|  |  |
| --- | --- |
| **Prova Prática - Desenvolvedor Mobile - Android** | |
| **Nome:**  Gabriel da Silva Esteves | **Data:** 16/04/2019 |
| **Vaga:** | **Nota:** |

Programação

1. **O que são design patterns? Cite dois exemplos.**

R: Design patterns ou padrões de projetos, foram criados com a finalidade de ajudar e simplificar o desenvolvimento de softwares. Exemplos: Observer, Abstract Factory.

1. **Qual a importância dos princípios SOLID (Single responsibility/Open–closed/Liskov substitution/Interface segregation/Dependency inversion) no desenvolvimento de uma solução?**

R: Os princípios de SOLID auxiliam os desenvolvedores a criarem um código de qualidade, separando as responsabilidades, reduzindo o acoplamento das classes e definindo melhor a relação entre elas, desta forma, podemos ter uma alta redução na complexidade do código fonte, criando assim um código limpo e de qualidade para possíveis manutenções e testes no código.

1. **Explique os design patterns Singleton, Factory, e Chain of Responsibility Singleton?**

R: Singleton: tem como objetivo, criar classes que tenham somente uma única instância. Factory: tem o objetivo de fornecer uma interface para criar vários(fábrica) objetos sem especificar as suas classes concretas. Chain of Responsibility Singleton: tem o objetivo de criar uma cadeia de objetos que irão servir como receptores para uma solicitação. Separando assim o seu remetente e seu destinatário.

1. **O que é complexidade algorítmica, e qual a sua importância na programação?**

R: A complexidade algorítmica está preocupada com a performance dos algoritmos, tentando determinar a velocidade de uma algoritmo para executar uma determinada tarefa. Ela tem uma importância muito grande quando pensamos em experiência de usuários, principalmente com aplicativos que não tem uma boa resposta de algo na tela, ou seja, algo visível para o usuário, para indicar que algo está acontecendo no aplicativo.

1. **Explique o que são funções anônimas? Dê um exemplo**

R: Funções Anônimas são funções que são passadas como se fossem um objeto qualquer, elas podem ser atribuídas a variáveis e são passadas para camadas superiores. Exemplo: expressões lambdas no Kotlin.

Android SDK

1. **O que são Activities?**

R: São as classes gerenciadoras da interface do usuário(UI), quando um aplicativo é inicializado e vemos a primeira tela do App, o que estamos vendo na verdade é uma activity.

1. **O que são Fragments?**

R: Fragments podem ser definidos como um fragmento de uma tela, sendo muito utilizado para apresentar, uma reutilização de funcionalidades na tela de um dispositivo, podendo servir até para modularizar as Activities.

1. **Qual a diferença entre Activities e Fragments?**

R: A diferença entre os dois é que um Fragment precisa de uma Activity para ser visualizado, sendo que uma Activity funciona independente do uso de um Fragment.

1. **Explique o que são Intents?**

R: São objetos utilizados para a solicitação de uma ação entre os componentes de um aplicativo.

1. **Descreva o ciclo de vida de um aplicativo Android?**

R: OnCreate: É a primeira função chamada em uma activity, carregando o XML da activity. OnStart: É a função chamada logo após o onCreate, normalmente também é chamada quando uma activity que estava em background volta para a Main Thread. OnResume: É a função chamada após o onStart na inicialização de uma activity e quando uma activity que estava em background volta a ter foco. Activity running: Activity está funcionando. OnPause: É a função chamada assim que uma activity vai perder o foco na Main Thread, normalmente quando uma nova activity vai ser chamada. OnStop: É a função chamada quando uma activity fica fora da Main Thread, ou seja, tem uma outra activity funcionando na frente dela. OnDestroy: É a função chamada quando uma activity vai ser destruída, ou seja, para de funcionar. Se a activity for chamada novamente terá que ser construída uma nova, refazendo o ciclo da activity novamente. OnRestart: É a função chamada antes do onStart quando uma activity que estava em background vai voltar a ter foco na Main Thread.

1. **Qual o papel do Bundle?**

R: O Bundle geralmente é utilizado para passar dados entre Activities no Android.

1. **O que acontece com um aplicativo quando a orientação do aparelho é mudada (por exemplo, de Portrait para Landscape)?**

R: Quando isso acontece, o onCreate é chamado novamente, se tivermos dados na tela, iremos perde-los, então devemos fazer algumas configurações para que isso não aconteça e não de uma experiência desagradável ao usuário. Medidas a se fazer: Opção 1. Podemos configurar o aplicativo para funcionar em apenas uma orientação (Portrait ou Landscape). Opção 2. Podemos configurar o aplicativo para que ele não chame novamente o onCreate, e assim podemos usar o aparelho com o mesmo layout nas duas orientações. Opção 3. Podemos manter o estado da tela, possibilitando utilizar layouts diferentes de acordo com a orientação do aplicativo, desde que sejam mantidos os componentes (text views, list views, etc) e os ids.

1. **Como preservar o estado do aplicativo quando ele entra em background?**

R: Podemos utilizar a biblioteca do RxJava para preservar os estados da aplicação, fazendo uso continuo das propriedades do RxJava que trabalha muito bem em background, podendo nos ajudar a realizar várias tarefas e até mesmo chamadas de serviços em background, e quando o aplicativo voltar ao foco no aparelho não perderíamos dados na tela.

1. **Networking: Quais as responsabilidades da classe ConnectivityManager?**

R: 1. Monitora o estado de conexão com uma rede (WI-FI, 4G, 3G, etc), podendo notificar o aplicativo caso tenha alterações na rede. 2. Tentativa de reconexão com outra rede, caso a conectividade com uma rede seja perdida. 3. Permite que os aplicativos selecionem as redes de trafego de dados que serão utilizadas.

1. **Explique qual a função do Gradle no ambiente de desenvolvimento Android?**

R: O Gradle serve para automatizar o processo de build de um aplicativo, desde forma, dentro dos arquivos do Gradle, podemos definir as dependências e bibliotecas que serão utilizadas no projeto, ainda nos possibilita personalizar o arquivo Gradle de cada modulo do aplicativo, possibilitando assim o funcionamento de arquivos Gradles diferentes para cada modulo.

1. **Descreva o que fazem as seguintes libs: Retrofit, Picasso, Shimmer, Gson e Stream (libs usadas no projeto Editara)?**

R: Retrofit: É uma biblioteca utilizada para fazer requisições http através da lib OKHttp, no padrão RESTFul, trabalhando junto com libs de serialização de dados Json como a lib Gson. Picasso: É uma biblioteca utilizada para o carregamento de imagens dentro dos aplicativos. Shimmer: Biblioteca utilizada para criar o efeito Shimmer no carregamento de componentes, como listas e ProgressBar. Gson: Biblioteca utilizada para a conversão (serialização e desserialização) de dados Json. Stream: Não conheço essa biblioteca.

Arquitetura

1. **Explique o que é MVC (Model-View-Controller)?**

R: É uma arquitetura utilizada para separar a aplicação em 3 camadas. A camada da View é onde fica a interação com o usuário. A camada do Controller, é responsável por receber as solicitações dos usuários que vem da camada da View e repassa as solicitações para a camada do Model, também repassa dados do Model para a View exibir na tela. Na camada do Model, fica a manipulação e as validações dos dados.

1. **Explique o que é MVVM (Model-View-ViewModel)?**

R: É uma arquitetura que visa separar a responsabilidades de uma aplicação em camadas bem distintas, onde a camada da View não conhece a camada do Model, e vice versa, porém a camada da View conhece a camada do ViewModel, e vice versa, elas se comunicam através de um mecanismo chamado Data Binding, desde forma, uma View pode implementar mais de um ViewModel, e o ViewModel se comunica com a camada do Model, fazendo solicitações, atualizações ou inserção de dados no Model.

1. **Opcional: Explique a arquitetura VIPER (View, Interactor, Presenter, Entity, and Router)?**

R:

1. **Opcional: Explique o conceito de Clean Architecture?**

R: É uma arquitetura criada para ser independente de frameworks e altamente testável em qualquer nível da aplicação, separando bem as suas regras de negócios, camada de UI, camada de acesso ao banco de dados e agentes externos, como serviços. Sendo possível assim testar até mesmos as regras de negócios de forma fácil e sem depender de outras camadas.

Networking

1. **O que é REST/RESTFul?**

R: Pode ser considerada um modelo de arquitetura de comunicação web, que usa protocolos HTTPs para se comunicar com Web Services na web, sem ter a necessidade de criar os seus próprios protocolos.

1. **Explique os seguintes conceitos: GET, POST, DELETE e PUT?**

R: GET: protocolo utilizado para obter dados de um serviço (recurso). Exemplo: obter dados de um usuário. POST: protocolo utilizado para criar um novo recurso no serviço. Exemplo: criação de login. DELETE: protocolo utilizado para excluir um determinado recurso de um serviço. Exemplo: exclusão de um usuário. PUT: protocolo utilizado para substituir dados de um determinado serviço. Exemplo: alteração de login.

GitFlow

1. **Explique o que são as operações de checkout e commit?**

R: Checkout: comando utilizado para a sincronização de dados no seu repositório. Commit: comando utilizado para salvar as alterações realizadas dentro das branches, apontando para um ponto especifico do desenvolvimento da sua aplicação.

1. **O que são branches?**

R: Podemos definir Branches como uma ramificação no desenvolvimento de uma aplicação, onde ela trabalha de forma independente das outras, muito utilizadas no desenvolvimento paralelo de funcionalidades para uma aplicação. Exemplo: Temos o um aplicativo na Google Play, o código fonte está na nossa Branch Master, mas estamos desenvolvendo novas funcionalidade para a aplicação, vamos criar a nossa Branch Develop, que está com o mesmo código da Master nesse momento, a partir desse ponto criamos as nossas branches paralelas partindo da Develop, com as funcionalidades que iremos desenvolver, quando terminarmos o desenvolvimento damos o commit nas Branches e mandamos elas para a Branch de Develop, fazendo com que a Branch de Develop centralize o código com essas funcionalidades novas antes de serem criadas as versões de release da aplicação.

1. **O que faz o comando git pull?**

R: Busca os dados da Branch no repositório remoto da aplicação e já sincroniza/baixa os dados na sua Branch local, podendo fazer o merge dos dados da Branch remota no código que está sendo desenvolvido localmente.

1. **O que faz o comando git push?**

R: Ele manda os dados da sua Branch local para a Branch remota dentro do repositório web. Se for a Branch Master precisa ter acesso de escrita no repositório web.

1. **O que é merge?**

R: É a ação de juntar duas Branch na mesma Branch, por exemplo, podemos juntar a nossa Branch Develop na nossa Branch Master, de forma direta ou através de um Pull Request.

1. **Como funciona rebase? Dê um exemplo.**

R: O rebase permite que você reescreva o histórico do projeto, podendo criar novos commits para cada commit na ramificação original. Isso pode ser um problema enorme, visto que, você perde todas as referências de commits mesclados (merge) do seu código, o que impossibilita a reversão, a correção ou até mesmo um hotfix a partir de um determinado commit, podendo gerar um problema de tempo bem grande para realizar uma alteração simples e rápida em uma parte do código.

1. **Opcional: qual o conceito de squash?**

R: É uma pratica utilizada para juntar vários commits em um único commit, eliminado coisas desnecessárias para o projeto, proporcionando assim commits mais limpos e sem muitos históricos desnecessários.

**\*\*\*Apresentar algum App que desenvolveu!**

Projeto:

<https://github.com/Gabriel-Esteves/JavaPop_Android>