**Exemplos de Questões para PS Android.**

**Quer Respostas Completas.**

1. **Qual a finalidade do arquivo AndroidManifest.xml?**

Centralizar as informações e permissões pertencentes ao aplicativo.

1. **Qual a finalidade de uma Activity?**

Onde toda a interação do usuário acontecerá. Uma janela, responsável pela interface gráfica do usuário. Janelas da aplicação.

1. **Cite exemplos de Persistência (armazenar os dados e recuperar depois) no Android.**

Os principais exemplos de Persistências são: SharedPreferes, File (em arquivo) e SQLite.

1. **O que é uma View? Justifique sua resposta.**

São os componentes gráficos. É uma superclasse de qualquer componente gráfico do Android. Ex: TextView herda de View.

1. **Cite exemplos de 3 layouts (são views que organizam outras views dentro dela).**

FrameLayout, LinearLayout, RelativeLayout, AbsoluteLayout.

1. **Qual a diferença entre LinearLayout e FrameLayout?**

Linear: Organizar em orientação específica, horizontal e vertical.

Frame: Tudo que jogar dentro dele, ocupa o espaço disponível.

Table: Organiza as views em Forma de tabelas.

Relative: Relativo uma da outra. Organiza as views uma do lado da outra, uma embaixo da outra.

Absolute: Organiza em uma posição específica em relação a x e y.

1. **Cite o ciclo de vida de uma Activity.**

OnCreate () – OnStart () – OnResume () – OnPause () – OnRestart () – OnStop () – OnDestroy ().

1. **Explique a função de um Adapter.**

Tem a função de adaptar dados de um objeto específico dentro de uma determinada View.

1. **Quando podemos utilizar um ProgressDialog?**

Serve para dar feedback ao usuário para mostrar que uma ação está acontecendo e está demorando.

1. **O que é um Estilo?**

Centralizar formatações padrões que é possível reaproveitar em outros componentes.

1. **O que é um tema?**

É uma combinação de estilos. Agrupa vários estilos e aplica-os com suas formatações. Ex: Todo botão vai ter uma formatação específica.

1. **Como aplicar o conceito de Herança em um estilo? \***

<style name: “botão”>

</style>

<style name: “botão.cancelar”> (Exemplo dois botões, só que um tem uma função cancelar, ele não deixa de ser um botão, mas tem uma função diferente, porém herda de ***BOTÃO.***

</style>

1. **Em qual arquivo colocamos itens para um menu?**

Colocamos itens no Menu.XML.

1. **Qual a finalidade de um inflate?**

A finalidade de um inflate é inflar (Ler) um xml e transformá-lo em um objeto Java.

1. **Qual a diferença entre AcionBar e ToolBar?**

A diferença é:

*ActionBar:* Já vem por padrão na activity, é limitada, é sempre fixa.

*ToolBar:* Colocar mais de uma numa activity, manipular a sua posição. Conseguimos personalizá-la. Precisa definir qual é a padrão, já que pode-se inserir mais de uma.

1. **Qual a finalidade de um Fragment?**

A finalidade de um fragment é reaproveitar porções/pedaços de uma activity em outras activitys.

1. **Pasta RES**

Armazena os recursos utilizados pelo app (imagens, layouts de tela, itens de menu, folhas de estilo, textos e etc..)

1. **Layouts**

O layout define a estrutura visual para uma interface do usuário, como a IU de uma atividade ou de um widget de aplicativo.

**SQLite**

É uma biblioteca em linguagem C, open source, que implementa um banco de dados SQL embutido, ou seja, não tem processo servidor separado, lendo e escrevendo diretamente no arquivo de banco de dados no disco.

**Notification**

Uma notificação é uma mensagem que pode ser exibida ao usuário fora da IU normal do aplicativo. Quando você diz ao sistema para emitir uma notificação, ela primeiro aparece como um ícone na **área de notificação**.

Permitem disparar determinadas ações que notificam o usuário sobre a ocorrência de um evento - como o recebimento de uma mensagem de texto;

**ApiRetrofit**

A Retrofit é uma API desenvolvida pela Square seguindo padrão REST, fornecendo um padrão simples de implementação para transmissão de dados entre aplicação e servidor, que faz uso do JSON.

**BroadcastReceivers**

Os Broadcast Receivers são componentes responsáveis por receber e tratar eventos (ou broadcasts) provenientes do sistema ou de outras aplicações.

São os receptores das Intents. Uma vez que um Intent é enviado algum Broadcast Receiver deverá ser acionado;

**ListView**

Utilizado para listar os dados.

**NavigationDrawer**

Utilizamos um Navigation Drawer quando queremos montar menus deslizantes, bastante utilizados em aplicativos Android.

**AsyncTask**

Muitas vezes, precisamos processar dados e tarefas pesadas que levam alguns segundos ou até mesmo minutos para serem concluídos. Quando fazemos isso na thread principal, o APP poderá ficar travado até o final do processamento e consequentemente frustrar o usuário. Para resolver este tipo de problema, a plataforma Android nos disponibiliza o AsyncTask, uma classe abstrata que encapsula todo o processo de criação de Threads e Handler.

[**SharedPreferences**](https://github.com/fernandomazola-student/2TINA-Android-SharedPreferences)

As SharedPreferences são utilizadas em situações onde não há necessidade de criação de um banco de dados, ou até mesmo quando há pouco número de dados a ser armazenado.

**Intent**

Um objeto Intent carrega informações que o sistema Android usa para determinar o componente a iniciar (como o nome exato do componente ou categoria do componente que deve receber o intent), além de informações que o componente receptor usa para realizar a ação adequadamente (como a ação a tomar e os dados a usar). São mensagens que podem ser trocadas entre aplicações ou entre diversas Activities da mesma aplicação;

**Bluetooth**

Muitas vezes precisamos de uma comunicação entre dispositivos. Podemos optar por enviar ou receber dados através de redes Wi-Fi e 3G. Porém, o Android também disponibiliza uma API para comunicação via Bluetooth, uma alternativa para transmissão de dados em curta distância.

**RecyclerView**

Com o passar do tempo o componente ListView mostrou-se não tão performático para listagem de muitos dados. Outros problemas que os desenvolvedores tinham ao utilizar o componente ListView era fazer o uso de animações ao inserir ou remover elementos da listagem, listar dados em formato de grid, etc. Com isso o Google introduziu junto com o Material Design o componente RecyclerView que resolve estes problemas citados

**PickersAndDialogs**

Android fornece controles para o usuário escolher um momento ou escolher uma data como diálogos ready-to-use.

**ActionBar**

A barra do aplicativo, também conhecida como a barra de ações, é um dos elementos de design mais importantes das atividades do seu aplicativo, pois ela fornece uma estrutura visual e elementos interativos que são familiares para os usuários.

**ToolBar**

Em sua forma mais básica, a barra de ações exige o título da atividade em um lado e um menu flutuante no outro. Mesmo nesse formato simples, a barra do aplicativo fornece informações úteis para os usuários e ajuda a dar uma aparência consistente aos aplicativos Android.

**Activities -** representam a camada visual da aplicação onde estão contidos os controles de interface (botões, caixas de texto, etc) chamados de Views;

**Services -** componentes não visuais que executam processamento em background;

**Content Providers -** utilizados para armazenamento de informações que podem ser compartilhadas entre aplicações distintas.

**Strings.xml**

No arquivo strings.xml todos os textos de uma aplicação podem ser criados (rótulos, mensagens de erro, textos explicativos, etc...)

**CAIXAS DE DIÁLOGO**

As caixas de diálogo são um recurso interessante para comunicar ao usuário sobre eventos importantes da aplicação; Todas as caixas de diálogo são descendentes diretos ou indiretos da classe Dialog e algumas já são padrão: Também podem ser criadas caixas de diálogo personalizadas estendendo a classe pai Dialog.

• AlertDialog: mensagens de alerta genéricas (conforme já visto);

• ProgressDialog: mostram progresso de uma ação;

• DatePickerDialog: permitem a seleção de uma data;

• TimePickerDialog: permitem a seleção de hora;