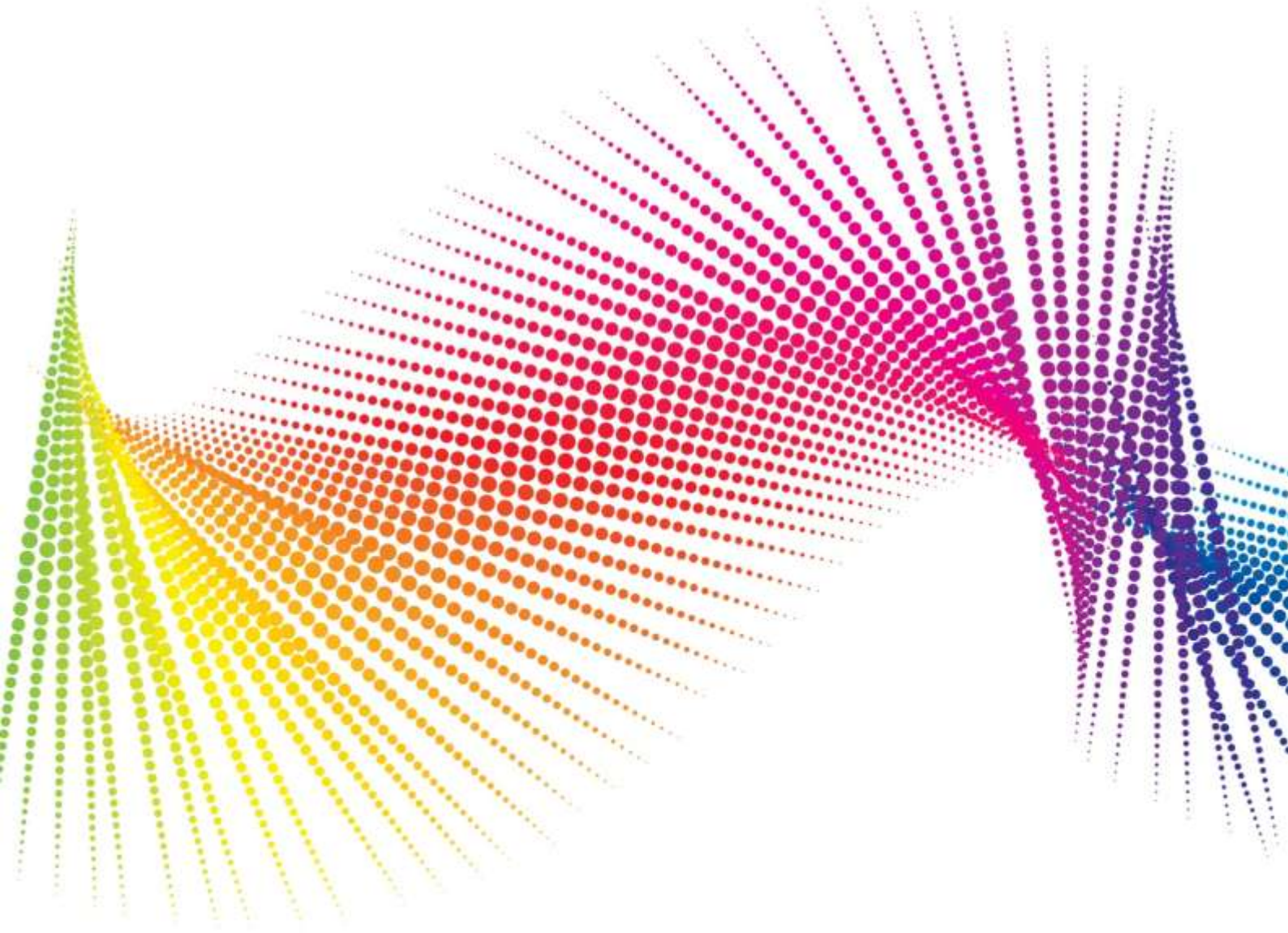
Computação Móvel

Aula 14



Este material é parte integrante da disciplina oferecida pela UNINOVE.
O acesso às atividades, conteúdos multimídia e interativo, encontros virtuais, fóruns de discussão e a comunicação com o professor devem ser feitos diretamente no ambiente virtual de aprendizagem UNINOVE.

Uso consciente do papel.
Cause boa impressão, imprima menos.

Aula 14: Filtros de mensagens no android

Objetivo: Compreender como utilizar os filtros oferecidos pelo sistema android, a fim de obter uma melhora significativa nos acessos às estruturas presentes na construção das aplicações.

Filtros

Para informar ao sistema quais implicit intents podem executar atividades, serviços e receptores de broadcast, e podem ter um ou mais filtros de intents. Cada filtro descreve a capacidade do componente ou um conjunto de intents que o componente está disposto a receber. São filtros de intents que separam um tipo desejado, enquanto descartam os indesejáveis, mas somente implicit intents não requeridos (aquelas não nomeadas na classe de destino). Uma explicit intents é sempre entregue a seu componente, não importa o que ela contenha, o filtro não é consultado. As implicit intents são entregues a um componente apenas se puder passar através de um dos filtros.

Um componente tem filtros separados para cada trabalho que pode realizar, para cada face que pode apresentar para o usuário. Por exemplo, a atividade NoteEditor do aplicativo Pad tem dois filtros - um para iniciar com uma nota específica que o usuário pode visualizar ou editar e outro para começar com uma nota nova, em branco, que o usuário pode preencher e salvar.

Um filtro de intent é uma instância da classe intentfilter. No entanto, desde que o sistema android deva saber sobre os recursos de um componente antes que ele possa iniciá-lo, filtros de intent geralmente não são criados em código Java, mas sim no arquivo de manifesto do aplicativo (AndroidManifest.xml), com elementos <intent-filter>. A única exceção seriam filtros para receptores de broadcast, registrados dinamicamente, chamando Context.registerReceiver; são diretamente criados como objetos intentfilter.

Um filtro tem campos que são similares à ação, dados e campos de categoria de um objeto intent. Uma implicit intents é testada contra o filtro em todas as três áreas.

Para ser entregue ao componente que possui o filtro, ele deve passar por todos os três testes. Se falhar somente uma delas, o sistema android não irá entregá-lo ao componente - pelo menos não com base nos filtros. No entanto, uma vez que um componente pode ter vários filtros de intents, uma intent que não passa através de um dos filtros pode fazê-lo através de outro.

Cada um dos três testes é descrito mais detalhadamente a seguir:

Teste de ação

Um elemento <intent-filter> no arquivo de manifesto lista ações como subelementos <Action>. Por exemplo:

```
<intent-filter . . . >
    <action android:name="com.example.project.SHOW_CURRENT" />
    <action android:name="com.example.project.SHOW_RECENT" />
    <action android:name="com.example.project.SHOW_PENDING" />
    . . .
</intent-filter>
```

Como mostrado, enquanto um objeto intent nomeia uma única ação, um filtro pode listar mais de uma. A lista não pode ser vazia, um filtro deve conter pelo menos um elemento <action> ou ele vai bloquear todas as intents.

Para passar esse teste, a ação especificada no objeto de intent deve corresponder a uma das ações listadas no filtro. Se o objeto ou o filtro não especifica uma ação, os resultados são os seguintes: se o filtro não consegue listar todas as ações, não há nada que corresponda à intent, por isso, todas as intents falham no teste. Por outro lado, um objeto intent que não especifica um recurso, automaticamente passa no teste, desde que o filtro contenha pelo menos uma ação.

Teste da categoria

Um elemento <intent-filter> também lista categorias como subelementos. Por exemplo:

```
<intent-filter . . . >
    <category android:name="android.intent.category.DEFAULT" />
    <category android:name="android.intent.category.BROWSABLE" />
    . . .
</intent-filter>
```

Note que as constantes descritas anteriormente para as ações e para as categorias não são usadas no arquivo de manifesto. São utilizados os valores de cadeia completos. Por exemplo, o "android.intent.category.BROWSABLE" acima corresponde à constante CATEGORY_BROWSABLE, mencionada anteriormente. Da mesma forma, a "android.intent.action.EDIT" corresponde à constante ACTION_EDIT.

Para a intenção de passar no teste de categoria, cada uma no objeto de intent deve corresponder a outra no filtro. O filtro pode listar outras categorias, mas não pode omitir nenhuma das que estão na intent.

Em princípio, portanto, um objeto intent sem categorias deve sempre passar nesse teste, independentemente do que está no filtro. No entanto, com uma exceção, o android trata todas as implicit intents passadas para startActivity() como se contivessem pelo menos uma categoria: "android.intent.category.DEFAULT" (a constante CATEGORY_DEFAULT). Assim, as atividades que estão dispostas a receber as implicit intents devem incluir "android.intent.category.DEFAULT" em seus filtros. Filtros com configurações "android.intent.action.MAIN" e "android.intent.category.LAUNCHER" são exceção. Eles marcam as atividades que começam novas tarefas e que são representados na tela principal. Podem incluir "android.intent .category.DEFAULT " na lista de categorias, mas não precisa.

Teste de dados

Como a ação e as categorias, a especificação para um filtro de dados de intent está contido em um subelemento. Nesses casos, ele pode aparecer várias vezes ou não. Por exemplo:

```
<intent-filter . . . >
    <data android:mimeType="video/mpeg" android:scheme="http" . . . />
    <data android:mimeType="audio/mpeg" android:scheme="http" . . . />
    . . .
</intent-filter>
```

Cada elemento <data> pode especificar um URI e um tipo de dados (MIME tipo de mídia). Existem atributos separados para host, schema, port e path para cada parte da URI: scheme://host:port/path. Por exemplo, no seguinte URI, content://com.example.project:200/folder/subfolder/etc

O scheme é "content", o host é "com.example.project", o port é "200" e o path é "pasta/subpasta/etc". O host e port juntos constituem a autoridade URI; se um host não for especificado, o port é ignorado.

Cada um desses atributos é opcional, mas eles não são independentes um do outro. Para uma autoridade ser significativa, um scheme também deve ser especificado.

Para que uma aplicação permita a passagem de um determinado tipo de dado ela realiza uma comparação da informação com os filtros especificados, a fim de determinar se correspondem a dados validos. Estes podem ser URIs (Uniform Resource Identifier) ou dados no objeto intent que pertençam no URI. As regras são as seguintes:

Um objeto intent que não contém nem URI, nem um tipo de dados, passa no teste somente se o filtro também não especificar qualquer tipo de URI ou dados.

O objeto intent que contém um URI, mas nenhum tipo de dados (um tipo não pode ser inferido a partir da URI), passa no teste somente se o seu URI corresponder a um URI no filtro, e o filtro também não especificar um tipo de dados.

Esse será o único caso de URI como mailto: e tel: que não se referem a dados reais.

Um objeto intent que contém um tipo de dados, mas não uma URI, passa no teste somente se o filtro de lista tiver o mesmo tipo de dados e, da mesma forma, não especificar um URI.

Quando um objeto intent possui dados provenientes de um URI ou algum tipo de dados provenientes de um URI, estes são comparados com os filtros declarados, havendo correspondência entre os dois. Os dados são aceitos pela aplicação, além disso, pode ser necessário que apenas parte do URI passe pelo filtro, o conteúdo ou o arquivo associado ao URI, ou ainda, em outra situação, que não haja nenhum filtro associado ao URI em questão. Nesses casos, presume-se que os dados que não possuem filtros são válidos e podem ser suportados pala aplicação.

Casos comuns

A última regra acima, para o teste de dados, reflete a expectativa de que os componentes são capazes de obter dados locais de um arquivo ou provedor de conteúdo. Portanto, seus filtros podem listar apenas um tipo de dados e não precisa nomear explicitamente o conteúdo: e arquivo: schemas. Esse é um caso típico. Um elemento <dados> como o seguinte, por exemplo, mostra o android que o componente pode obter os dados de imagem a partir de um provedor de conteúdo e exibi-lo:

```
<data android:mimeType="image/*" />
```

Como a maioria dos dados disponíveis é dispensado por provedores de conteúdo, filtros que especificam um tipo de dados, mas não um URI, são, talvez, o mais comum.

Outra configuração comum são filtros com um schema e um tipo de dados. Por exemplo, um elemento <data>, como o seguinte, diz que o componente pode obter dados de vídeo a partir da rede e exibi-lo:

```
<data android:scheme="http" android:type="video/*" />
```

Considere, por exemplo, o que faz o aplicativo do navegador quando o usuário segue um link em uma página web.

Ele primeiro tenta exibir os dados (como poderia se a ligação foi para uma página HTML). Se não puder exibi-los, ele reúne uma implicit intent com o tipo de schema e de dados, e tenta iniciar uma atividade que pode fazer o trabalho.

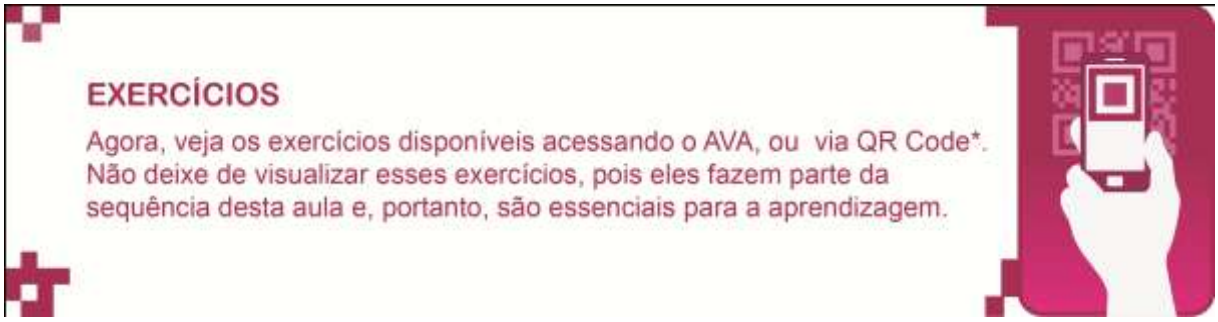
A maioria dos aplicativos também tem uma maneira de recomeçar, sem uma referência a todos os dados particulares. Atividades que podem iniciar as aplicações têm filtros com "android.intent.action.MAIN" especificado como ação. Se eles estão representados no inicio dos aplicativos, também especificam a categoria "android.intent.category.LAUNCHER":

```
<intent-filter . . . >
    <action android:name="android.intent.action.MAIN" />
    <category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
```

Próxima aula

Uma vez que já temos o conhecimento sobre o funcionamento dos principais filtros utilizados na arquitetura android, podemos dar o próximo passo em nosso apredizado. Na próxima aula (aula 14) aprenderemos a manipular a guarda e a manutenção de dados no android.

Agora que você já estudou esta aula, resolva os exercícios e verifique seu conhecimento. Caso fique alguma dúvida, leve a questão ao Fórum e divida-a com seus colegas e professor.



***** O QR Code é um código de barras que armazena links às páginas da web. Utilize o leitor de QR Code de sua preferência para acessar esses links de um celular, tablet ou outro dispositivo com o plugin Flash instalado.

REFERÊNCIAS

LECHETA, Ricard R. *Android*: aprenda a criar aplicações para dispositivos móveis com o android SDK. 2. ed. São Paulo: Editora Novatec, 2010.

ROGERS, Rick; LOMBARDO, John; MEDNIEKS, Zigurd; MEIKE, Blake. *Desenvolvimento de aplicações Android*. São Paulo: Editora Novatec, 2009.