表、获得报表的行数等。
  - 任务组件接口：为了方便用户编写自定义任务脚本，系统对常见任务进行了封装，为用户提供一些实用的组件。

说明

- 关于JavaScript的介绍，可自行上网学习相关资料。

引入Java对象

如何引入Java对象？
例如：Packages.java.io.File引用了Java的io包中File对象。 要在JavaScript中使用该Java对象，可用如下写法，new 和Packages都是可以被省略的（因为Rhino定了了一个变量java等同于Packages.java，所以才可以省略Packages）：

```
//相当于: var frame = new Packages.java.io.File("filename");

var frame = java.io.File("filename");
```

我们也可以像Java代码中一样把这个对象引用进来：

```
importClass (java.io.File);
var file = File("filename");
```

如果要将整个包下的所有类都引用进来可以用importPackage，然后就可以直接这个包下面的类：

```
importPackage(java.io);

var file = File("filename");
```

如果只需要在特定代码段中引用某些包，可以使用JavaImporter搭配JavaScript的with关键字，如：

```
var MyImport = JavaImporter(java.io.
File);
with (MyImport) {
var myFile = File("filename");
}
```

用户自定义的包也可以被引用进来，不过这时候Packages引用不能被省略：

```
importPackage(Packages.
tony);
var hello = HelloWorld();
hello.sayHello();
```

注意：这里只有public 的成员和方法才会在JavaScript中可见。对于非public的成员，例如对 hello.s的引用将得到undefined。

常用开发技巧

1，任务配置界面配置的任务实际是可以查看其对应的任务脚本的（界面上的【查看运行脚本】），很多时候可以基于这个脚本改成实际的需求，这个脚本一般使用的是任务组件，譬如自定义导出文件的名字，批量发送邮件之类。

2，无法使用1说的，可以参考现有示例（本文中开始说的几个都可），基本都是按新的需求重新组合，如果涉及到报表可能就需要了解SDK对象调用。

3，最重要实际还是需要结合示例，充分理解自定义任务脚本。

系统内置对象

系统内置对象说明：

对象	对象描述

connector	系统内置的连接对象，连接到计划任务服务器，可直接使用。例如：
	• 在SDK对象中调用： `var tempReport = new Report(connector);`
	• 在任务组件中调用：
	`var tempResourceHandle = execute("openResource", {`
	`clientConnector: connector,`
	`reportId: "xxxxxxxxxx"`
	`});`
	• 调用扩展包中的module方法，这样逻辑复杂可以考虑写在扩展包，任务直接调用：
	`connector.remoteInvoke("CustomModule", "syncLDAPUsers", []); //后面中括号是参数,有的话就传,没有的话就是个空数组`

logger	系统内置的写日志对象。有三个方法： • logger.debug • logger.error • logger.info
execute	execute函数用于执行内置任务组件，一般可以通过任务配置界面配置，然后查看运行脚本看execute的示例，详见任务组件说明。

SDK对象调用

说明
Smartbi服务器端SDK通过JAVA API提供七项服务接口，可以在自定义任务中调用这些API。

服务对象	描述
AnalysisReportService	提供多维分析相关操作功能
CatalogService	提供资源目录树的访问功能等
GraphicReportService	提供图形分析报表访问功能
ManageReportService	提供业务报表相关操作功能
UserManagerService	提供用户相关操作。包括：读取/维护用户信息、读取/维护组信息、读取/维护角色信息、为用户和组分配角色等

具体的方法以及帮助请参考《**JAVA API文档**》。

常用接口说明

自定义任务中常用到如下接口：

说明	调用接口		

读取资源节点		`var reportId = 'I2c9410a623cc37d10123cc8f90930187';`
		`var catalogService = new CatalogService(connector);`
		`var reportElem = catalogService.getCatalogElementById(reportId);`
		`var reportAlias = reportElem.getAlias();`	

创建资源定义	多维分析	`var tempReportService = new AnalysisReportService(connector);` `var clientId = tempReportService.openAnalysisReport(reportId)`	

	电子表格	**var** tempReport = new SSReport(connector); tempReport.open(reportId);	

参数相关	电子表格	//获取参数列表
		
		var paramList = tempReport.getParamList();
		
		//设置参数值
		tempReport.setParamValue(param.id, param.value, param.displayValue);	

	多维分析	**var** paramList = tempReportService.getParameters(clientid); tempReportService.setParamValue(clientId, param.id, param.value, param.displayValue); **var** enumParamValues = tempReportService.getParamStandbyValue(clientId, param.id);	
刷新报表	多维分析	tempReportService.executeQuery(clientId)	
	电子表格	无接口，导出时会自动刷新报表。	

导出报表到本地文件	多维分析	//第二个参数为String类型，表示导出格式，支持格式为EXCEL、CSV、TXT、PDF、WORD;
		tempReportService.doExport(clientId, 'MHT', ',' , outputStream, '');	

	电子表格	//第二个参数为String类型，表示导出格式，支持格式为PDF、PNG、WORD、EXCEL、EXCEL2007、HTML;
		tempReport.export('EXCEL', outputStream);	

常用参数说明如下：

- **connector**：连接对象，为系统内置对象，无需创建可直接使用。
- **clientId**：String字符串类型，指报表的句柄ID。
- **reportId**：String字符串类型，指报表资源的唯一标识ID。
- **reportName**：String字符串类型，指报表资源的名称。
- **pageId**：String字符串类型，指页面资源的唯一标识ID。
- **param.id**，**param.value**，**param.displayValue**：String字符串类型，分别指参数ID、参数真实值、参数显示值。
- **outputStream**：OutputStream文件流类型，指输出文件流。

SDK调用示例

示例：以下脚本实现"打开灵活分析报表，输出该报表的总行数"。

```
// 导入将会使用的java类与java包
importClass(java.lang.System);
importPackage(Packages.smartbi.sdk.service.simplereport);

// 创建分析查询对象
var reportId = 'I2c949e8e1ac2d5e6011ac380971301b8';
var tempReportService = new SimpleReportService
(connector);
var tempReport = tempReportService.openReport(reportId);
// 执行分析报表，获取总行数
var totalRow = tempReport. execute(100);
tempReport.close();

logger.info('总行数为: '+ totalRow);
```

注：在自定义任务中connector客户端连接对象是系统预先创建好的，在使用时直接调用就可以了，完整的示例可见通过计划任务自动同步用户机构和角色。

任务组件说明

Smartbi的系统SDK为任务的开发者提供了最基本的API，例如：打开一个报表、设置参数、刷新等。但这些API比较底层和基本、粒度比较细，往往需要多步调用才能完成一个常规操作。
为了方便用户编写自定义任务脚本，系统对常见任务进行了封装，为用户提供一些实用的组件。一般内置组件都会有对应的任务配置界面，然后【查看运行脚本】看对应组件的示例。

组件接口说明

一个组件由三个部分构成：输入参数Input、输出结果Ouput、执行方法execute。
系统内置了如下组件，供用户调用：

Execute命令解释	Input输入参数	Output输出结果

OpenResource 打开资源组件，可以打开任意组件类型。	• connector：客户端连接对象。 • reportId：String类型，报表资源的ID。	• resourceHandle：资源句柄。
EnumerateParamValues 参数枚举组件。	• resourceHandle：系统连接对象。 • byEveryParam：boolean类型，是否枚举true/flase。 • paramsSetting：IParamValue[]，参数缺省值。	• getParamValues：获得参数枚举值的迭代器。
FillData 填充数据组件。	• connector：客户端连接对象。 • tableId：String类型，表ID。	• isSucceeded：是否成功。

ExportResource
导出资源组件。 根据输入的参数迭代器，可以导出一份或多份参数组合的报表执行结果。
	• connector：客户端连接对象。
	• resourceHandle：资源句柄。
	• paramSettingIterator：参数迭代器。
	• exportSetting：IExportSetting类型，导出设置。
	  • delimiter：String类型，分隔符。
	  • folderDepth：int类型，目录深度。
	  • height：String类型，用于仪表分析设置高度。
	  • paramValueTypeInReportName：String。
	  • pathId：String类型，路径。
	  • resourceBasePath：String类型。
	  • userDefinedFolderName：String类型，用于指定目录名。
	  • width：String类型，用于仪表分析设置宽度。
	• taskName：String类型，任务名。
	• exportType：String类型，导出类型。
		• file：输出的临时文件。
		• folder：输出的临时目录。

SendToFile
发送到文件组件。
	• file：File[]类型，要发送的文件。
	• sendSetting：ICopySetting类型，发送设置。
	  • filename：String类型，目标文件名。
	  • path：String类型，目录路径。
		• 无。

SendToMail
发邮件组件。
	• connector：客户端连接对象。
	• files：File[]类型，要发送的邮件附件列表。
	• paramValueMap：String类型，参数值。
	• sendSetting：IMailSetting类型，发邮件设置信息。
	  • doZip：boolean类型，是否压缩。
	  • mailList：String类型，收件人地址列表，用分号（;）分隔。
	  • ccMailList：String类型，抄送地址列表，用分号（;）分隔。
	  • bccMailList：String类型，密送地址列表，用分号（;）分隔。
	  • text：String类型，邮件正文。
	  • title：String类型，邮件标题。
	• taskName：String类型，任务名。
		• 无。

说明

- OpenResource和ExportResource支持的资源类型包括：灵活分析、仪表分析、多维报表、门户页面。

组件调用示例

用户在自定义任务JavaScript脚本中通过execute标准函数来调用组件，该函数接收两个参数：

- 第一个参数是要调用组件的命令名称；
- 第二个参数是该命令的输入参数（Javascript表示方式），可能存在多个参数，用{ }括起。表示如下：

```
{ 参数1名称: 参数值,
参数2名称: 参数值,
参数3名称: 参数值
}
```

组件的示例，一般都可以通过任务配置界面配出，然后【查看示例脚本】，请见报表拆分并批量发送邮件中的5.使用自定义定制任务小技巧。
示例：以下脚本实现"打开分析报表，枚举"产品目录参数"参数，以Excel格式导出文件至C:/task/目录"。

```
//初始化参数设置：枚举"产品目录参数"，固定"开始日期"和"结束日期"参数值。
var reportParamSetting = [
{id:"OutputParameter.I2c90903e114ef1af01114f2ed1e40097.产品目录参数", value:"$枚举值$",displayValue:"$枚举值$"},
{id:"OutputParameter.I2c90903e114ef1af01114f2ed1e40097.开始日期", value:"1996-01-17",displayValue:"1996-01-17"},
{id:"OutputParameter.I2c90903e114ef1af01114f2ed1e40097.结束日期", value:"2011-03-03",displayValue:"2011-03-03"}
];

//打开报表资源。
var tempResourceHandle = execute("openResource", {
clientConnector: connector,
reportId: "I2c90903e114f6f9601114f70e09d000e"
});

//设置报表参数。
var enumerateParamValuesOutput = execute('enumerateParamValues', {
resourceHandle: tempResourceHandle.resourceHandle,
byEveryParam: true,
paramsSetting: reportParamSetting
});

//以EXCEL格式导出报表资源。
var exportReportOutput = execute('exportResource', {
connector: connector,
resourceHandle: tempResourceHandle.resourceHandle,
paramSettingIterator: enumerateParamValuesOutput.getParamValues(),
exportSetting: {"delimiter":"", "pathId":""},
taskName: taskName,
exportType: "EXCEL"
});

//输出文件到C:/task/目录。
var outputFile = exportReportOutput.folder;
var sendToFileOutput = execute('sendToFile', {
file: outputFile,
sendSetting: {path: ' C:/task/'}
});
```