MercadoLibre - Examen Mobile Developers - Ciro Gonzalez

Es **importante** que las respuestas las escribas con tus palabras. Si consultás alguna fuente de información porque desconoces un tema, te pedimos que por favor nos indiques cuáles son.

**Conceptos generales de programación**

* Definir que es una Clase y que es una instancia de una clase

Es una representación de un conjunto de objetos. Puede tener atributos y métodos. Puede ser o no Abstracta. Una instancia de esa clase es cuando se “materializa” la clase. Pueden ser iguales o no sus atributos pero igual van a ser una instancia de la misma clase.

* Explique los siguientes conceptos:
* Herencia
* Polimorfismo
* Encapsulamiento

Son tres pilares de la POO.

Herencia: Las Clases pueden heredar de una Clase Padre. Pueden heredar métodos y también atributos.

Polimorfismo: Cuando una clase hereda un método de su clase madre y lo ejecuta a su manera. Dos clases heredan el mismo método de una clase madre, y cada una lo ejecuta de manera distinta aunque sea el mismo método.

Encapsulamiento: Permite “ocultar” las responsabilidades de cada objeto. O sea qué es lo que hace/puede hacer. No hace falta saber el COMO hace las cosas sino el QUE hace.

* ¿Qué es un patrón de diseño?. Ejemplificar con patrones usados en la plataforma

Son soluciones a problemas que son recurrentes y ya conocidos. Hay distintos tipos de patrones de diseño.

MVC (Modelo-Vista-Controlador). Es con el que estoy familiarizado. La vista se encarga de mostrar la información, que se la pide al Controlador. El controlador se encarga de establecer la lógica y pide a un DAO que le traiga toda la información que necesita. El mismo controlador puede ser utilizado en varias vistas

MVP(Modelo-Vista-Presentador). Investigando entendí que la principal diferencia con MVC es que el Presentador se comunica con la vista mediante interfaces, y la Vista con el Presentador mediante los métodos de una instancia de dicho presentador. Y que cada presentador responde a una sola Vista. (Stackoverflow)

* ¿Qué es una interfaz?. Compare la misma con una clase abstracta

Es como un contrato. Cuando se crea una interfaz se le escriben métodos. Estos métodos van a ser obligatoriamente implementados en las clases que implementen la interfaz. La principal diferencia con la clase abstracta es que la clase abstracta tiene atributos y las interfaces no.

* ¿Qué es un Thread? Describa brevemente qué consideraciones hay que tener cuando 2 threads acceden a un mismo recurso

Es un hilo de ejecución. Puede haber mas de un Thread. Hay que tener precauciones con la sincronización de los threads en caso de que haya mas de uno, y también con el “Deadlock” que viene de la mano de la sincronización para que uno no se quede esperando la respuesta de otro que nunca llegará.

* ¿Qué es una Exception?, ¿cómo se puede manejar?

Son errores en ejecución en alguna instrucción de la app. Se pueden manejar con el Try-Catch

**Programación y Java** (puede haber más de una rta correcta)

* equals() y hashCode()
* Son equivalentes
* No tienen relación \*
* Dos objetos iguales para equals deben tener el mismo hashcode \*
* Dos objetos distintos pueden tener el mismo hashcode \*
* Son fundamentales para meter objetos en un hashSet
* Synchronized...
* Es para evitar que más de un hilo ejecute una sección del código al mismo tiempo
* Es una palabra reservada en Java \*
* Se puede usar como modificador de una variable de instancia \*
* Se puede usar como modificador de un método de instancia
* Se puede usar como modificador de una clase
* Se puede usar para sincronizar sólo algunas líneas dentro de un método.
* HTTP (HyperText Transfer Protocol)
* Es un programa para navegar internet
* Es un protocolo stateless \*
* Para un request solo corresponde un response \*
* Permite transferir únicamente Hipertexto.
* GET y POST Son equivalentes
* HTTP siempre viaja encriptado para evitar que alguien pueda ver el contenido.
* Explique qué entiende por REST y de un ejemplo(Varias páginas de internet. No lo sabía)

Es un Estilo de arquitectura a la hora de transferencia de datos entre Cliente y Servidor. Utiliza HTTP. Para que sea REST tiene que aplicar lo siguiente:

-Cliente-servidor.

-Stateless.

-Interfaz Uniforme: Los recursos se exponen a traves de URL. Hacen referencia a los recursos, no a acciones. Post,Put,GetDelete.

Cacheables: Sucede en el cliente. Cachea peticiones.

Multicapa: Al cliente no le interesa si está conectado o no al servidor o a otra capa intermedia.

* Enumere tres headers de HTTP y su funcionalidad (Ayuda de google.)

Server: Tipo de servidor HTTP empleado

Expires: Hasta cuando es válida esta response

Cache-control: Cuanto tiempo y que cosas cachear en el servidor

* ¿Qué son y qué diferencias hay entre un Thread (Hilo) y un Proceso?
* Son sinónimos
* Un proceso puede tener múltiples threads \*
* Un thread puede tener múltiples procesos
* Los procesos no comparten espacio de memoria \*
* Los threads no comparten espacio de memoria
* ¿Cómo se logra la ejecución de más de 1 hilo en un procesador con un solo core (núcleo)?
* No es posible
* Se particiona el core en partes para cada hilo \*
* Se turnan los hilos para correr en el mismo core en espacios de tiempo pequeños. \*
* Se turnan entrando uno al terminar el trabajo del anterior.
* Se generan nuevos cores a demanda, para cada hilo.

**Conocimientos de Android**

* ¿Qué es una Activity? Definir su ciclo de vida.

Es una pantalla donde el usuario puede interactuar con la app. Se lee asocia un layout que es lo que se va a ver en pantalla. Del lado del código se le asignan las cosas que se pueden hacer en ella. onCreate, onStartm onResume, onPause, onStop, onDestroy

* ¿Puedo llamar a una Activity y esperar un resultado?

No entendí la pregunta. Supongo que refiere al startActivityForResult() y a onActivityResult(). En caso de ser eso, si. Se llama a una actividad para pedirle ALGO pero la otra actividad tambien tiene que estar diseñada para que envíe esa data.

* ¿Qué es un Fragment? Explicar las diferencias con respecto a un Activity.

Es una parte de una interfaz del usuario. A diferencia de una activity estos tienen como ventaja que se pueden reutilizar, se los puede usar para modularizar una Activity, se les puede agregar transiciones, son útiles a la hora de tratar pantallas de tamaños grandes como las tablets.

* ¿Cómo se comunican un Fragment con una activity? ¿Es posible la comunicación en ambos sentidos?

Mediante interfaces declaradas en el Fragment. La activity las implementa y utiliza el método declarado en el Fragment. Se llama a este método desde el fragment y la Activity lo ejecuta.

No mediante la misma interfaz. La activity le puede pasar al fragment data mediante el setArguments cuando lo crea por ejemplo.

* ¿Qué son los recursos y cómo funcionan?

Es la parte gráfica, los XML, los layuouts, los menu, los drawables. Accedemos mediante la clase R. Inflamos las vistas mediante código.

* Qué es y para que se utiliza: AndroidManifest, Intent, Service, AsyncTask, Handler, LayoutInflater, Adapter y Gradle.

AndroidManifest: Es donde estan declaradas todas las Activities. Donde se ponen los permisos como por ejemplo el acceso a internet. También donde se ve reflejado el icono de la app, entre otras cosas.

Intent: Se utiliza para ir de una Activity a otra. Se crea un Intent y se le pasa como parametro la Activity en la que estamos y la Activity a la que llamamos.

Service: Es un componente que lo puede iniciar otro componente de la app. Se comporta en segundo plano, sin interfaz.

AsyncTask: Va a ejecutarse en un nuevo Thread que se genera para esa AsyncTask.

Handler: Estan asociados a un Thread. Permite poner en cola los procesos. (Google – Android dev)

LayoutInflater: Lo utilizamos para mostrar o “inflar” un layout en una View. Por ejemplo una appbar o una celda custom de un recycler o spinner.

Adapter: Lo usamos para manejar la data de una vista. Como si fuese un puente. Como por ejemplo un recyclerview. Los podemos crear como una clase aparte o usar alguno directamente como un ArrayAdapter.

Gradle: Es donde van a estar todas las dependencias y librerías que utilicemos. Tambien que version de sdk vamos a usar y el numero de version.

**Patrones de diseño Android**

* ¿Qué es el ActionBar y qué uso puedo darle?

Es la barra superior de la app. Se le pueden agregar botones / Texto que responden al ser presionados en caso que se desee.

* ¿Qué diferentes tipos de navegación se utilizan regularmente en Android? Enumerarlas y definirlas.

No se si entendí bien la pregunta.

Actualmente todo se esta pasando a una navegación manejada desde la parte inferior, con una Bottom Navigation Bar. Una barra en la parte inferior de la app con botones y que cada uno te lleva a una pantalla distinta.

Sino también hay apps que todavía siguen con el Navigation Drawer que es un slide desde la izquierda que hace aparecer menúes y que cada uno te lleva a una pantalla distinta.

Si no se utilizan ninguno de estos tipos tal vez la app tiene un flujo de pantallas que va de A a B si o si, como en el caso del ejercicio C que es un pago, y la navegación entre pantallas es mediante un solo boton que va de fragment en fragment hasta terminar.

* ¿Que es el Android Support Library?. ¿Qué beneficios podemos obtener al utilizarlo?

Es una librería en la que hay funciones que no vienen integradas y se pueden aplicar en distintas versiones de API.

**Ejercicios de código**

**Ejercicio A**

Dada la siguiente Activity, explique las líneas de código numeradas:

public class EjercicioActivity extends Activity {

@Override

public boolean onCreateOptionsMenu(Menu menu) { **//1**

getMenuInflater().inflate(R.menu.menu\_ejemplo); **//2**

return true;

}

@Override

public boolean onOptionsItemSelected(MenuItem item) { **//3**

switch (item.getItemId()) {

case android.R.id.home: **//4**

// TODO

return true;

default:

return super.onOptionsItemSelected(item);

}

}

}

Línea 1) ¿Para qué se utliiza dicho método?

El método va a crear el menu que se le pase en la vista con el .inflate()

Línea 2) ¿Qué realiza dicha línea de código?

Obtiene el inflater, le pasan el xml del menu y lo genera en la vista.

Línea 3) ¿Cuál es el propósito de dicho método?

Dependiendo que item se presione del menú va a hacer determinada cosa.

Línea 4) ¿Qué se recomienda hacer dentro de dicho Case?

Volver al home de la app (No se si entendí bien la pregunta)

**Ejercicio B**

Definir qué realiza el siguiente código:

String nombre = "Roberto";

Intent intent = new Intent(this, CualquierActivity.class);

intent.putExtra("nombre", nombre);

startActivity(intent);

Crea una variable String y le asigna el valor “Roberto”. Luego crea un intent y se le pasa como parámetro la activity en la que esta parado y como segundo parametro la Activity a la que vamos a ir. Luego le asigna al intent un “extra” con el .putExtra() con su clave/valor para enviar la data a la otra activity a la que vamos con el intent. Finalmente con el startActivity(intent) vamos a la otra actividad.

**Ejercicio C (mayor importancia) :** <https://github.com/cirogg/MLTecnica_Fragments_Ciro>

Con este ejercicio se busca que muestres conocimientos a la hora de crear un proyecto en Android, ver si usas librerías y cómo planteas la solución. **Es importante que cuando respondas el cuestionario, la respuesta a este ejercicio sea un link a Github, Bitbucket o algún repositorio de código.**

MercadoPago posee APIs abiertas a la comunidad para que cualquier desarrollador las consuma y pueda tener pagos en su aplicación. Pero las APIs necesitan de una clave pública para identificar quién es el que está invocando a la API, además de ofrecer customizaciones para esa persona en particular. En este ejercicio, vas a utilizar la siguiente public\_key: **444a9ef5-8a6b-429f-abdf-587639155d88**.

A la hora de realizar un pago con tarjeta de crédito, se debe especificar el monto, el medio de pago, el banco y la cantidad de cuotas, además de una identificación segura (token) de la tarjeta con la que se está realizando el pago.

Te pedimos que realices una secuencia de pantallas realizando distintos API Calls para obtener todos los datos (exceptuando el token). El input de la pantalla actual son todos los seleccionados por el usuario en las pantallas anteriores.

La secuencia de pantalla debe ser:

* **Pantalla de ingreso de monto.**
* **Selección de medio de pago** (de tipo **credit\_card**). Los medios de pago puede ser: Visa, Mastercard, American Express, etc. **Mostrar nombre e imagen.** Para acceder a estos medios de pago, se consulta a la siguiente URL: [https://api.mercadopago.com/v1/payment\_methods?public\_key=PUBLIC\_KEY](https://api.mercadopago.com/v1/payment_methods?public_key=444a9ef5-8a6b-429f-abdf-587639155d88) (Método GET).
* **Selección de banco**. Los bancos pueden ser: ICBC, Santander, Galicia, entre otros. **Mostrar nombre e imagen.** Estos operan con diversos medios de pago de pago, por lo que debes consumir una API para poder saber cuáles son los bancos disponibles para el medio de pago seleccionado. [https://api.mercadopago.com/v1/payment\_methods/card\_issuers?public\_key=PUBLIC\_KEY HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/card\_issuers?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&payment\_method\_id=visa"& HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/card\_issuers?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&payment\_method\_id=visa"payment\_method\_id=MEDIO\_DE\_PAGO\_SELECCIONADO](https://api.mercadopago.com/v1/payment_methods/card_issuers?public_key=444a9ef5-8a6b-429f-abdf-587639155d88&payment_method_id=visa) (Método GET).
* **Selección de cuotas**. Luego de tener el medio de pago y las el banco seleccionados, el usuario debe seleccionar la cantidad de cuotas en las que desea pagar el monto ingresado en el paso 1. La API provee un ***recommended\_message*** que resuelve el mensaje que se debe mostrar al usuario con la cantidad de cuotas, el valor de cada cuota y el monto final. **Mostrar el recommended\_message.** [https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=PUBLIC\_KEY HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"& HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"amount=MONTO\_DEL\_PASO\_1 HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"& HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"payment\_method\_id=MEDIO\_DE\_PAGO\_SELECCIONADO HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"& HYPERLINK "https://api.mercadopago.com/v1/payment\_methods/installments?public\_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment\_method\_id=visa&issuer.id=288"issuer.id=ISSUER\_SELECCIONADO](https://api.mercadopago.com/v1/payment_methods/installments?public_key=444a9ef5-8a6b-429f-abdf-587639155d88&amount=15&payment_method_id=visa&issuer.id=288) (Método GET).

Utilizar un esquema de callbacks para cada servicio utilizado. Modularizar la llamada para recibir base\_url, uri y query\_params.

El ejercicio debe volver a la primer pantalla y mostrar un mensaje de alerta con los valores seleccionados en el flujo de pantallas solicitado.

No es necesario realizar **todo** el modelo de clases, pero se valorará la calidad de la solución entregada y la profundidad del tema.

Recordá entregar el proyecto en un **repositorio**.

***La resolución del ejercicio es totalmente libre, todo lo que hagas de más va a sumar en tu proceso de selección.***

**Buena suerte y gracias por tu tiempo!**