

# Föreläsning 2

- Activity
- Fragment
- Referenser till Views
- Händelsehantering
- Workshop

Design och implementering av program med

- \* Activity för att initiera program och hålla Fragment-objekt
- \* UI och händelsehantering i Fragment-objekt
- \* Controller-klass med logik

# Viktiga UI-klasser

## **View**

Superklass för alla UI-komponenter, synliga (controls / widgets) och osynliga som layouts.

## **ViewGroup**

Subklass till View. Kan innehålla flera Views. Ett exempel är LinearLayout

## **Fragment**

Ungefär som en JPanel i java. Innehåller ett eget UI med komponenter och händelsehantering.

## **Activity**

Ungefär som en JFrame i java. Representerar ett fönster på skärmen.

# Activity

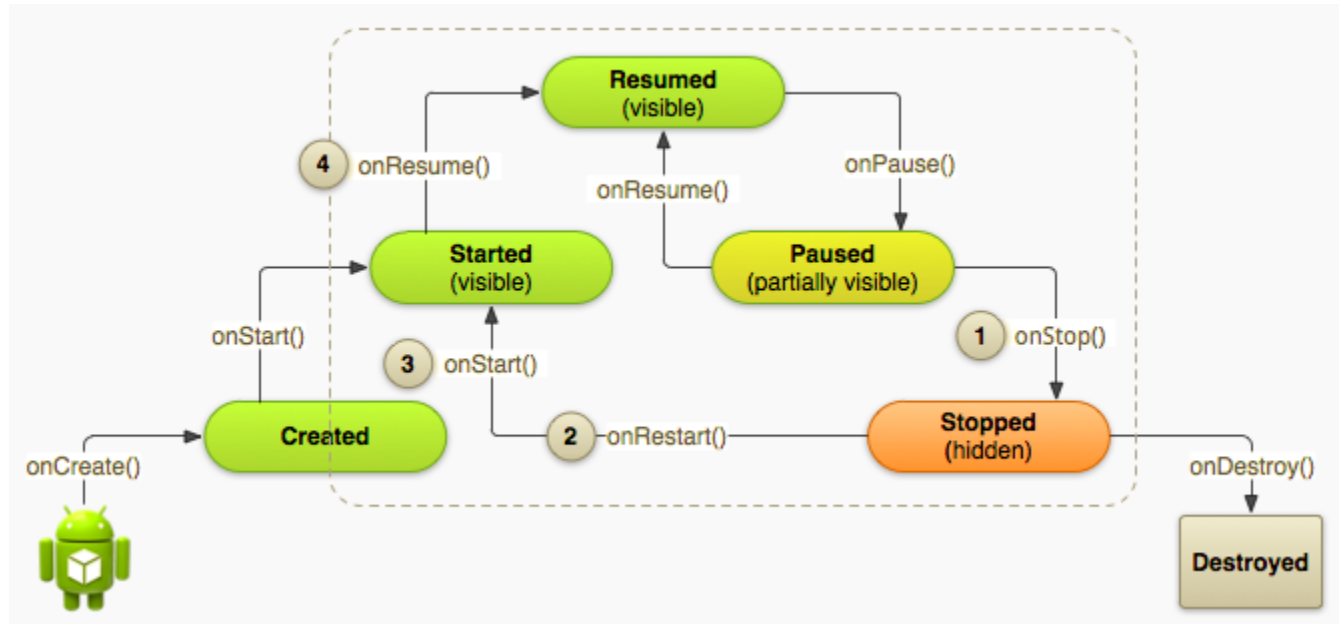
- Ett fönster i applikationen
- Applikationen startar i en Activity
- Innehåller UI ofta bestående av Fragment
- Innehåller referens till resurser
- Klassen ska ärva Activity / AppCompatActivity

```
public class App extends Activity {  
  
}
```

- En Activity ska registreras i Manifestet

```
:  
<application ...>  
:  
  <activity ...>  
    <intent-filter>  
      :  
      </intent-filter>  
    </activity>  
  :  
</application>
```

# Activity - lifecycle



En Activity kan vara i tre olika states:

- **Active** - synlig  
onCreate(), onStart() och onResume() är anropade
- **Paused** – delvis synlig  
Efter Active, då onPause() anropats
- **Stopped** – osynlig  
Efter Paused, då onStop() anropats

# Activity - lifecycle

## **onCreate(Bundle savedInstanceState)**

Initialisera Activity och skapa UI

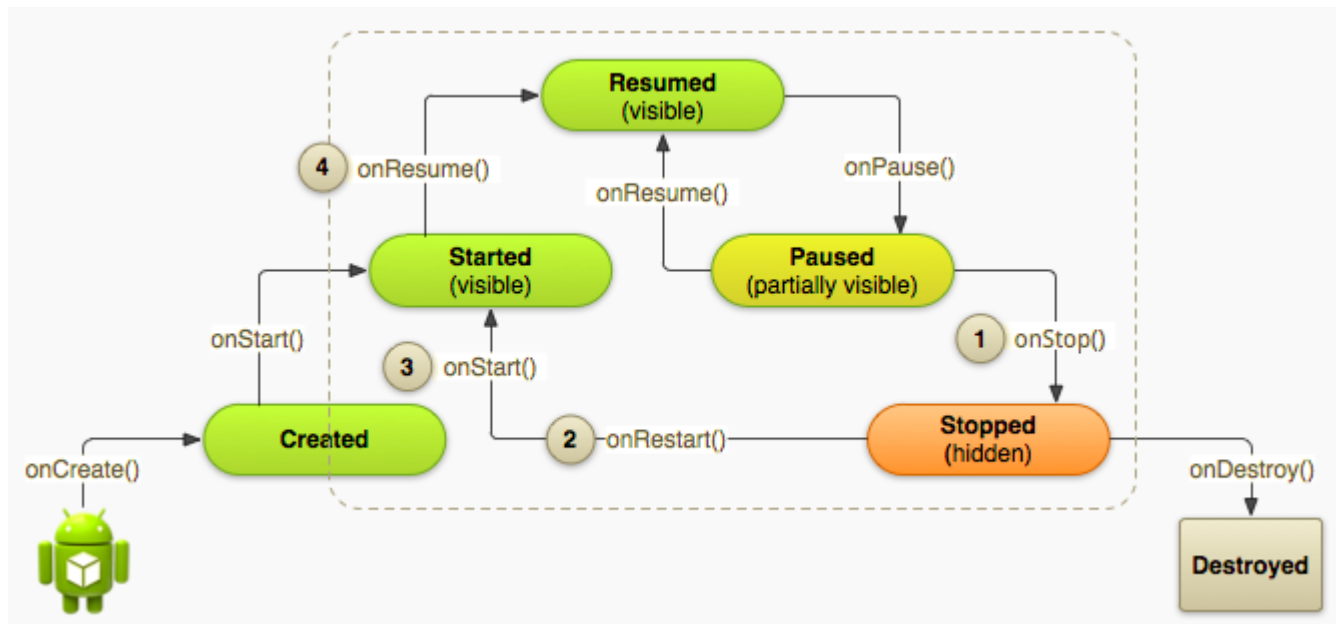
Eventuellt aktivera resurser som avaktiverats vid avbrott av Activity (savedInstanceState!=null)

## **onRestart(), onStart()**

Aktivera resurser som avaktiverats av onStop()

## **onResume()**

Aktivera resurser som avaktiverats i onPause()



# Activity - lifecycle

## **onSaveInstanceState(Bundle savedInstanceState)**

Spara undan representation av Ulets state i en Bundle-instans. Denna Bundle är argument till bl.a. onCreate då activityn instansieras på nytt.

## **onPause()**

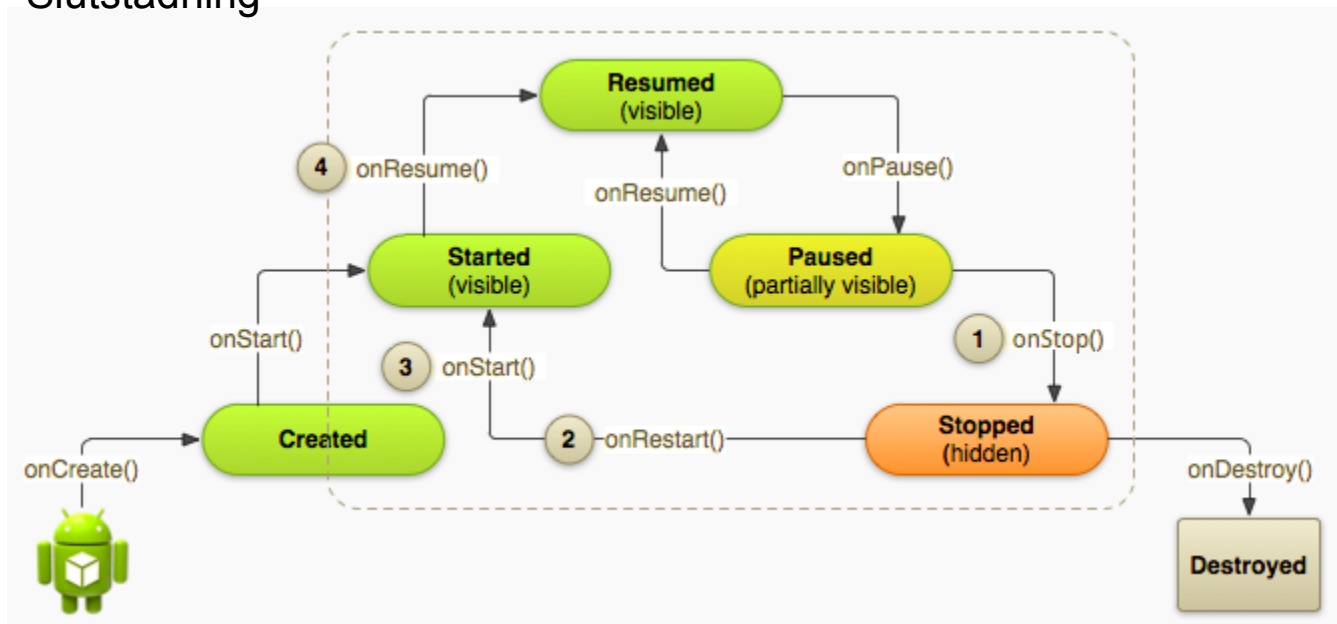
Avaktivera resurser som ej behövs. Spara nödvändig data.  
Snabb exekvering krävs!

## **onStop()**

Avaktivera resurser som ej avaktiverats i onPause

## **onDestroy()**

Slutstädning



# Activity – UI, Views och händelser

- Ett fönster i applikationen
- Applikationen startar i en Aktivitet
- Innehåller appens UI vilket ofta består av ett eller flera Fragment

```
public class AnActivity extends Activity {  
    private Button btnHello;  
    private TextView tvInfo;  
  
    protected void onCreate(Bundle savedInstanceState) {  
        super.onCreate(savedInstanceState);  
        setContentView(R.layout.activity_main);  
        initializeComponents();  
        registerListeners();  
    }  
  
    private void initializeComponents() {  
        tvInfo = (TextView)findViewById(R.id.tvInfo);  
        btnHello = (Button)findViewById(R.id.btnHello);  
    }  
}
```

- onCreate, onRestart, onStart och onResume inleds med ett anrop till superklassens motsvarighet av metoden.
- onSaveInstanceState, onPause, onStop och onDestroy avslutas med ett anrop till superklassen motsvarighet av metod.

# Activity – UI, Views och händelser

- Ett fönster i applikationen
- Applikationen startar i en Activity
- Innehåller appens UI vilket ofta består av ett eller flera Fragment

```
public class AnActivity extends Activity {
    :
    private void registerListeners() {
        btnHello.setOnClickListener(new BL());
    }

    private class BL implements OnClickListener {
        int index=0;
        String[] info = {"Hello","GoodBye"};

        public void onClick(View v) {
            index = (index+1) % 2; // index = 0,1,0,1,0,1 osv
            tvInfo.setText(info[index]);
        }
    }
}
```



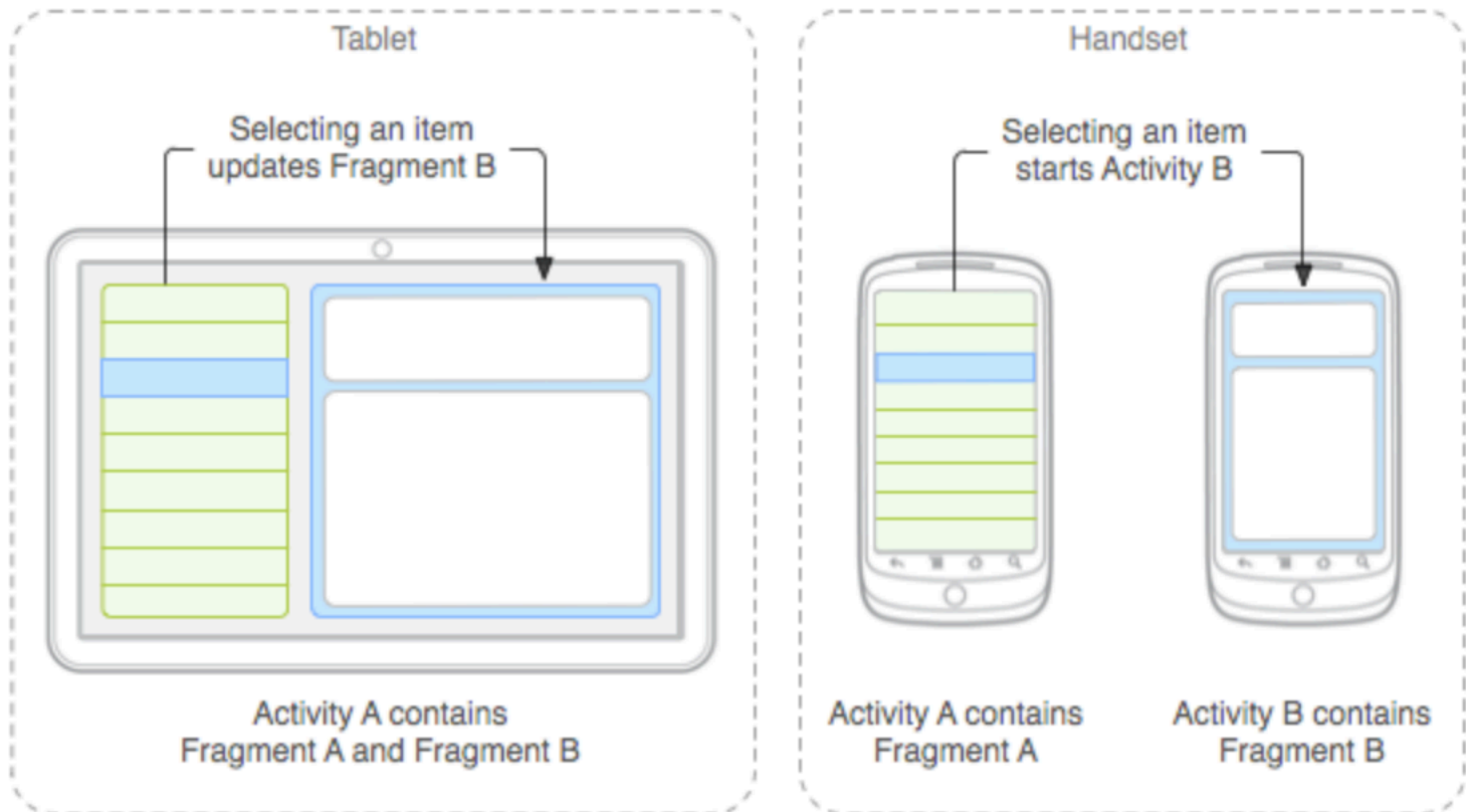
# Fragment

- Ett fragment representerar ofta en del av Uiet i en Activity.
- Flera fragment kan kombineras för att bygga upp Uiet i en Activity. Samma Fragment-klass kan användas i olika Activities
- Som en panel i java
- Har eget UI och händelsehantering
- Bör vara väl inkapslad
- Bör ej innehålla logik
- Har tillgång till referens till sin Activity, metoden getActivity()
- Klassen ska ärva Fragment eller subclass till Fragment

```
public class Frag extends Fragment {  
  
}
```

# Fragment

- Fragment kan användas för att stödja olika skärmstorlekar..
- En stor skärm kan kanske visa flera fragment medan en mindre endast kan visa ett fragment åt gången.

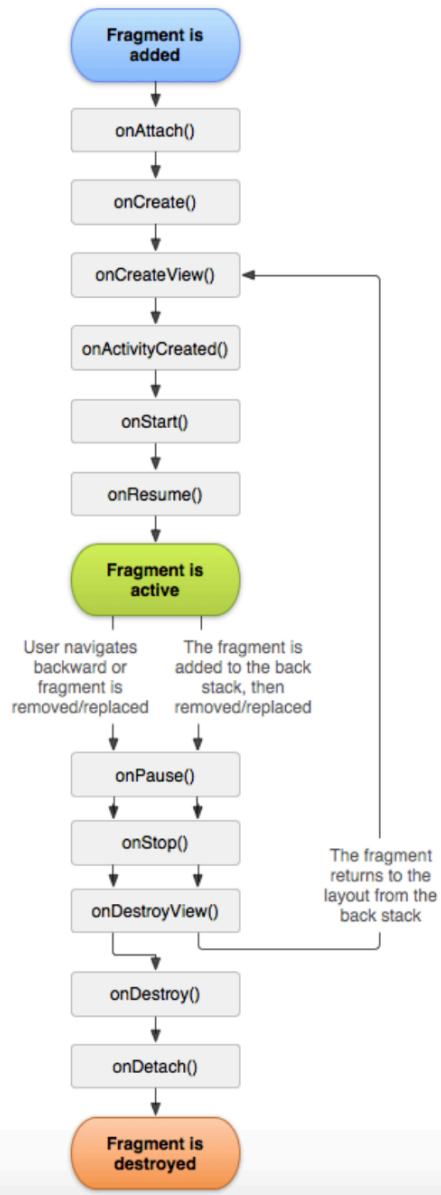


# Fragment - lifecycle

En Fragment har metoder som motsvarar metoderna i Activity-klassen. Dessutom tillkommer det några metoder.

- **onAttach(Activity activity)**  
anropas då fragmentet är kopplat till activityn
- **onCreate(Bundle savedInstanceState)**  
viss initialisering av fragmentet
- **public View onCreateView(LayoutInflater inflater, ViewGroup container, Bundle savedInstanceState)**  
skapa och returnerar Ulet som är associerat till fragmentet. Denna metod finns endast i synliga fragments.
- **onStart()** och **onResume()**  
Metoder lämpliga för att aktivera resurser och för sena initialiseringar.

Superklassens motsvarighet anropas i början av metoderna. Gäller ej onCreateView!

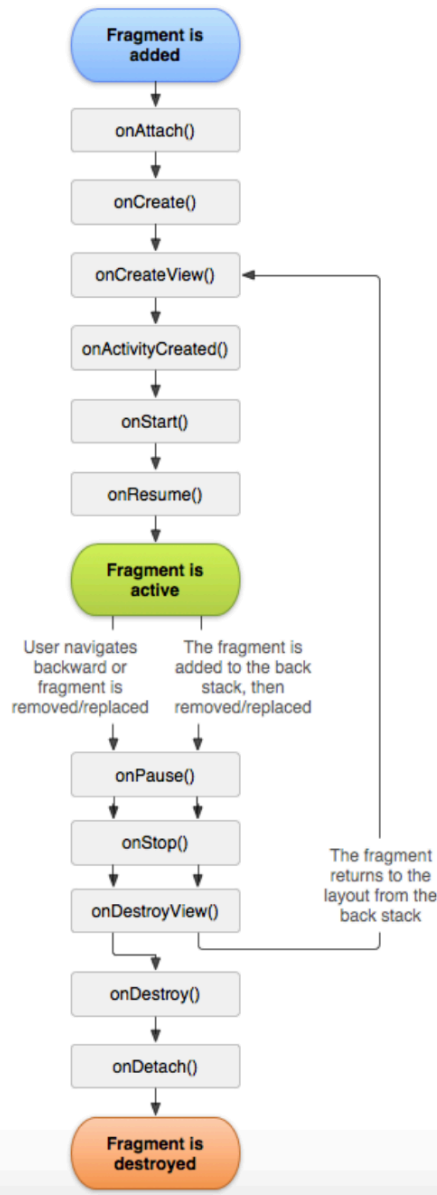


# Fragment - lifecycle

En Fragment har metoder som motsvarar metoderna i Activity-klassen.

- **onPause()**  
Avaktivera resurser
- **onSaveInstanceState(Bundle savedInstanceState)**  
Spara undan fragmentets dynamiska tillstånd i Bundle-instansen så att fragmentet ska kunna återskapas av ny aktivitet. Bundle-instansen levereras till bl.a. onCreate och onCreateView i det nya fragmentet.
- **onStop()**  
Avaktivera resurser som ej avaktiverats i onPause
- **onDestroyView()**  
fragmentet kan avaktivera resurser kopplade till dess UI
- **onDestroy()**  
slutstädning
- **onDetach()**  
skiljer fragment från activityn

Superklassens motsvarighet anropas i slutet av metoderna.



# Fragment – i en Activity

Fragment kan placeras i en layout med fragment-taggen. Så här kan Activity-klassens layoutfil se ut:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/
android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical" >

    <fragment
        android:id="@+id/viewer_fragment"
        android:name="se.mah.tsroax.staticfragment.ViewerFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

    <fragment
        android:id="@+id/input_fragment"
        android:name="se.mah.tsroax.staticfragment.InputFragment"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" />

</LinearLayout>
```

# Fragment – UI, Views och händelser

- I *onCreateView* skapas fragmentets UI

```
public class AFragment extends Fragment {
    private Button btnHello;
    private TextView tvInfo;

    public View onCreateView(LayoutInflater inflater,
                             ViewGroup container,
                             Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.viewer,
                                     container, false);

        initializeComponents(view);
        initializeResources();
        return view;
    }

    private void initializeComponents(View view) {
        tvInfo = (TextView)view.findViewById(R.id.tvInfo);
        btnHello = (Button)view.findViewById(R.id.btnHello);
    }
}
```

# Fragment – UI, Views och händelser

- Då nödvändiga referenser finns kan t.ex. lyssnare kopplas

```
public class AFragment extends Fragment {
    private Controller controller;

    :
    private void registerListeners() {
        btnHello.setOnClickListener(new BL());
    }

    public void setInfo(String str) {
        tvInfo.setText(str);
    }

    private class BL implements OnClickListener {
        public void onClick(View v) {
            controller.newInfo();
        }
    }
}
```

# FragmentManager

Activityklassen kan hantera sina Fragment med ett FragmentManager-objektet.

- Det går att få referens till olika Fragment som finns i Activityn.
- Det går att ändra fragment som är synligt i en container

```
FragmentManager fm = getSupportFragmentManager();
```

```
ViewerFragment viewer =  
    (ViewerFragment) fm.findFragmentById(R.id.viewer_fragment);  
InputFragment input =  
    (InputFragment) fm.findFragmentById(R.id.input_fragment);
```

```
FragmentTransaction fragTransaction = fragmentManager.beginTransaction();
```

```
// bl.a. add, remove och replace här
```

```
fragTransaction.commit();
```



# Workshop

Ett sten, sax och påse-spel bestående av

- En Activity-klass (MainActivity)
- Två Fragment-klasser (InputFragment och ViewerFragment)
- En controller-klass (RPSController)
- En datorspelare-klass (RPSPlayer)

