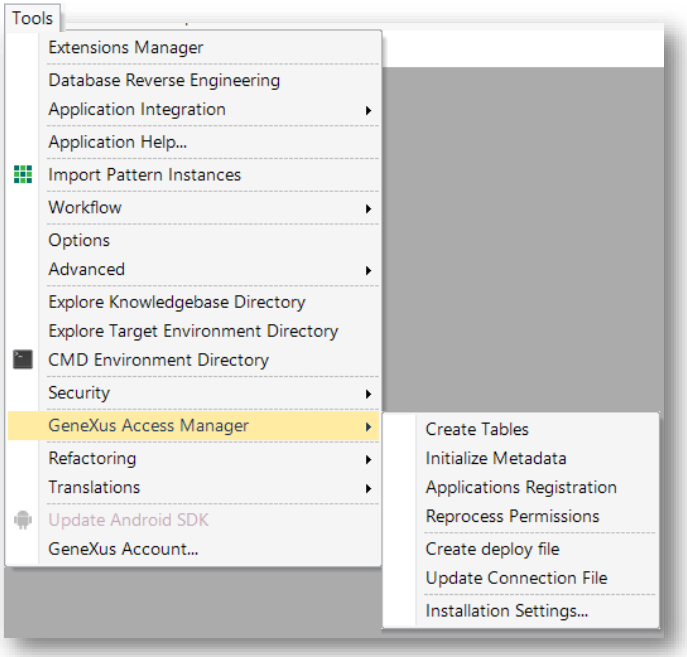
# GAM Components

Nicolas Adrién | GeneXus Training

Integration with GeneXus

| Backend | Data Store | Frontend |
|---|---|---|

GAM suporta plataformas para executar seu ambiente tanto web como móvel.
É possível definir a linguagem a ser utilizada no Backend, os sistemas de gerenciamento de bases de dados e a plataforma a ser utilizada no FrontEnd. Tudo a partir da UI de GeneXus.
Mais adiante entraremos em profundidade em cada um deles.



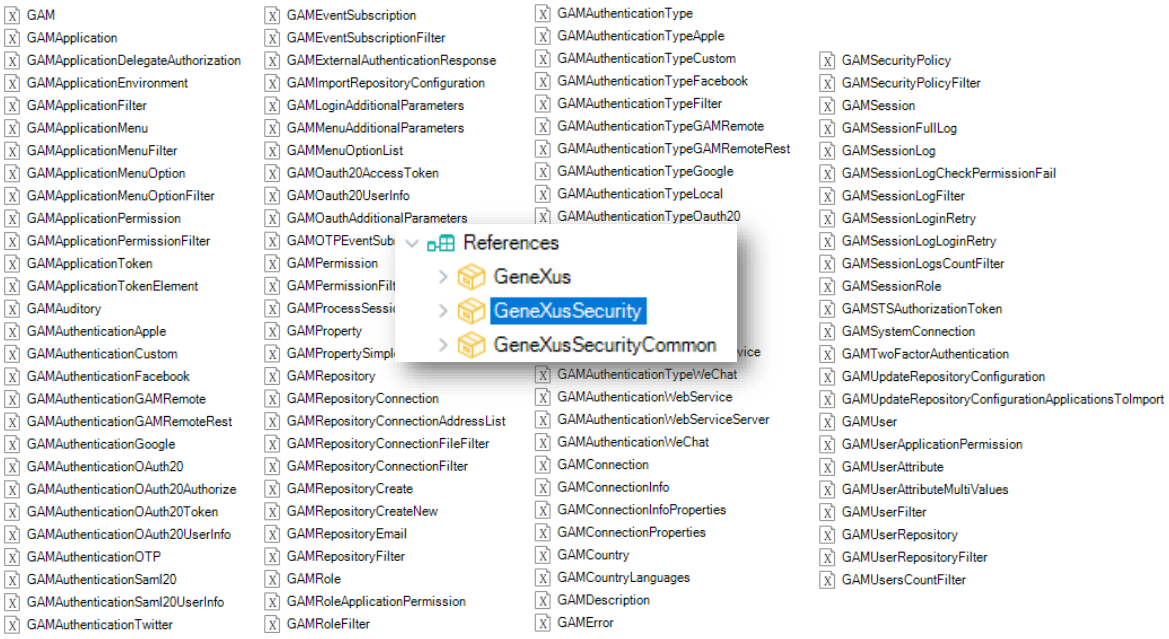

No menu Tools/GeneXus Access Manager podemos acessar as diferentes funcionalidades que nos oferece o GAM dentro do GeneXus.

A partir dali podemos:

- Criar as tabelas que utiliza GAM
- Inicializar os metadados
- Registrar as aplicações que utilizarão GAM
- Reprocessar as permissões das mesmas
- Criar o arquivo de deploy
- Modificar o arquivo de conexões

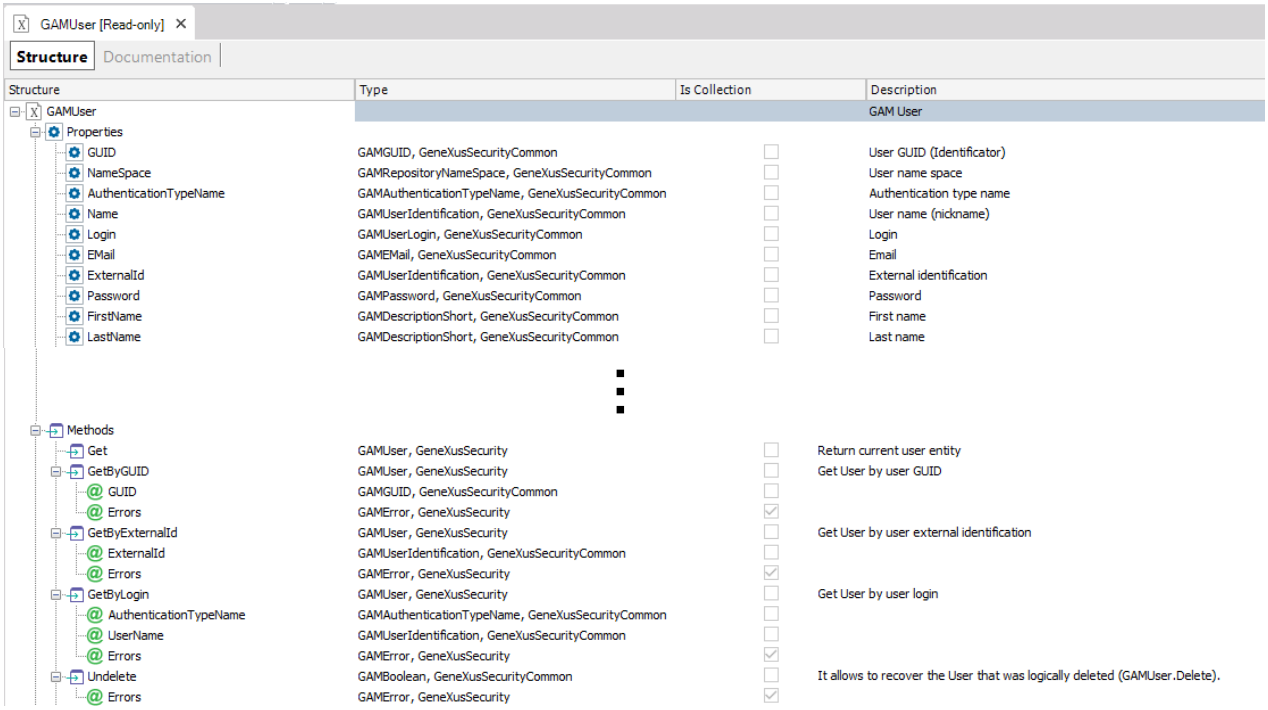
E por fim,

- Configurar a instalação do mesmo




No item References do menu contextual da KB, dentro de GeneXusSecurity podemos encontrar todos os External Object que nos fornece GAM, como vemos em tela.

Vejamos a estrutura de um como exemplo.




Aprofundando-se por exemplo, no External Object GAMUser, vemos que sua estrutura (igual a todos os outros) é composta por propriedades e métodos. Dado que este External Object é bastante grande, foi simplificado o conteúdo a ser exibido neste slide para que fique visível o conteúdo explicado. A partir de GeneXus podem ser vistas todas as propriedades e métodos que ele contém.

Podemos notar que para cada propriedade e método, por sua vez temos o tipo, se é coleção ou não, e descrição que nos servem de ajuda no momento de querer utilizá-los.

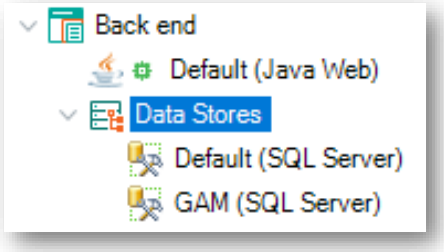
Vejamos um exemplo de uso.

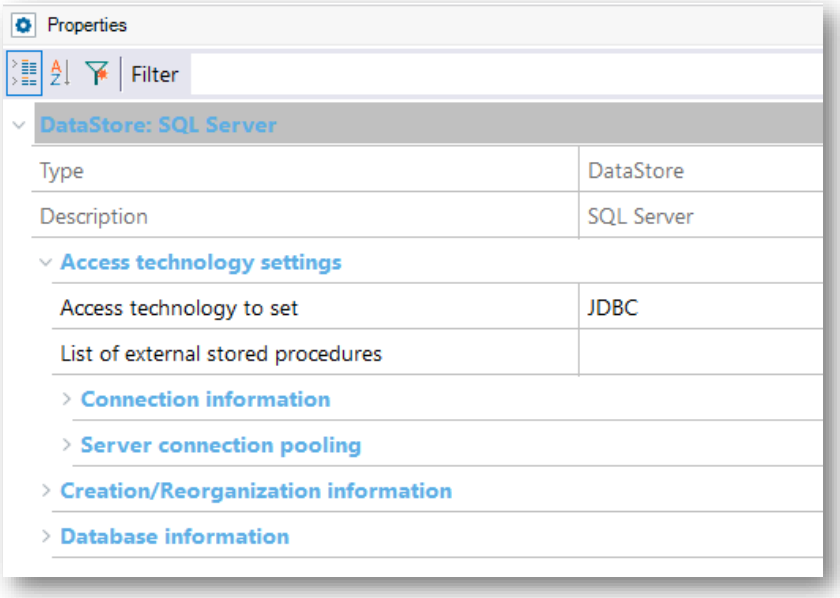

```
&User.Name = "John"

&Name = &User.GetName()
```

Para utilizar o External Object devemos criar uma variável deste tipo (no nosso caso GAMUser) e para utilizá-la no código, simplesmente são atribuídos valores aos seus atributos ou propriedades, ou são utilizados seus métodos para obter os dados já carregados.

Para utilizar o restante dos External Object, segue o mesmo fluxo e metodologia.




No momento de habilitar GAM na KB, é gerado um Data Store para GAM para armazenar sua informação junto ao da Aplicação, onde é possível configurar a tecnologia a ser utilizada, a informação de conexão e servidor, etc.

Platforms

| Backend | Data Store | Frontend |
| --- | --- | --- |



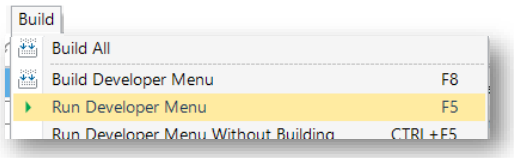


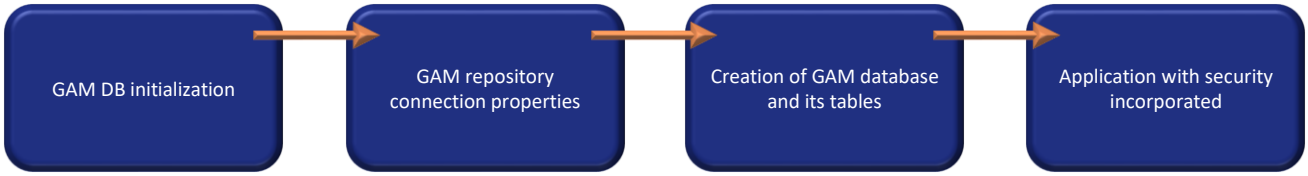


...


Entrando em mais detalhes sobre as diferentes plataformas que suporta GAM, temos que:
Para Backend, pode ser gerado nas linguagens Java, C# e .NET Core.
Para o Data Store, são oferecidas muitas opções onde podem ser destacadas PostgreSQL, SQL Server, MySQL, entre outras.
Finalmente, para Frontend é possível utilizar versões web, como Java ou Angular, e versões móveis, como Android e Apple.

Databases



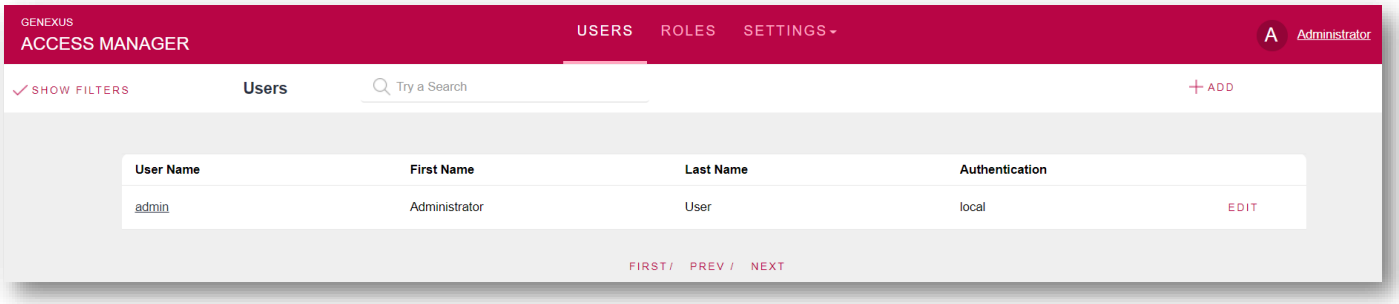




Após executar o Developer Menu, é estabelecida uma conexão com a base de dados especificada no Data Store de GAM, usando as propriedades de conexão dele, verificando a existência de algumas tabelas e a versão do GAM.

Dado que estas tabelas não existem, são criadas as tabelas da base de dados GAM. Então acontece o seguinte:

Primeiro são atribuídas algumas propriedades relacionadas à conexão com o repositório GAM, com seus valores predeterminados.
Em seguida, é criada a base de dados GAM e todas as suas tabelas. Antes de criar as tabelas, é perguntado ao usuário se deseja criar a estrutura da base de dados GAM.

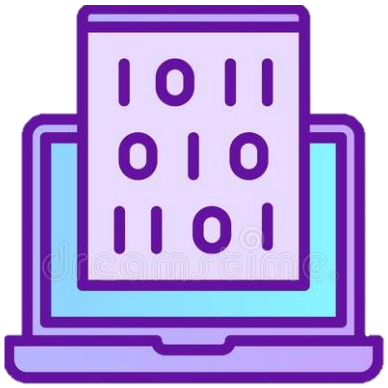
# Backend



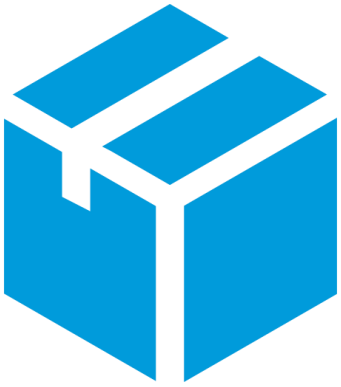

O Backend do GAM é uma Aplicação GAM definida automaticamente com a criação dos metadados GAM, durante o processo de inicialização.

O objetivo desta aplicação é configurar e gerenciar todos os conceitos relacionados ao GAM.

Backend



Compiled

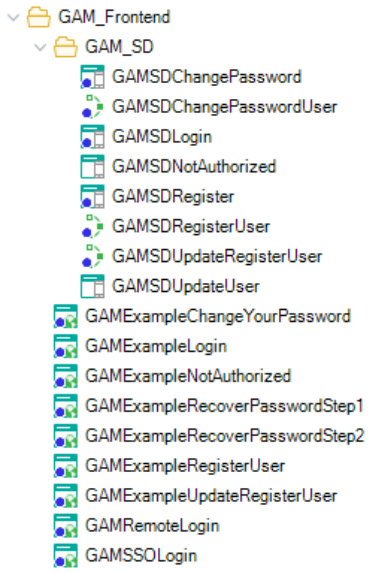


XPZ

Existem duas versões do Backend.
A primeira é a compilada. Por padrão, é criada ao habilitar GAM na KB.
A segunda versão é por XPZ. Um XPZ refere-se aos arquivos utilizados pelo Manager de KBs para trocar objetos entre bases de conhecimento. O nome vem da extensão destes arquivos. Neste caso, pode ser incorporado GAM à KB importando este xpz que vem na instalação de GeneXus.

Estes objetos são úteis, pois são exemplos de uso da API GAM.
Podem ser modificados à vontade pelo desenvolvedor GeneXus se não forem atendidos alguns requisitos (para isso está disponível a API GAM).

## Frontend




No momento de habilitar GAM, são importados para a KB todos os objetos de exemplo do Frontend.

Os exemplos de GAM são objetos web e painéis SD que utilizam a API GAM, e sua finalidade é ajudar o usuário GeneXus a aprender a usar esta API.
Outra finalidade importante destes objetos é ajudar o usuário a começar a utilizar GAM, pois incluem objetos para login, registro, alteração de senhas, redirecionamento em caso de erro de Autorização, entre outros.

Todos estes objetos são consolidados na KB durante o processo de ativação do GAM e são colocados na pasta Exemplos do GAM; toda vez que é instalado um novo build ou upgrade do GAM, o usuário pode decidir se deseja atualizar esses exemplos com o look and feel da aplicação, por exemplo.

GAMRegisterUser GAMSDRegisterUser

Uma boa prática consiste em criar cópias dos exemplos para serem utilizados na aplicação. Desta forma, perante uma atualização do GeneXus (e, portanto, do GAM), as mudanças introduzidas não serão substituídas.

Outra recomendação é eliminar aqueles painéis que não serão utilizados. Um exemplo disto é o auto registro que veremos mais adiante.
Se na aplicação os usuários forem registrados por um administrador, é recomendável remover os painéis GAMRegisterUser e GAMSDRegisterUser.

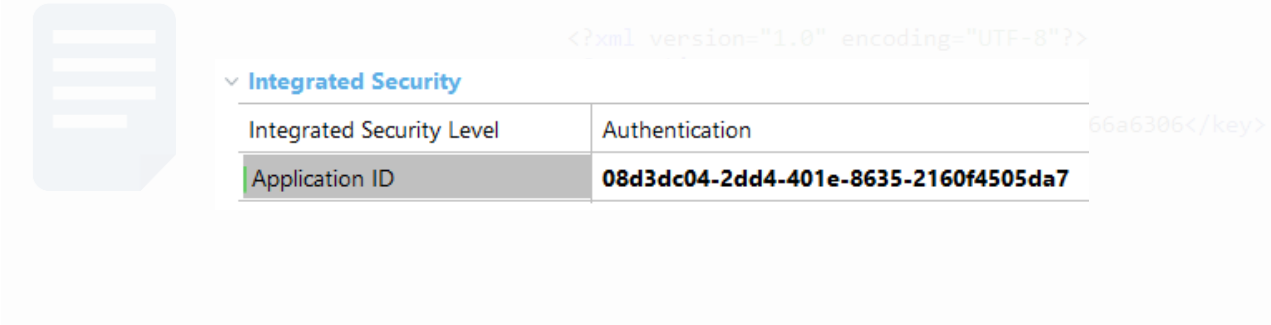
API


GAM oferece uma API que permite aos usuários manipular tipos de dados e métodos para adicionar segurança às aplicações GeneXus (tanto aplicações Web como aplicações móveis).

Quando a segurança integrada está habilitada na KB, são incorporados objetos externos para permitir que o usuário interaja com a API do GAM. Os objetos externos são a maneira de acessar a API do GAM. Todos eles estão distribuídos em um módulo chamado GeneXusSecurity, como dissemos anteriormente.

As APIs são as utilizadas pelos painéis de exemplo. A partir delas, é possível realizar todas as ações disponíveis no Backend do GAM.

Settings




application.gam

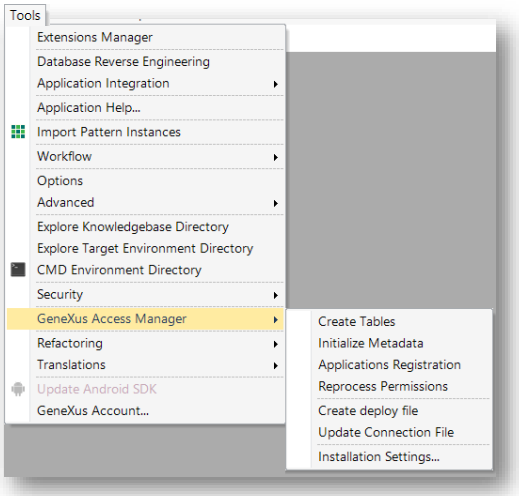
```
<?xml version="1.0" encoding="utf-8"?>
<Application>
    <Id>08d3dc04-2dd4-401e-8635-2160f4505da7</Id>
</Application>
```

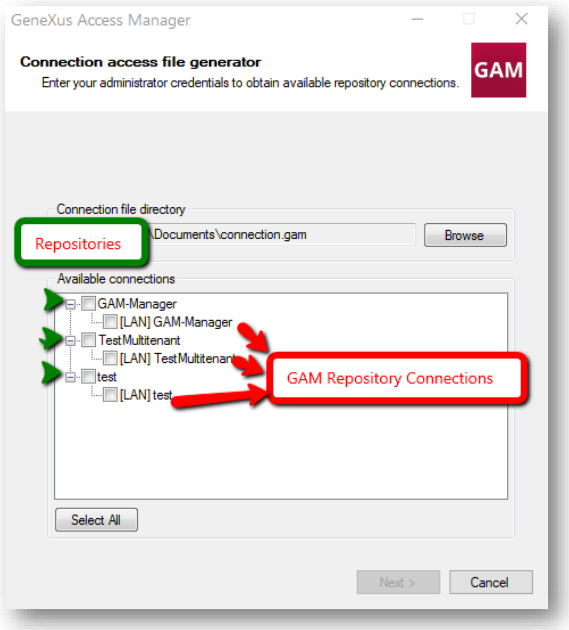
GAM gera diferentes arquivos com configurações para a conexão com repositórios.
Estes arquivos são connection.gam e application.gam.

**connection.gam** é um arquivo de configuração do GAM que contém a chave associada a cada Repositório ao qual deseja-se conectar. A informação de conexão associada à chave deve existir na base de dados do GAM (deve ter sido criada previamente usando a API do GAM).
No caso de Java, este arquivo é copiado para a raiz da aplicação web no servidor de servlets (isto é a partir do upgrade 3 de Evolution 2).
No caso de NET vai no diretório virtual (ou seja, o diretório web).

**application.gam** é um arquivo de configuração do GAM que contém o id da Aplicação WEB GAM da KB, especificado na propriedade Application Id e é utilizado para que GAM identifique qual aplicação web está executando.
Durante o período de criação de protótipos, é transferido automaticamente para a pasta WEB-INF no caso das aplicações Java e para o diretório virtual no caso das aplicações NET.
Ao levar a aplicação para produção, deve ser incluído este arquivo na implementação.
O ID que contém deve existir em uma aplicação definida na Base de Dados GAM, e é configurável a partir do IDE de GeneXus através das propriedades do Environment na seção Integrated Security, como vemos em tela.

## Settings - Tools




Existem ferramentas do GAM para modificar a informação dos arquivos de configuração para diferentes ambientes, como Teste, Pré-Produção ou Produção, com a finalidade de facilitar a edição deles.

Para o arquivo **connection.gam**, existe a ferramenta **GamDeployTool** que pode ser executada a partir do IDE de GeneXus, selecionando a opção *GAM - Update Connection File*.
Esta ferramenta destina-se ao uso pelos administradores do Repositório do GAM Manager.
Ao usá-la, são adicionadas entradas em determinadas tabelas da base de dados GAM dependendo da seleção do usuário, quem pode selecionar entre as conexões de Repositório GAM existentes de cada Repositório.

Ao finalizar o processo, o arquivo é gerado no endereço especificado para, em seguida, ser copiado para a aplicação web.

## Settings

client.cfg

```
EnableIntegratedSecurity=1
IntegratedSecurityLoginWeb=GAMExampleLogin
IntegratedSecurityNotAuthorizedWeb=GAMExampleNotAuthorized

DataSource1=GAM
DataSource2=DEFAULT
```

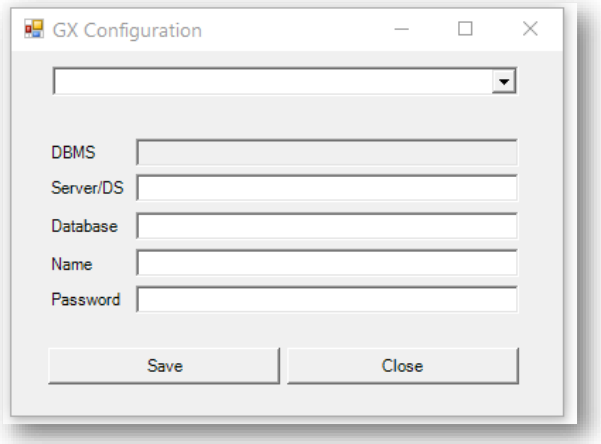
web.config

```
<add key="DataStore1" value="GAM" />
<add key="DataStore2" value="Default" />

<add key="EnableIntegratedSecurity" value="1" />
<add key="IntegratedSecurityLoginWeb"
value="gamexamplelogin,objects" />
<add key="IntegratedSecurityNotAuthorizedWeb"
value="gamexamplenotauthorized,objects" />
```

Outros tipos de arquivos de configuração são o **client.cfg** e **web.config**.
Estes dependem do gerador que está sendo utilizado:
Em aplicações Java, é utilizado client.cfg (onde se aplica para aplicações web e de 2 níveis).
Em aplicações .Net, é utilizado web.config

Para o **client.cfg** e **web.config**, no momento de habilitar GAM em uma base de conhecimento, são definidas neles as linhas que vemos em tela.
Correspondem a estabelecer em 1 a segurança integrada e definir os objetos de exemplo de Login e Não Autorizado.
Além disso, é adicionado um DataSource novo que corresponde ao GAM, deixando em segundo lugar o DEFAULT da aplicação.

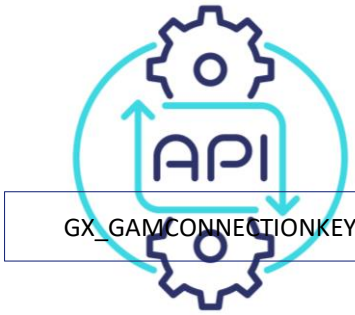



Quanto às ferramentas para estes dois últimos arquivos, temos:

Para o arquivo **web.config**, existe uma tool feita por GeneXus chamada GxConfig, que facilita a criação do arquivo através da interface gráfica que vemos em tela.

Para o arquivo **client.cfg**, existem diferentes maneiras de facilitar a configuração dele que variam de acordo com a versão de GeneXus que está sendo utilizada.

Settings




GX_GAMCONNECTIONKEY

GX_GAMCONNECTIONKEY

connection.gam

Em vários cenários, uma prática comum é ler a informação de configuração das variáveis de ambiente, em vez dos arquivos de configuração.

Desde o upgrade #5 de GeneXus 17, a chave de conexão GAM pode ser especificada através de uma variável de ambiente chamada GX_GAMCONNECTIONKEY.
Isto permite, por exemplo, que ao realizar um deploy para Docker seja possível configurar qual será a Key de conexão que se deseja usar para essa instância do contêiner.

Ao tentar acessar qualquer API do GAM a partir de uma aplicação, a primeira coisa que se faz é verificar se está configurada essa variável de ambiente para obter os dados de conexão do GAM. Se não está, busca-se o arquivo connection.gam para obter esta Key.

Na Wiki de GeneXus é descrito um documento que detalha os cenários em que se utiliza a prática de utilizar variáveis de ambiente, suas vantagens e como fazê-lo.

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