RESPOSTAS DO *APLM2023_24-T02-Android-questions*

Nome: **Antonewton Emanuel Lungoge Quima**
Número de matrícula: **20210907**

1. **API Level no Android**:
   Concordo. O API Level é um valor inteiro que identifica a revisão da API do framework de uma versão específica do Android. Ele afeta a portabilidade porque determina quais recursos e funcionalidades estão disponíveis para o desenvolvedor, o que significa que uma aplicação pode não funcionar corretamente em dispositivos com uma versão do Android inferior ao nível de API exigido.

2. **Componentes importantes do Android:**
   Além dos "Serviços", dois componentes principais são:

   1. **Atividade (Activity)**: Representa uma única tela com uma interface de usuário. É onde os elementos visuais da aplicação são exibidos.
   2. **Receptor de Broadcast (Broadcast Receiver):** Permite que o sistema ou outras aplicações enviem mensagens ou notificações para a aplicação, como eventos do sistema ou outras mudanças.

3. **Necessidade do Android em relação aos sistemas operativos de desktop**:
   O Android é projetado para dispositivos móveis, com recursos específicos para gerenciamento eficiente de bateria, recursos limitados de hardware e uma interface tátil. Os sistemas de desktop não são adequados porque são otimizados para dispositivos com maior capacidade de processamento e uma interface baseada em teclado e mouse.

4. **Interface do usuário no componente "Serviço":**
   Não, um "Serviço" no Android não possui uma interface de usuário. Ele é usado para executar operações em segundo plano, sem interagir diretamente com o usuário.

5. **Ficheiro** *AndroidManifest.xml*:
   Sim, o `AndroidManifest.xml` especifica os serviços de hardware e software necessários e as bibliotecas externas que uma aplicação requer. Se algum serviço de software não for declarado, a aplicação não poderá utilizá-lo, resultando em falhas ou comportamentos indesejados.

6. **Mecanismo de broadcast e objeto Intent**:
   Não, o mecanismo de broadcast não especifica uma atividade específica a ser acionada, mas sim um evento que outras partes do sistema podem receber. O objeto `Intent` contém informações sobre a ação a ser realizada e os dados necessários para essa ação.

7. **Iniciar atividades de outras aplicações**:
   Sim, uma atividade pode iniciar outra de uma aplicação diferente usando um `Intent`. Por exemplo, uma aplicação pode abrir o navegador para exibir uma URL.

8. **Pilha de atividades (back stack)**:

Sim, as atividades no Android são organizadas em uma pilha. A lógica segue o princípio "último a entrar, primeiro a sair" (LIFO), onde a atividade que está no topo da pilha é a primeira a ser fechada.

9. **Novo início de tarefa ou tela inicial**:
   Quando o usuário inicia uma nova tarefa ou vai para a tela inicial, a tarefa anterior entra em segundo plano e é pausada, mas permanece na memória para ser retomada posteriormente.

10. **Externalização de recursos como imagens e strings**:
    Sim, o Android recomenda externalizar esses recursos para facilitar a manutenção, suporte a múltiplos idiomas e adaptar a aplicação a diferentes tamanhos de tela e densidades.

11. **Três estados de uma atividade no Android:**
    1. **Resumido (Resumed)**: A atividade está em execução e visível para o usuário.
    2. **Pausado (Paused)**: A atividade ainda está visível, mas não em primeiro plano.
    3. **Parado (Stopped):** A atividade está em segundo plano e não é visível.

12. **Visibilidade e primeiro plano de uma atividade**:
    Uma atividade está visível entre as chamadas *onStart()* e *onStop()*. Está em primeiro plano entre *onResume()* e *onPause()*.

13. **Finalidade do AsyncTask**:
    O *AsyncTask* permite executar operações em segundo plano e atualizar a interface do usuário com os resultados. Por exemplo, pode ser usado para carregar dados de uma API sem bloquear a interface.

14. **Android UI toolkit e thread-safety**:
    Não, o toolkit de interface do Android não é thread-safe. Isso significa que apenas a thread principal (UI thread) pode manipular elementos da interface do usuário.