Uso de la API del GAM

Recordemos que las APIs son propiedades y métodos que el GAM disponibiliza para que las aplicaciones que quieran utilizarlo puedan interactuar con él, haciendo posible la comunicación con la base de datos del GAM, que es la que contiene la información de los usuarios, roles, etc.

Vamos a considerar dos nuevos requerimientos específicos para esta aplicación:

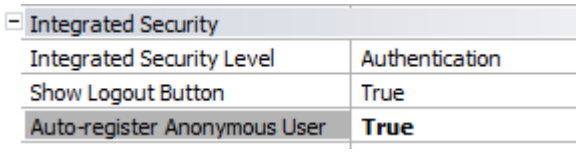
1. Dar la posibilidad de que los asistentes al evento puedan registrarse en cualquier momento, creándose un usuario que les permita, entre otras cosas, poder seleccionar conferencias como favoritas y consultar esas conferencias favoritas desde cualquier dispositivo o computadora con conexión a internet (a través de la aplicación SD o Web).
   Si antes de registrarse habían elegido conferencias favoritas, éstas quedarán asociadas al dispositivo utilizado en el momento de la selección. Pero si a partir de allí la persona decide registrarse, automáticamente las conferencias favoritas del dispositivo serán asociadas al nuevo usuario, sin tener que hacer nada; y..

2. Que sólo para el caso de usuarios autorizados se puedan ver y ejecutar operaciones de CRUD desde el propio dispositivo. Por ejemplo, sobre los Speakers. En definitiva, se deben poder asociar roles especiales a usuarios especiales.

Además, querremos que la aplicación sea segura: es decir, que no se puedan ejecutar los business components expuestos como servicios rest (ni los demás servicios) si no se lo hace a través de la aplicación.

Para conseguir todo esto, prenderemos la autenticación a nivel del objeto main (que así será heredada por todos sus objetos referenciados). Pero para lograr que el usuario no tenga que loguearse necesariamente, habilitaremos la auto-registración como usuario anónimo.

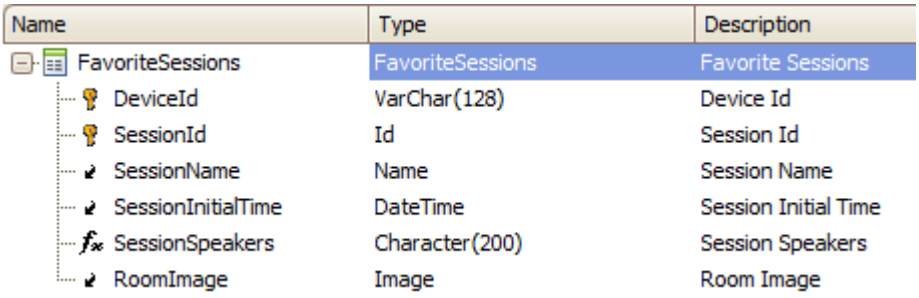
Para eso vamos al dashboard EventDay y en la propiedad **Auto-register Anonymous User** le ponemos el valor **True**.



Esto creará un usuario en la base de datos del GAM que tendrá como identificador, el identificador del dispositivo. Será transparente para el usuario, quien creerá no estar logueado, cuando en verdad lo está con el usuario anónimo. Por tanto, todo lo que se seleccione como favorito de esta forma, quedará asociado a su dispositivo. Para lograr esto, así como que un usuario no anónimo pueda elegir favoritos, debemos modificar la transacción FavoriteSessions.

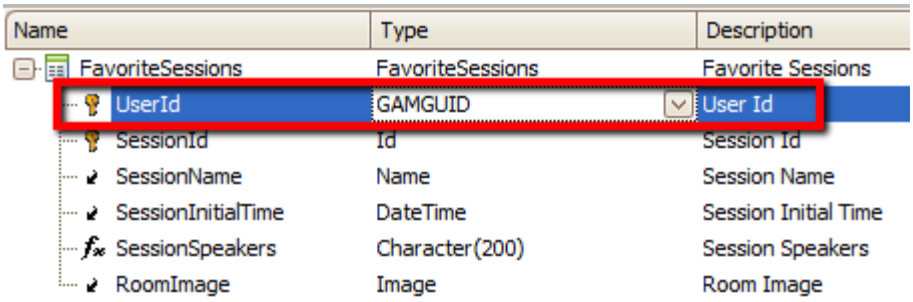
Antes de aplicar el GAM, como no teníamos información de usuarios no tuvimos otra alternativa que realizar el marcado de conferencias favoritas por dispositivo.

Si abrimos la transacción FavoriteSessions, vemos que la clave primaria es compuesta y formada por los atributos DeviceId y SessionId, es decir que la lista de favoritos será dependiente del dispositivo y para diferentes dispositivos la lista será distinta.



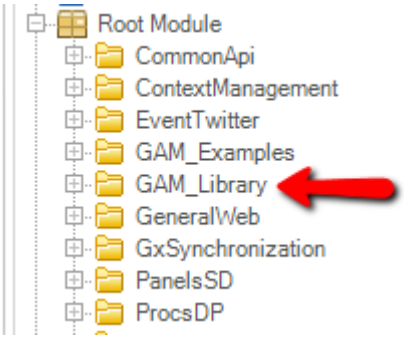
Pero ahora querremos que los favoritos no sean por dispositivo, sino que sean por usuario, de modo de que si el usuario marcó en su dispositivo personal ciertas conferencias, pueda verlas luego si accede a la aplicación web del Evento.

Para cumplir este objetivo, empecemos con modificar la estructura de la transacción FavoriteSessions, sustituyendo al atributo DeviceId por un atributo UserId, que lo definimos del tipo GAMGUID, para poder utilizar el identificador de usuario de GAM.



La idea es que al elegir una conferencia como favorita, se utilice el Id del usuario como parte de la clave que la identifica. Para lograr esto, utilizaremos la API del GAM, para recuperar el identificador del usuario logueado y utilizaremos ese valor para almacenar las sesiones favoritas.

Recordemos que las Api's se encuentran dentro de la carpeta GAMLibrary



A través del objeto externo GAMUser podremos utilizar los métodos disponibles y obtener, entre otras cosas, el identificador del usuario asociado.

A continuación vamos a la solapa Patterns y elegimos el Work With for Smart Devices. Vamos al nodo List y seleccionamos el Grid1. En sus conditions ponemos: UserId = GAMUser.GetId(); para que sólo se muestren las sesiones favoritas que corresponden al usuario logueado y salvamos.

Ahora abrimos el WorkWithDevicesSession y vamos a la Section(General). En los eventos, ubicamos al evento "SessionFavorite". Vemos que la interacción con la base de datos, para saber si una sesión es favorita o para fijarla como tal, la realizan los métodos IsFavoriteSession y SetFavoriteSession respectivamente.

```
Event 'SessionFavorite'
    Composite

        &IsFavoriteSession = IsFavoriteSession(SessionId)
        If &IsFavoriteSession
            ButtonFavorite.Class = 'Button.FavoriteDisabled'
        else
            ButtonFavorite.Class = 'Button.FavoriteEnabled'
        endif

        SetFavoriteSession( SessionId )

        Refresh
    endcomposite

Endevent
```

Abrimos primero el método SetFavoriteSession.

```
&ClientInformationID = ClientInformation.Id
&FavoriteSessions.Load(&ClientInformationID, &SessionID)

if &FavoriteSessions.Fail()
    &FavoriteSessions = new()
    &FavoriteSessions.DeviceId = &ClientInformationID
    &FavoriteSessions.SessionId = &SessionID
    &FavoriteSessions.Save()
else
    &FavoriteSessions.Delete()
endif

commit
```

Vemos que en la primera línea se utiliza la API ClientInformation para recuperar la identificación del dispositivo Smart Devices.

Vamos a sustituir esa línea por una llamada a la API del GAM, para recuperar el identificador del usuario logueado. Escribimos: `&UserId = GAMUser.GetId()`. Luego damos botón derecho sobre la variable &UserId y seleccionamos Add Variable.

Ahora sustituimos donde dice ClientInformationId por UserId… y salvamos.

```
&UserId = GAMUser.GetId()
&FavoriteSessions.Load(&UserId, &SessionID)

if &FavoriteSessions.Fail()
    &FavoriteSessions = new()
    &FavoriteSessions.UserId = &UserId
    &FavoriteSessions.SessionId = &SessionID
    &FavoriteSessions.Save()
else
    &FavoriteSessions.Delete()
endif

commit
```

Luego abrimos el método IsFavoriteSession.

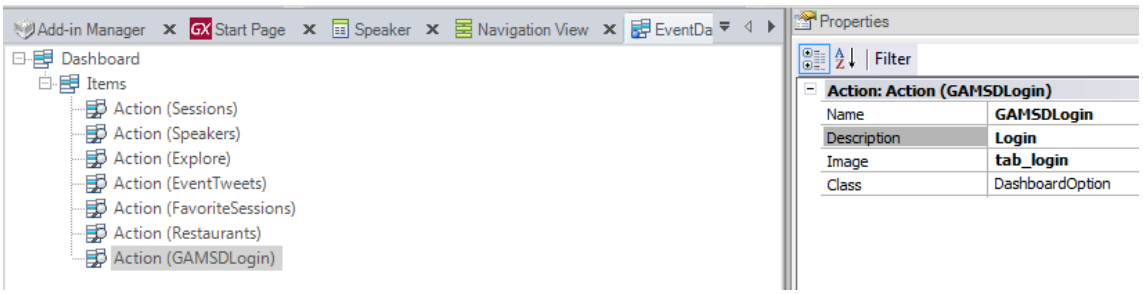
```
&IsFavorite = false
&ClientInformationID = ClientInformation.Id
for each FavoriteSessions
    where DeviceId = &ClientInformationID.Trim()
    &IsFavorite = True
endfor
```

Y cambiamos la línea donde carga el ClientInformationId y la línea del where, poniendo UserId en lugar de ClientInformationId. Salvamos…

```
&IsFavorite = false
&UserId = GAMUser.GetId()
for each FavoriteSessions
    where UserId = &UserId
    &IsFavorite = True
endfor
```

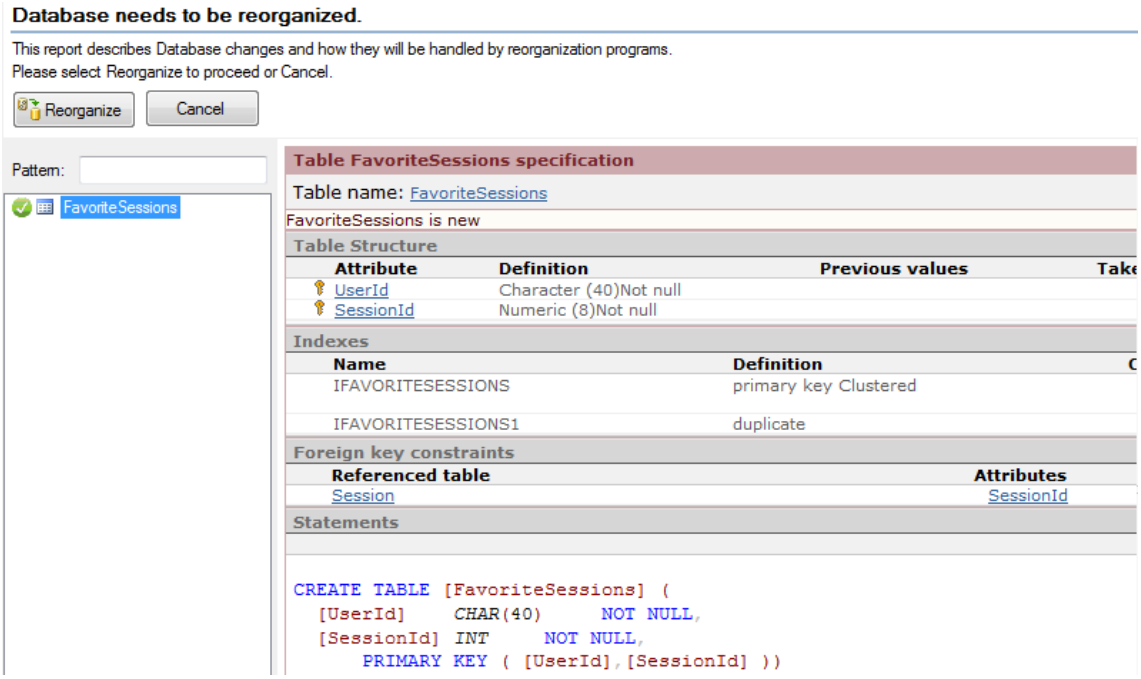
Ahora vamos a agregar un botón en el dashboard, que al presionarlo nos invoque al panel SD de login que el GAM construyó, y que contiene una opción para registrarse.

Abrimos el dashboard EventDay, vamos a Items, damos botón derecho y elegimos Add/Action. Luego elegimos al SD panel GAMSDLogin y en sus propiedades, seleccionamos la imagen tab_login.



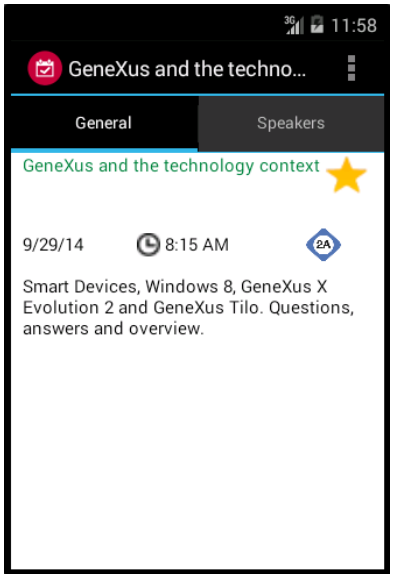
Presionemos F5, para ver esto en funcionamiento.

Vemos que se va a reorganizar la tabla FavoriteSession. Presionamos Reorganize.

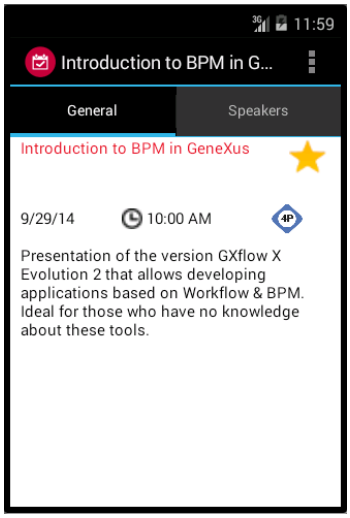


Como no nos hemos logueado, ingresamos a la aplicación con el usuario anónimo.

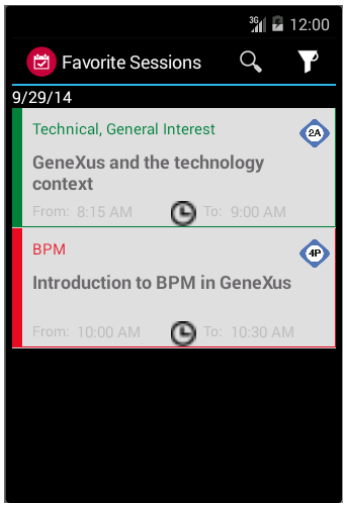
Ahora vamos a Sessions, elegimos la primera conferencia y la marcamos como favorita.



Vamos nuevamente a la lista de conferencias, seleccionamos la segunda y la marcamos también como favorita.



Ahora vamos al dashboard y elegimos Favorites. Vemos que aparecen marcadas como favoritas las dos conferencias que elegimos.



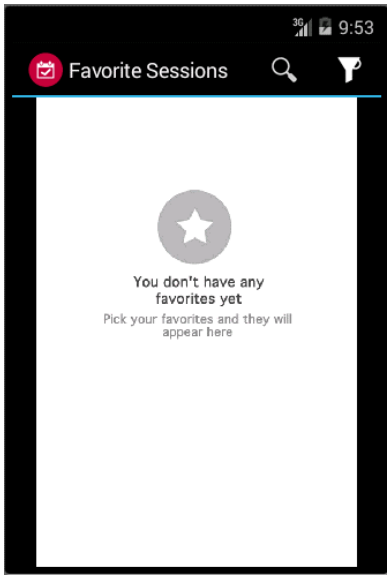
Ahora nuevamente en el dashboard, elegimos Login, presionamos los tres puntos, elegimos Register y nos registramos. Ingresamos el usuario “jsmith”, de nombre “John” y apellido “Smith” y su mail es jsmith@example.com. La contraseña es “jsmith123”, reingresamos “jsmith123” y confirmamos.

Si volvemos a los Favoritos, vemos que están las mismas conferencias, ya que automáticamente se asignaron del usuario anónimo, al usuario “jsmith”.

Volvemos al dashboard, presionamos el botón de Menu y elegimos Logout.

Y nos logueamos con el usuario “admin” y con clave “admin123”.

Si vamos a Favoritos, vemos que el usuario admin no tiene favoritos, ya que ahora las conferencias serán dependientes del usuario que se loguea a la aplicación y no del dispositivo que esté usando para ejecutarla.



Ahora queremos implementar el segundo requerimiento…

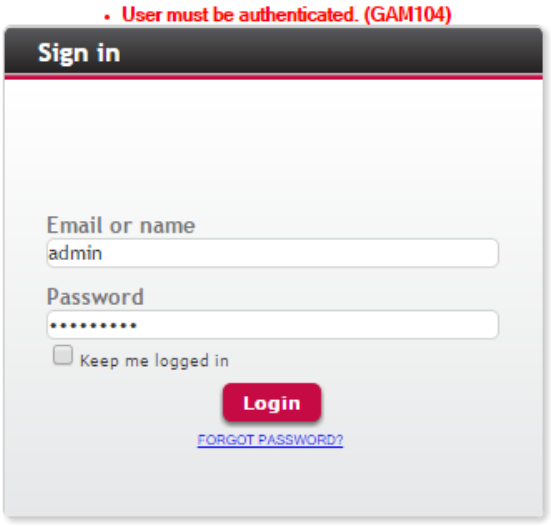
Anteriormente hemos mencionado que nos interesaba diferenciar las operaciones que los usuarios podían realizar en la aplicación EventDay, de acuerdo a su rol, para individualizar aquellas que corresponden al backend móvil.

Un usuario cualquiera podrá ingresar al sistema pero no podrá ver las opciones para insertar, modificar o eliminar oradores. Para poder realizar estas operaciones, deberá loguearse con un perfil de organizador del evento.

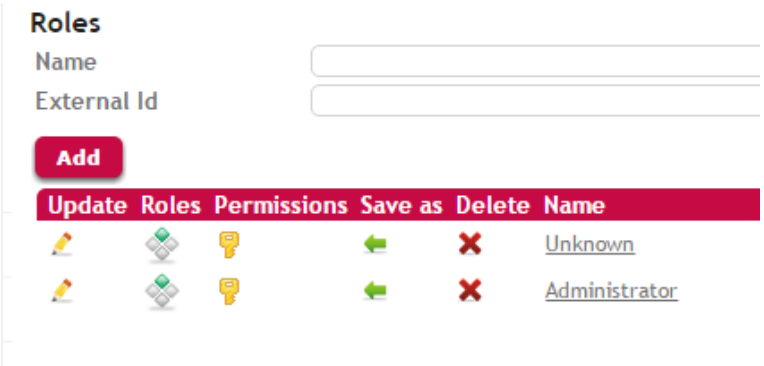
Vamos a comenzar definiendo el rol para organizadores y un usuario que tenga dicho rol.

Para ello, ejecutemos el backend web que nos brinda el GAM.

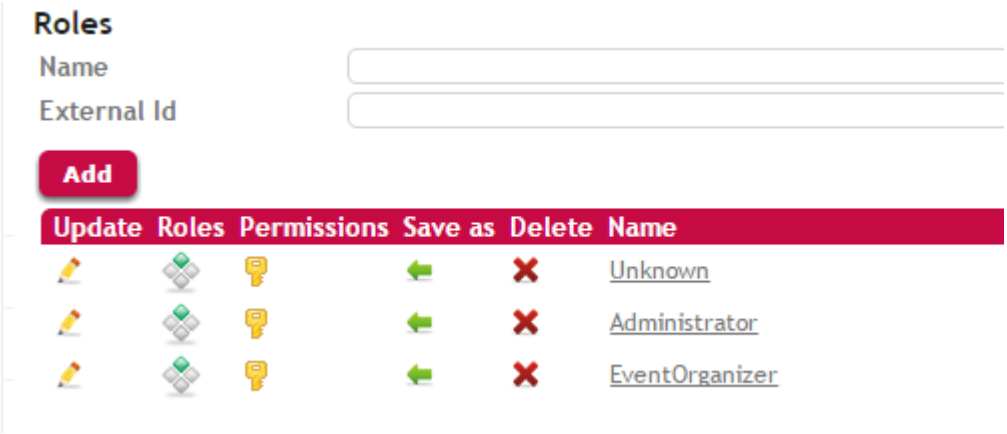
En GeneXus, vamos a View y elegimos Show QR Codes. Expandimos la sección de links y seleccionemos GAMHome. Nos logueamos con el usuario “admin” y password “admin123”.



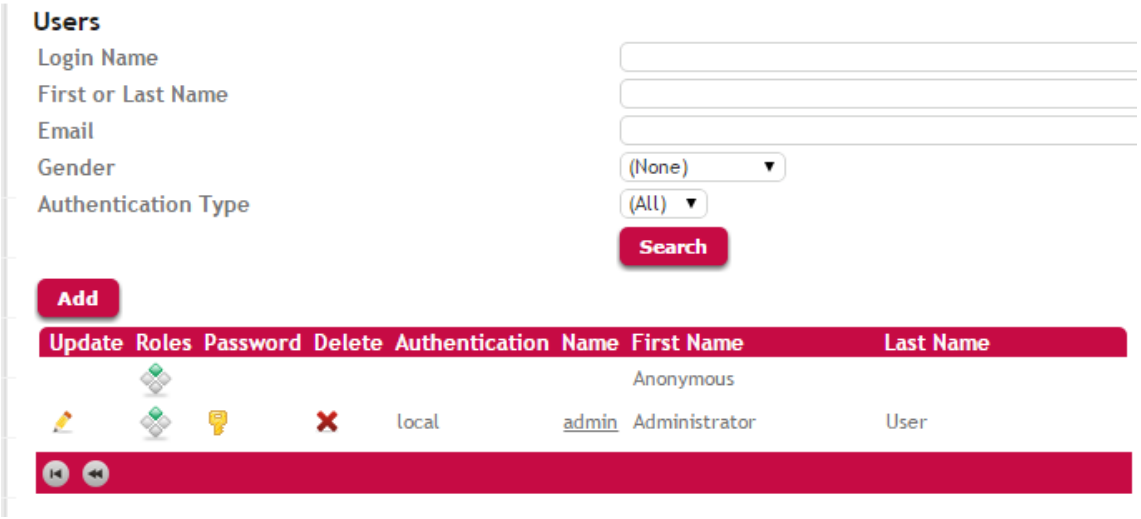
Elegimos la opción “Roles” y vemos que contamos con los roles “Unknown” y “Administrator”, que se han definido automáticamente cuando hemos habilitado el GAM.



Vamos a crear ahora un nuevo rol de nombre EventOrganizer, así que presionamos el botón Add, como nombre usamos EventOrganizer y como descripción “Event Organizer”. Presionamos Confirm.



Pasemos ahora a editar los usuarios. Vamos a la opción Users.

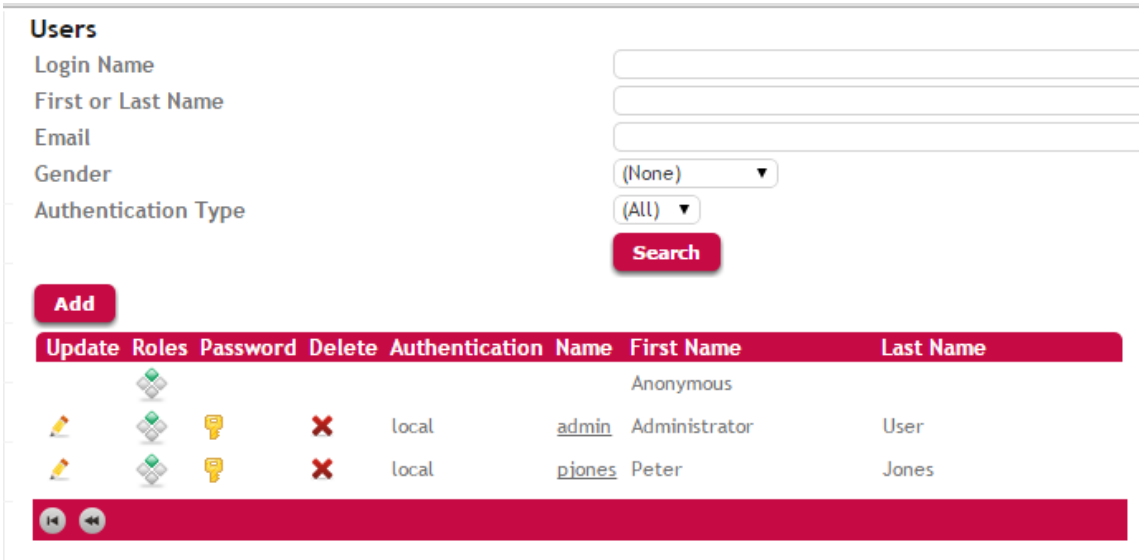


Los usuarios anonymous y admin, han sido creados automáticamente. El usuario admin fue basado en el rol: administrator. Vamos ahora a crear el usuario pjones, con el rol: EventOrganizer

El usuario es pjones, el mail pjones@example.com, el nombre Peter y su apellido Jones.

La password será “pjones123”, y la confirmamos “pjones123”.

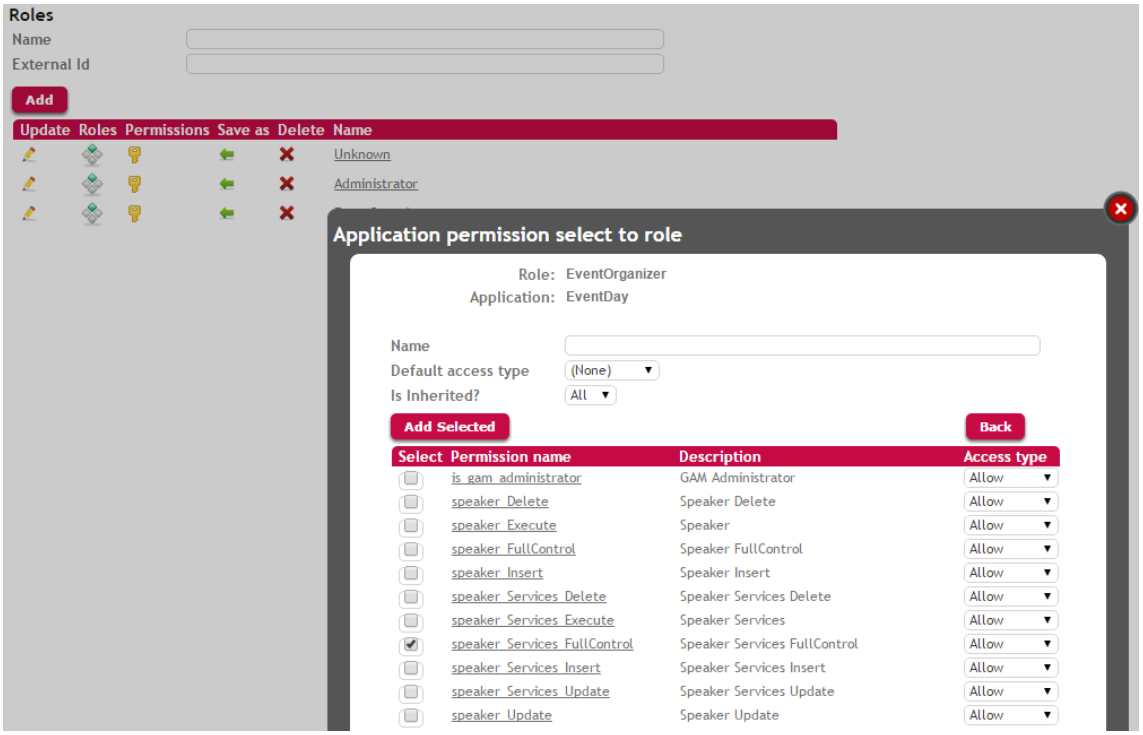
Vamos ahora a asociarle el nuevo rol. Ubicamos la opción Role y elegimos EventOrganizer.



Para asignar permisos sobre ciertas operaciones de Speakers, tenemos que aumentar el nivel de seguridad sobre el Business Component Speaker, para que no sólo requiera autenticación, sino también autorización. Así que vamos a la transacción Speaker y asignamos a la propiedad **Integrated Security Level** el valor **Authorization**.

Vamos al link GAMHome y nos logueamos con el usuario “admin” y password “admin123”.

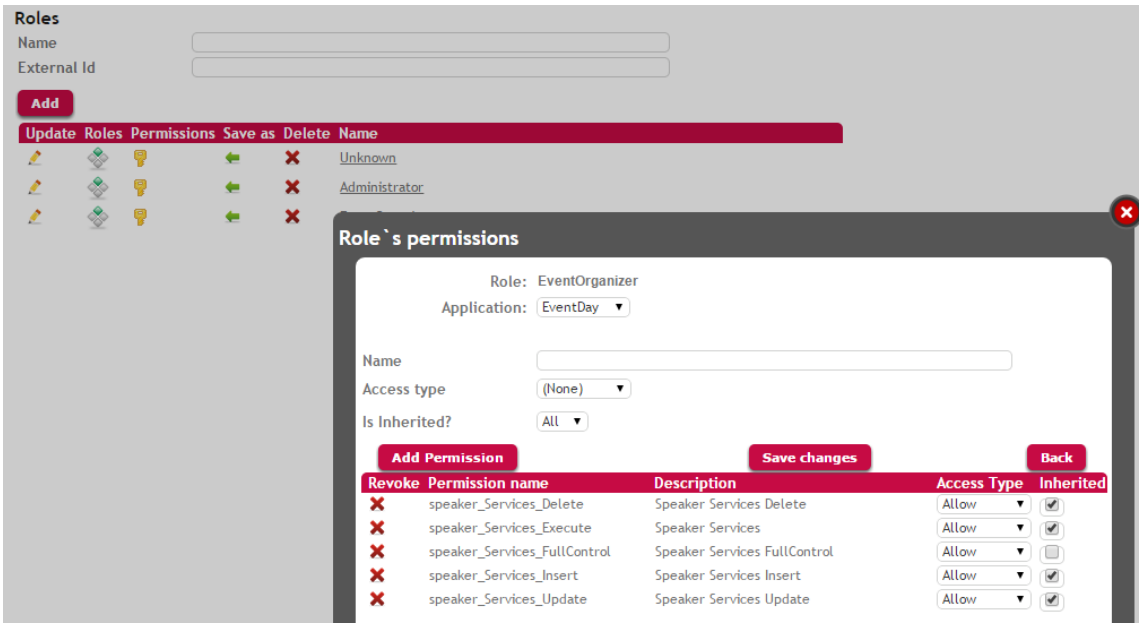
Seleccionamos Roles y en el rol EventOrganizer presionamos Permissions. Seleccionamos la aplicación EventDay (notemos que la aplicación SD tiene el mismo nombre que el dashboard) y luego presionamos Add Permission.



Vemos que los permisos son solamente relacionados al objeto Speakers, porque fue al único objeto que le configuramos el nivel de seguridad en Authorization.

Seleccionamos **speaker_Services_FullControl**.

Esto agrega a este rol, permisos de ejecución, inserción, modificación y eliminación sobre la aplicación.



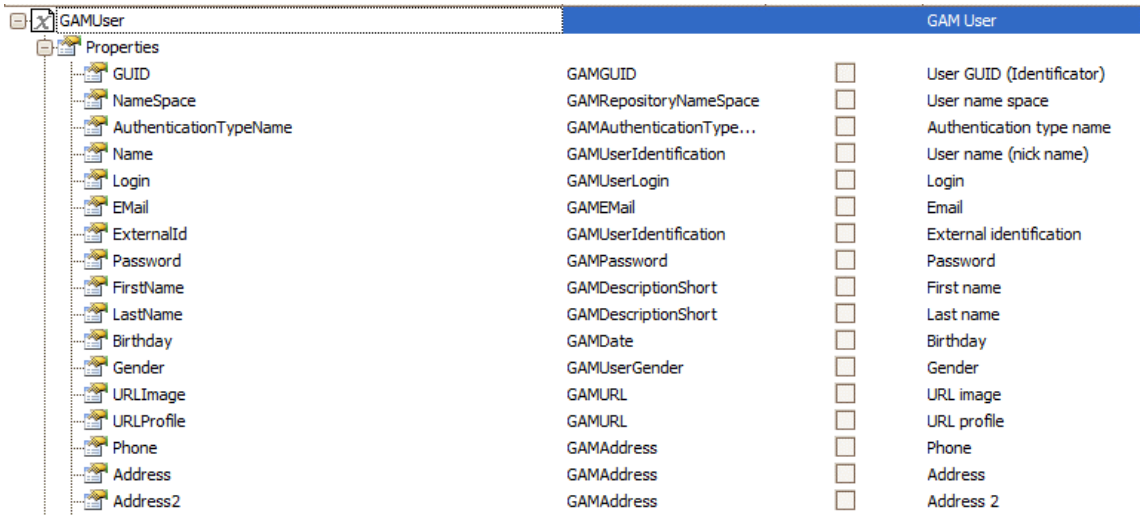
Si queremos que este rol tenga los mismos permisos sobre la aplicación web, hay que hacer lo mismo para esa aplicación.

Ahora abrimos el Work With Speakers.

La propuesta es modificar este objeto para que tenga en cuenta el rol del usuario logueado, de modo que si el rol es “EventOrganizer” mostremos los botones de Insert, Update y Delete y si no lo es, los ocultemos.

Seleccionamos el nodo List y creamos al evento Start. Para identificar el rol utilizaremos la API **GAMuser** del GAM. Entre los métodos del objeto externo GAMUser utilizaremos el que nos devuelve los roles que tiene el usuario asociado.

Vamos a definir una variable de nombre &GAMUser en el objeto Work With Speakers, cuyo tipo de datos GAMUser coincide con este objeto externo, de modo que su estructura coincidirá con las propiedades del objeto externo y tendrá disponibles todos los métodos para trabajar con su información.



Ahora en el evento Start, inicializamos esa variable &GAMUser, con lo devuelto por el método Get() de la API. De esta forma obtenemos toda la información del usuario logueado (su GUID, Name, Password, etc.)

Al definir una variable del tipo de datos del objeto externo, estaremos definiendo una variable con cierta estructura, pero deberemos usar el método Get para que se recupere la instancia del usuario en memoria y acceder así a las propiedades y métodos del usuario que fue creado.

```
Event Start
    &GAMUser = GAMUser.Get()
```

En particular, podremos obtener todos los roles del usuario, a través del método GetAllRoles, el cual devolverá una colección de elementos del tipo GAMRole, que es también un objeto externo. El tipo GAMRole tiene entre sus propiedades una llamada Name, que corresponde al nombre del rol.

Si bien en nuestro caso cada usuario tiene solamente un rol asociado, en el futuro podría tener más, así que vamos a tener que recorrer esa colección de roles para buscar si existe el de nombre “EventOrganizer” que es el que nos interesa.

De modo que vamos a definir una nueva variable:

- &GAMRole: de tipo de dato GAMRole, para recorrer los roles de esa colección.
- Y una &isEventOrganizer, booleana, para saber si existe o no el rol que nos interesa.

Volvamos al evento Start para completar el código.

Escribimos: For, la variable &GAMRole in &Gamuser.GetAllRoles…. y entre paréntesis debemos indicar una variable para guardar los posibles errores que se puedan suceder. Escribimos &Errors y la marcamos como una colección.

A continuación escribimos lo que queremos hacer con cada rol de la colección, que en definitiva será saber si su nombre es "EventOrganizer", en cuyo caso ya no querremos seguir iterando.

Escribimos esto y cerramos con Endfor.

```
Event Start
    &GAMUser = GAMUser.Get()
    for &GAMRole in &GAMUser.GetAllRoles(&GAmError)
        &isEventOrganizer = False
        If &GAMRole.Name = 'EventOrganizer'
            &isEventOrganizer = True
            Exit
        endif
    endfor

Endevent
```

Ahora si el usuario tiene un rol "EventOrganizer" mostraremos el botón de insert y de lo contrario lo ocultaremos.

```
Event Start
    &GAMUser = GAMUser.Get()
    for &GAMRole in &GAMUser.GetAllRoles(&GAmError)
        &isEventOrganizer = False
        If &GAMRole.Name = 'EventOrganizer'
            &isEventOrganizer = True
            Exit
        endif
    endfor
    If &isEventOrganizer
        ButtonInsert.Visible = 1
    else
        ButtonInsert.Visible = 0
    endif
Endevent
```

Ahora seleccionamos el código del evento Start, lo copiamos, vamos a la Section(General) y lo pegamos en la ventana de eventos. También copiamos las variables del nodo List a la Section(General).

Luego modificamos la parte de mostrar y ocultar los botones para hacer referencia a los botones de Update y Delete.

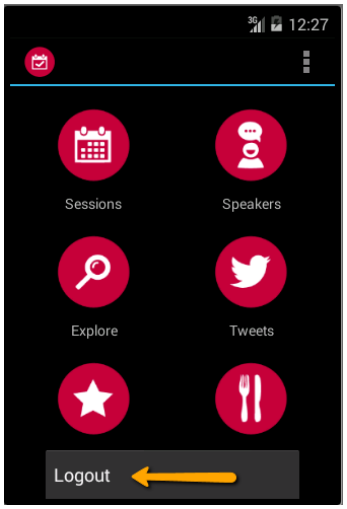
```
Event Start
    &GAMUser = GAMUser.Get()
    for &GAMRole in &GAMUser.GetAllRoles(&GAmError)
        &isEventOrganizer = False
        If &GAMRole.Name = 'EventOrganizer'
            &isEventOrganizer = True
            Exit
        endif
    endfor
    If &isEventOrganizer
        ButtonUpdate.Visible = 1
        ButtonDelete.Visible = 1
    else
        ButtonUpdate.Visible = 0
        ButtonDelete.Visible = 0
    endif
Endevent
```

Veamos lo que acabamos de definir, en ejecución.

Presionemos F5.

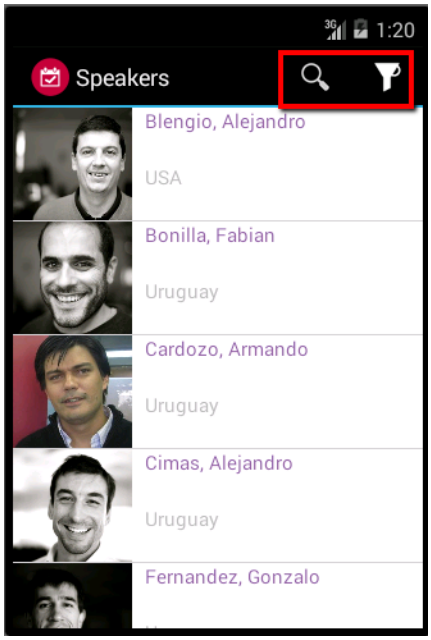
Vamos al emulador de Android y ejecutamos la aplicación.

Ahora nuevamente en el dashboard, nos logueamos con el usuario “jsmith”.

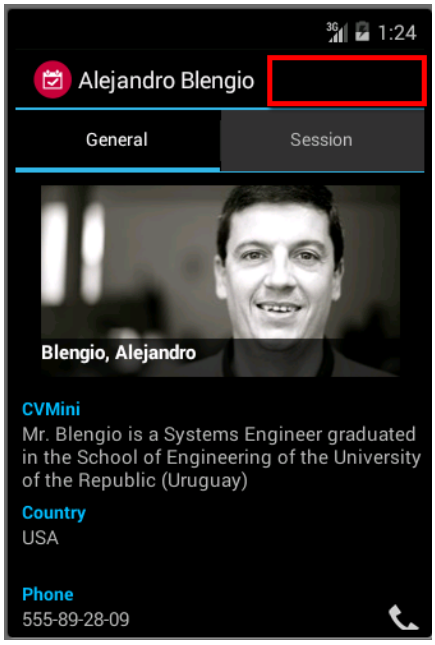


Recordemos que “jsmith” es un usuario común, no es organizador del evento. Ahora en el dashboard, demos tap sobre el icono de Speakers.

Una vez que vemos la lista de oradores, vemos que ahora la opción de Insert no aparece.

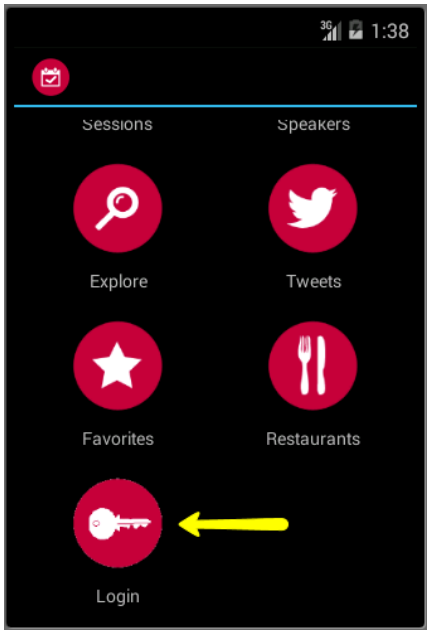


Seleccionemos un orador y vemos que los botones de modificar o de eliminar el orador, tampoco aparecen.

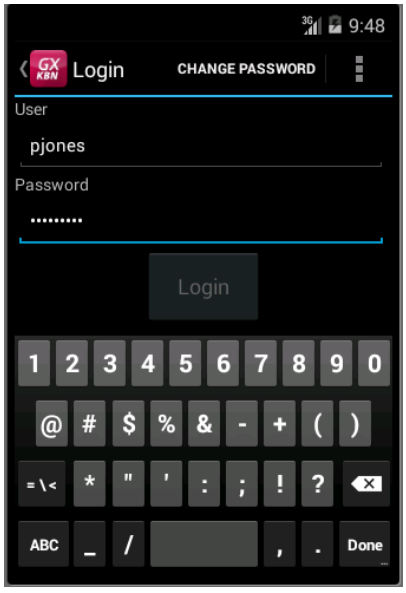


Ahora vamos a ingresar al sistema con un usuario con permisos de EventOrganizer. Vamos al dashboard y nos deslogueamos.

Ahora hacemos tap sobre el icono de Login.

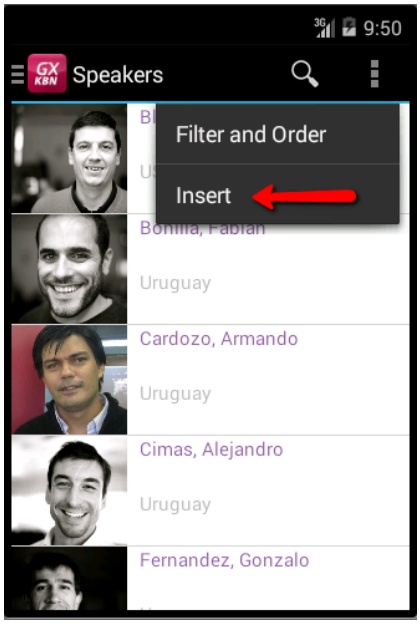


Nos logueamos con el usuario “pjones” y contraseña “pjones123”. Recordemos que al usuario “pjones” le asignamos el rol de “EventOrganizer”.

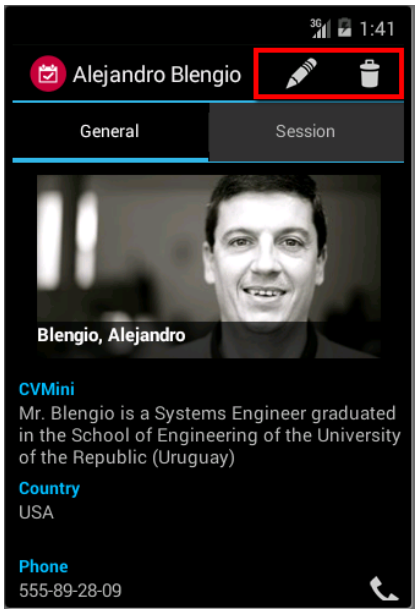


En el dashboard seleccionemos Speakers.

En la pantalla de Work With Speakers, vemos que arriba a la derecha aparecen los tres puntos, damos tap y vemos que ahora aparece la opción para insertar un orador nuevo.



Si seleccionamos un orador, vemos que ahora también podremos modificarlo o eliminarlo.



Con esto, hemos visto la utilidad de acceder a las API's del GAM en tiempo de ejecución para poder acceder a la información del perfil del usuario que se loguea en el sistema y restringir el acceso a determinadas operaciones, en función de su rol.

En este caso hemos modificado por código la interfase, ocultando o mostrando los botones dependiendo de los permisos de usuario, que pudimos obtener gracias a las API's del GAM. Pero además al haber aplicado el GAM, nos hemos asegurado que nadie pueda acceder a los servicios REST en el servidor web, aun conociendo la URL de los mismos, si no se autentica.

Puede encontrar más información de los métodos y propiedades de la API del GAM, en el siguiente link en pantalla.

http://wiki.genexus.com/commwiki/servlet/hwikibypageid?16535