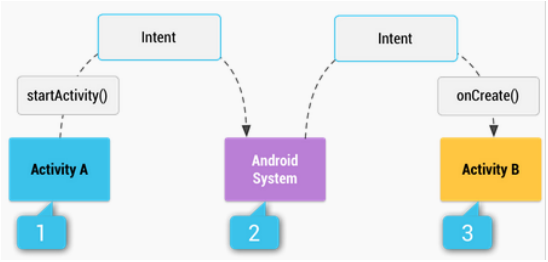
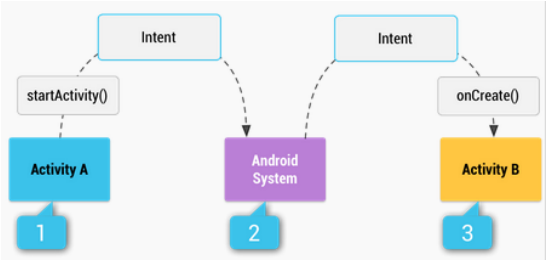


Interactuando con otras aplicaciones

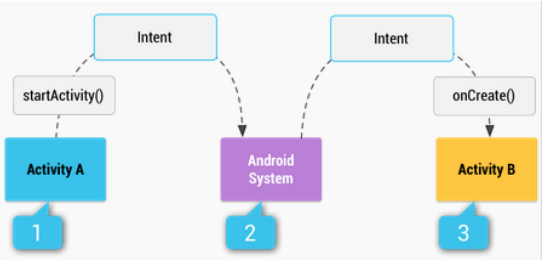
Intents implícitos

Tipos de Intent



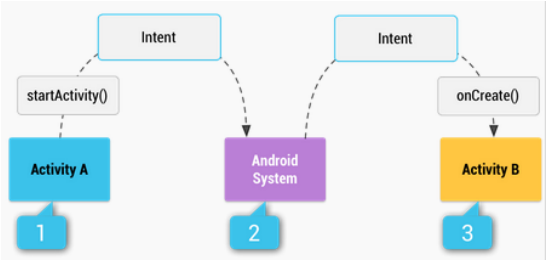
- Existen dos tipos de *intent*:
  - Explícitos: llaman a componentes de la propia aplicación (unidad 23_Comunicando_actividades).
  - **Implícitos**: llaman a componentes ajenos a la propia aplicación (actividades de otras aplicaciones, servicios del sistema,…).
- Si el componente llamado es una actividad, ambos tipos la llaman usando los métodos startActivity() o startActivityForResult() ya estudiados.

Intent implícitos



- Llaman a componentes ajenos a la aplicación en curso (otras aplicaciones, servicios del sistema,…) para realizar una ACCIÓN. Además, a veces, es necesario enviar DATOS para que dicha acción se pueda realizar:
  - por ejemplo, si nuestro Intent "llama" a un navegador (la acción) deberá enviar una URL (el dato)
  - en cambio, si nuestro Intent "llama" a la cámara de fotos (la acción) no es necesario enviar ningún dato
- Las acciones suelen solicitar tareas abstractas ("quiero hacer una foto" o "quiero enviar un mensaje", etc.) que se resolverán reutilizando componentes de aplicaciones sin tener que especificar una en especial.

Intent filter



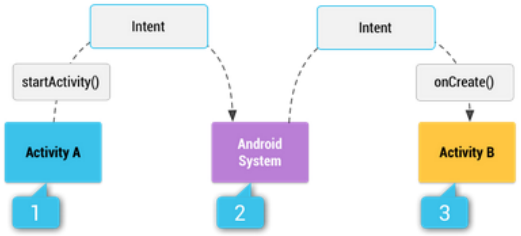
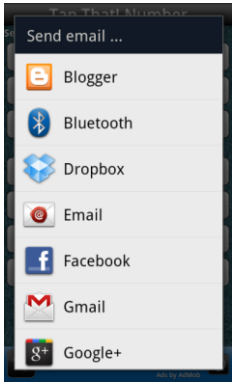
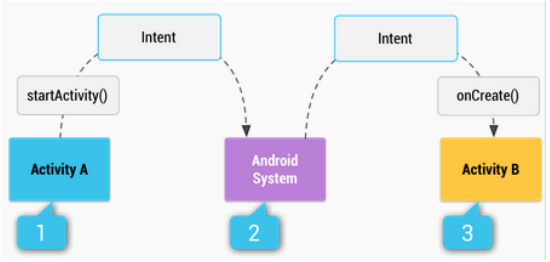


Figure 1. Illustration of how an implicit intent is delivered through the system to start another activity: **[1]** *Activity A* creates an `Intent` with an action description and passes it to `startActivity()`. **[2]** The Android System searches all apps for an intent filter that matches the intent. When a match is found, **[3]** the system starts the matching activity (*Activity B*) by invoking its `onCreate()` method and passing it the `Intent`.



- Las intenciones implícitas se resuelven en tiempo de ejecución, de forma que el sistema mirará cuantas aplicaciones tienen la posibilidad de ejecutar ese tipo de acción y elegirá la adecuada para dar respuesta.
- ¿Cómo encuentra el sistema? Comparando el contenido del Intent creado con los "intent filters" declarados en el archivo de manifiesto de las aplicaciones instaladas en el dispositivo. Si hay alguna coincidencia, el sistema inicia ese componente encontrado.
- Si encuentra varias, el sistema puede preguntar al usuario a través de qué aplicación quiere lanzar la intención. Por ejemplo, cuando pedimos enviar un email, muestra la lista de aplicaciones instaladas válidas para esa tarea.

Partes de un Intent implícito



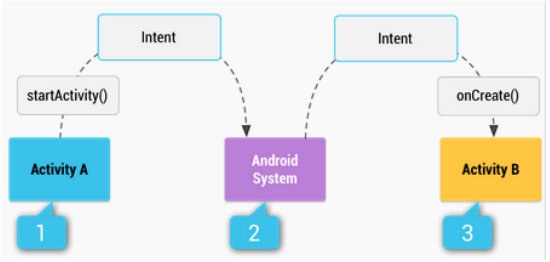
- Esenciales:
  - Acción
  - Datos

```
Intent i = new Intent(Intent.ACTION_CALL, Uri.parse("tel:976491015"));
```

- Secundarias:
  - Tipo de datos
  - Categoría
  - Extras

```
Intent i= new Intent(Intent.ACTION_SEND);
i.setType("text/plain");
i.putExtra(Intent.EXTRA_SUBJECT, "esto es el asunto");
i.putExtra(Intent.EXTRA_TEXT, "esto es el texto del correo");
i.putExtra(Intent.EXTRA_EMAIL, new String[] { "bsoler@iespabloserrano.com"});
```

Acción



- Es una **cadena de caracteres** donde indicamos la acción a realizar (ver, editar, llamar,…).

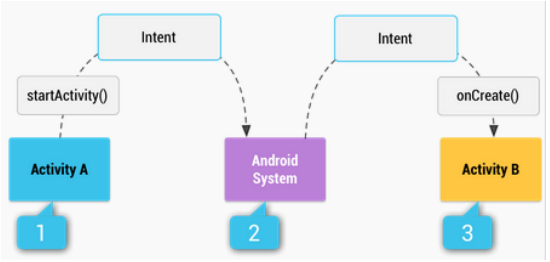
```
Intent i= new Intent();
i.setAction(Intent.ACTION_SEND);
```

=

```
Intent i= new Intent(Intent.ACTION_SEND);
```

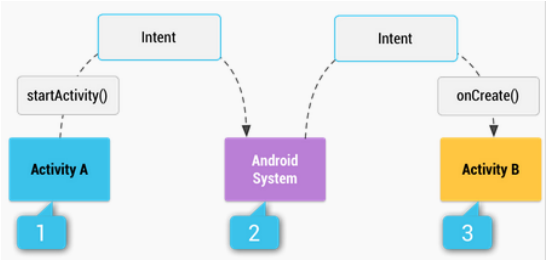
- La clase Intent define una serie de constantes para acciones genéricas.
- No obstante, además de estas podemos definir acciones propias.

Ejemplos de acciones genéricas



Constante	Componente Objetivo	Action (Acción)
ACTION_CALL	actividad	Iniciar una llamada de teléfono.
ACTION_EDIT	actividad	Mostrar datos para que el usuario los edite.
ACTION_MAIN	actividad	Se ejecuta como la actividad inicial de una tarea, sin input de datos ni output.
ACTION_SYNC	actividad	Sincronización de datos entre servidor y el dispositivo móvil.
ACTION_BATTERY_LOW	broadcast receiver	Un aviso de que la batería está baja.
ACTION_HEADSET_PLUG	broadcast receiver	Unos auriculares han sido insertados o sacados del dispostivo.
ACTION_SCREEN_ON	broadcast receiver	La pantalla ha sido encendida.
ACTION_TIMEZONE_CHANGED	broadcast receiver	La configuración del huso horario ha sido cambiada.

Constantes más utilizadas



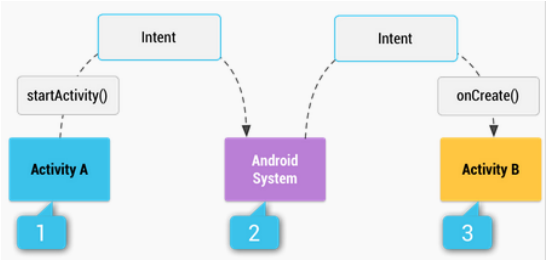
- ACTION_MAIN
- ACTION_VIEW
- ACTION_ATTACH_DATA
- ACTION_EDIT
- ACTION_PICK
- ACTION_CHOOSER
- ACTION_GET_CONTENT
- ACTION_DIAL
- ACTION_CALL
- ACTION_SEND
- ACTION_SENDTO
- ACTION_ANSWER
- ACTION_INSERT
- ACTION_DELETE
- ACTION_RUN
- ACTION_SYNC
- ACTION_PICK_ACTIVITY
- ACTION_SEARCH
- ACTION_WEB_SEARCH
- ACTION_FACTORY_TEST

Acciones para lanzar actividades

Acciones para lanzar broadcast

- ACTION_TIME_TICK
- ACTION_TIME_CHANGED
- ACTION_TIMEZONE_CHANGED
- ACTION_BOOT_COMPLETED
- ACTION_PACKAGE_ADDED
- ACTION_PACKAGE_CHANGED
- ACTION_PACKAGE_REMOVED
- ACTION_PACKAGE_RESTARTED
- ACTION_PACKAGE_DATA_CLEARED
- ACTION_UID_REMOVED
- ACTION_BATTERY_CHANGED
- ACTION_POWER_CONNECTED
- ACTION_POWER_DISCONNECTED
- ACTION_SHUTDOWN

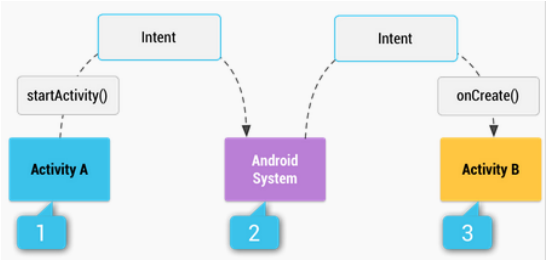
Datos



- Son con lo que trabajará el componente llamado.
- Hay que expresar estos datos por medios de URI. Ejemplos de datos:
  - tel:976491015
  - http://www.iespabloserrano.es
  - content://contacts/people

```
Intent i = new Intent(Intent.ACTION_VIEW, Uri.parse("http://www.iespabloserrano.es"));
```

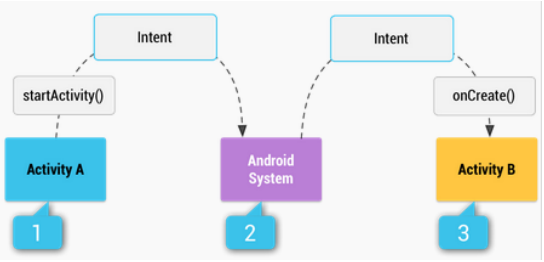
Tipo de datos



- Cuando se pasa un URI al crear nuestro objeto Intent, el sistema determina el tipo MIME adecuado requerido por la intención.
- Cuando no se especifica el URI, debemos utilizar setType() para especificar el tipo de datos (MIME) asociados y así el sistema pueda para determinar qué tipo de componentes deben recibir la intención.
- Ejemplos de tipos MIME son text/xml, image/jpeg, audio/mp3…

```
Intent i= new Intent(Intent.ACTION_SEND);
i.setType("text/plain");
```

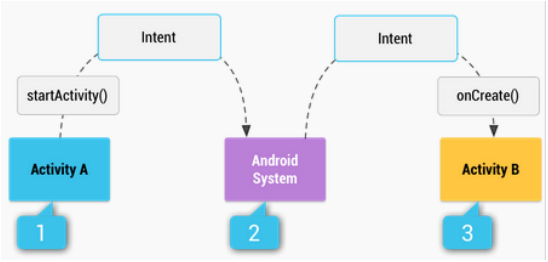
Ejemplos pares acción/datos



Some examples of action/data pairs are:

- `ACTION_VIEW` ***content://contacts/people/1*** -- Display information about the person whose identifier is "1".
- `ACTION_DIAL` ***content://contacts/people/1*** -- Display the phone dialer with the person filled in.
- `ACTION_VIEW` ***tel:123*** -- Display the phone dialer with the given number filled in. Note how the VIEW action does what is considered the most reasonable thing for a particular URI.
- `ACTION_DIAL` ***tel:123*** -- Display the phone dialer with the given number filled in.
- `ACTION_EDIT` ***content://contacts/people/1*** -- Edit information about the person whose identifier is "1".
- `ACTION_VIEW` ***content://contacts/people/*** -- Display a list of people, which the user can browse through. This example is a typical top-level entry into the Contacts application, showing you the list of people. Selecting a particular person to view would result in a new intent { `ACTION_VIEW` ***content://contacts/people/N*** } being used to start an activity to display that person.

Extras

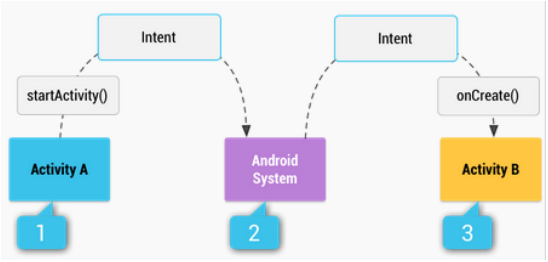


- Información adicional que será recibida por el componente llamado.
- Está formada por un conjunto de pares variable/valor que se almacenan en objetos de la clase Bundle.
- Existen constantes para extras genéricos.

```
Intent i= new Intent(Intent.ACTION_SEND);
i.setType("text/plain");
i.putExtra(Intent.EXTRA_SUBJECT, "esto es el asunto");
i.putExtra(Intent.EXTRA_TEXT, "esto es el texto del correo");
i.putExtra(Intent.EXTRA_EMAIL, new String[] { "bsoler@iespabloserrano.com"});
```

- EXTRA_ALARM_COUNT
- EXTRA_BCC
- EXTRA_CC
- EXTRA_CHANGED_COMPONENT_NAME
- EXTRA_DATA_REMOVED
- EXTRA_DOCK_STATE
- EXTRA_DOCK_STATE_HE_DESK
- EXTRA_DOCK_STATE_LE_DESK
- EXTRA_DOCK_STATE_CAR
- EXTRA_DOCK_STATE_DESK
- EXTRA_DOCK_STATE_UNDOCKED
- EXTRA_DONT_KILL_APP
- EXTRA_EMAIL
- EXTRA_INITIAL_INTENTS
- EXTRA_INTENT
- EXTRA_KEY_EVENT
- EXTRA_ORIGINATING_URI
- EXTRA_PHONE_NUMBER
- EXTRA_REFERRER
- EXTRA_REMOTE_INTENT_TOKEN
- EXTRA_REPLACING
- EXTRA_SHORTCUT_ICON
- EXTRA_SHORTCUT_ICON_RESOURCE
- EXTRA_SHORTCUT_INTENT
- EXTRA_STREAM
- EXTRA_SHORTCUT_NAME
- EXTRA_SUBJECT
- EXTRA_TEMPLATE
- EXTRA_TEXT
- EXTRA_TITLE
- EXTRA_UID

Categorías



- CATEGORY_DEFAULT
- CATEGORY_BROWSABLE
- CATEGORY_TAB
- CATEGORY_ALTERNATIVE
- CATEGORY_SELECTED_ALTERNATIVE
- CATEGORY_LAUNCHER
- CATEGORY_INFO
- CATEGORY_HOME
- CATEGORY_PREFERENCE
- CATEGORY_TEST
- CATEGORY_CAR_DOCK
- CATEGORY_DESK_DOCK
- CATEGORY_LE_DESK_DOCK
- CATEGORY_HE_DESK_DOCK
- CATEGORY_CAR_MODE
- CATEGORY_APP_MARKET

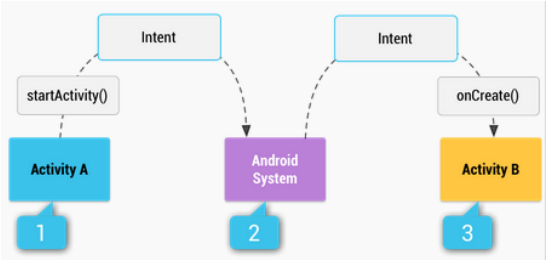
- Complementa a la acción ya que proporciona información adicional sobre el tipo de componente que ha de ser lanzado.
- En la clase Intent se definen una serie de categorías genéricas que podemos utilizar.

Constante	Significado
CATEGORY_BROWSABLE	La actividad objetivo puede ser llamada con seguridad por el navegador para mostrar datos referenciados por un link — por ejemplo, una imagen o un mensaje de correo electrónico.
CATEGORY_GADGET	La actividad puede ser embebida dentro de otra que sea un host de gadgets.
CATEGORY_HOME	La actividad muestra la pantalla de inicio, la primera pantalla que el usuario ve cuando el dispositivo se enciende o cuando se hace click en la tecla HOME.
CATEGORY_LAUNCHER	La actividad puede ser la actividad inicial de una tarea y está en el nivel más alto del lanzador de la aplicación.
CATEGORY_PREFERENCE	La actividad objetivo es un panel preferente.

- El número de categorías fijadas en el Intent puede ser amplio pero la mayoría de las veces no es necesario fijarlo, ya que las actividades llamadas desde intent implícitos con startActivity() o startActivityForResult() obligatoriamente deben fijar en su *intent filter* :

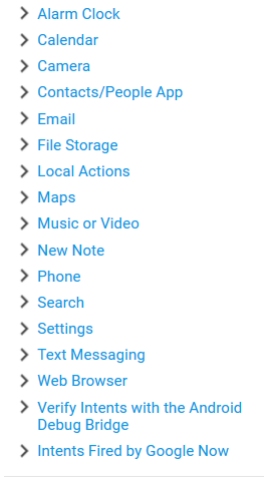
```
<category android:name="android.intent.category.DEFAULT" />
```

Ejemplos de creación de intenciones implícitas



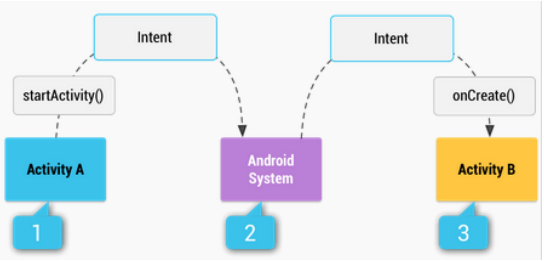

```
Intent i = new
    Intent(Intent.ACTION_VIEW, Uri.parse("http://www.iespabloserrano.es/"));
Intent i = new  Intent(Intent.ACTION_VIEW, Uri.parse("geo:41.656313,-0.877351"));
Intent i = new  Intent("android.media.action.IMAGE_CAPTURE");
Intent i = new Intent(Intent.ACTION_CALL, Uri.parse("tel:976491015"));
Intent i = new Intent(Intent.ACTION_DIAL, Uri.parse("tel:976491015"));
```

- Más ejemplos "oficiales" muy recomendables:



Para simular llamadas, puedes abrir dos emuladores y sus nº de puerto simulan los nº de teléfono (por ejemplo 5554 y el 5556) .

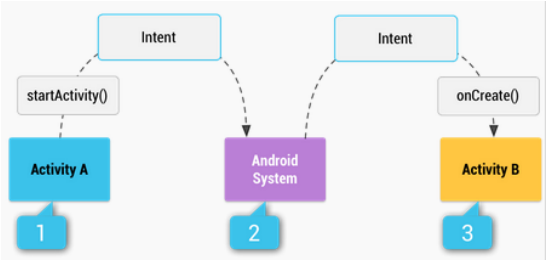
Usando emuladores



- Si ejecutas esto ejemplos en un emulador "normal" es muy posible que el botón mandar Correo o Google Maps no funcione. La razón es que no hay ninguna aplicación instalada en el emulador que sea capaz de realizar este tipo de acciones.
- Para resolver algunos de estos problemas, crea un dispositivo virtual con Google API (para ello has debido instalar desde el SDK Manager las GoogleAPIs de la plataforma que te interesa para el emulador). Estos dispositivos incorporan además de las API de Android, algunas (no todas) de las API de Google, como la de Google Maps.

Android 2.3.3 (API 10)			
SDK Platform	10	2	Installed
Samples for SDK	*10*	*1*	*Not installed*
Intel x86 Atom System Image	*10*	*2*	*Not installed*
Google APIs	*10*	*2*	*Not installed*

Comprobar existencia de aplicación y escoger preferida



- Antes de lanzar la intención deberíamos comprobar que existe una aplicación que la puede manejar usando el método resolveActivity():

```
if (intent.resolveActivity(getPackageManager()) != null) {
        startActivity(intent);
    }
```

- Si existe más de una y deseamos que aparezca un selector (es lo que hace por defecto, a menos que se haya definido aplicación por defecto, por ej. Gmail como correo)

```
// Verify the intent will resolve to at least one activity
if (intent.resolveActivity(getPackageManager()) != null) {
        String title = getResources().getString(R.string.chooser_title);
        // Create intent to show chooser
        Intent chooser = intent.createChooser(intent, title);
        startActivity(chooser);
}
```

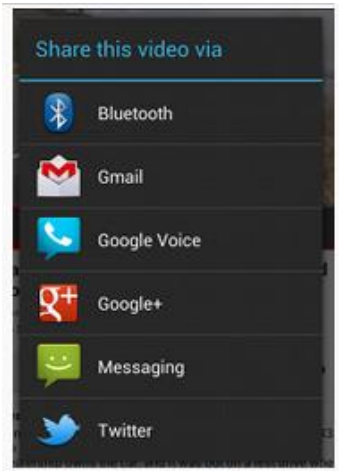
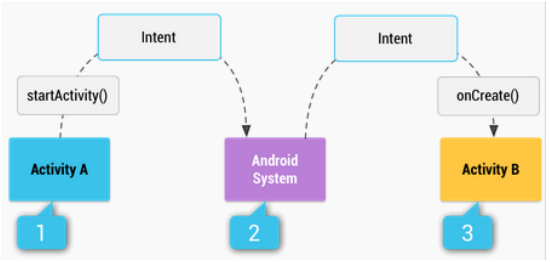


Figura 2. Un diálogo selector.

Permisos

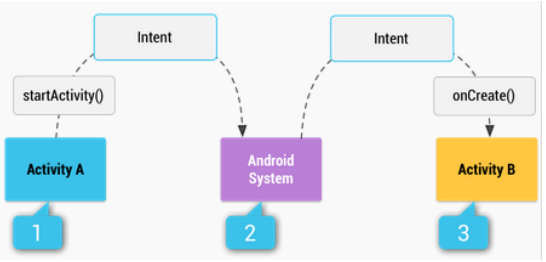


- En muchos ejercicios veremos como se ha de declarar en el AndroidManifest los permisos que necesita la aplicación (llamar por teléfono o acceder a Internet, etc).
- Cuando estas acciones las realiza directamente un Intent implícito, no siempre hay que pedir estos permisos pero es conveniente hacerlo para asegurarnos el correcto funcionamiento.

```
<uses-permission android:name="android.permission.CALL_PHONE" />
<uses-permission android:name="android.permission.CAMERA" />
<uses-permission android:name="android.permission.WRITE_CONTACTS" />
<uses-permission android:name="android.permission.READ_CONTACTS" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
```

- Desarrollado en la siguiente unidad

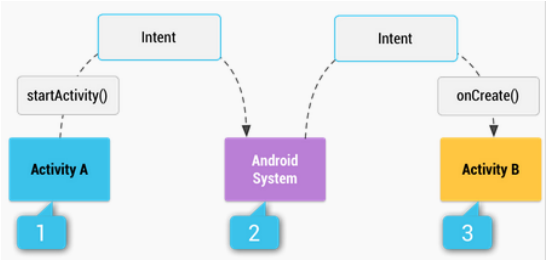
Acciones propias



- También pueden usarse acciones propias definidas en otras aplicaciones nuestras.
- En este caso, en la acción ha de indicarse el nombre completo de la aplicación a la que en el fichero Manifest se le ha fijado con "intent filter" dicha acción en alguna de sus actividades.
- Por ejemplo:

  com.pdm.proyecto_xx.NUESTRA_ACCION.

Aplicación llamada: Intent Filter

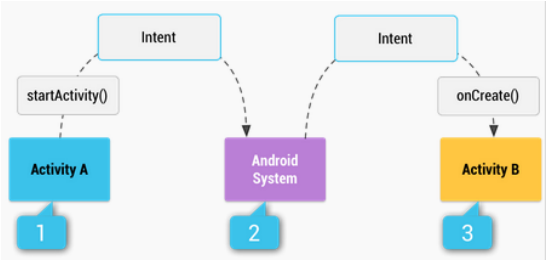


- Como ya se ha dicho, el sistema Android determinará que aplicaciones son las adecuadas para responder a un Intent implícito y en caso de que haya más de una aplicación que pueda responder a la necesidad, se le lanzará al usuario un cuadro de diálogo en el cuál podrá elegir cuál aplicación es la más conveniente para él.
- Esta selección de aplicaciones se basa en el uso de Intent Filters que se definen en el archivo AndroidManifest.xml utilizando la etiqueta <intent-filter> .
- Para reaccionar a un determinado Intent implícito, un componente de la aplicación debe estar registrado a través de un IntentFilter en el archivo AndroidManifest.xml para este evento. Por ejemplo, una actividad de una aplicación desarrollada por nosotros que tenga capacidad de enviar emails (nuestro "GmailPDM" ☺!) debería registrar:

```
<activity ...>
    <intent-filter>
        <action android:name="android.intent.action.SEND" />
        <data android:type="*/*" />
        <category android:name="android.intent.category.DEFAULT" />
    </intent-filter>
```

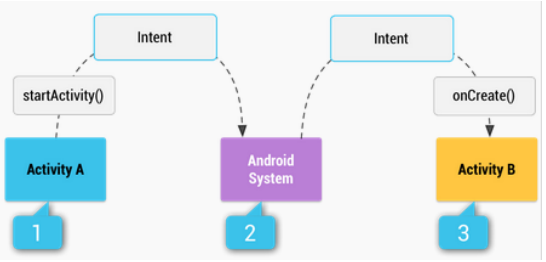
- Recuerda que en caso de que el componente no defina un intent-filter únicamente podrá ser llamado a través de un Intent explícito.
- Además una actividad puede tener más de un *intent-filter* y un *intent-filter* más de una acción.

Obteniendo la información del Intent desde la actividad llamada



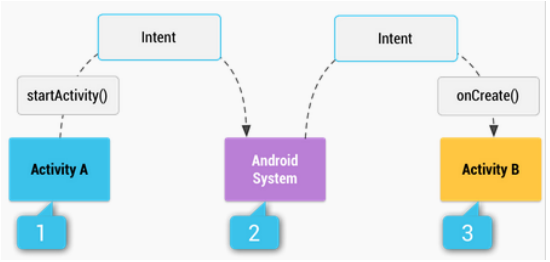
- Para obtener el Intent en nuestra actividad que se haya mandado llamar utilizamos el método **getIntent()**:
  - Intent intent=getIntent();
- Podemos recibir la información desde el Intent utilizando los métodos **getAction()**, **getData()** y **getExtras()**:
  - Bundle bundle = intent.getExtras();

Otros usos de Intent



- En este tema y en el anterior, solo hemos estudiado las intenciones para lanzar actividades ya que es su uso principal pero tienen otros que veremos más adelante.
- Por ejemplo Android utiliza las intenciones de broadcast para enviar eventos del sistema como cambios en la carga de la batería, conexiones de red, llamadas entrantes, sms entrantes,...

Prácticas propuestas



- Realiza la hoja de ejercicios Ejercicios_05_Comunicando_aplicaciones