

ANDROID platforma

Internet i Android servisi

RSS primer

Mobilni i distribuirani informacioni sistemi

Mr Bratislav Predić

2012. godina



Internet - RSS



- RSS (RDF Site Symmary)
 - Web feed format za objavljivanje čestih izmena u nekom sadržaju
blog stavke, vremenska prognoza, vesti...
- RSS dokument (web feed ili channel) sadrži tekst, sažetak, datum i autora
- RSS je publish – subscribe koncept
- RSS feed-ovi se često *agregiraju* tako da su dostupni na jednom mestu
- Atom Syndication format i RSS2.0 su XML standardi koji se danas koriste za web feed-ove (Google, iTunes, CNN ...)



Internet – Web feed



- Bitni elementi
 - Channel
 - Item
- Svaki channel može imati proizvoljan broj item-a
- Item je kao članak u novinama
- Elementi item-a
 - title
 - description
 - author
 - category
 - pubDate
 - link
 - comments
 - source
 - ...



Internet – RSS primer



```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
  <channel>
    <title>RSS Title</title>
    <description>This is an example of an RSS feed</description>
    <link>http://www.someexamplesdomain.com/main.html</link>
    <lastBuildDate>Mon, 06 Sep 2010 00:01:00 +0000 </lastBuildDate>
    <pubDate>Mon, 06 Sep 2009 16:45:00 +0000 </pubDate>

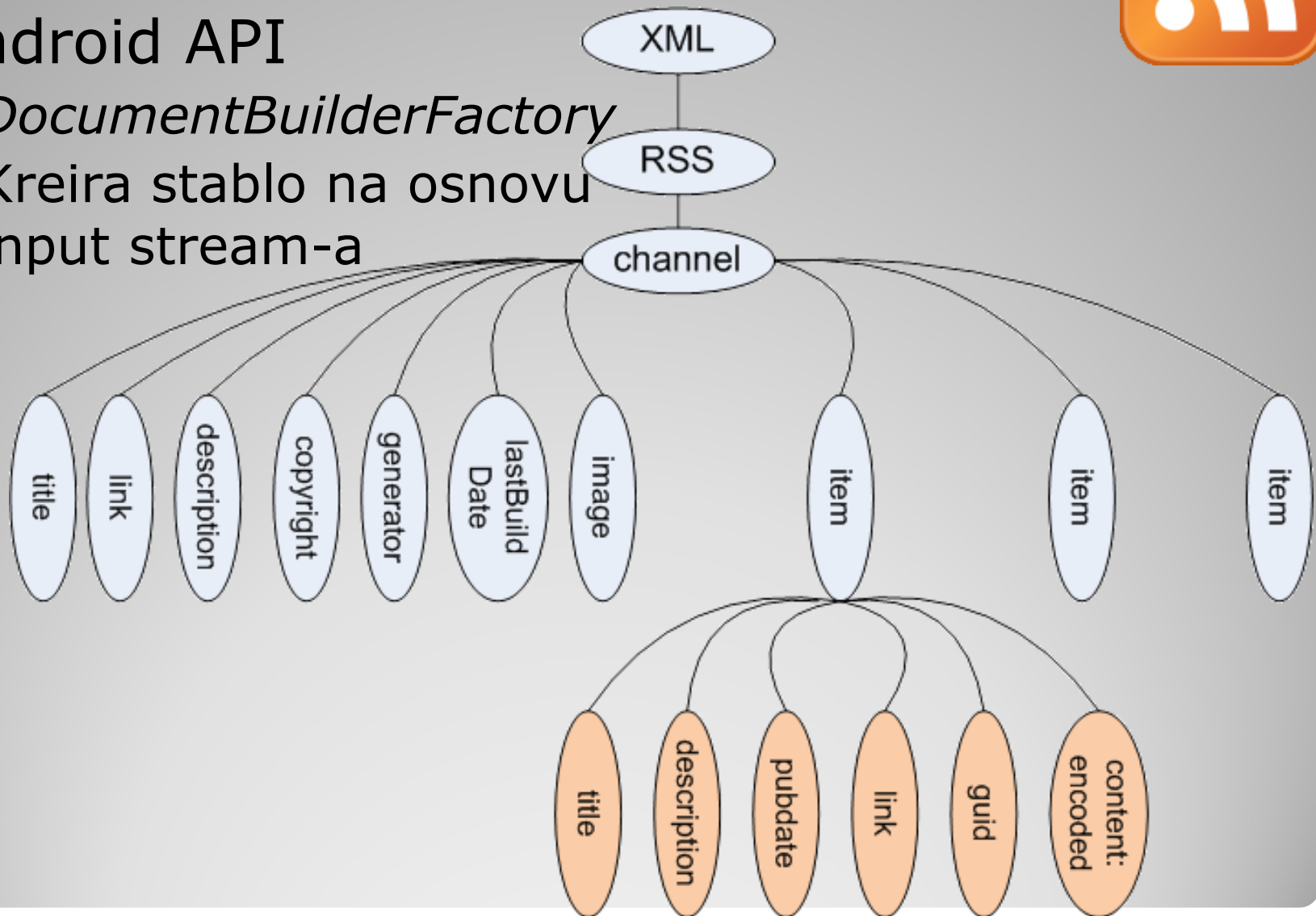
    {
      <item>
        <title>Example entry</title>
        <description>Here is some text containing an interesting
        description of the thing to be described. </description>
        <link>http://www.wikipedia.org/</link>
        <guid>unique string per item</guid>
        <pubDate>Mon, 06 Sep 2009 16:45:00 +0000 </pubDate>
      </item>
    }

  </channel>
</rss>
```

Internet – RSS stablo



- Android API
 - *DocumentBuilderFactory*
 - Kreira stablo na osnovu input stream-a



Internet – Dom i Http klase

- Document Object Model (DOM) se primenjuje na HTML i XML dokumente
- Primer Android DOM manager-a

```
DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();  
DocumentBuilder db = dbf.newDocumentBuilder();  
Document dom= db.parse(someHttpInputStream);
```

- Android za rad sa HTTP protokolom koristi:
 - Standardni *java.net* paket i *URLConnection*
 - Apache *HttpClient* biblioteku
- Za pristup Internet-u je potreban permission
`android.permission.INTERNET`
- Sva mrežna komunikacija obavezno u posebnu nit (Honeycomb ovo forsira)



Internet – Apache HttpClient

- Apache HttpClient biblioteka sadrži
 - *DefaultHttpClient* – standardni HttpClient
 - *AndroidHttpClient*
Implementacija prekonfigurisana za Android
- **AndroidHttpClient**
 - Podržava SSL i GZIP kompresiju podataka
 - Threadsafe je

```
HttpClient client = new DefaultHttpClient();
HttpGet request = new HttpGet("http://www.vogella.com");
HttpResponse response = client.execute(request);
// Get the response
BufferedReader rd = new BufferedReader(new InputStreamReader(
    response.getEntity().getContent()));
String line = "";
while ((line = rd.readLine()) != null) {
    textView.append(line);
}
```

Internet – RSS primer aplikacija

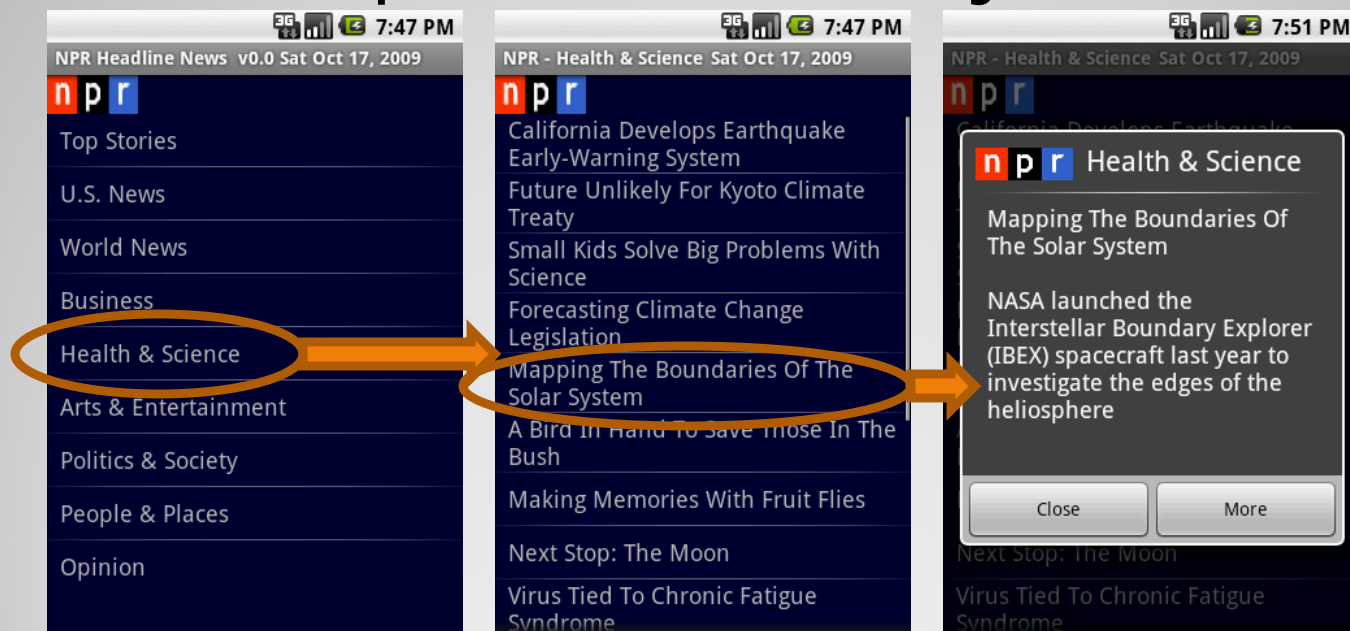
- Primer aplikacija će prikazivati podatke sa jednog kanala broadcaster-a National Public Radio (NPR)



- Jedan feed je na adresi
<http://www.npr.org/rss/rss.php?id=1001>

Internet – RSS primer aplikacija

- Glavne teme (topic-e) prikazujemo u listi iz koje korisnik bira jednu
- Sledeći ekran prikazuje naslove (item-e)
- DialogBox prikazuje sažetak
- Možemo da prikažemo detalje



Internet – RSS primer aplikacija

- Link unutar item-a koristimo kako bi ga prikazali u WebKit aktivnosti
- Bitne karakteristike primera:
 - URL
<http://www.npr.org/rss/rss.php?id=1007>
 - Za mrežnu komunikaciju koristimo HttpComponents API
 - Po preuzimanju feed-a kreiramo dokument sa DOM kretiranim na osnovu XML dokumenta
 - Prolazimo kroz stablo i tražimo *<item>* stavke
 - Iz svakog *item*-a izvlačimo dodatne podatke *tile, description, link, datepublication...*



Internet – RSS primer aplikacija

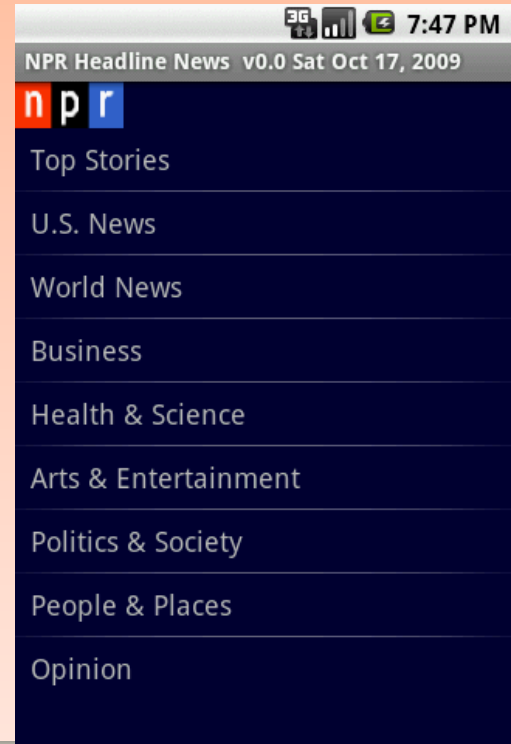
- Preuzimamo RSS feed i parsujemo ga u DOM

```
URL url = new URL(urlAddress);
URLConnection connection;
connection = url.openConnection();
HttpURLConnection httpConnection = (HttpURLConnection) connection;
int responseCode = httpConnection.getResponseCode();
if (responseCode == HttpURLConnection.HTTP_OK) {
    InputStream in = httpConnection.getInputStream();
    DocumentBuilderFactory dbf =
DocumentBuilderFactory.newInstance();
    DocumentBuilder db = dbf.newDocumentBuilder();
    Document dom = db.parse(in);
    Element docEle = dom.getDocumentElement();
    NodeList nl = docEle.getElementsByTagName("item");
    if ((nl != null) && (nl.getLength() > 0)) {
        for (int i = 0; i < nl.getLength(); i++) {
            dissectNode(nl, i);
        } // for
    } // if
}
```

Internet – RSS primer aplikacija

- Main layout

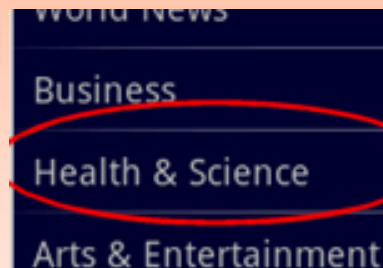
```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:id="@+id/widget28"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    android:background="#ff000033"
    xmlns:android="http://schemas.android.com/apk/res/android"
>
    <ImageView
        android:layout_width="70px"
        android:layout_height="30px"
        android:layout_x="2px"
        android:layout_y="2px"
        android:background="@drawable/logo_npr">
    >
</ImageView>
<ListView
    android:id="@+id/myListView"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
>
</ListView>
</LinearLayout>
```



Internet – RSS primer aplikacija

- Layout za list item

```
<?xml version="1.0" encoding="utf-8"?>
<TextView
    xmlns:android=http://schemas.android.com/apk/res/android
    android:id="@android:id/text1"
    android:layout_width="fill_parent"
    android:layout_height="wrap_content"
    android:gravity="center_vertical"
    android:paddingLeft="10dip"
    android:textSize="18sp"
    android:minHeight="40sp"
/>
```



World News
Business
Health & Science
Arts & Entertainment

- Potrebne su još tri klase
 - Dve aktivnosti za prikaz listi tema i item-a
 - Jedna klasa za memorisanje item-a



Internet – RSS primer aplikacija

```
Public class AndroNPR extends Activity {
    ArrayAdapter<String> aa;
    ListView myListView;
    Context context;
    SingleNewsItem selectedNewsItem;
    String [] myUrlAddress= {
        "http://www.npr.org/rss/rss.php?id=1001",
        "http://www.npr.org/rss/rss.php?id=1003",
        "http://www.npr.org/rss/rss.php?id=1004",
        "http://www.npr.org/rss/rss.php?id=1006",
        "http://www.npr.org/rss/rss.php?id=1007",
        "http://www.npr.org/rss/rss.php?id=1008",
        "http://www.npr.org/rss/rss.php?id=1012",
        "http://www.npr.org/rss/rss.php?id=1021",
        "http://www.npr.org/rss/rss.php?id=1057"
    };
    String [] myUrlCaption= {
        "Top Stories",
        "U.S. News",
        "World News",
        "Business",
        "Health & Science",
        "Arts & Entertainment",
        "Politics & Society",
        "People & Places",
        "Opinion"
    };
};
```



Internet – RSS primer aplikacija

```
String [] myUrlAddress2= new String[myUrlAddressCaption.length];
String [] myUrlCaption2= new String[myUrlAddressCaption.length];
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);
    context= getApplicationContext();
    this.setTitle("NPR Headline News v0.0 "+ niceDate() );
    myListView= (ListView)this.findViewById(R.id.myListView);
    myListView.setOnItemClickListener(new OnItemClickListener() {
        public void onItemClick(AdapterView<?> _av, View _v, int _index, long _id) {
            String urlAddress= myUrlAddress[_index];
            String urlCaption= myUrlCaption[_index];
            Intent NprNewsDetailsIntent= new Intent(AndroNPR.this,
                                                    NprNewsDetails.class);

            Bundle myData= new Bundle();
            myData.putString("urlAddress", urlAddress);
            myData.putString("urlCaption", urlCaption);
            NprNewsDetailsIntent.putExtras(myData);
            startActivity(NprNewsDetailsIntent);
        }
    });
    int layoutID= R.layout.my_simple_list_item_1;
    aa = new ArrayAdapter<String>(this, layoutID, myUrlCaption);
    myListView.setAdapter(aa);
}
```

Internet – RSS primer aplikacija

```
public static String niceDate() {  
    DateFormatSymbols dfs= newDateFormatSymbols();  
    String shortWeekdaysArray[] = dfs.getShortWeekdays();  
    String shortMonthArray[] = dfs.getShortMonths();  
    Calendar cal = Calendar.getInstance(Locale.US);  
    int dd = cal.get(Calendar.DAY_OF_MONTH);  
    int mm = cal.get(Calendar.MONTH);  
    String mmText = shortMonthArray[mm];  
    int yy = cal.get(Calendar.YEAR);  
    int wd = cal.get(Calendar.DAY_OF_WEEK);  
    String wdText= shortWeekdaysArray[wd];  
    return( wdText+ " " + mmText+ " " + dd+ ", " + yy);  
} // niceDate  
} // AndroNPR
```

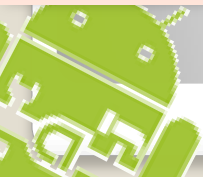
- Statička metoda za lepo formatiranje trenutnog datuma



Internet – RSS primer aplikacija

```
public class NprNewsDetails extends Activity {
    ArrayList<SingleNewsItem> newsList =
        new ArrayList<SingleNewsItem>();
    ArrayAdapter<String> aa;
    ListView myListView;
    String urlAddress= "";
    String urlCaption= "";
    SingleNewsItem selectedNewsItem;
    Context context = getApplication();

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        myListView = (ListView)this.findViewById(R.id.myListView);
        Intent myLocalIntent = getIntent();
        Bundle myBundle = myLocalIntent.getExtras();
        urlAddress = myBundle.getString("urlAddress");
        urlCaption = myBundle.getString("urlCaption");
        String todayStr = AndroNPR.niceDate();
        this.setTitle("NPR -" + urlCaption + " \t" + todayStr);
        myListView = (ListView)this.findViewById(R.id.myListView);
    }
}
```



Internet – RSS primer aplikacija

```
myListView.setOnItemClickListener(new OnItemClickListener() {
    public void onItemClick(AdapterView<?> _av, View _v,
                            int _index, long _id) {
        selectedNewsItem= newsList.get(_index);
        showNiceDialogBox(selectedNewsItem, context);
    }
});
} // onCreate

@Override
protected void onResume() {
    super.onResume();
    try{
        URL url = new URL(urlAddress);
        URLConnection connection = url.openConnection();
        HttpURLConnection httpConnection= (HttpURLConnection) connection;
        int responseCode= httpConnection.getResponseCode();
        if(responseCode == HttpURLConnection.HTTP_OK) {
            InputStream in = httpConnection.getInputStream();
            DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
            DocumentBuilder db = dbf.newDocumentBuilder();
            Document dom= db.parse(in);
            Element docEle= dom.getDocumentElement();
            NodeList nl= docEle.getElementsByTagName("item");
```

Internet – RSS primer aplikacija

```
        if((nl != null) && (nl.getLength() > 0)) {
            for(int i= 0; i< nl.getLength(); i++) {
                dissectNode(nl, i);
            }// for
        }// if
    }// if
    int layoutID = R.layout.my_simple_list_item_1;
    ArrayAdapter<SingleNewsItem> aaNews=
        new ArrayAdapter<SingleNewsItem>(this, layoutID, newsList);
    myListView.setAdapter(aaNews);
} catch (MalformedURLException) {
    e.printStackTrace();
} catch (IOException) {
    e.printStackTrace();
    Toast.makeText(context, "Trouble!!!", 1).show();
} catch (ParserConfigurationException) {
    e.printStackTrace();
} catch (SAXException) {
    e.printStackTrace();
}
}
} // onResume
```



Internet – RSS primer aplikacija

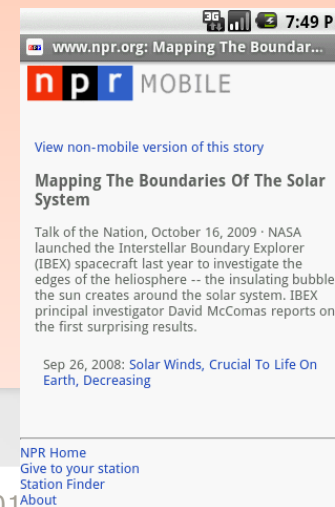
```
public void dissectNode(NodeList nl, int i){
    try{
        Element entry = (Element) nl.item(i);
        Element title = (Element) entry.getElementsByTagName("title").item(0);
        Element description = (Element) entry.getElementsByTagName
                                ("description").item(0);
        Element pubDate = (Element) entry.getElementsByTagName
                                ("pubDate").item(0);
        Element link = (Element) entry.getElementsByTagName
                                ("link").item(0);

        String titleValue = title.getFirstChild().getNodeValue();
        String descriptionValue = description.getFirstChild().getNodeValue();
        String dateValue = pubDate.getFirstChild().getNodeValue();
        String linkValue = link.getFirstChild().getNodeValue();
        SingleNewsItem singleItem= new SingleNewsItem(
            dateValue, titleValue, descriptionValue, linkValue);
        newsList.add(singleItem);
    } catch (DOMException) {
        e.printStackTrace();
    }
} //dissectNode
```



Internet – RSS primer aplikacija

```
public void showNiceDialogBox(SingleNewsItem selectedNewsItem,
                             Context context){
    try{
        final Uri myLink = Uri.parse(selectedNewsItem.getLink());
        AlertDialog.Builder myBuilder= newAlertDialog.Builder(this);
        myBuilder
            .setIcon(R.drawable.logo_npr)
            .setTitle(urlCaption)
            .setMessage( selectedNewsItem.getTitle() + "\n\n"
                        + selectedNewsItem.getDescription() + "\n")
            .setPositiveButton("Close", null)
            .setNegativeButton("More", new OnClickListener() {
                public void onClick(DialogInterface dialog, int whichOne) {
                    Intent webIntent = new Intent( Intent.ACTION_VIEW, myLink);
                    startActivity(webIntent);
                }
            })//setNegativeButton
            .show();
    } catch(Exception e) {
        e.printStackTrace();
    }
    //showNiceDialogBox
} //NprNewsDetails
```



Internet – RSS primer aplikacija

- Klasa za memorisanje item-a

```
public class SingleNewsItem{
    private String pubDate;
    private String title;
    private String description;
    private String link;

    public String getPubDate() { return pubDate; }
    public String getTitle() { return title;}
    public String getDescription(){ return description; }
    public String getLink() { return link; }
    public SingleNewsItem( String _pubDate, String _title, String _description,
                           String _link) {
        pubDate = _pubDate;
        description = _description;
        title = _title;
        link = _link;
    }

    @Override
    public String toString() {
        return title;
    }
}
```

Internet – RSS primer aplikacija

- Manifest aplikacije

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="matos.internet"
    android:versionCode="1"
    android:versionName="1.0.0">
    <application android:icon="@drawable/star_big_on"
        android:label="@string/app_name">
        <activity android:name=".AndroNPR"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <activity android:name=".NprNewsDetails" >
        </activity>
    </application>
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-sdk android:minSdkVersion="4" />
</manifest>
```



Android servisi

- Servis je komponenta aplikacija koje se izvršava u pozadini bez interakcije sa korisnikom i traje neodređeno vreme
- I servisi, kao i ostali objekti aplikacije se izvršavaju u **GLAVNOJ** niti (UI thread) aplikacije
- Ako servis treba da obavlja neku dugu operaciju **MORA** da kreira novu nit za tu operaciju
- Servis se u manifestu označava <service> tag-om
- Servis se startuje sa *Context.startService()*



Android servisi

- Višestruki pozivi *Context.startService()* se ne ugnježdavaju već se samo više puta poziva metoda *onStart()* klase servisa
- Bez obzira koliko puta je “startovan” servis zaustavlja se samo jednim pozivom *Context.stopService()* ili *stopSelf()*
- Kao i sve komponente Android aplikacije i servis ima životni ciklus
- Životni ciklus servisa je kraći u odnosu na životni ciklus aktivnosti (postoje samo tri metode za override)



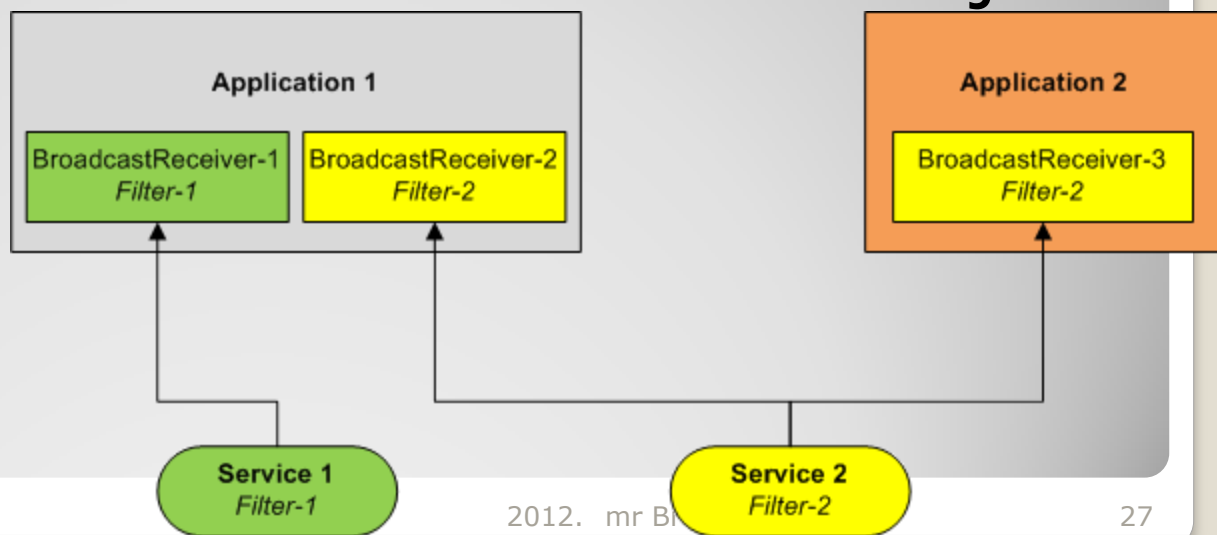
Android servisi

- Životni ciklus Android servisa
 - Tri public metode
 - `void onCreate()`
 - `void onStart(Intent intent)`
 - `void onDestroy()`
- Ceo životni ciklus servisa je između poziva *onCreate()* i *onDestroy()*
- Sva inicijalizacija ide u metodu *onCreate()* kao kod aktivnosti
- Na primer: music player bi u *onCreate()* kreirao nit za puštanje zvučne datoteke, a u *onDesstroy()* bi uništio tu nit



Android servisi

- **Broadcast receiver-i**
- Ovo su klase koje osluškiju Intent-e koji su **emitovani**
- Ne bave se Intent-ima koji su poslati konkretnoj aplikaciji/aktivnosti
- Broadcast Intent-e sistem isporučuje svim registrovanim broadcast receiver-ima koji ih obrađuju **sekvencijalno**



Android servisi – broadcast receiver

- Broadcast receiver se može registrovati kao zainteresovan za neki Intent na dva načina
 - Programski
Context.registerReceiver()
 - Iz manifest-a
<receiver> tagom
- Broadcast receiver ima jedan callback metod

```
void onReceive(Context curContext, Intent broadcastMsg)
```

- Broadcast receiver je aktivan samo do izvršava *onReceive* metod
- Preko parametra *broadcastMsg* ova metoda dobija Intent koji je uhvaćen



Android servisi – broadcast receiver

- Potrebne izmene manifest-a
 - Svaki servis mora da ima `<service>` tag
 - Svaki *BroadcastReceiver* kao nezavisna klasa mora da ima `<receiver>` tag
 - Za svaku od ovih komponenti mogu da se definišu `<intent-filter>` tagovi

```
<application android:icon="@drawable/icon" android:label="@string/app_name">

  {
    <activity android:name=".MyServiceDriver2">
      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>
    </activity>
  }

  {
    <service android:name="MyService2" />
  }

  {
    <receiver android:name="MyBroadcastReceiver">
      {
        <intent-filter>
          <action android:name="matos.action.GOSERVICE2" />
        </intent-filter>
      }
    </receiver>
  }

</application>
```



Android servisi – broadcast receiver

- Dva tipa broadcast-a
 - Normalan
Šalju se sa *Context.sendBroadcast()*, potpuno su asinhroni i **svi** receiver-i se izvršavaju u slučajnom redosledu sekvencijalno
 - Uređeni broadcast
Šalju se sa *Context.sendOrderedBroadcast()*, obrađuju ih redom receiver-i pri čemu svaki receiver može da propagira broadcast dalje ida da prekine lanac
 - Redosled receiver-a se definiše atributom *android:priority* odgovarajućeg *intent-filter* tag-a
 - Svi receiver-i sa istim prioritetom se izvršavaju u proizvoljnom redosledu



Android servisi – tipični koraci

- Česta situacija je da aktivnost interaguje sa servisom, tipični koraci:
- Startovanje servisa

```
Intent intentMyService = new Intent(this, MyService3.class);  
Service myService = startService(intentMyService);
```

- Definisanje intent filtra i receiver-a

```
IntentFilter mainFilter = new  
    IntentFilter("matos.action.GOSERVICE3");  
BroadcastReceiver receiver = new MyMainLocalReceiver();  
registerReceiver(receiver, mainFilter);
```

- Implementirati receiver i metodu

```
public void onReceive(Context localContext, Intent callerIntent)
```



Android servisi – tipični koraci

- Unutar servisa kreira intent

```
Intent myFilteredResponse = new  
    Intent("matos.action.GOSERVICE3");
```

- Dodaje podatke kao extras

```
Object msg = new Integer(13);  
myFilteredResponse.putExtra("myServiceData", msg);
```

- Šalje intent svim registrovanim receiver-ima

```
sendBroadcast(myFilteredResponse);
```

- Aktivnost zaustavlja servis

```
stopService(newIntent(intentMyService) );
```

- Deregistrujemo receiver

```
unregisterReceiver(receiver);
```


Primer servisa sa komunikacijom

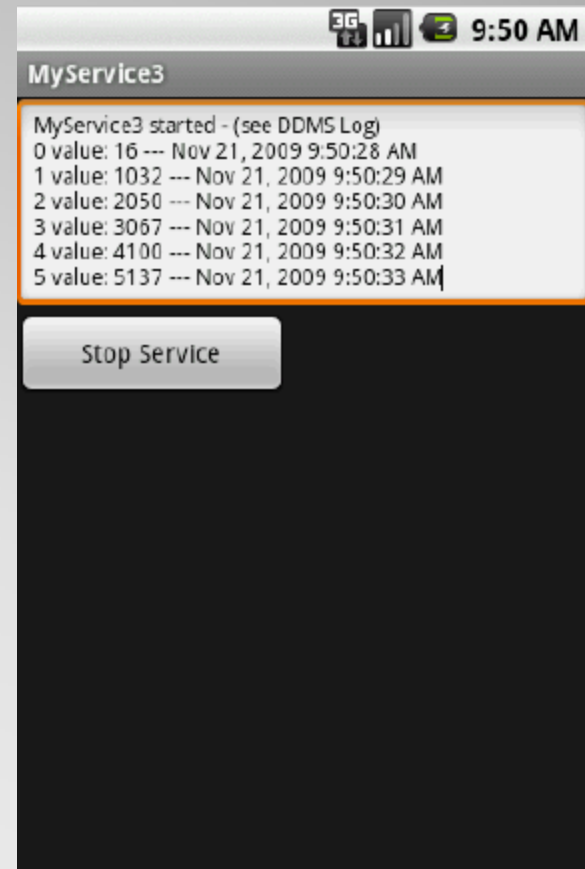
- Glavna aktivnost startuje servis i registruje receiver
- Servis obavlja dugačak zadatak, pa ga izvršavamo u posebnom procesu
- Kada se završi zadatak servis dodaje poruku u intent
- Kreirani intent servis šalje korišćenjem nekog filtra: *matos.action.GOSERVICE3*
- BroadcastReceiver definisan unutar aktivnosti hvata intent i prikazuje sadržaj poruke
- Korisnik kroz glavnu aktivnost zaustavlja servis



Primer servisa sa komunikacijom

- Layout

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout
    android:id="@+id/widget32"
    android:layout_width="fill_parent"
    android:layout_height="fill_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
>
    <EditText
        android:id="@+id/txtMsg"
        android:layout_width="fill_parent"
        android:layout_height="120px"
        android:textSize="12sp"
    >
    </EditText>
    <Button
        android:id="@+id/btnStopService"
        android:layout_width="151px"
        android:layout_height="wrap_content"
        android:text="Stop Service"
    >
    </Button>
</LinearLayout>
```



Primer servisa sa komunikacijom

- Manifest

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android=http://schemas.android.com/apk/res/android
    package="cis493.demos"
    android:versionCode="1"
    android:versionName="1.0.0">
    <uses-sdk android:minSdkVersion="4"></uses-sdk>
    <application android:icon="@drawable/icon"
        android:label="@string/app_name">
        <activity android:name=".MyServiceDriver3"
            android:label="@string/app_name">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category
                    android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
        <service android:name="MyService3">
        </service>
    </application>
</manifest>
```

Primer servisa sa komunikacijom

- Glavna aktivnost

```
Public class MyServiceDriver3 extends Activity {
    TextViewtxtMsg;
    Button btnStopService;
    ComponentName service;
    Intent intentMyService;
    BroadcastReceiver receiver;

    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.main);
        txtMsg= (TextView) findViewById(R.id.txtMsg);
        intentMyService= new Intent(this, MyService3.class);
        service = startService(intentMyService);
        txtMsg.setText("MyService3 started -(see DDMS Log)");
        btnStopService = (Button) findViewById(R.id.btnStopService);
        btnStopService.setOnClickListener(new OnClickListener() {
            public void onClick(View v) {
                try{
                    stopService(new Intent(intentMyService) );
                    txtMsg.setText("After stopingService: \n"+
                                   service.getClassName());
                } catch(Exception e) {e.printStackTrace();}
            }
        });
    }
}
```

Primer servisa sa komunikacijom

- Glavna aktivnost (cont.)

```
// register & define filter for local listener
IntentFilter mainFilter = new
    IntentFilter("matos.action.GOSERVICE3");
receiver = new MyMainLocalReceiver();
registerReceiver(receiver, mainFilter);
} // onCreate

@Override
protected void onDestroy() {
    super.onDestroy();
    try{
        stopService(intentMyService);
        unregisterReceiver(receiver);
    } catch (Exception e) {
        Log.e("MAIN3-DESTROY>>>", e.getMessage() );
    }
    Log.e("MAIN3-DESTROY>>>", "Adios");
} // onDestroy
```

Primer servisa sa komunikacijom

- Glavna aktivnost (receiver)

```
public class MyMainLocalReceiver extends BroadcastReceiver{
    @Override
    public void onReceive(Context localContext, Intent callerIntent) {
        String serviceData = callerIntent.getStringExtra("serviceData");
        Log.e("MAIN>>>", serviceData+ " -receiving data "
                + SystemClock.elapsedRealtime() );
        String now = "\n"+ serviceData+ " ---"
                + new Date().toLocaleString();
        txtMsg.append(now);
    }
} //MyMainLocalReceiver
} //MyServiceDriver4
```

- Treba kreirati i sam servis



Primer servisa sa komunikacijom

- Servis

```
Public class MyService3 extends Service {  
    boolean isRunning= true;  
  
    @Override  
    public IBinder onBind(Intent arg0) {  
        return null;  
    }  
  
    @Override  
    public void onCreate() {  
        super.onCreate();  
    }  
  
    ....  
}
```



Primer servisa sa komunikacijom

```
@Override
public void onStart(Intent intent, int startId) {
    super.onStart(intent, startId);
    Log.e("<<MyService3-onStart>>", "I am alive-3!");
    // Dug zadatak ide u poseban thread
    Thread triggerService = new Thread ( new Runnable(){
        long startingTime = System.currentTimeMillis();
        long tics= 0;
        public void run() {
            for(int i=0; (i< 120) & isRunning; i++) { //max 10 minuta
                try{
                    tics = System.currentTimeMillis() - startingTime;
                    Intent myFilteredResponse =
                        new Intent("matos.action.GOSERVICE3");
                    String msg = i + " value: "+ tics;
                    myFilteredResponse.putExtra("serviceData", msg);
                    sendBroadcast(myFilteredResponse);
                    Thread.sleep(1000);
                } catch(Exception e) { e.printStackTrace(); }
            } //for
        } //run
    });
    triggerService.start();
} //onStart
```


Primer servisa sa komunikacijom

- Uništavanje servisa

```
@Override
public void onDestroy() {
    super.onDestroy();
    Log.e("<<MyService3-onDestroy>>", "I am dead-3");
    isRunning= false;
} //onDestroy
} //MyService3
```

- Preko bool promenljive zaustavljamo thread koji je kreiran u servisu

