G5ServiceAPI E ODATA plugin

Estas APIs permitem integração com a G5 via requisições REST. E também integração com o ODATA da plataforma G7. A API de integração com G5, recebe e retorna os parâmetros no formato JSON, facilitando o desenvolvimento. Possui facilitadores que encapsulam a complexidade das requisições, deixando transparente o uso para o desenvolvedor. E a API ODATA serve para encapsular funções de integração com o ODATA da plataforma G7

A API é composta por duas camadas. Uma é uma aplicação Java (Servlet) que tem o papel de tratar as chamadas de webservices para a g5 convertendo os parâmetros JSON para XML no envio dos dados e vice-versa no retorno. Outra são os componentes javascript para utilização no front-end.

Como usar

A aplicação Java deve ser disponibilizada em um servidor Java com acesso disponível a partir da plataforma G7 via HTTPS. A mesma pode estar publicada no ambiente do cliente ou em qualquer outro servidor Java. Porém esse servidor deve ter acesso à infra dos webservices da g5 do cliente.

Para fazer deploy dessa aplicação, basta colocar o war (G5ServiceAPI.war) na pasta autodeploy do domínio do glassfish.

Serão disponibilizados dois servlets. G5ServiceAPI/G7Rest e G5ServiceAPI/G5Rest.

**Front-end angularjs**

Deve ser incluído na página os seguintes plugins js:

Dependências

|  |
| --- |
| <script type="text/javascript" src="https://platform.senior.com.br/ecm-frontend/assets/libs/angular/angular.js"></script> <script type="text/javascript" src="https://platform.senior.com.br/ecm-frontend/assets/libs/angular-odata-resources/build/odataresources.min.js"></script> <script type="text/javascript" src="https://cdn.senior.com.br/platform/1.7.1/js/workflow-cockpit.js"></script> |

Componentes das APIs

|  |
| --- |
| <script type="text/javascript" src="js/g5service.js"></script>  <script type="text/javascript" src="js/odataservice.js"></script> |

Configurando as APIs

No controller do angular o G5Service e o ODataServiceUtil devem ser inicializados conforme segue:

|  |
| --- |
| var wsserviceurl = "http://[server]:[port]";  var apig5resturl = "https://[server]:[port]/G5ServiceAPI/G5Rest";  G5Service.config(wsserviceurl, apig5resturl, “usuario”, “token”, “\_”);  ODataServiceUtil.config(“[odataurl]”, “token”); |

Conforme demonstrado, a API solicita o usuário e o token (tokenG7). Essas informações serão utilizadas para autenticação com a soluções solicitadas.

O último parâmetro do G5Service “\_”, é opcional. Alguns casos o separador é a "/". Checar na WSDL [g5-senior-services/rubi**\_**Synccom | g5-senior-services/rubi**/**Synccom]. Esse parâmetro permite ajustar esse caso.

Utilizando no workflow G7, os parâmetros de autenticação e a url do ODATA são fornecidos pela API do workflow (ver documentação do workflow para detalhes).

No workflow seria algo assim:

|  |
| --- |
| workflowCockpit({  init: \_init,  onSubmit: \_saveData,  onError: \_rollback  });  function \_init(data, info) {  //obtendo o usuário logado  info.getUserData().then(function(user) {  $scope.user = user.subject;  }).then(function() {  //obtendo o token da plataforma na inicialização  info.getPlatformData().then(function(plataformData) {  //configurando as APIs  var wsserviceurl = "http://[server]:[port]";  var apig5resturl = "https://[server]:[port]/G5ServiceAPI/G5Rest";  G5Service.config(wsserviceurl, apig5resturl, $scope.user, plataformData.token);  ODataServiceUtil.config(plataformData.serviceUrl, plataformData.token);  ... |

Utilizando o ODATAServiceUtil

Para salvar dados utiliza-se o método saveData.

|  |
| --- |
| function \_saveData(data, info) {  $scope.valid();  var isNew = info.isRequestNew();  return ODataServiceUtil.saveData( $scope.entityName, data.processInstanceId, $scope.entity, isNew ).then(function(result) {  return result;  },function(error){  $scope.showError(error);  });  } |

O parâmetro entityName, é o nome do formulário cadastrado no ECM. Para essa forma de uso, precisa ser cadastrado um formulário no workflow que será utilizado como fonte de dados para o serviço do ODATA. O parâmetro processInstanceId servirá como identificador único do registro e o atributo entidade será o registro propriamente dito a ser inserido. Ele deve possuir os atributos conforme definição do formulário ECM, inclusive os parâmetros devem ser dos mesmos tipos. O último parâmetro indica se é um novo registro ou um existente para a API decidir por um insert ou update.

O resultado da função é o retorno de um objeto estruturado conforme esperado pelo workflow.

A função valid deve tratar os erros e disparar uma exception caso ocorra algum para aportar o envio dos dados para o workflow.

|  |
| --- |
| $scope.valid = function(){  if($scope.entity.versao == "0"){  var msg = 'Os dados informados são inválidos.'  throw Error(msg);  }  } |

Para se obter os dados salvos no ODATA, utiliza-se o método getData.

|  |
| --- |
| if (!info.isRequestNew()) {  return ODataServiceUtil.getData($scope.entityName, data.processInstanceId)  .then(function(result) {  $scope.entity = result;  return result;  }, function(error){  $scope.showError(error);  });  } |

Da mesma forma, passa-se o nome da entidade e o identificador único e o retorno é o objeto gravado.

Utilizando o G5Service

Possui um método para cada módulo da G5

|  |
| --- |
| callBSService  callHRService  callQLService  callSMService  callRubiService  callTRService |

Exemplo chamando um serviço do Administração de Pessoal:

|  |
| --- |
| G5Service.callRubiService("com.senior.g5.rh.fp.db", "GetDBDate").then(  function(result) {  $scope.entity.dbdate = result.pmReturnGetDateTime;  },  function(error) {  alert(error);  }  ); |

Exemplo chamando um serviço que recebe parâmetros:

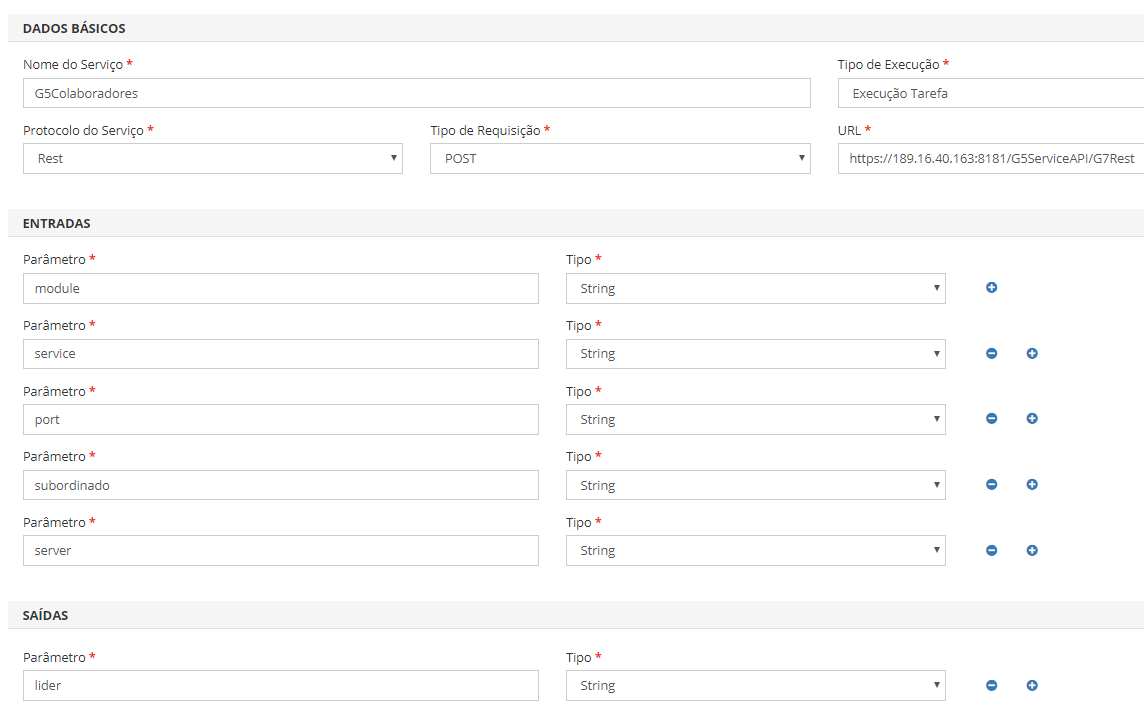
|  |
| --- |
| var params = {  pmToAddress: "destinatario@dominio.com.br",  pmSubject: "Envio e-mail G5Service",  pmBody: "Corpo do e-mail"  };  G5Service.callRubiService("com.senior.g5.rh.fp.email","SendMail", params).then(  function(result) {  //alert("email enviado");  },  function(error) {  $scope.showError(error);  }  ); |

Extras

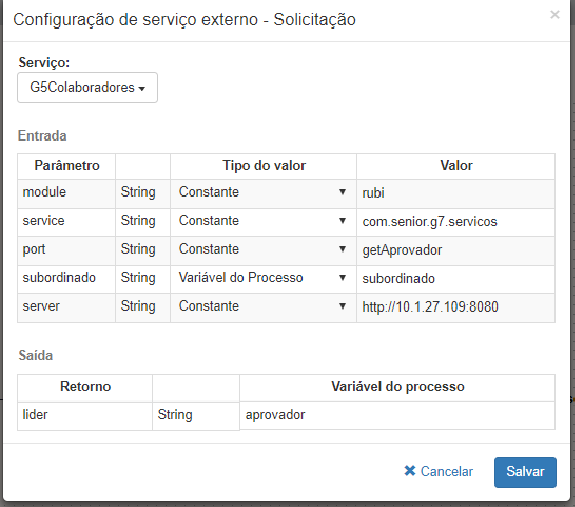
Utilizando a API como serviço externo e fonte de dados em tarefas do fluxo no workflow G7

**Configurando serviço externo:**

Os parâmetros server, module, service e port são obrigatórios, os demais são parâmetros do serviço. A URL tem que ser acessível da núvem.



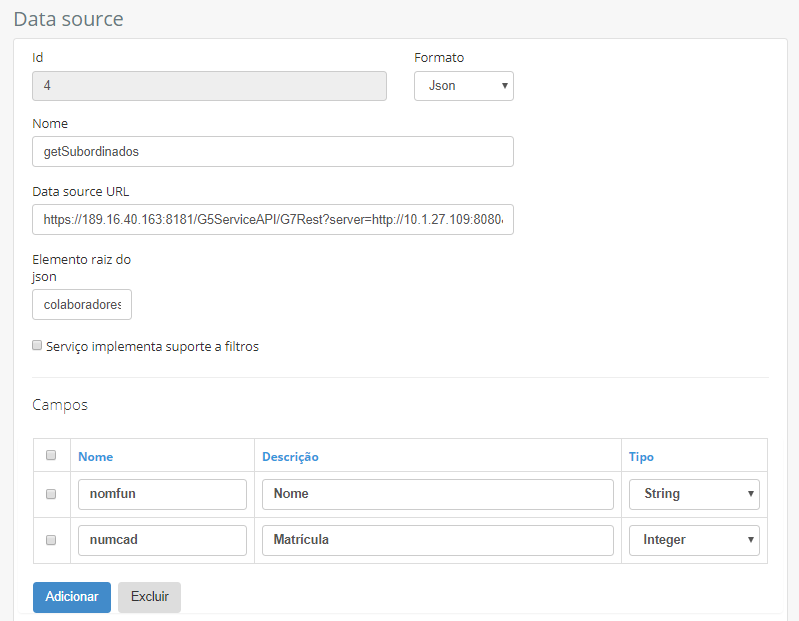
**Utilizando o serviço externo no processo:**



Após a execução do serviço “getAprovador” a variável do processo “aprovador”, será alimentada com o atributo “lider” do serviço.

**Configurando fonte de dados:**

Os parâmetros server, module, service e port são obrigatórios e são enviados na URL (GET)



Exemplo de URL: https://189.16.40.163:8181/G5ServiceAPI/G7Rest?server=http://10.1.27.109:8080&module=rubi&service=com.senior.g7.servicos&port=getColaboradores&empresa=1

**No formulário configurar o campo como datasource:**

