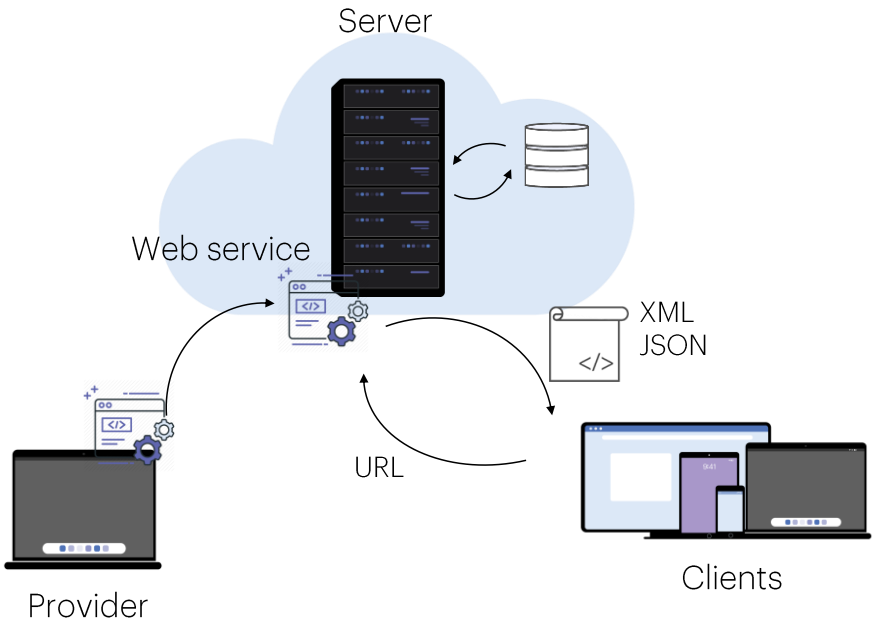
Web Services em GeneXus

Introducción

GeneXus™

A seguir, veremos o que são os web services e como podemos usar esses serviços em uma aplicação GeneXus.

Definição



Os Web Services são programas que fornecem funcionalidades úteis para outros programas e estão localizados em servidores web para que possam ser localizados e chamados por meio de uma rede, geralmente a Internet.

Quando publicamos os serviços do backend de nossa aplicação no servidor web de forma que possam ser acessados por outros sistemas, passam a ser web services. Para facilitar o acesso a estes serviços, são utilizados padrões que definem o mecanismo de interação com os mesmos e o formato da informação recebida.

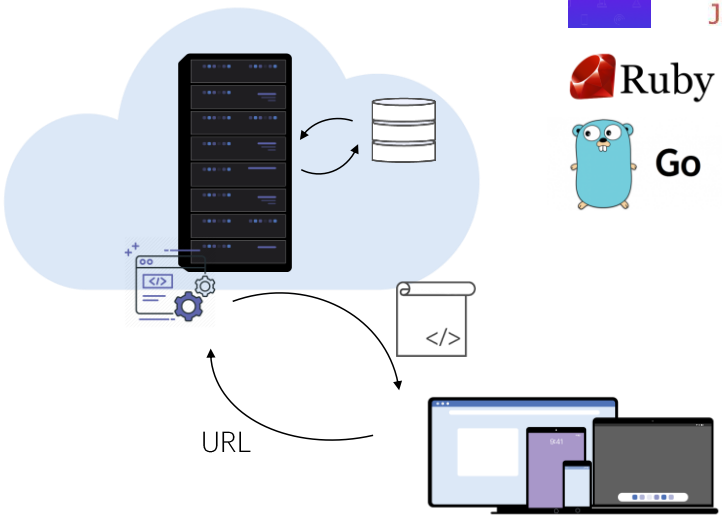
O provedor de serviços "publica" um Web Service em um servidor e as aplicações cliente "consomem" o Web Service publicado.

Para acessar o serviço, a aplicação cliente utiliza sua localização (URL) para chamá-lo e, eventualmente, envia os parâmetros necessários.

Em troca, recebe as informações retornadas, geralmente como uma estrutura no formato XML ou JSON.

Protocolos dos web services

Protocol / Standard	
SOAP	WSDL
REST	OpenAPI



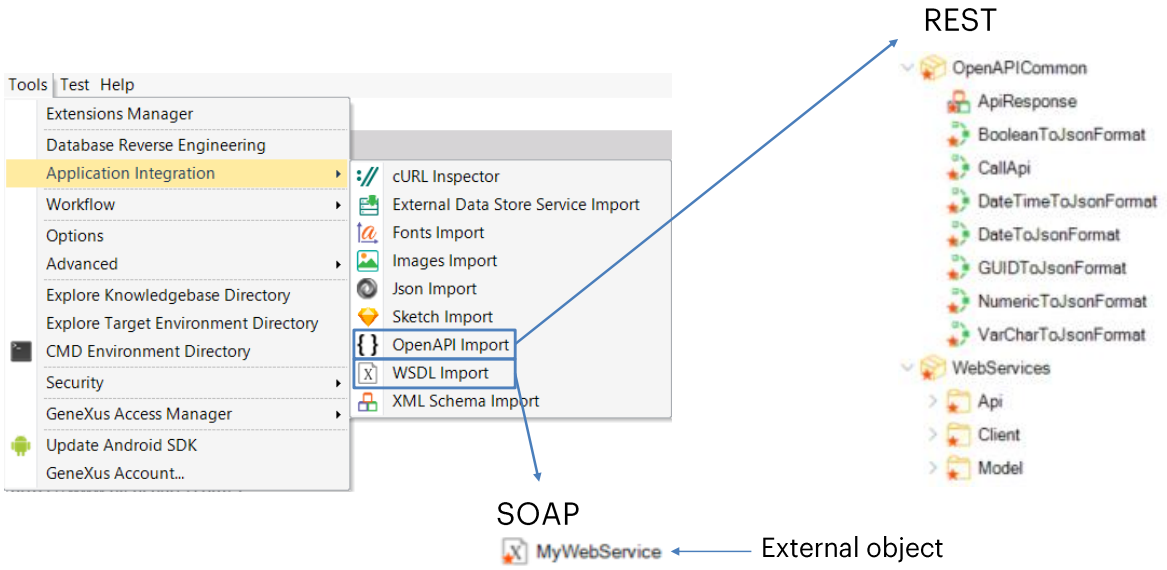
Os Web Services podem ser desenvolvidos seguindo padrões diferentes, os mais comuns no momento são SOAP e REST. Cada padrão define como as informações das funções disponíveis no web service devem ser publicadas.

Os web services SOAP utilizam uma definição escrita em WSDL (Web Services Description Language), enquanto os serviços REST utilizam o padrão OpenAPI.

Para acessar um Web Service publicado, precisamos saber sua localização e importar sua definição para ter acesso às funções disponíveis no serviço web.

O GeneXus permite consumir Web Services que foram desenvolvidos em qualquer ferramenta de programação ou plataforma, com protocolos SOAP ou REST.

Como consumir um web service em GeneXus

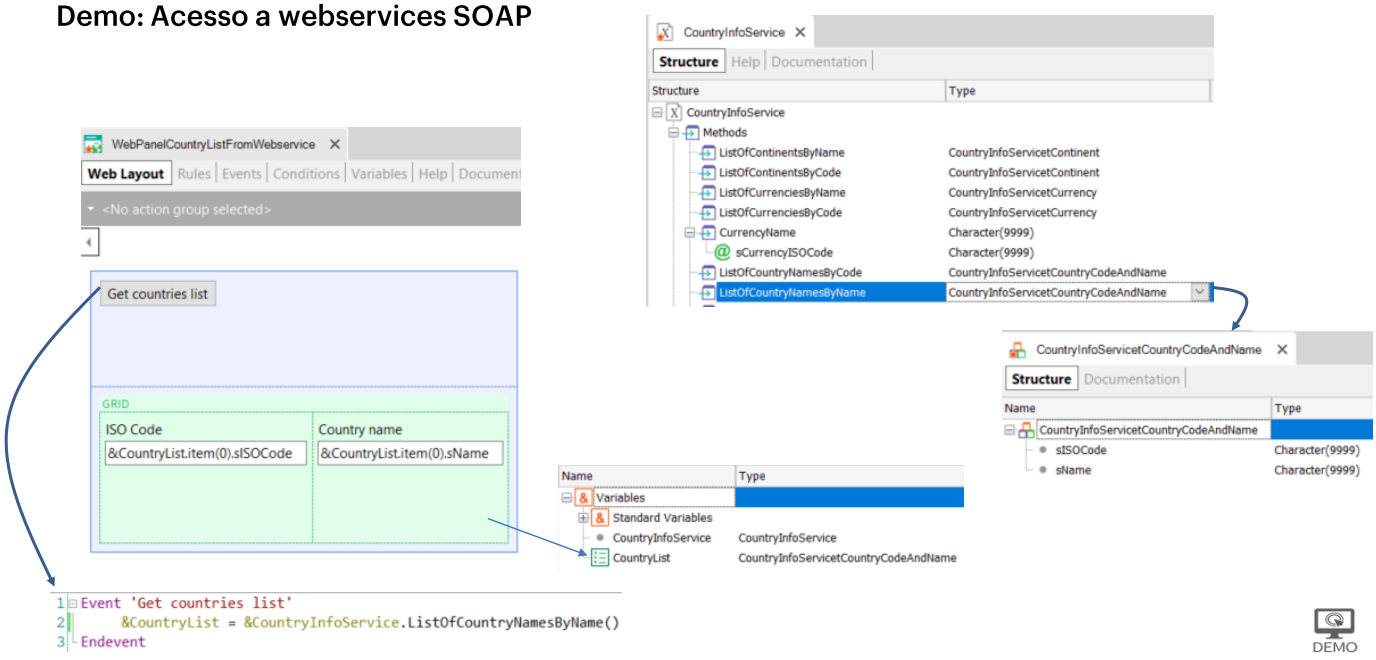


Para integrar um Web service em uma aplicação GeneXus, vamos para Tools/Application Integration, escolhemos WSDL Import se o web service seguir o protocolo SOAP e OpenAPI Import se o webservice tiver arquitetura REST.

Isto disparará um wizard que irá variar dependendo do tipo de serviço web escolhido. Ao concluir, se o webservice for SOAP, GeneXus criará automaticamente um Objeto Externo associado ao Web Service e os tipos de dados estruturados necessários para gerenciar seus dados.

Se em vez disso fosse REST, será criada automaticamente uma série de objetos GeneXus, (que normalmente incluiremos dentro de um módulo, por exemplo, WebServices) e que nos permitirão executar o serviço diretamente através desses objetos em nossa KB. O wizard deixará em uma pasta API os programas a serem chamados e em uma pasta Model os SDTs para manipular os dados.

Demo: Acesso a webservices SOAP

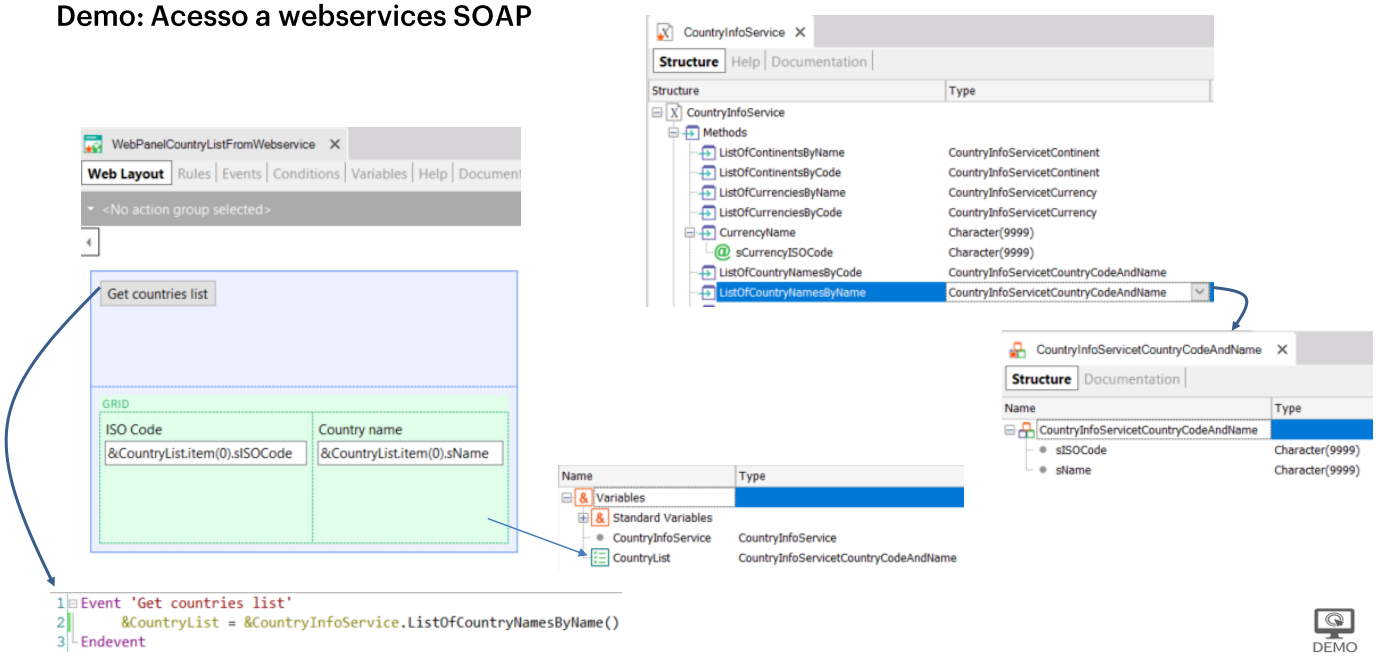


Comecemos importando para nossa aplicação uma lista de países a partir de um webservice SOAP.
Para isso acessamos o menu Tools/Application Integration/WSDL import e escrevemos a url que vemos em tela (http://webservices.oorsprong.org/websamples.countryinfo/CountryInfoService.wso?WSDL) , e pressionamos Next.
Vemos que foi encontrado um serviço chamado CountryInfoService. Para ordenar o que importamos, vamos salvar os objetos no folder SOAPWebService, deixamos o prefixo sugerido e pressionamos Next.
Clicamos no sinal "+" e, em seguida, fazemos o mesmo no nó Service Description.
Observamos que se abre uma lista das funções oferecidas pelo webservice, como, por exemplo: ListOfCountryNamesByName para obter a lista de nomes de países, CountryCurrency para obter a moeda de um país, CountryFlag para obter a imagem de sua bandeira, etc.

Pressionamos Import para que GeneXus importe a definição do webservice e vemos que a janela de output nos informa que estão sendo importados uma série de tipos de dados estruturados e outros componentes. Ao finalizar, verificamos que aparece no KBNavigator a pasta SOAPWebService e se vemos seu conteúdo, encontramos o objeto externo CountryInfoService e uma série de SDTs que nos permitirão armazenar as informações recebidas a partir de cada método do serviço.

Se clicamos duas vezes sobre CountryInfoService, podemos ver que todos os métodos oferecidos pelo webservice CountryInfoService foram incorporados ao objeto externo, detalhando os parâmetros necessários de cada método e que tipo de dados retorna. Particularmente nos interessa o método LisfOfCountryNamesByName, que retornará uma lista de países ordenada por nome.

Demo: Acesso a webservices SOAP



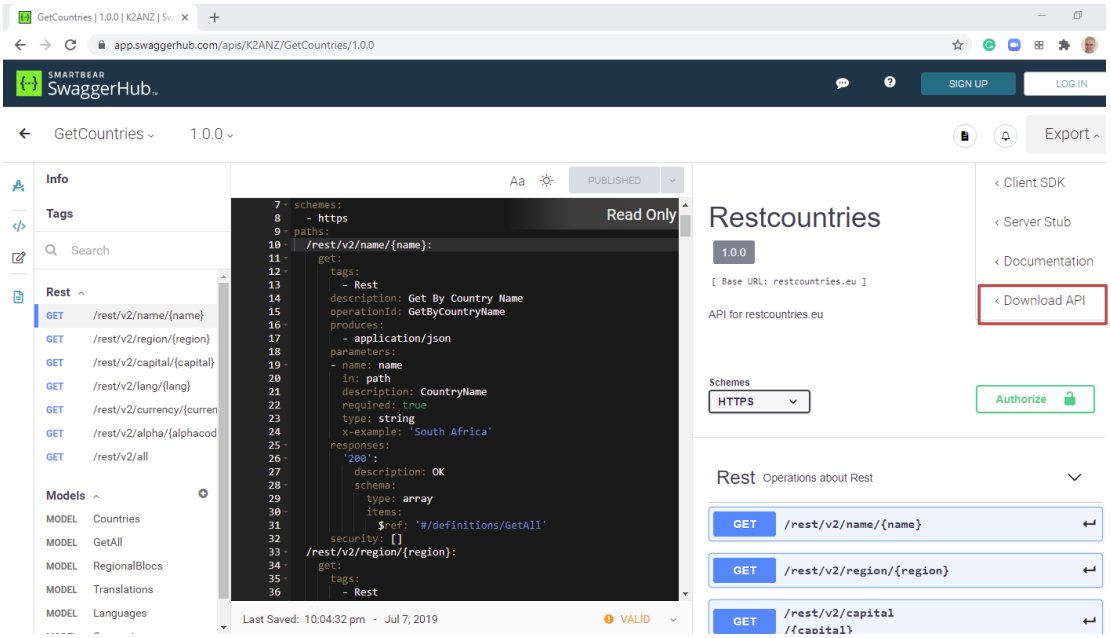
Agora criamos um webpanel denominado WebPanelCountryListFromWebService. Em suas variáveis, criamos uma variável de nome CountryInfoService que automaticamente fica do tipo de dados do objeto externo.
Se voltamos à definição do objeto externo, vemos que o tipo de dados retornado pelo método LisfOfCountryNamesByName é um SDT chamado CountryInfoServicetCountryCodeAndName. Se o abrimos, vemos que armazena o código ISO do país e o nome.
Voltamos ao webpanel, criamos uma variável CountryList, do tipo de dados do SDT CountryInfoServicetCountryCodeAndName e a marcamos como coleção.
Agora no form, arrastamos um botão com o nome do evento: Get country list, damos um duplo clique sobre ele e no evento inserimos a variável &CountryList e lhe atribuímos a variável &CountryInfoService. Se escrevemos ponto, vemos que temos acesso a todos os métodos do webservice, então escolhemos ListOfCountryNamesByName.
Voltamos ao form e arrastamos a variável CountryList baseada no sdt e pressionamos OK.
Executamos....

Se abrimos o webpanel WebPanelCountryListFromWebService e pressionamos o botão Get country list, vemos que obtemos a lista de países em ordem alfabética com o Código ISO de cada um, exatamente como esperávamos.

Podemos então usar estes dados em nossa aplicação da agência de viagens como lista de seleção para filtrar por um país, ou em diferentes usos.

Demo: Acesso a webservices REST com protocolo OpenAPI

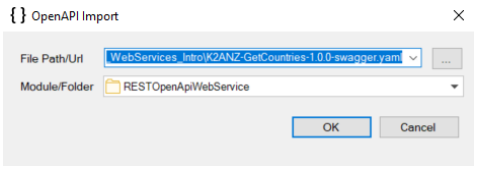
GeneXus atualmente pode importar OpenAPI versão 2.0

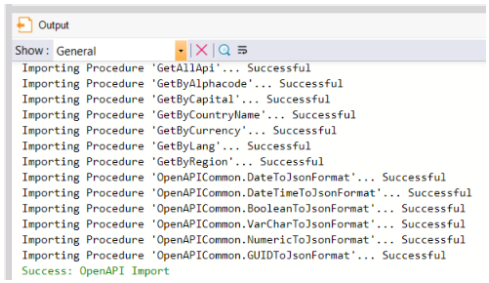


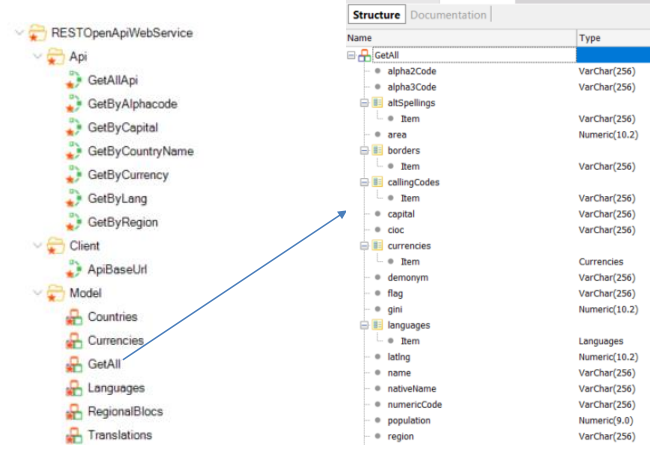
DEMO

Vejamos o caso de importação de um webservice REST que cumpre com a especificação OpenAPI, também conhecida como especificação Swagger.
Obtemos o exemplo da página: https://app.swaggerhub.com, uma api chamada GetCountries. Escolhemos a opção Download API e baixamos o arquivo .yaml.

Demo: Acesso a webservices REST com protocolo OpenAPI








```
GetAllApi(&GetAllOUT, &HttpMessage, &IsSuccess)
```

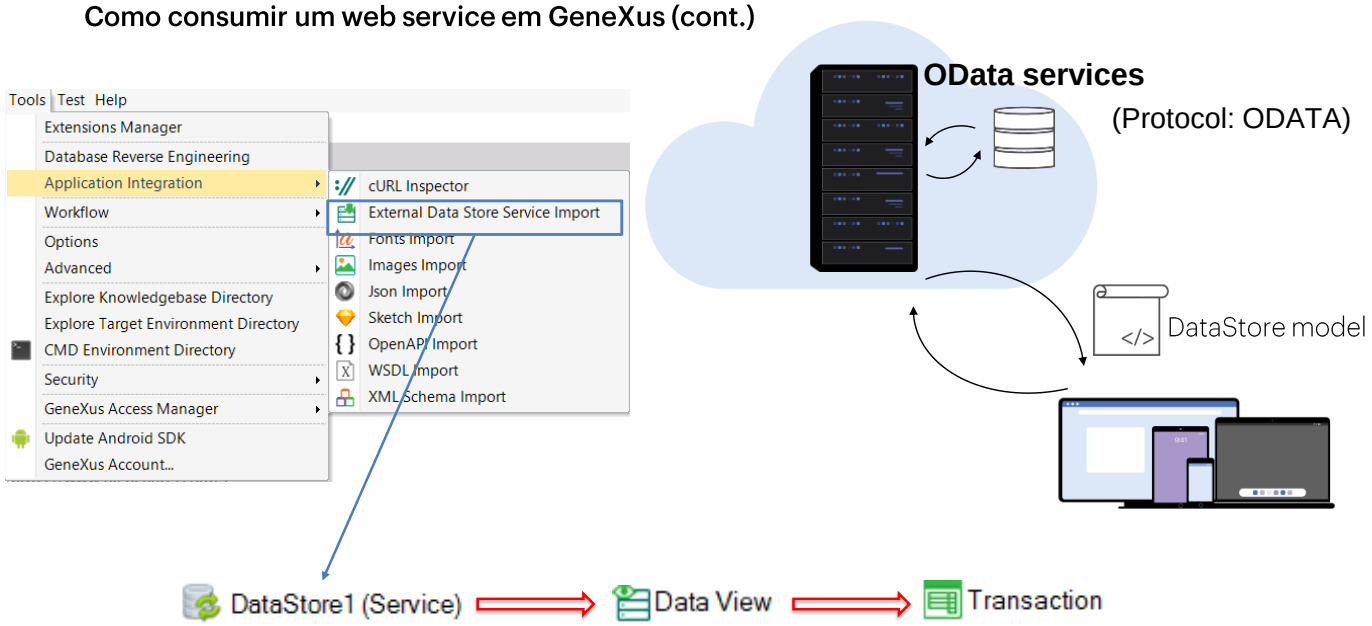
DEMO

Em seguida, vamos para Tools/Application Integration/OpenApi import e escolhemos o arquivo .yaml que baixamos. Como folder RESTOpenApiWebService e pressionamos Ok.
Vemos na janela de Output que são importados vários elementos e ao finalizar abrimos a pasta de destino. Verificamos que foram criadas as pastas Api que contém os métodos que podemos invocar, a pasta Client com o método ApiBaseURL e a pasta Model que contém vários SDTs que são os tipos de dados retornados pelos métodos anteriores.

Se abrimos o SDT chamado GetAll, podemos ver todos os dados que podemos recuperar dos países. E se quisermos obter a lista dos países, podemos invocar o método GetAllApi que nos retornará os dados nas variáveis &GetAllOut (coleção de elementos GetAll), &HttpMessage e a variável booleana &IsSuccess.

Em seguida, podemos percorrer a coleção &GetAllOut para obter os dados dos países.

Como consumir um web service em GeneXus (cont.)



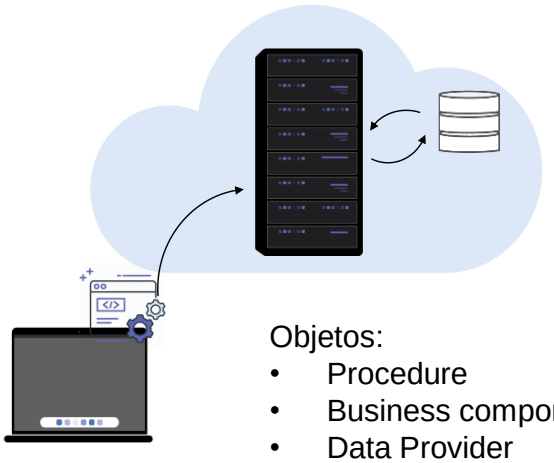
Um tipo especial de serviços web são os chamados OData services. Estes serviços utilizam o protocolo Open Data Protocol (Odata) projetado para fornecer operações para inserir, modificar ou excluir registros de uma base de dados, por meio de um serviço web.

Como em um webservice Soap ou Rest tinha que importar sua definição primeiro, para consumir um serviço Odata nós devemos primeiro importar o modelo com as entidades da base de dados incluídas no serviço.

Para fazer isso, vamos a Tools/Application Integration e escolhemos External DataStore Service Import. Como o processo envolve a criação de um DataStore do tipo Serviço para associá-lo ao datastore externo, só podemos usar esta funcionalidade no GeneXus Full.

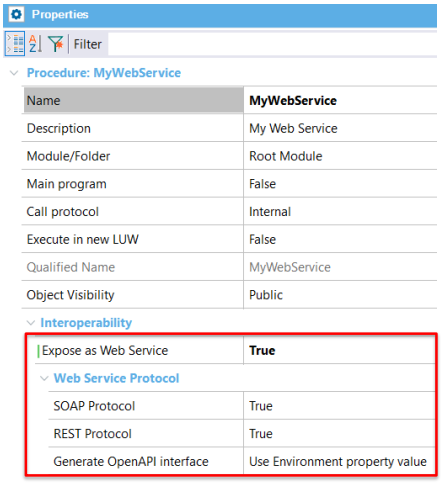
Depois de executado o wizard, serão criados tantos objetos transação como entidades que estavam no modelo e poderemos trabalhar com elas como se fossem qualquer outra transação da nossa aplicação.

Como publicar um web service com GeneXus



Objetos:

- Procedure
- Business component
- Data Provider



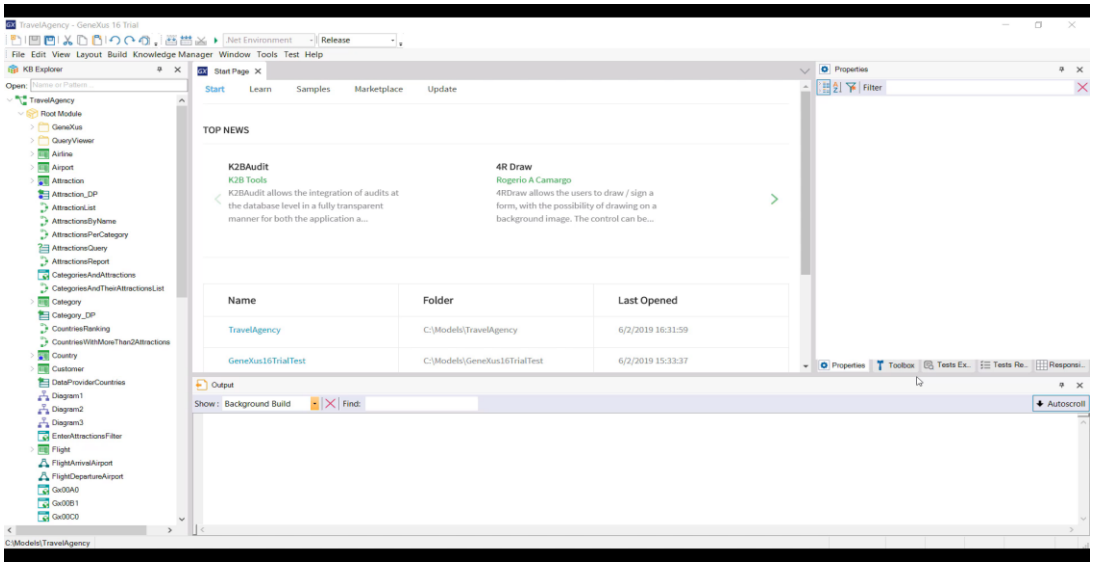
O GeneXus também permite criar web services e publicá-los em um servidor web.

Os objetos GeneXus que podem ser expostos como um web service são os procedimentos, os business components e os data providers.

Para expor um destes objetos como web service, atribuímos a propriedade Expose as Web Service > True e, em seguida, poderemos especificar se queremos usar o protocolo SOAP, REST ou ambos.

Se queremos que nosso serviço forneça operações de CRUD em uma base de dados, também podemos gerar as informações necessárias seguindo o protocolo Odata

Exemplo de uso de um web service REST



DEMO

[DEMO: https://youtu.be/bvmt0Gjcxpw]

Como exemplo do que foi visto, criaremos um web service REST que insere uma nova companhia aérea na base de dados e o chamaremos a partir de nossa aplicação.

Para fazer isso, primeiro abrimos a transação Airline e colocamos sua propriedade Business Component em True. Em seguida, criaremos um objeto procedimento que receba por parâmetro os dados de uma companhia aérea e insere um novo registro na tabela Airline.

Chamamos o procedimento CreateNewAirlineWS e definimos uma regra Parm com 2 variáveis de entrada: &AirlineName e &AirlineDiscountPercentage. Definimos estas duas variáveis usando Add Variable e então, na seção Variables, definimos mais uma variável, Airline do tipo business component Airline.

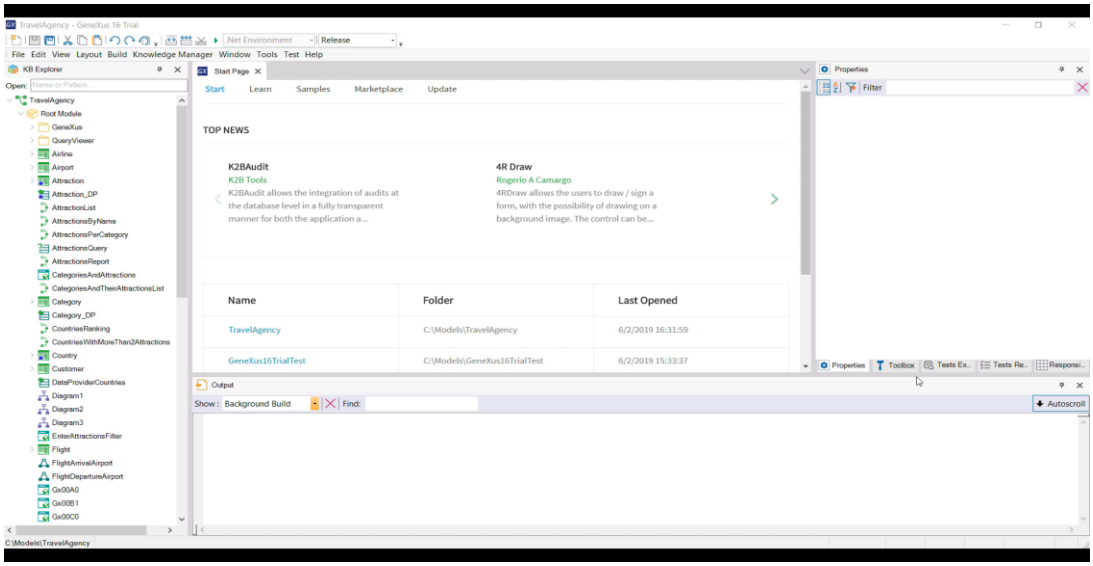
No source, escrevemos o código para inserir uma companhia aérea com os dados recebidos por parâmetro. Não precisamos definir valor para o AirlineId, uma vez que é autonumerado.

Agora vamos para as propriedades do objeto procedimento e atribuímos **Expose as Web Service** em True, **SOAP Protocol** em False, **REST Protocol o deixamos** em True e **Generate OpenAPI interface** em Yes. Este último permitirá que seja gerada a definição de nosso webservice no padrão OpenAPI.
Clicamos com o botão direito no procedimento e escolhemos Build With This Only, com o que o serviço será publicado em nosso servidor web, neste caso em nossa máquina local. Na janela de Output, vemos uma mensagem que confirma que foi gerada a documentação da API Rest do webservice, com a definição do mesmo.

Antes de importar o webservice, criaremos um novo módulo, que chamamos de WebServices,

para que tudo seja salvo nesse módulo.

Exemplo de uso de um web service REST



DEMO

[DEMO: https://youtu.be/bvmt0GjcxpW]

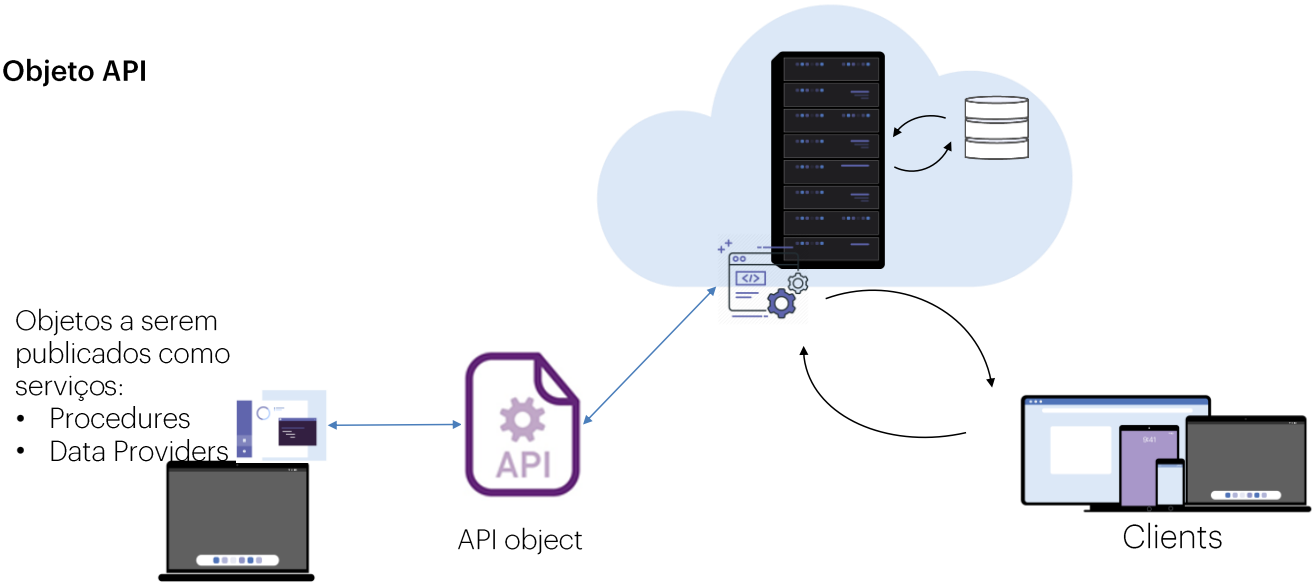
Para importar o webservice, vamos a Tools/Application Integration e escolhemos OpenAPI Import. No File Path / URL escrevemos: C:\Models\TravelAgency\CSharpModel\Web\default.yaml que é onde foi gerada a documentação da API Rest e escolhemos como destino o módulo WebServices. Vemos que tudo foi importado corretamente e se abrimos o módulo WebServices, na pasta Api está o procedimento importado. Também vemos que na pasta Model existe um SDT chamado CreateNewAirlineWSInput, onde se o abrimos vemos que os parâmetros que precisamos passar para o serviço estão disponíveis aqui.

Para provar que o web service funciona corretamente, criamos um web panel chamado CreateNewAirlineUsingWS. Em seguida, arrastamos um botão para o form e chamamos o evento Create airline. Agora vamos para as variáveis e definimos uma variável &CreateNewAirlineWSInput que é automaticamente definida a partir do tipo do SDT.
Para ter um feedback do resultado da execução do webservice, também criamos uma variável &IsSuccess e outra &HttpMessage.

Clicamos duas vezes no botão e no evento carregamos os membros do SDT, então chamamos o webservice passando o SDT como parâmetro e por último escrevemos o seguinte código para dar mensagens por tela

Pressionamos F5... Vemos as companhias aéreas que temos inseridas... E executamos o webpanel que acabamos de criar.
Pressionamos o botão e vemos que o serviço nos informa que a companhia aérea foi criada corretamente.
Para verificar, vamos para a transação Airline... E vemos que a companhia aérea que queríamos foi realmente adicionada.

Objeto API



Veremos agora outro mecanismo para publicar objetos como serviços, utilizando o objeto API.

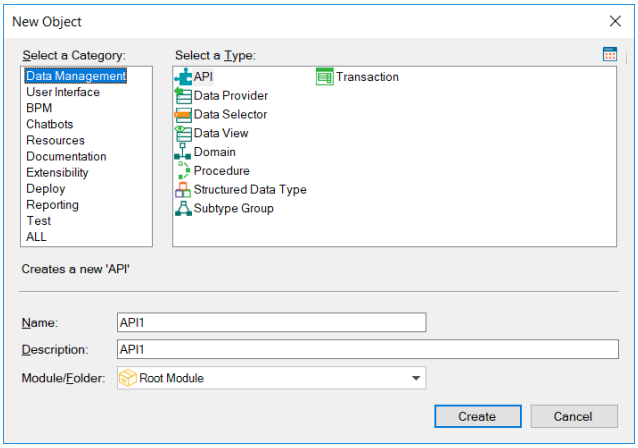
O objeto API (seu nome vem de Application Programming Interface) é um objeto GeneXus que nos permite definir uma interface de acesso por programação a alguns objetos de nossa aplicação, como procedimentos ou data providers.
Isto significa que uma aplicação externa à nossa poderá acessar estes objetos publicados como web services.

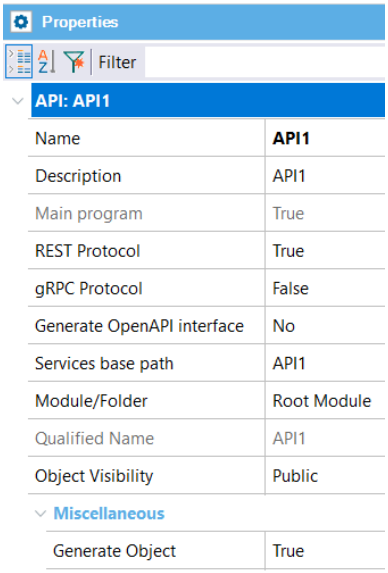
O objeto API adiciona uma camada intermediária que separa a interface dos detalhes de implementação, o que permite que futuras alterações na programação nos objetos não afetem a maneira como são chamados pelas aplicações externas.

Por exemplo, é feito um mapeamento entre o nome interno do objeto na KB e o nome com o qual ele é exposto como serviço, bem como também é possível alterar o tipo e nome dos parâmetros ou alterar o path de acesso, sem que isso afete a forma como o serviço é invocado.

Esta abstração fornece uma flexibilidade importante porque podemos evoluir nossos serviços sem forçar as aplicações que os utilizam a mudar seu código para se adaptar a essas mudanças.

Objeto API (cont.)

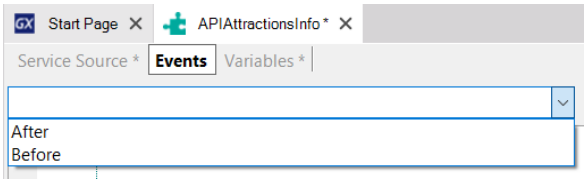


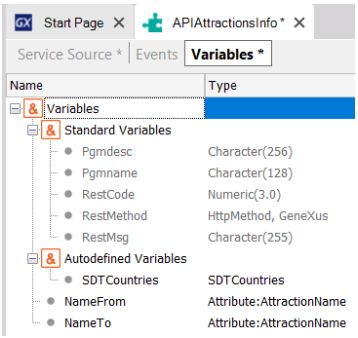


Para criar um objeto API, vamos para a categoria Data Management e selecionamos o tipo API. Em suas propriedades podemos definir em tempo de desenho seu nome, endereço, protocolo a ser utilizado, entre outras coisas.

O protocolo gRPC é um moderno framework open source desenvolvido pela Google, que permite fazer chamadas para procedimentos remotos com um alto desempenho e de forma bidirecional.

Objeto API (cont.)





O objeto API tem três seções: Service Source, Events e Variables.

Em Service Source se define mediante uma sintaxe declarativa, o nome de cada serviço a ser publicado, com os respectivos parâmetros e o nome do objeto GeneXus em nossa KB que estamos expondo como serviço, com seus respectivos parâmetros.

No exemplo, vemos que o data provider que criamos anteriormente denominado RankingCountriesWithAttractionsQty, o estamos publicando como serviço com o nome RankingCountriesAttractions e os parâmetros que expomos são os mesmos do objeto.
O mesmo para a lista AttractionsByName que publicamos com o nome ListAttractionsByName.

Esta definição nos permite no futuro alterar o nome ou os parâmetros do objeto GeneXus, sem que altere a definição de como está publicado este objeto.

Nos eventos dispomos do evento Before e After, com o que podemos realizar ações que se executem antes ou depois da invocação ao objeto como serviço.

Nas variáveis existem algumas do tipo padrão, que nos permitem definir ou alterar características do objeto API em tempo de execução.

Mais informações sobre Web Services

WSDL:https://wiki.genexus.com/commwiki/servlet/wiki?6181
OpenAPI:https://wiki.genexus.com/commwiki/servlet/wiki?31864
OData: https://wiki.genexus.com/commwiki/servlet/wiki?40713
API Object:https://wiki.genexus.com/commwiki/servlet/wiki?46151

Para mais informações sobre webservices em GeneXus, siga os seguintes links da Wiki.

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications