Wykład 12
Broadcast Receivers, Pending Intents

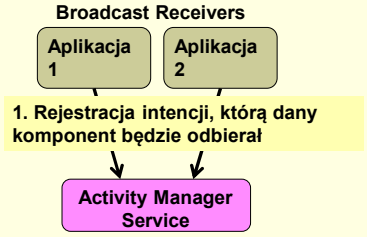
Programowanie Urządzeń Mobilnych

Dr inż. Damian Raczyński

Broadcast Receivers

BroadcastReceiver to komponent, który reaguje na zdarzenia związane z broadcast'em

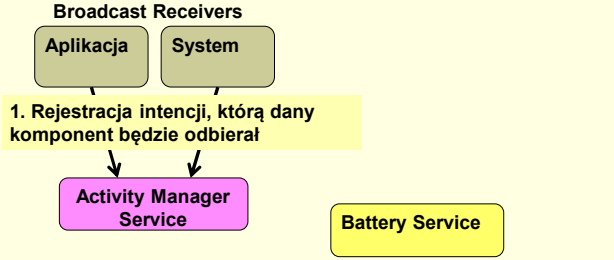
Zdarzenia są zaimplementowane jako intencje (Akcja, Dane, kategoria)



Broadcast Receivers

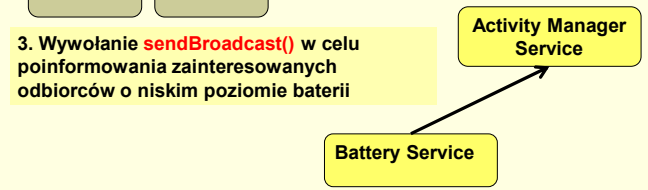
BroadcastReceiver to komponent, który reaguje na zdarzenia związane z broadcast'em

Zdarzenia są zaimplementowane jako intencje



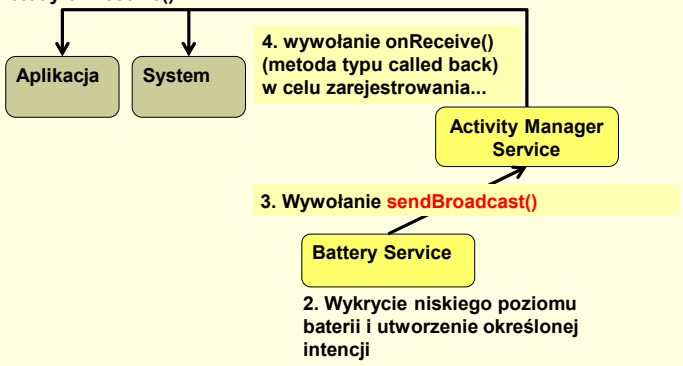
Broadcast Receivers

Zdarzenia są rozgłaszane w całym systemie



Broadcast Receivers

W przypadku, gdy nastąpi określone zdarzenie, Intencje są rozgłaszane do wszystkich „pasujących" odbiorców poprzez ich metody onReceive()



Broadcast Receivers

Odbiorcy komunikatów są ograniczeni w odniesieniu do czynności, które mogą wykonać w systemie.

- Przykładowo nie mogą wyświetlać okna dialogowego,
-Nie można ich bindować do usługi... .

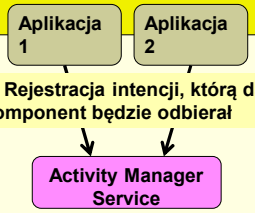
Z założenia komponenty te nie powinny wykonywać swojego kodu przez długi okres czasu (powinny wykonać szybko czynność związaną z reakcją na komunikat).

6

Broadcast Receivers

Istnieją dwa sposoby na zarejestrowanie odbiorców komunikatów:
Statycznie – poprzez publikację odpowiednich informacji w pliku manifestu:

```
<receiver android:name=„App$NazwaBrReceiver" exported=„false">
        <intent-filter>
                <action
android:name=„com.android.phone.ACTION_HANG_UP_ONGOING_CALL" />
                <action
android:name=„com.android.phone.ACTION_SEND_SMS_FROM_NOTIFICATION"/>
        </intent-filter>
</receiver>
```



7

Broadcast Receivers

Istnieją dwa sposoby na zarejestrowanie odbiorców komunikatów:
Dynamicznie – rejestracja z wykorzystaniem metody Context.registerReceiver()

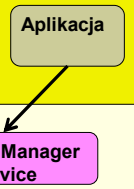
```
final BroadcastReceiver mReceiver = new PhoneAppBroadcastReceiver();

...

IintentFilter intentFilter =
        new IntentFilter(Intent.Action_AIRPLANE_MODE_CHANGED);

...

registerReceiver(mReceiver, intentFilter);
```



8

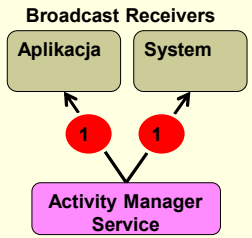
Broadcast Receivers

Kilka mechanizmów rozgłaszania wiadomości

Normal – wysyłany z wykorzystaniem Context.sendBroadcast(), sposób jest całkowicie asynchroniczny

Sposób ten związany jest z zasadą, że wszyscy odbiorcy są powiadamiani o zdarzeniu równolegle (teoretycznie – zależy od liczby rdzeni ...).

Wiadomość jest wysyłana w pętli do wszystkich odbiorców, którzy są zadeklarowani (send, send, send ...)



9

Broadcast Receivers

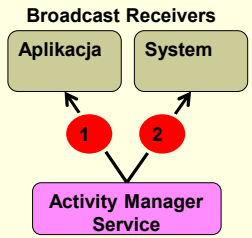
Kilka mechanizmów rozgłaszania wiadomości

Ordered – wysyłanie za pomocą Context.sendOrderedBroadcast(), wiadomość jest dostarczana do jednego odbiorcy w danym czasie

System oczekuje aż pierwsza wiadomość zostanie dostarczona i dopiero wtedy wysyła wiadomość do następnego odbiorcy

1. **Wyślij(1)**
2. **CzekajNaPotwierdzenie(1)**
3. **Wyślij(2)**
4. **czekajNaPotwierdzenie(2)**

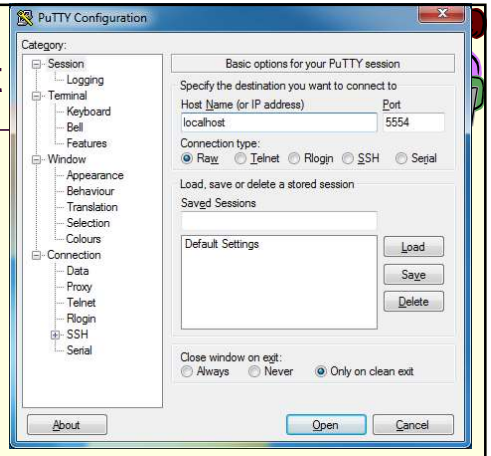
...



10

Broadcast

Prosty przykład:
putty
localhost 5554

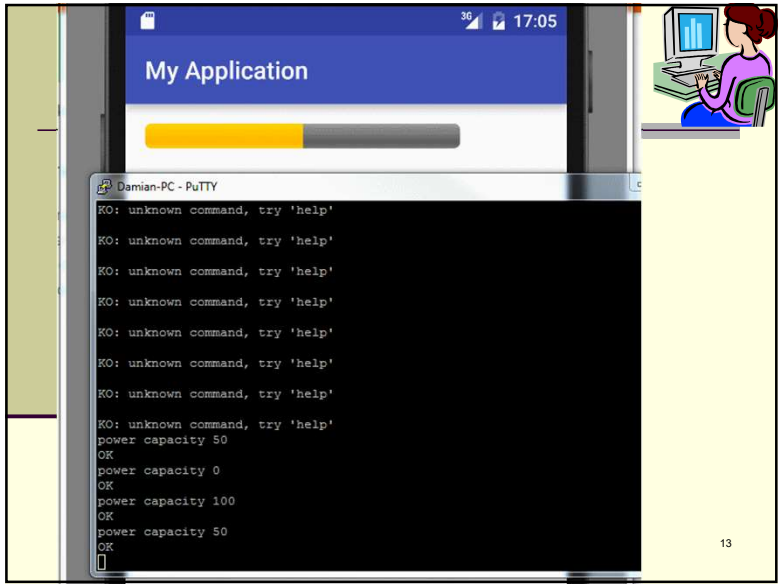


power capacity 100
power capacity 2

11

Broadcast Receivers

```
public class MainActivity extends AppCompatActivity {
    private BroadcastReceiver battery = new BroadcastReceiver() {
        @Override
        public void onReceive(Context c, Intent i) {
            int level = i.getIntExtra("level", 0);
            ProgressBar pb = (ProgressBar)
findViewById(R.id.progressBar);
            pb.setProgress(level);
        }
    };
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        IntentFilter filter=new
IntentFilter(Intent.ACTION_BATTERY_CHANGED);
        registerReceiver(battery, filter);
    }
}
```



Broadcast Receivers

Usługa mierzenia czasu

14

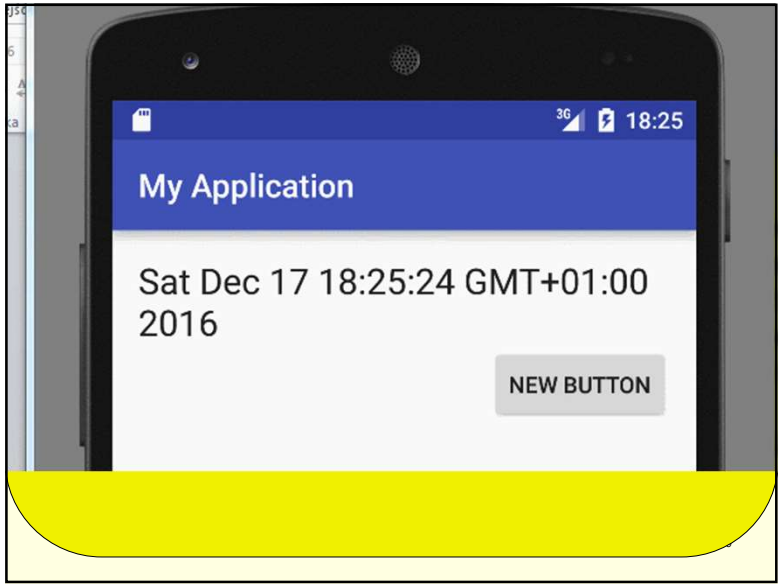
```
public class StoperService extends Service {
    class Stoper extends Thread{
        public long delay;
        @Override
        public void run(){
            while(!isInterrupted()){
                Date czas= new Date();
                String tekst = czas.toString();
                Intent intent = new Intent(Intent.ACTION_ANSWER);
                intent.putExtra("czas", tekst);

intent.setPackage("com.example.administrator.myapplication");
                sendBroadcast(intent);
                try {
                    Thread.sleep(delay);
                } catch (InterruptedException e) {
                    break;
                }           }        }    }
    Stoper stoper;
    public int onStartCommand(Intent intent, int flags, int
startid){
        stoper = new Stoper();
        stoper.delay = intent.getLongExtra("delay", 100);
        stoper.start();
        return START_NOT_STICKY;
    }
    @Override
    public void onDestroy(){
```

```
@Override
    public void onDestroy(){
        super.onDestroy();
        stoper.interrupt();
    }
    @Nullable
    @Override
    public IBinder onBind(Intent intent) {
        return null;
    }
}
```

```
public class StoperReceiver extends BroadcastReceiver {
    public StoperReceiver(TextView textView){
        super();
        t=textView;
    }
    TextView t;
    @Override
    public void onReceive(Context context, Intent intent) {
        String czas= intent.getStringExtra("czas");
        t.setText(czas);
    }
}
```

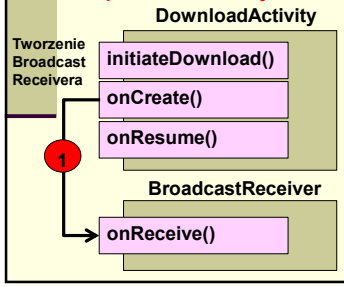
```
protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        IntentFilter filter=new IntentFilter(Intent.ACTION_ANSWER);
        TextView t = (TextView) findViewById(R.id.textView);
        StoperReceiver s = new StoperReceiver(t);
        registerReceiver(s, filter);
        Intent intent = new Intent(MainActivity.this,
StoperService.class);
        intent.putExtra("delay", 1000L);
        startService(intent);
        Button b = (Button) findViewById(R.id.button);
        b.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                Intent intent = new Intent(MainActivity.this,
StoperService.class);
                stopService(intent);
            }
        });
    }
```



Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik

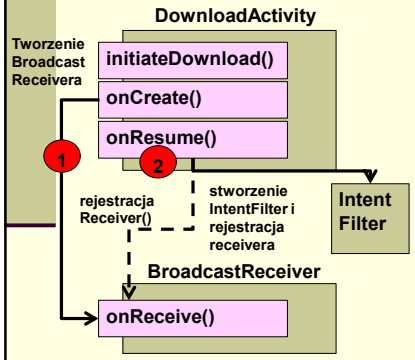
- Aktywność `DownloadActivity` tworzy i rejestruje `BroadcastReceiver` wraz z obiektem `IntentFilter` skonfigurowanym z wykorzystaniem `ACTION_COMPLETE`
- Usługa `DownloadService` rozgłasza `ACTION_COMPLETE` z powrotem do aktywności



20

Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



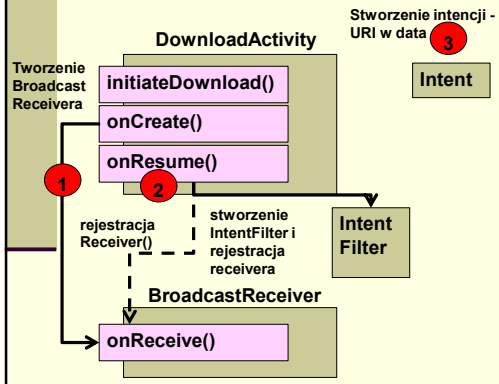
Przy każdym pauzowaniu aktywności odbiorcy są usuwani!!!

Dlatego rejestracja musi odbywać się w metodzie onResume!

21

Broadcast Receivers

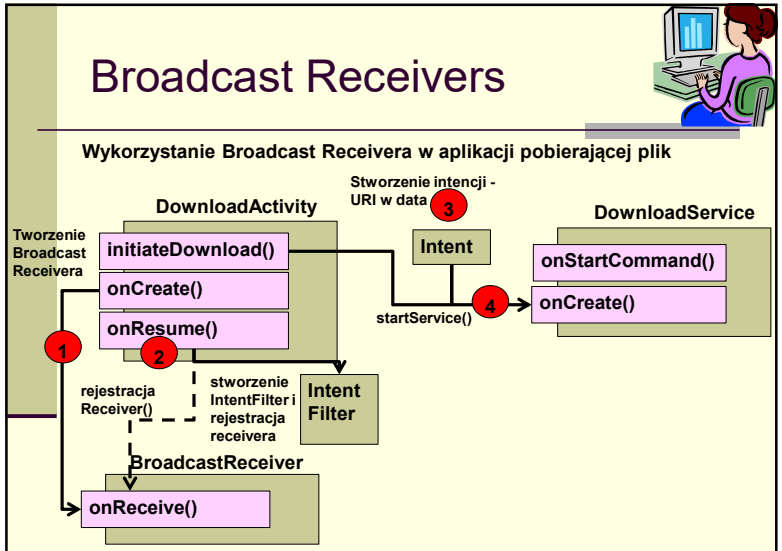
Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



22

Broadcast Receivers

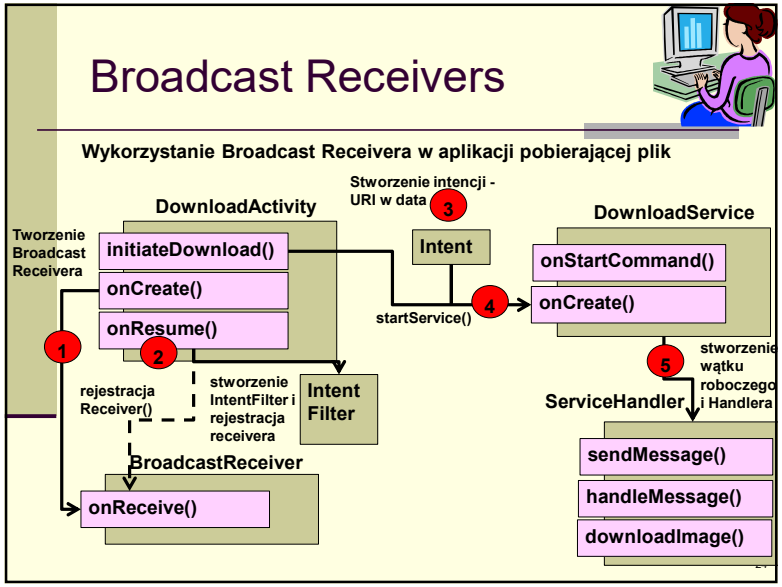
Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



23

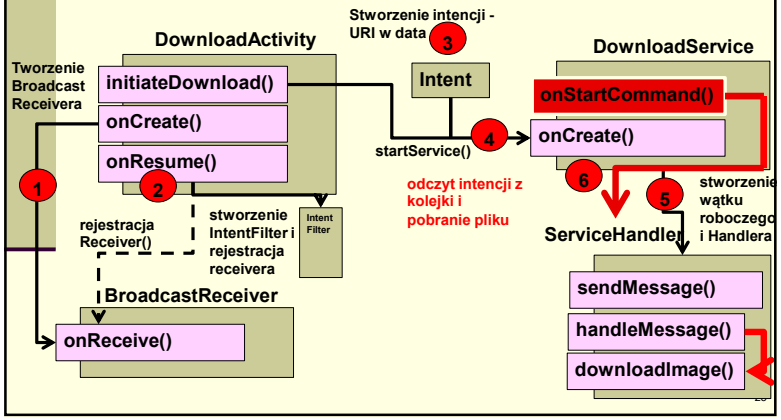
Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



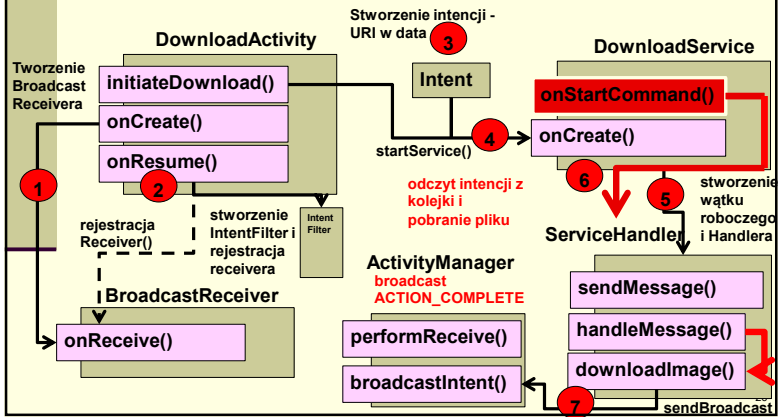
Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



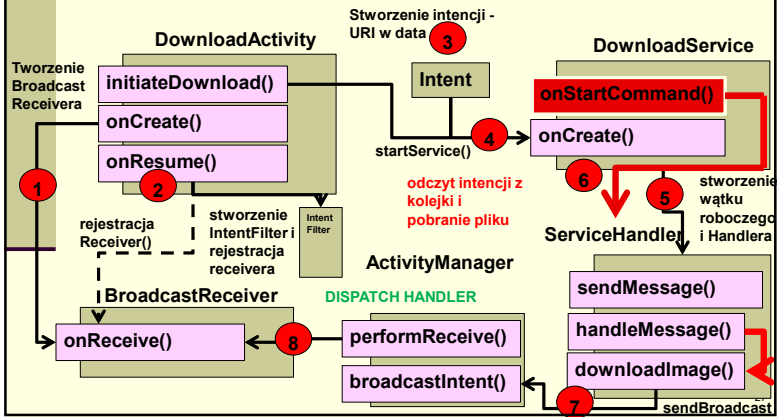
Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



Broadcast Receivers

Wykorzystanie Broadcast Receivera w aplikacji pobierającej plik



Broadcast Receivers

Aktywność zawiera instancję BroadcastReceiver'a:

```
public class DownloadActivity extends Activity{
  private BroadcastReceiver onEvent;
  public void onCreate(Bundle onSavedInstance){
    onEvent= new BroadcastReceiver(){
      public void onReceive(Context context, Intent intent){
        // metoda otrzymuje Intencje wysłane przez sendBroadcast()
        // metoda pobiera obiekt Context – kontekst hanldera
        // broadcast receivera.
        String path= intent.getStringExtra(RESULT_PATH);
        // określa ścieżkę z wykorzystaniem pola Extra Intencji
        displayImage(path);  // Wyświetla obrazek
      }
    };
  }

      ...
}
```

Broadcast Receivers

rejestracja filtru intencji w metodzie onResume:

```
public class DownloadActivity extends Activity{
      ...
  //Metoda rejestruje BroadcastReceiver'a przy uaktywnieniu
aktywności:
  public void onResume(){
    super.onResume();
    IntentFilter filter = new IntentFilter(ACTION_COMPLETE);
    registerReceiver(onEvent, filter);
  }

  //Usunięcie BroadcastReceivera z elementów zarejestrowanych przed
  //przejściem w stan zatrzymania:
  public void onPause(){
    super.onPause();
    unregisterReceiver(onEvent);
  }
      ...
}
```

Broadcast Receivers

Metody inicjalizujące pobieranie pliku

```
public class DownloadActivity extends Activity{
      ...
  public void initiateDownload(View v){
    Intent intent = new Intent(DownloadActivity.this,
                               DownloadService.class);
    ...
    intent.putExtra(PACKAGE_NAME, getPackageName());
    // Przekazujemy w Extra nazwę pakietu w celu uruchomienia
    // DownloadService i utworzenia właściwego broadcastu
    startService(intent);
  }
...
}
```

Broadcast Receivers

Usługa DownloadService

```
public class DownloadService extends Service{
  ...

  private final class ServiceHandler extends Handler{
    ...
    public void downloadImage(Intent intent){
      // Z poprzednich przykładów wędruje tu kod ściągający obrazek
      // pod określoną ścieżkę

      Intent replyIntent = new Intent(ACTION_COMPLETE);
      replyIntent.putExtra(RESULT_PATH, pathname);
      String packageName= intent.getStringExtra(PACKAGE_NAME);
      replyIntent.setPackage(packageName);

      sendBroadcast(replyIntent);
    }
  }
```

Broadcast Receivers

PROBLEMY:

Przestrzeń nazw intencji jest globalna – może to powodować konflikty

registerReceiver() pozwala dowolnej aplikacji wysyłać broadcast'y do odbiorców

W przypadku, gdy odbiorca jest zarejestrowany w pliku manifestu i są do niego przyporządkowane odpowiednie filtry, każda inna aplikacja może wysyłać broadcast'y to aplikacji - aby temu zapobiec można ustawić opcję: android:exported=„false" – wtedy odbiorca działa wyłącznie w ramach naszej aplikacji.

sendBroadcast() umożliwia każdej innej aplikacji otrzymanie wysyłanej wiadomości! Broadcast'y mogą zostać adresowane do pojedynczej aplikacji z wykorzystaniem metody Intent.setPackage();

32