Questionário Programação Mobile

1. O que caracteriza um dispositivo móvel?

Resposta: Dispositivos móveis são equipamentos eletrônicos portáteis, como smartphones e tablets, caracterizados por conectividade, sensores, e recursos multimídia. Seu uso vai desde comunicação e entretenimento até automação e controle de dispositivos. Podemos definir como qualquer aparelho portátil que, por meio de recursos de computação, execute funções e processos, como: reproduzir mídia, navegar na internet, acessar e-mail, localização GPS, calculadora, calendário etc.

2. Qual a diferença entre smartphones e featurephones?

Resposta: Smartphones oferecem um sistema operacional completo (Android, iOS), lojas de aplicativos, multitarefa e suporte a apps complexos. Featurephones são celulares com funcionalidades básicas, sem loja de apps, geralmente com teclado físico e sistema fechado, ou seja, é possível definir um *feature phone* como um celular simples, cujas finalidades incluem a realização de chamadas e o envio de mensagens de texto. Sem a possibilidade de tirar fotos com a qualidade de um smartphone, baixar um aplicativo e, no caso de alguns modelos, de acessar à Internet, esses aparelhos são iguais aos modelos lançados no início dos anos 2000. Já os smartphones ou telefones inteligentes. Trata-se de um telemóvel que oferece funções semelhantes às de um computador e que se destaca pela sua conectividade

3. Cite 3 sensores comuns em smartphones.

Resposta: GPS, acelerômetro, giroscópio.

4. Quais os principais sistemas operacionais móveis?

Resposta: Android (Linux-based, apps em Java/Kotlin), IOS (apps em Swift/Objective-C), Harmony OS, KaiOS e outros sistemas menores.

5. O que são WebApps?

Resposta: Trata-se de uma aplicação de software que roda na internet, em vez de funcionar com base em sistemas operacionais. Uma aplicação web funciona com base na infraestrutura da internet. Um app feito para a internet está livre da

necessidade de compatibilidade com elementos específicos de um computador ou dispositivo. Ele opera de forma descentralizada e móvel.

6. O que são apps nativos?

Resposta: Aplicativos desenvolvidos especificamente para o sistema operacional usando sua linguagem e SDK (Os SDKs disponibilizam bibliotecas para acesso a hardware (câmera, GPS), conectividade (Bluetooth, Estudo Completo - Tecnologias de Dispositivos Móveis e J2ME Wi-Fi), interface gráfica, armazenamento, sensores, redes, etc. Exemplos: Android SDK, Apple Core APIs.) nativos.

7. Dê um exemplo de app híbrido.

Resposta: Aplicativo desenvolvido com Flutter ou React Native.

8. Quais os modelos de monetização de apps?

Resposta:

Vendas Direta (Pay-Once) - O modelo Pay-Once permite que o usuário pague um valor único para fazer o download do aplicativo. Este modelo oferece receita imediata e não exige pagamentos recorrentes**.**

Grátis (Freemium) - O modelo freemium é uma combinação entre “grátis” e “premium”. Ele oferece o aplicativo gratuitamente, mas com funcionalidades limitadas. Para acessar recursos mais avançados, o usuário deve pagar por um upgrade.

Assinatura (Subscription-Based) - A monetização por assinatura se tornou uma das mais populares, pois garante uma fonte constante de receita. O usuário paga uma taxa recorrente, geralmente mensal ou anual, para acessar conteúdo exclusivo ou serviços contínuos.

Compras no app (in-app Purchases) - O modelo In-App Purchases permite que os usuários adquiram itens virtuais, funcionalidades extras ou melhorias dentro do próprio aplicativo.

Financiamento Coletivo (Crowdfunding) - O crowdfunding envolve o financiamento coletivo para desenvolver um app. Usuários podem apoiar financeiramente o projeto em troca de benefícios, como acesso antecipado ou versões exclusivas.

Patrocínio e Publicidade in-app - A publicidade in-app é uma das estratégias mais populares de monetização. Ela permite que os desenvolvedores ganhem dinheiro exibindo anúncios dentro do app.

Comissões sobre Vendas (Revenue Share) - No modelo de comissões sobre vendas, o desenvolvedor do aplicativo recebe uma parte da receita gerada pelas vendas realizadas dentro da plataforma.

Licenciamento de Software (Software Licensing) - O modelo de licenciamento de software permite que os desenvolvedores ofereçam o uso do aplicativo por meio de licenças, normalmente para empresas ou usuários avançados.

Modelos Baseados em Dados(Data Monetization) - A monetização baseada em dados envolve o uso ou venda de informações coletadas dos usuários.

Gamificação e Recompensas (Reward-Based Monetization) - Esse modelo usa a gamificação para incentivar o usuário a realizar ações dentro do aplicativo, como completar tarefas, interagir com a plataforma ou fazer compras.

Modelo de Pay-Per-Use - O modelo pay-per-use (pague para usar) cobra dos usuários apenas pelos serviços ou funcionalidades que eles utilizam dentro do aplicativo.

9. Cite duas bibliotecas ou APIs comuns em Android.

Resposta: LocationManager (GPS), Camera API.

10. Para que serve uma API em dispositivos móveis?

Resposta: Para permitir o acesso a funcionalidades do sistema ou hardware, como câmera, GPS, armazenamento, etc.

11. O que é J2ME?

Resposta: é uma versão da plataforma Java projetada para dispositivos com recursos limitados, como celulares antigos, PDAs e sistemas embarcados.

12. Quais são as quatro camadas da arquitetura J2ME?

Resposta:

Application (Aplicação) - MIDlets - Aplicações desenvolvidas para J2ME, com a classe principal chamada de MIDlet.

Configuration (Configuração) - Composta por dois tipos **- CLCD (Connected Limited Device Configuration)** e **CDC (Connected Device Configuration) -** define o conjunto básico de bibliotecas e a JVM (Java Virtual Machine) disponíveis para um tipo específico de dispositivo. São responsáveis por fornecer um ambiente mínimo de execução, garantindo compatibilidade entre dispositivos de mesma categoria.

Profile (Perfil) - Os perfis são conjuntos de APIs específicas que adicionam funcionalidades sobre uma configuração. Enquanto a configuração (Configuration) fornece a base do ambiente Java, o perfil adiciona funcionalidades extras para diferentes categorias de dispositivos. **Um perfil depende de uma configuração (CDC ou CLDC)** e fornece APIs para interface gráfica, rede, armazenamento e outras funções. **Exemplos de perfis - MIDP (Mobile Information Device Profile) e Foundation Profile.**

Packets Opcional (Pacotes Opcionais) - Além das configurações (CLDC e CDC) e dos perfis (MIDP, Foundation Profile), o J2ME permite o uso de APIs opcionais para adicionar funcionalidades específicas.

Principais pacotes opcionais (JSRs - Java Specification Requests):

● JSR-82 - Bluetooth API - Permite comunicação via Bluetooth em dispositivos móveis.
● JSR-75 - FileConnection API - Fornece acesso ao sistema de arquivos local do dispositivo.
● JSR-135 - Mobile Media API - Suporte para reprodução de áudio, vídeo e imagens.
● JSR-179 - Location API - API para acesso a GPS e serviços de localização

JVM (Máquina virtual do Java) - A máquina virtual que executa os aplicativos Java em dispositivos com recursos limitados. Vai depender da configuração do dispositivo em específico. Se for CLCD em dispositivos com baixa memória entre 160 KB e 512 KB de RAM é **KMV (Kilobyte Virtual Machine)**. Se for CDC em dispositivos com pelo menos 2 MB de RAM é **CVM (Compact Virtual Machine)**.

13. O que é MIDlet?

Resposta: Aplicações desenvolvidas para J2ME sendo a Classe principal de um aplicativo J2ME, que herda de **`javax.microedition.midlet.MIDlet`**.

14. O que significa CLDC?

Resposta: Connected Limited Device Configuration – para dispositivos com recursos limitados.

15. O que significa CDC?

Resposta: Connected Device Configuration – para dispositivos mais potentes, como set-top boxes.

16. O que é MIDP?

Resposta: Mobile Information Device Profile – define as APIs para dispositivos móveis.

17. Qual máquina virtual J2ME geralmente é usada em CLDC?

Resposta: KVM - Kilo Virtual Machine.

18. Quais métodos precisam ser implementados por um MIDlet?

Resposta: `startApp()`, `pauseApp()`, `destroyApp(boolean)`.

19. Para que serve o método `destroyApp()`?

Resposta: Encerra o MIDlet, liberando recursos.

20. O que diferencia CLDC de CDC?

Resposta: CLDC é para dispositivos com menos memória e processamento; CDC para dispositivos mais potentes.

21. O que é a classe Canvas?

Resposta: O Canvas é utilizado para criar interfaces gráficas personalizadas. Diferente de Formou List, o Canvas permite que o desenvolvedor desenhe diretamente na tela. Isso é útil para criar jogos e interfaces gráficas mais avançadas, ou seja, é a Classe que permite desenhar na tela e controlar eventos personalizados.

22. O que é um Displayable?

Resposta: Componente visual que pode ser mostrado em um dispositivo. é a classe responsável pela exibição das telas no dispositivo. Ela determina qual tela o usuário está visualizando, seja uma tela de formulário, uma lista ou outro componente gráfico.

23. Cite 3 tipos de telas em J2ME.

Resposta: `Form`, `List`, `TextBox`.

24. Qual classe controla os eventos dos comandos em J2ME?

Resposta: CommandListener

25. Como se insere dados do usuário em J2ME?

Resposta: através de `TextBox`, `TextField` em `Form`, ou escolha com `ChoiceGroup`.

26. Como tratar exceções em J2ME?

Resposta: Com `try/catch`, como em qualquer aplicação Java. Principais exceções:

IOException: Relacionada a falhas na comunicação, como problemas na rede ou na leitura de arquivos.

RecordStoreException: Ocorre ao manipular dados no RMS, indicando erros de acesso ou corrupção do banco de dados.

SecurityException: Surge quando a aplicação tenta acessar recursos protegidos sem as permissões adequadas

27. O que é `Command` em J2ME?

Resposta: Representa ações como "OK", "Sair", etc.

28. Como adicionar um comando em uma tela?

Resposta: Usando `addCommand()` e registrando um `CommandListener`

29. Qual a diferença entre `Form` e `Canvas`?

Resposta: `Form` é baseado em componentes prontos; `Canvas` permite controle total do layout.

30. Qual método desenha na tela em `Canvas`?

Resposta: `paint(Graphics g)`.

31. O que define a estrutura de um aplicativo J2ME?

Resposta: Uma classe que herda de MIDlet e implementa os métodos do ciclo de vida.

32. Como é o ciclo de vida de um MIDlet?

Resposta: `startApp()` - inicializa a aplicação `pauseApp()` - bloqueia a aplicação - `destroyApp(boolean)` - fecha a aplicação se true.

33. Como se detecta eventos de teclado em `Canvas`?

Resposta: Usando os métodos `keyPressed()`, `keyReleased()`

34. Como se faz a navegação entre telas?

Resposta: Usando `Display.getDisplay(this).setCurrent(tela);`

35. O que é o RMS em J2ME?

Resposta: O armazenamento de dados em J2ME é baseado no Record Management System (RMS), que oferece um banco de dados simples e orientado a registros. Essa estrutura permite a persistência de informações entre sessões de uso da aplicação, sendo essencial para o desenvolvimento de aplicativos móveis que necessitam armazenar dados do usuário.

36. Para que serve o método `repaint()`?

Resposta: Solicita a atualização da tela em um Canvas.

37. Como capturar toques na tela em dispositivos com touch?

Resposta: Usando coordenadas x/y no `pointerPressed()`

38. Quais métodos podem ser sobrescritos em Canvas para controle?

Resposta: `paint()`, `keyPressed()`, `pointerPressed()`, `pointerDragged()`.

39. Qual biblioteca é usada para acesso à rede?

Resposta: javax.microedition.io

40. Quais são os benefícios de usar Form em vez de Canvas?

Resposta: Menor complexidade; mais fácil para formulários e entrada de dados.