ANDROID

ANDROID STUDIO

O ANDROID é um sistema operacional projetado para dispositivos móveis. Os seus aplicativos podem ser desenvolvidos usando JAVA, uma das linguagens de programação mais usadas no mundo. Para isso, é necessário um JDK, um SDK e uma IDE chamada ANDROID STUDIO. O JDK proporciona aos programadores desenvolverem aplicativos com a linguagem JAVA, enquanto o SDK proporciona o desenvolvimento nativo para o sistema operacional ANDROID. O ANDROID STUDIO, por sua vez, é o ambiente de desenvolvimento integrado mais usado para essa tarefa.

COMPONENTES

Os aplicativos são compostos por quatro classes de componentes, as atividades, os serviços, os provedores de conteúdo e os receptores de transmissão. As atividades são telas que fornecem uma interface de interação com o aplicativo, os serviços são componentes que são executados em segundo plano. Os provedores de conteúdo controlam o acesso aos dados do aplicativo e os receptores de transmissão recebem mensagens do sistema ANDROID acionando eventos.

ACESSO AOS ELEMENTOS

Durante a compilação de um aplicativo, o código fonte e os recursos são combinados em um arquivo APK. Automaticamente, é gerada uma classe contendo todas as referências de recursos necessários para acessar os elementos da interface. Essa classe, nunca deve ser alterada pelo programador, pois o próprio SDK se encarrega fazer as atualizações necessárias.

LAYOUTS

O ANDROID coloca alguns dos seus recursos em uma pasta chamada RESOURCES. Os LAYOUTS nesse caso são chamados de LAYOUT RESOURCES. Para que os aplicativos possam se adaptar aos diversos tamanhos de telas existentes, usamos o elemento CONSTRAINT LAYOUT. Através dele, podemos criar recursos complexos de interface. Toda a construção da interface pode ser modelada através da ferramenta LAYOUT EDITOR.

ARMAZENAMENTO

O ANDROID oferece algumas opções para que os dados dos aplicativos fiquem salvos de forma persistente, proporcionando a recuperação após o encerramento. SHARED PREFERENCES é uma dessas opções, com ela podemos armazenar dados em uma estrutura de chaves e valores. Para que possamos salvar dados, devemos chamar o método EDIT que retornará um EDITOR. Com o EDITOR criado é possível salvar os dados. Para cada tipo de dado que se deseja salvar, existe um método correspondente. Após os dados serem direcionados para o EDITOR, devemos então realmente efetuar a gravação dos valores no arquivo através do método APPLY.

TRABALHANDO COM NOVAS ATIVIDADES

Ao usarmos os aplicativos instalados em nosso dispositivo móvel, podemos perceber que eles apresentam mais de uma tela de interação. Dispor de várias telas nos proporciona organizar e oferecer mais funções para nossos aplicativos. Para que possamos trabalhar com mais de uma atividade, devemos conhecer o INTENT, que nada mais é do que um elemento de mensagem que pode ser usado para requerer uma ação de outro componente do aplicativo. Em outras palavras, uma atividade é um componente do aplicativo que representa uma tela, e para que ela possa trocar dados e até mesmo ativar outro, é necessário usar um INTENT, que carrega as informações usadas pelo sistema para determinar qual componente será ativado. Nós podemos adicionar essas informações através dos métodos.

FRAGMENTS

O elemento que nos permite separar uma atividade em partes é conhecido como FRAGMENT, e foram criados justamente para combinar essas várias partes de um aplicativo. Com o propósito de aproveitar todo o espaço disponível em uma tela grande, vários deles podem ser mostrados ao mesmo tempo. Podemos acessar o arquivo referente ao recurso de LAYOUT FRAGMENT para adicionar novos elementos, seguindo o mesmo padrão de construção de um LAYOUT para uma atividade. Em seguida, é necessário preparar a atividade para receber todos os FRAGMENTS, e acessar o recurso referente a ela e adicionando um elemento FRAME LAYOUT, que reserva um determinado espaço da tela que receberá um determinado FRAGMENT.

BANCO DE DADOS LOCAL

Os aplicativos são capazes de armazenar um conjunto de dados através de um banco de dados local, todas essas informações devem permanecer disponíveis para consultas. O ANDROID nos oferece uma API chamada SQLITE e uma ferramenta chamada ROOM, para trabalharmos com armazenamento de dados relacionais. O SQLITE é uma poderosa ferramenta, porém ela requer bastante esforço do programador. Para solucionar esse problema, temos a ferramenta ROOM que torna mais simples o trabalho com SQLITE, mas ainda garante o seu aproveitamento.

BANCO DE DADOS NA NUVEM

O FIREBASE é uma plataforma que oferece diversas tecnologias para colaborar com o processo de desenvolvimento de aplicativos. Podemos combinar essas tecnologias para oferecer mais funções para os usuários. Dentre elas podemos destacar o REALTIME DATABASE que consiste em um banco de dados NOSQL hospedado na nuvem. Por ser não relacional, tem otimizações e funções diferentes de um banco de dados relacional. Nele, os dados são armazenados como JSON e sincronizados em tempo real com todos os usuários conectados.

ACESSANDO O REALTIME DATABASE

Para que possamos acessar os dados, devemos recuperar uma instância do banco de dados. Os programadores podem usar o FIREBASE DATABASE, para ter o ponto de entrada para acessar o REALTIME DATABASE. Assim, os mesmos devem chamar pelo método GET INSTANCE. Usando o FIREBASE DATABASE, acessamos o DATABASE REFERENCE, onde podemos ler ou gravar dados.

INSERINDO DADOS NO REALTIME DATABASE

Como o DATABASE REFERENCE, também podemos inserir dados no banco de dados. Usando o método CHILD, criamos ou acessamos referências dentro do REALTIME DATABASE. O método PUSH é responsável por criar uma referência no REALTIME DATABASE, mas, esse método gera uma chave exclusiva para o dado que será armazenado. O método SET VALUE armazena o dado na referência determinada, convertendo automaticamente para JSON.

LENDO DADOS DO REALTIME DATABASE

Como o DATABASE REFERENCE, ainda podemos estabelecer uma referência para lermos dados do REALTIME DATABASE. O método ORDER BY CHILD ordena resultados pelo valor informado no parâmetro. Usando o método ORDER BY KEY podemos ordenar todos os resultados de uma referência de acordo com as suas chaves. O método EQUAL TO recebe como parâmetro uma STRING e reduz os resultados da referência de acordo com o parâmetro informado.

RECUPERANDO DADOS DO REALTIME DATABASE

O CHILD EVENT LISTENER é responsável por controlar mudanças numa referência do banco, e com ele podemos detectar sempre que um objeto for adicionado, atualizado ou removido.

Para usar o CHILD EVENT LISTENER devemos selecionar uma referência e chamar pelo método ADD CHILD EVENT LISTENER. Quando o criamos, o aplicativo receberá os dados sempre que as mudanças nas referências do REALTIME DATABASE ocorrerem. Mas, o usuário poderá sair do aplicativo e navegar para outras telas. Portanto, é importante informar ao aplicativo que pare de ouvir as mudanças do banco de dados. Devemos então, chamar pelo método REMOVE EVENT LISTENER e passar como parâmetro o LISTENER que será removido.