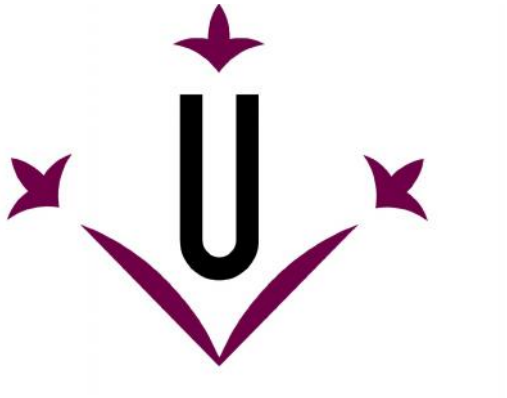

Universitat de Lleida

Escola Politècnica Superior

Máster Universitario en Ingeniería Informática

Auditoria y Certificación de la Calidad

ANEDIN Software Labs S.L.

Super Heroes Agenda

Alumnos:

Anna Blanc

Eduardo Gutiérrez

Indira Lanza

Contenido

Índice de imágenes

1. Descripción general de la aplicación

1.1. Objetivo general

El objetivo de la aplicación es crear una agenda de contactos de SuperHeroes. Para entrar en la aplicación después de dar de alta a un usuario, con nombre y contraseña, se accede a la agenda de SuperHeroes. Así se podrán gestionar los diferentes contactos de la agenda.

1.2. Descarga e instalación

1. Descargar el zip del repositorio master.
2. Descomprimir
3. En un workspace de eclipse, importar el proyecto descomprimido.
4. Compilar y ejecutar el proyecto.
5. En la máquina virtual de eclipse o si se quiere, se puede instalar el apk generado en un dispositivo móvil.

1.3. Pantalla de login y dar de alta un usuario

Nada más ejecutar la aplicación se mostrará la típica pantalla de login en que se comprobará el usuario y la contraseña. Si son correctos se accede a la pantalla principal de la aplicación. En caso contrario se notificará que la contraseña no es la correcta.

Para crear un usuario solo hace falta poner un usuario que no esté guardado y poner la contraseña, y dar al botón Entrar. A continuación si todo ha ido bien, aparecerá una notificación en la barra comunicando que el alta del usuario se ha producido de forma correcta. A partir de ese momento se podrá volver a acceder a la aplicación con el usuario y la contraseña dada de alta.

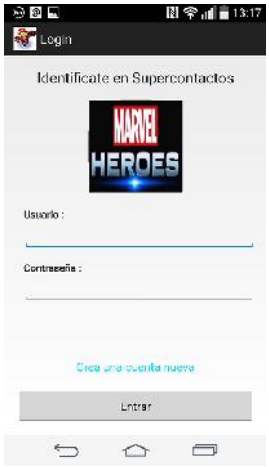

Imagen 1. Pantalla login

1.4. Pantalla principal y gestión de usuarios

En la pantalla principal se muestra un listado con los contactos que existen actualmente en la base de datos con una foto, su nombre y su dirección.

1.5. Dar de alta un contacto nuevo

A través del menú de la aplicación, opción Añadir contacto, se puede dar de alta un contacto. Se mostrará una pantalla para rellenar los distintos campos que aparecen (nombre, dirección, teléfono, mail...). Los campos mail y nombre son obligatorios, el resto son optativos. Una vez concluido se pulsa aceptar. Aparecerá una notificación en la barra comunicando que el contacto se ha dado de alta de forma correcta. Seguidamente se regresará a la página principal con el nuevo contacto incluido ya.

1.6. Eliminar un contacto

También se puede eliminar contactos que no se quieran utilizar más. En la pantalla principal, utilizando el menú contextual sobre el elemento de la lista que se quiera eliminar. Se selecciona eliminar contacto y éste se borrará de la base de datos.

1.7. Visualizar contacto

También se pueden visualizar todos los datos correspondiente a un contacto: de nuevo en el menú contextual que aparece al presionar sobre el elemento que se desee hasta que aparezca el menú contextual, entonces se selecciona la opción Ver contacto y se podrán visualizar todos los datos del contacto. Para volver a la pantalla principal pulsar el botón Back.

Imagen 2. Pantalla principal

1.8. Acerca De

Se puede ver un vídeo descriptivo e información de la aplicación entrando desde la pantalla principal, en el menú de la aplicación, opción Acerca de. Para volver a la pantalla principal pulsar el botón back.

1.9. Salir

Para salir de la aplicación solo se tiene que hacer back desde la pantalla principal.

2. Análisis y Diseño de Requisitos

En el presente apartado se van a definir los requisitos funcionales de la aplicación, actores, diagrama y descripción de los casos de uso.

- Requisitos funcionales
    - Gestionar una agenda de contactos
    - Gestionar contactos
    - Gestionar la seguridad del sistema
- **Actor:** Usuario
- Casos de Uso del sistema
    - Autenticar usuario
    - Registrar usuario
    - Gestionar contactos

2.1. Diagrama de casos del uso del sistema

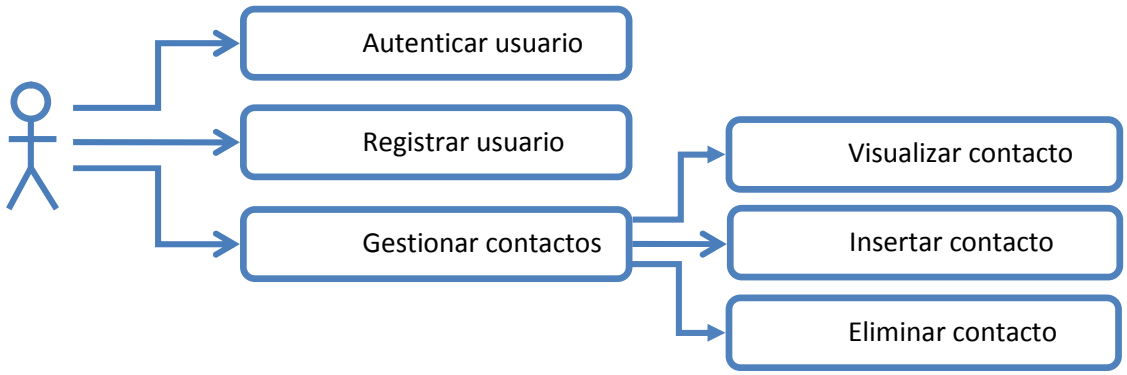

Imagen 3. Diagrama casos de uso

2.2. Descripción de los Casos de Uso del Sistema

2.2.1. Caso de Uso Autenticar Usuario

Caso de Uso	Autenticar usuario
Actores	Usuario
Tipo	Esencial
Precondición	Estar registrado en la base de datos
Postcondición	El usuario logra acceder al sistema
Propósito	Autenticarse en la aplicación
Resumen	
El usuario se autentica en la aplicación, para ello debe ingresar su nombre de usuario y contraseña. Una vez autenticado accede a las funciones de la aplicación y se muestra una notificación sobre el éxito de la acción. En caso de introducir una contraseña errónea se muestra una notificación. En caso de no existir el usuario que se autentica el sistema lo registra automáticamente.	
Caso de Uso Normal	
1. El usuario para autenticarse ingresa los datos correspondientes a nombre y contraseña, y hace clic en el botón aceptar. 2. El usuario accede al sistema	

Cursos Alternos
• El usuario no existe en la base de datos. o Una vez comprobados los datos de acceso, si el usuario no se encuentra en la base de datos la aplicación lo registra de forma automática (Caso de Uso Registrar Usuario) y accede.
• Si el usuario existe y la contraseña es incorrecta o Se muestra un mensaje de alerta para que el usuario rectifique la contraseña.

Importancia	Alta

2.2.2. Caso de Uso Registrar Usuario

Caso de Uso	Registrar Usuario
Actores	Usuario
Tipo	Esencial
Precondición	No estar registrado en la base de datos
Postcondición	Se registra un usuario en el sistema
Propósito	Insertar un usuario en el sistema

Resumen
El usuario comienza el proceso de registro, insertando los campos correspondientes a nombre y contraseña. El sistema notifica el éxito o no de la acción.
Caso de Uso Normal
1. El usuario selecciona la opción de registro . 2. El usuario inserta su nombre y contraseña, y hace clic en el botón aceptar. 3. Se registra el usuario en la base de datos y se muestra notificación sobre el éxito de la acción.

Importancia	Alta

2.2.3. Caso de Uso Gestionar Contactos

El Caso de Uso Gestionar Contactos se divide en tres casos de usos:

- Insertar Contacto
- Eliminar Contacto
- Visualizar Contacto

Insertar Contacto

Caso de Uso	Insertar Contacto
Actores	Usuario
Tipo	Esencial
Precondición	El contacto no existe en la base de datos
Postcondición	Se inserta un nuevo contacto
Propósito	Registrar un nuevo contacto
Resumen	
El usuario registra un nuevo contacto en la agenda, para ello introduce los campos: nombre y apellidos, teléfono, email, dirección postal, sexo, referencia y redes sociales. El caso de uso concluye una vez selecciona el botón aceptar. El sistema muestra una notificación sobre el éxito de la acción.	
Caso de Uso Normal	
1. El usuario selecciona la opción de registro. 2. El usuario inserta su nombre y contraseña, y hace clic en el botón aceptar. 3. Se registra el usuario en la base de datos y se muestra notificación del éxito de la acción.	
Cursos Alternos	
• Si el actor decide no insertar el contacto puede cancelar la acción y el sistema muestra la pantalla anterior.	
Importancia	Alta

Eliminar Contacto

Caso de Uso	Eliminar Contacto
Actores	Usuario
Tipo	Esencial
Precondición	Existir al menos un contacto en la base datos
Postcondición	Se elimina un contacto
Propósito	Eliminar un contacto
Resumen	
El usuario accede a la lista de contactos, selecciona un contacto y haciendo clic en el botón “Eliminar” lo elimina.	
Caso de Uso Normal	
1. El usuario accede a la lista de contactos. 2. El usuario selecciona el contacto a eliminar manteniendo pulsada la pantalla del móvil. 3. El usuario selecciona la opción eliminar. 4. Se muestra la notificación de contacto eliminado.	
Cursos Alternos	
• El usuario cancela la acción.	
Importancia	Media

Visualizar Contacto

Caso de Uso	Visualizar Contacto
Actores	Usuario
Tipo	Esencial
Precondición	Existir al menos un contacto en la base datos
Postcondición	Se muestra un contacto
Propósito	Visualizar un contacto
Resumen	
El usuario accede a la lista de contactos, selecciona un contacto y visualiza sus datos en pantalla.	
Caso de Uso Normal	
1. El usuario accede a la lista de contactos. 2. El usuario selecciona el contacto a visualizar manteniendo pulsada la pantalla del móvil. 3. El usuario selecciona la opción “Visualizar”. 4. Se muestra la información del contacto	
Cursos Alternos	
• El usuario cancela la acción.	
Importancia	Media

3. Objetos del proyecto

Siendo que el proyecto ha sido desarrollado en eclipse para plataformas Android, los objetos java se encuentran en la carpeta SRC/COM/EXAMPLE/INFORMACION.

3.1. Jerarquía de los objetos

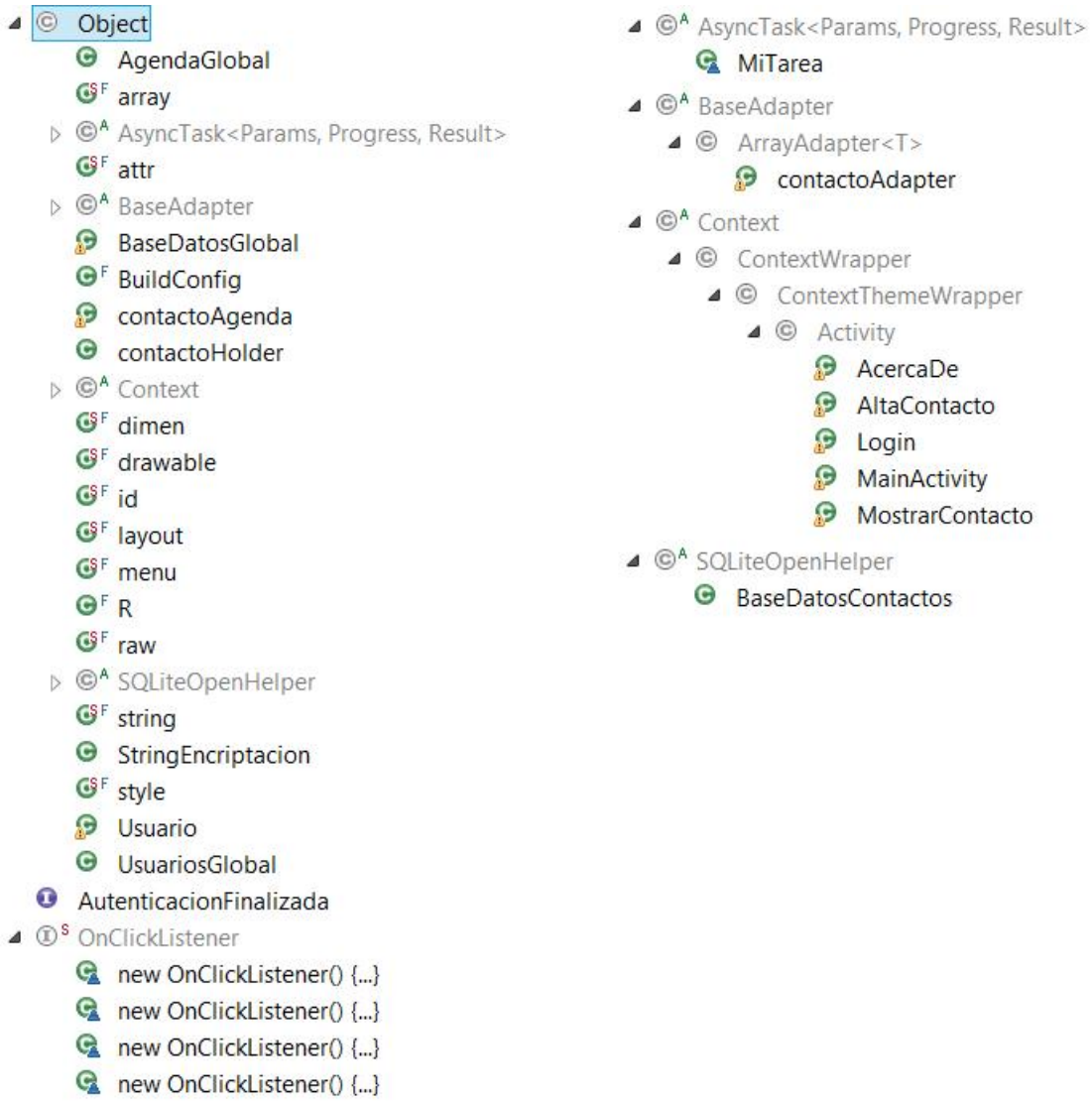

Imagen 4. Jerarquía de objetos

3.2. Detalle de los objetos del proyecto

A continuación se detallan los objetos java existentes.

AcercaDe

- **Descripción**: Muestra una pantalla con el autor, fecha y un vídeo de presentación. Se accede a ésta a través del menú, opción Acerca de.
- **Tipo**: Clase. Extiende la clase Activity.
- Declaraciones:
  - VideoView intro
- Operaciones:
  - onCreate(Bundle): void

AgendaGlobal

- **Descripción**: Clase usada para declarar un arraylist global de contactoAgenda, para poder así acceder desde cualquier otra.
- Tipo: Clase
- Declaraciones:
  - ArrayList<contactoAgenda> miAgenda
  - AgendaGlobal *instance*
- Operaciones:
  - getInstance(): AgendaGlobal
  - AgendaGlobal()

AltaContacto

- **Descripción**: Formulario para dar de alta contactos. Se accede a ésta a través del menú, opción Añadir contacto.

  Cada contacto consta de:

- Imagen asociada (una por defecto y una a elegir de un set de imágenes).
- Campos de texto (dirección, teléfono, mail, nombre).
- Spinner (tipo contacto familiar, laboral, amigo).
- Radio button: Hombre o mujer.
- Check buttons: (miembro google, Facebook, twitter, LinkedIn).

Cada contacto tiene que tener como mínimo un mail y nombre.

- **Tipo:** Clase. Extiende la clase Activity.
- Declaraciones:
  - contactoAgenda contactoActual
  - EditText edtxtNombre
  - EditText edtxtDireccion
  - EditText edtxtMail
  - EditText edtxtTelefono
  - ImageView imgContacto
  - Spinner tipoContacto
  - RadioButton sexo
  - NotificationCompat.Builder notification
  - PendingIntent pIntent
  - NotificationManager manager
  - Intent resultIntent
  - TaskStackBuilder stackBuilder
- Operaciones:
  - onCreate(Bundle): void
  - onCreateOptionsMenu(Menu): boolean
  - onOptionsItemSelected(MenuItem): boolean
  - onCheckboxClicked(View): void
  - OnRadioButtonClick(View): void
  - OnClickAceptar (View): void
  - comprobarContacto(contactoAgenda): boolean

- OnClickCancelar (View): void
- onCreateContextMenu(ContextMenu, View, ContextMenuInfo): void
- onContextItemSelected(MenuItem): boolean

AutenticacionFinalizada

- **Descripción**: Se utilitza una tarea asíncrona para volver al formulario que lo ha llamado.
- **Tipo:** Interfície.
- **Declaraciones**: onAcabeAutenticacion(Integer): void

BaseDatosContactos

- **Descripción**: Base de datos con la tabla contactos donde se almacenan todos los datos de cada contacto (nombre, dirección, teléfono, etc.).
- **Tipo:** Clase. Extiende la clase SQLiteOpenHelper.
- Declaraciones:
  - int VERSION_BASEDATOS
  - String NOMBRE_BASEDATOS
  - String TABLA = "contactos";
  - String CREAR_TABLA
- Operaciones:
  - borrarContacto(contactoAgenda): void
  - borrarContacto(String): void
  - insertarContacto(contactoAgenda): void
  - insertarContacto(SQLiteDatabase, String, String, String, String, int, int, int, int, int, String, int): void
  - insertarContacto(String, String, String, String,int, int, int, int, int sexo, String, int): void
  - modificarContacto(contactoAgenda): void
  - modificarContacto (String, String, String, String, int, int, int, int, int, String, int)
  - onCreate(SQLiteDatabase): void
  - onUpgrade(SQLiteDatabase, int, ing): void

- recuperarContacto(String): contactoAgenda
- recuperarTodosContactos(): ArrayList<contactoAgenda>

BaseDatosGlobal

- **Descripción**: Se crea un acceso global para el acceso a la base de datos.
- **Tipo:** Clase.
- Declaraciones:
  - BaseDatosContactos agendaBaseDatos
  - Context mCon
  - BaseDatosGlobal instance
- Operaciones:
  - getInstance(): BaseDatosGlobal
  - getInstance(Context): BaseDatosGlobal
  - BaseDatosGlobal()
  - BaseDatosGlobal(Global)

Login

- **Descripción**: Pantalla inicial donde los usuarios deben identificarse correctamente para poder acceder a la aplicación.

  El usuario debe poner su usuario y contraseña. Valida el usuario.
- **Tipo:** Clase. Extiende la clase Activity.
- Declaraciones:
  - String OPERACIÓN
  - String AlgoritmoEncriptacion
  - String PREFS_NAME

- EditText entradaUsuario
- EditText entradaPassword
- Button botonLogin
- TextView Mensaje
- TextView cuentaNueva
- Usuario usuarioaValidar
- NotificationCompat.Builder notification
- PendingIntent pIntent
- NotificationManager manager
- Intent resultIntent
- TaskStackBuilder stackBuilder

- Operaciones:
  - onCreate(Bundle): void
  - ClickCuentaNueva(View): void
  - ClicLogin (View): void

MiTarea

Implementada dentro de la clase Login.

- Descripción: Tarea asíncrona para realizar el login y validar el usuario y contraseña.
- **Tipo:** Clase. Extiende la clase AsyncTask.
- Declaraciones:
  - ProgressDialog mProgress
  - Usuario actual

- Operaciones:
    - onPreExecute(): void
    - doInBackground(Usuario...): Integer
    - onPostExecute(Integer): void

MainActivity

- **Descripción**: Pantalla principal.

  Se muestra una lista con los contactos existentes en la base de datos.

  Para cada uno de los contactos mostrados, existe la opción de consultarlo o eliminarlo.

- **Tipo:** Clase. Extiende la clase Activity. Implementa OnItemClickListener y OnItemLongClickListener.

- Declaraciones:
    - String EXTRA_MAIL
    - String EXTRA_POSITION
    - ListView lvContactos
    - contactoAdapter adapter
    - int PosicionActual

- Operaciones:
    - onCreate(Bundle): void
    - onCreateOptionsMenu(Menu): boolean
    - onOptionsItemSelected(MenuItem): boolean
    - salir(): void
    - onItemClick(AdapterView<?>, View, int, long) : void
    - onItemLongClick(AdapterView<?>, View, int, long): boolean

- onCreateContextMenu(ContextMenu, View, ContextMenuInfo) : void
- onContextItemSelected(MenuItem): boolean
- EliminarContacto(int) : void
- onAcabeInicializacion(): void
- onDestroy(): void
- void onStart(): void
- onRestart(): void

MostrarContacto

- **Descripción**: Pantalla que muestra todas las características del contacto.

Se accede a esta pantalla, a través del menú contextual existente en el listado de contactos de la pantalla MainActivity.

El listview de la pantalla principal le pasa la clave del contacto que se debe mostrar. Se accederá a la base de datos, para obtener todos los datos y se muestran por pantalla.

- **Tipo:** Clase. Extiende la clase Activity.
- Declaraciones:
  - String EXTRA_MAIL
  - String EXTRA_POSITION
  - contactoAgenda contactoActual
  - TextView edtxtNombre
  - TextView edtxtDireccion
  - TextView edtxtMail
  - TextView edtxtTelefono
  - Button botonCancelar

- Button botonAceptar
- ImageView imgContacto
- Spinner tipocontacto
- CheckBox chkFacebook
- CheckBox chkTwitter
- CheckBox chkGoogle
- CheckBox chkLinkedin
- RadioGroup sexo

- Operaciones:
  - onCreate(Bundle): void

StringEncriptacion

- **Descripción**: Clase usada para encriptar el usuario y la contraseña, y guardarlo en SharedPreferences.
- **Tipo:** Clase.
- Declaraciones:
  - String MD2
  - String MD5
  - String SHA1
  - String SHA256
  - String SHA384
  - String SHA512
- Operaciones:
  - String getStringMessageDigest(String, String): String
  - toHexadecimal(byte[]): String

Usuario

- **Descripción**: Clase usada para gestionar los usuarios de la aplicación.
- **Tipo:** Clase. Implementa la interface Parceable.
- Declaraciones:
    - String nombre
    - String contrasenya
    - String mail
    - int algoritmoEncriptacion
- Operaciones:
    - Usuario()
    - Usuario(String, String, String)
    - describeContents(): int
    - getNombre(): String
    - setNombre(String) : void
    - getContrasenya() : String
    - setContrasenya(String): void
    - getMail(): String
    - setMail(String): void
    - writeToParcel(Parcel, int): void

UsuariosGlobal

- **Descripción**: Clase global para poder acceder a todos los usuarios. Se usa un hash map donde la clave es la contraseña y cada elemento un usuario.
- **Tipo:** Clase.

- Declaraciones:
    - HashMap< String, Usuario> misUsuarios
    - UsuariosGlobal instance
- Operaciones:
    - getInstance(): UsuariosGlobal
    - UsuariosGlobal()

contactoAdapter

- **Descripción**: Clase necesaria para poder visualizar los contactos dentro del listview de la pantalla principal.
- **Tipo**: Clase. Extiende ArrayAdapter<contactoAgenda>
- Declaraciones:
    - Context context
    - ArrayList<contactoAgenda> datos
- Operaciones:
    - contactoAdapter(Context, ArrayList<contactoAgenda>)
    - getView(int, View, ViewGroup): View

contactoAgenda

- **Descripción**: Clase que describe las características de los contactos.
- **Tipo:** Clase. Implementa la interficie Serializable.
- Declaraciones:
    - String Nombre
    - String Direccion
    - String Telefono
    - String mail

- int miembroFacebook
- int miembroTwitter
- int miembroGoogle
- int miembroLinnkedin
- int sexo
- String tipoContacto
- int drawableImageID

- Operaciones:
  - contactoAgenda()
  - contactoAgenda(String, String, String, String)
  - contactoAgenda(String, String, String, String, int, int, int, int, int, String, int)
  - getDrawableImageID(): int
  - setDrawableImageID(int): void
  - getNombre(): String
  - setNombre(String) : void
  - getDireccion(): String
  - setDireccion(String): void
  - getTelefono(): String
  - setTelefono(String) : void
  - getMail(): String
  - setMail(String): void
  - isMiembroFacebook(): int
  - setMiembroFacebook(int) : void
  - isMiembroTwitter(): int
  - setMiembroTwitter(int) : void
  - isMiembroGoogle(): int
  - setMiembroGoogle(int): void
  - public int isMiembroLinnkedin(): int
  - setMiembroLinnkedin(int): void
  - isSexo(): int
  - setSexo(int): void
  - getTipoContacto() : String
  - toString(): String
  - setTipoContacto(String): void

contactoHolder

- **Descripción**: Almacena las referencias a los controles de la vista personalizada del ListView.
- Tipo: Clase
- Declaraciones:
    - ImageView imgContacto
    - TextView tvNombre
    - TextView tvMail