Föreläsning 3

- Ärva en View
- ListView och ArrayAdapter
- Anpassad rad i ListView
- Fragment i statisk layout
- Fragment i dynamisk layout
- Workshop
  Skifta synligt Fragment dynamiskt under exekvering

Ärva en View

Det går utmärkt att ärva en View, t.ex. en TextView. I exemplet konstrueras en TextView-komponent som alltid visar texten som rövarspråket.

```
public class TRLTextView extends TextView {
    private static String consonants = "bcdfghjklmnpqrstvwxzBCDFGHJKLMNPQRSTVWXZ";

    public TRLTextView(Context context) {
        super(context);
    }

    public TRLTextView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }

    public TRLTextView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
    }

    public void setText(CharSequence text, TextView.BufferType type) {
        StringBuilder str = new StringBuilder();
        for(int i=0; i<text.length(); i++) {
            if(consonants.indexOf(text.charAt(i)) >=0) {
                str.append(text.charAt(i));
                str.append('o');
                str.append(text.charAt(i));
            } else {
                str.append(text.charAt(i));
            }
        }
        super.setText(str.subSequence(0,str.length()),type);
    }
}
```

Skriva konstruktorer

Överskugga setText.

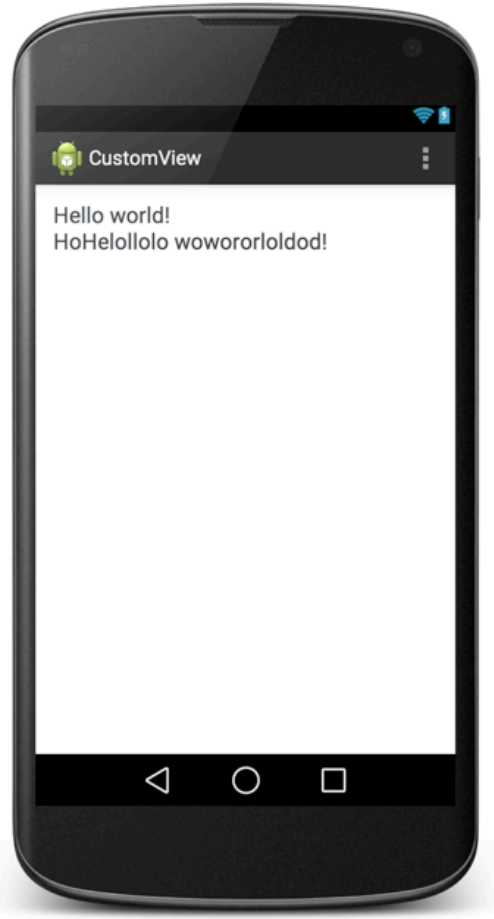
- Konstruera ny sträng
- Anropa superklassens setText

En ärvd View i en Layout

När den nya komponenten ska placeras I en layout måste man använda det fullständiga (kvalificerade) klassnamnet, t.ex.

se.mah.tsroax.f3.TRLTextView

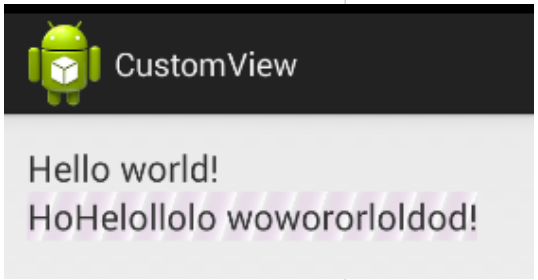
```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:paddingLeft="@dimen/activity_horizontal_margin"
    android:paddingRight="@dimen/activity_horizontal_margin"
    android:paddingTop="@dimen/activity_vertical_margin"
    android:paddingBottom="@dimen/activity_vertical_margin"
    android:orientation="vertical"
    tools:context=".MyActivity">

    <TextView
        android:text="@string/hello_world"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <se.mah.tsroax.f3.TRLTextView
        android:text="@string/hello_world"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />
```



Ärva en View

Lite färg på bakgrunden är alltid kul???

```
public class TRLTextView extends TextView {
    private static String consonants = "bcdfghjklmnpqrstvwxzBCDFGHJKLMNPQRSTVWXZ";
    private static Paint paint;

    public TRLTextView(Context context) {
        super(context);
        init();
    }

    public TRLTextView(Context context, AttributeSet attrs) {
        super(context, attrs);
        init();
    }

    public TRLTextView(Context context, AttributeSet attrs, int defStyle) {
        super(context, attrs, defStyle);
        init();
    }

    public void setText(CharSequence text, TextView.BufferType type) {...}

    private void init() {
        if(paint==null) {
            paint = new Paint();
            paint.setShader(new LinearGradient(0, 0, 30, 10, 0xD0A0D0, Color.WHITE, Shader.TileMode.REPEAT));
        }
    }

    @Override
    protected void onDraw(Canvas canvas) {
        canvas.drawPaint(paint);
        super.onDraw(canvas);
    }
}
```



Initiera ritverktyg

Överskugga onDraw

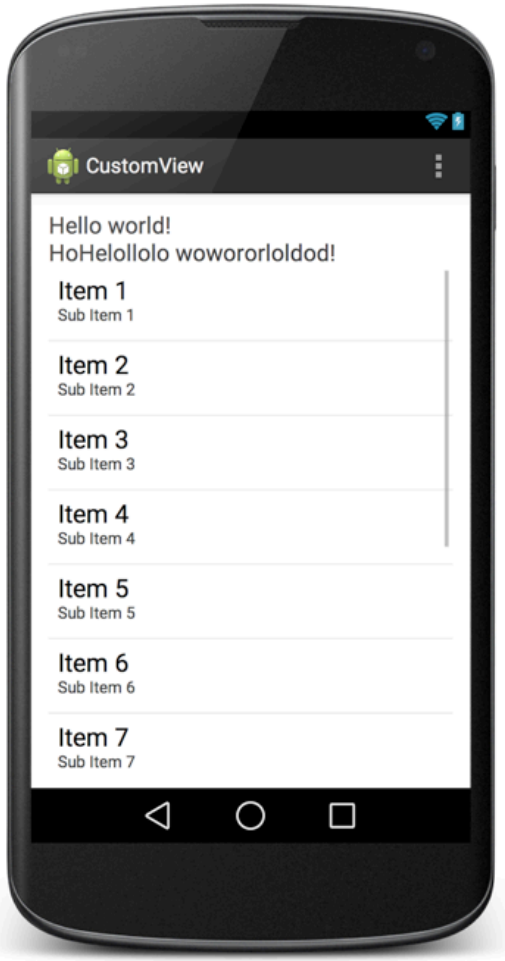
- Bakgrund (under text) ritas före super-anrop
- Förgrund (över text) ritas efter super-anrop

ListView och ArrayAdapter

ListView-komponenten används flitigt i applikationer för att visa information i listform.

ListView-komponenten behöver en ListAdapter för att fylla raderna I listan med information.

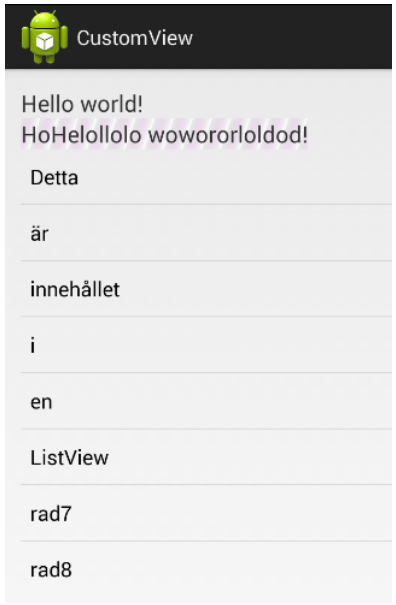
ListAdapter är ett interface med många metoder.
Klassen ArrayAdapter implementerar ListAdapter och är mycket användbar.



ListView och ArrayAdapter

En ListView måste deklareras i layout-filen.

```
        android:layout_height="wrap_content" />

    <se.mah.tsroax.f3.TRLTextView
        android:text="@string/hello_world"
        android:textSize="20sp"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content" />

    <ListView
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:id="@+id/lvTrl" />

</LinearLayout>
```



Man måste skapa en ArrayAdapter (med någon form av innehåll) och koppla adaptern till ListView-komponenten.

```
public class MyActivity extends Activity {
    private String[] content = {"Detta","är","innehållet","i","en","ListView",

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my);
        initializeComponents();
    }

    private void initializeComponents() {
        ListView lvTrl = (ListView)findViewById(R.id.lvTrl);
        lvTrl.setAdapter(new ArrayAdapter<String>(this,android.R.layout.simple_list_item_1,content));
    }
```

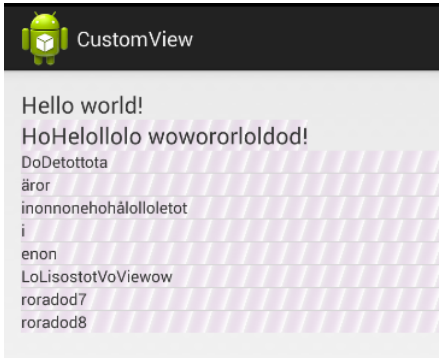
setAdapter sätter den ListAdapter som ListViewn ska använda

3 argument då ett ArrayAdapter-objekt skapas:

- Context (this)
- Layout för raden
- Data att visa (array eller List)

ListView och ArrayAdapter

Har man en TextView per rad i listan kan man göra sin egen layout att använda i ArrayAdaptern.




```
<?xml version="1.0" encoding="utf-8"?>
<se.mah.tsroax.f3.TRLTextView xmlns:android="http://schemas.android.com/apk
    android:layout_width="match_parent"
    android:layout_height="match_parent">

</se.mah.tsroax.f3.TRLTextView>
```

Sparad som trl_row.xml

Sedan anger man layouten då ArrayAdapter-objektet skapas.

```
public class MyActivity extends Activity {
    private String[] content = {"Detta","är","innehållet","i","en","ListView", "rad7","rad8"};

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my);
        initializeComponents();
    }

    private void initializeComponents() {
        ListView lvTrl = (ListView)findViewById(R.id.lvTrl);
        lvTrl.setAdapter(new ArrayAdapter<String>(this,R.layout.trl_row,content));
    }
```

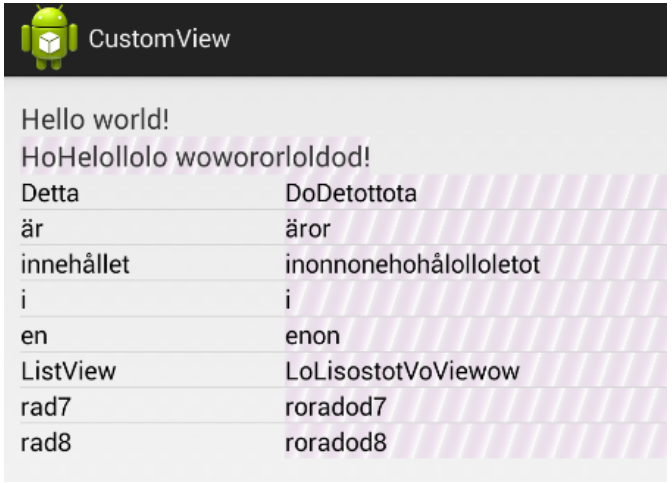
Ange layout-filen trl_row då ArrayAdaptern konstrueras.

Flera komponenter på en rad

En layoutfil måste skapas för raden

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:orientation="horizontal"
    android:layout_width="match_parent"
    android:layout_height="match_parent">

    <TextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:layout_weight="1"
        android:text="Medium Text"
        android:id="@+id/tvNormal" />

    <se.mah.tsroax.f3.TRLTextView
        android:layout_width="0dp"
        android:layout_height="wrap_content"
        android:textAppearance="?android:attr/textAppearanceMedium"
        android:layout_weight="2"
        android:text="Medium Text"
        android:id="@+id/ttvTrl" />
</LinearLayout>
```

Sparad som listview_item.xml



Flera komponenter på en rad

En klass som ärver ArrayAdapter behövs

```
public class TRLAdapter extends ArrayAdapter<String> {
    private LayoutInflater inflater;

    public TRLAdapter(Context context, String[] objects) {
        super(context, R.layout.listview_item, objects);
        inflater = (LayoutInflater)context.getSystemService(Context.LAYOUT_INFLATER_SERVICE);
    }

    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        TextView tvNormal;
        TRLTextView tvTrl;
        if(convertView==null) {
            convertView = (LinearLayout)inflater.inflate(R.layout.listview_item, parent, false);
        }
        tvNormal = (TextView)convertView.findViewById(R.id.tvNormal);
        tvTrl = (TRLTextView)convertView.findViewById(R.id.ttvTrl);
        tvNormal.setText(this.getItem(position));
        tvTrl.setText(this.getItem(position));
        return convertView;
    }
}
```

3 argument då superklass-konstruktor anropas:

- Context
- Layout för raden
- Data att visa (array eller List)

LayoutInflater-objekt för att skapa nya rader I listan

Rader återanvänds. Men om aktuell rad är null så måste en rad skapas. Därefter sätts värden I komponenterna på raden. Slutligen returneras rad-komponenten.

Metoden getView överskuggas.

Denna version av Adapter kommer att göras effektivare lite senare på kursen.

Flera komponenter på en rad

Slutligen ska ett TRLAdapter-objekt kopplas till ListView-komponenten

```
public class MyActivity extends Activity {
    private String[] content = {"Detta","är","innehållet","i","en","ListView", "rad7","rad8"};

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_my);
        initializeComponents();
    }

    private void initializeComponents() {
        ListView lvTrl = (ListView)findViewById(R.id.lvTrl);
        lvTrl.setAdapter(new TRLAdapter(this,content));
    }
```

Placera Fragment i statisk layout

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>
```

FragmentManager

Activityklassen kan hantera sina Fragment med FragmentManager-objektet. T.ex. går det att få referens till olika Fragment som finns i Activityn.

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

Med FragmentManager-objektet går det också att starta en FragmentTransaction vilket innebär att lägga till / ta bort / ersätta fragment i en container.

Placera Fragment dynamiskt i layout

Om man endast anger en (eller flera containrar) i layouten (ex FrameLayout eller LinearLayout) kan dessa vara hållare för Fragment under exekveringen. I layouten nedan är det två containrar som kan användas:

- "upper_container"
- "lower_container"

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

    <FrameLayout
        android:id="@+id/upper_container"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </FrameLayout>

    <FrameLayout
        android:id="@+id/lower_container"
        android:layout_width="match_parent"
        android:layout_height="wrap_content" >
    </FrameLayout>

</LinearLayout>
```

Placera Fragment dynamiskt i layout

Man gör s.k. Fragment Transactions vid dynamisk användning av Fragment i en layout.

En transaction börjar med anrop av metoden ***beginTransaction*** och avslutas med anrop till metoden ***commit***.

```
FragmentManager fragManager = getFragmentManager();
FragmantTransaction fragTransaction = fragManager.beginTransaction()

// bl.a. add, remove och replace här

fragTransaction.commit();
```

Placera Fragment dynamiskt i layout

```
<FrameLayout
    android:id="@+id/upper_container"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</FrameLayout>
```

Placera ett Fragment-objekt i containern med metoden ***add***.

```
FragmentTransaction transcation = fragManager.beginTransaction();
transaction.add( R.id.upper_container, fragment );
transaction.commit();
```

1. Starta först en Fragment transaction.
2. Därefter anropar du add-metoden med två (eller tre) argument :
   * Id till containern där Fragment-objektet ska placeras

     `R.id.upper_container`
   * Referens till Fragment-objektet

     `fragment`
   * En sträng, för att kunna erhålla referens till Fragment-objektet med hjälp av FragmentManager (findFragmentByTag(sträng)), kan anges som ett tredje argument.
3. Slutligen anropar du commit-metoden.

Ta bort Fragment dynamiskt ur layout

```
<FrameLayout
    android:id="@+id/upper_container"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</FrameLayout>
```

Ta bort Fragment från containern med metoden ***remove***

```
FragmentTransaction transcation = fragManager.beginTransaction();
transaction.remove( fragment );
transaction.commit();
```

Remove-metoden anropas med ett argument, refernes till Fragment-objektet som ska tas bort.

Ersätta Fragment dynamiskt i layout

```
<FrameLayout
    android:id="@+id/upper_container"
    android:layout_width="match_parent"
    android:layout_height="wrap_content" >
</FrameLayout>
```

Skifta Fragment i containern med metoden ***replace***

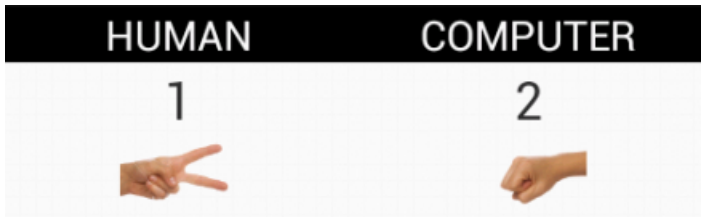
```
FragmentTransaction transcation = fragManager.beginTransaction();
transaction.replace( R.id.upper_container, fragment );
transaction.commit();
```

Replace-metoden anropas med två eller tre argument, precis som add-metoden.

Workshop

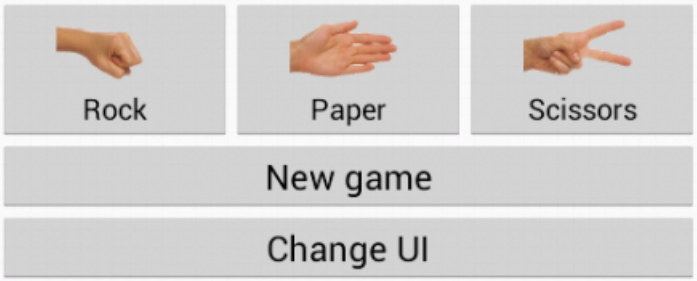
Sten, sax och påsespelet ska erbjuda användaren två olika viewer-fragments och två olika input-fragments.

ImageViewerFragment

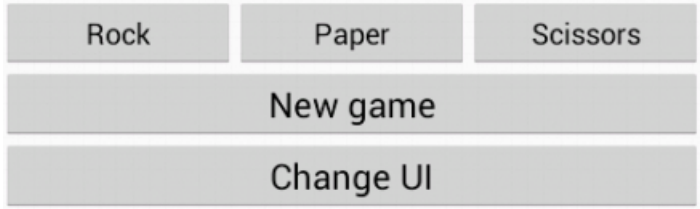


TextVewerFragment

ImageInputFragment



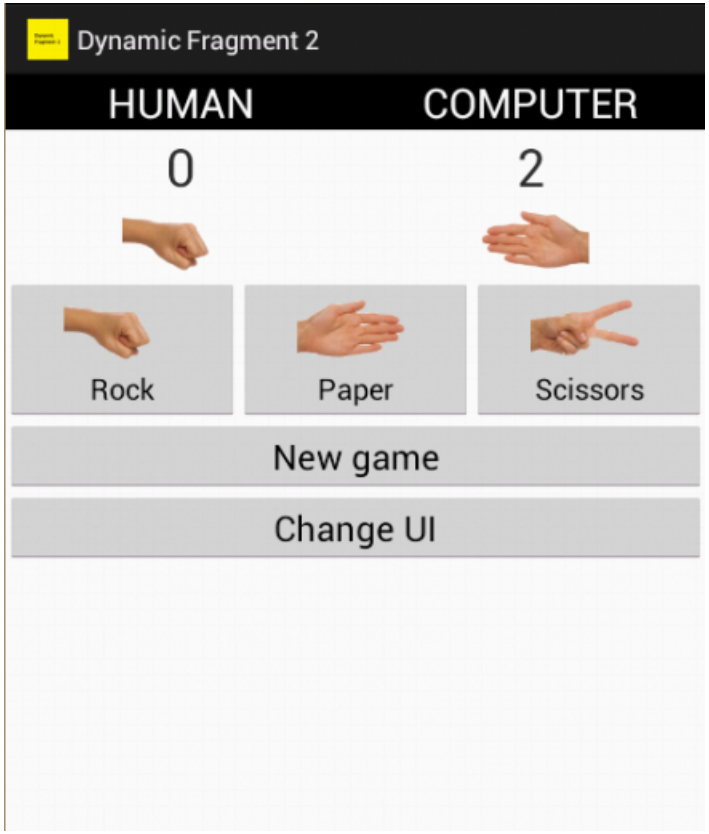
TextInputFragment

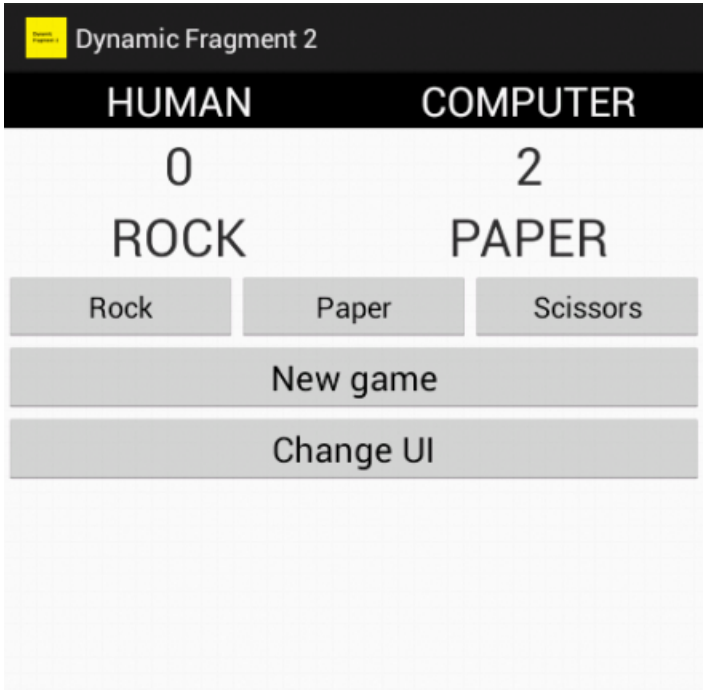


DynamicFragment2

Workshop

Användaren kan välja UI med bilder eller utan bilder genom klick på Change UI - knappen





Interface i programmet

Tre interface används i programmet. Detta för att enkelt kunna ersätta ett Viewer-objekt med ett annat, ett Input-objekt med ett annat och slutligen ett Controller-objekt med ett annat. Controller-objektet ersätts dock inte i denna workshop.

Interfacet Viewer

```
public interface Viewer {
    public void setController(Controller controller);
    public void setHumanPoints(int points);
    public void setComputerPoints(int points);
    public void setHumanChoice(int choice);
    public void setComputerChoice(int choice);
    public void humanWinner();
    public void computerWinner();
}
```

Interfacet Input

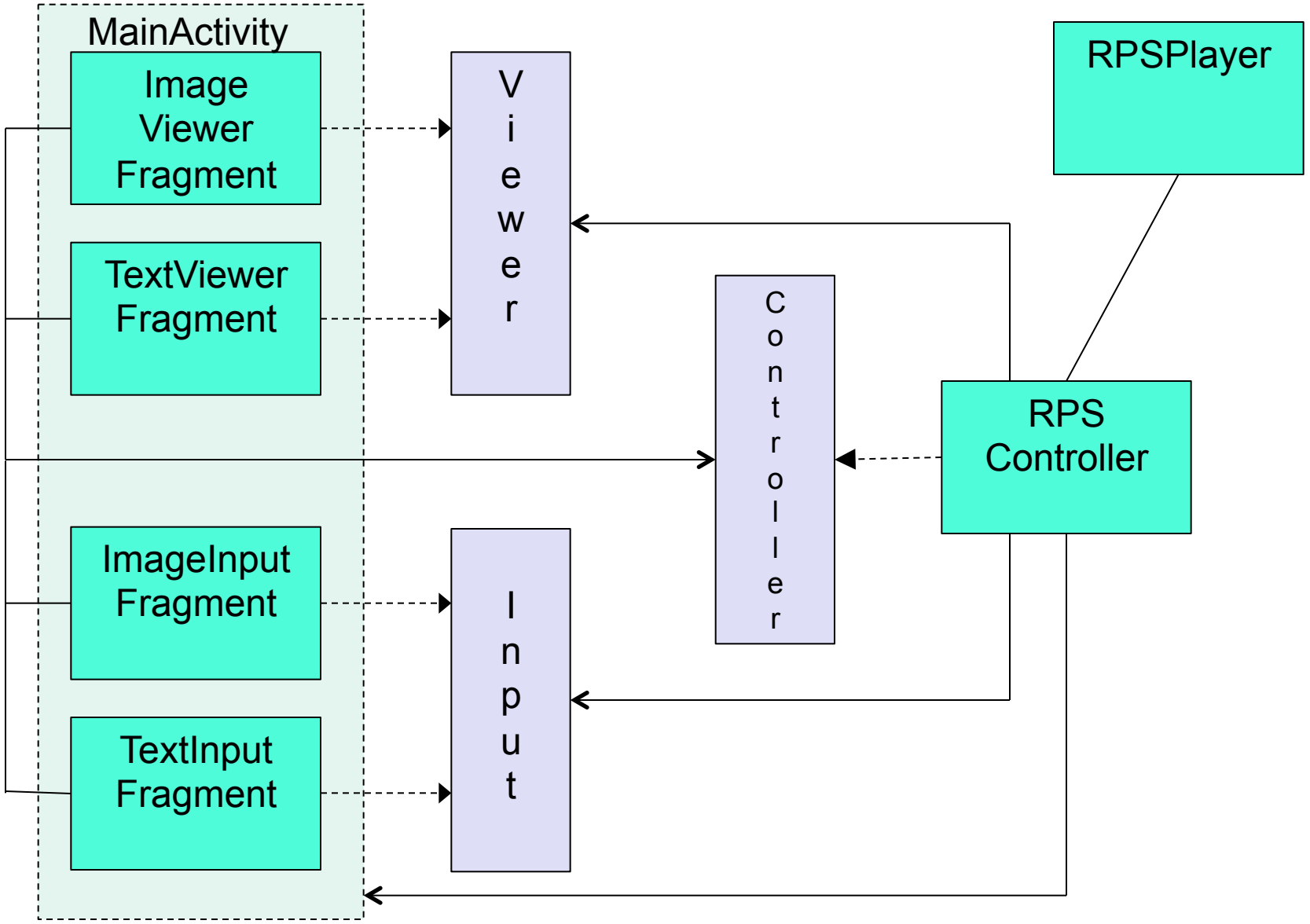
```
public interface Input {
    public void setController(Controller controller);
    public void enableChoiceButtons(boolean enabled);
}
```

Interface i programmet

Interfacet Controller

```
public interface Controller {

    public static final int ROCK = 0, PAPER = 1, SCISSORS = 2,
        EMPTY = 3, WINNER = 4, LOSER = 5;

    public void newGame();

    public void newChoice(int choice);

    public void updateViewer();

    public void changeUI();

}
```

Klassdiagram över programmet



MainActivity

```
public class MainActivity extends Activity {
    protected void onCreate(Bundle savedInstanceState) {
        :
    }

    public boolean onCreateOptionsMenu(Menu menu) {
        :
    }

    private void initController() {
        // instantierar objekt
    }

    public void setViewer(Viewer viewer) {
        FragmentManager fragManager = getFragmentManager();
        FragmentTransaction transaction = fragManager.beginTransaction();
        transaction.replace(R.id.upper_container, (Fragment)viewer);
        transaction.commit();
    }

    public void setInput(Input input) {
        FragmentManager fragManager = getFragmentManager();
        FragmentTransaction transaction = fragManager.beginTransaction();
        transaction.replace(R.id.lower_container, (Fragment)input);
        transaction.commit();
    }
}
```

RPSController

```
public class RPSController implements Controller {
    private MainActivity activity;
    private RPSPlayer computerPlayer;
    private Input input;
    private Viewer viewer;
    private int humanPoints, computerPoints;
    private int humanChoice, computerChoice;

    public RPSController(MainActivity activity, RPSPlayer computerPlayer) {
        this.activity = activity;
        this.computerPlayer = computerPlayer;
        viewer = new ImageViewerFragment();
        input = new ImageInputFragment();
        viewer.setController(this);
        input.setController(this);
        activity.setViewer(viewer);
        activity.setInput(input);
    }
    :
    private void rules(int humanChoice, int computerChoice) {…}
    private void winner() {…}
    private void initGame() {
        humanPoints = computerPoints = 0;
        humanChoice = computerChoice = Controller.EMPTY;
        input.enableChoiceButtons(true);
        updateViewer();
    }
```

RPSController

```
public class RPSController implements Controller {
    :
    public void newGame() {
        initGame();
    }

    public void newChoice(int humanChoice) {
        :
    }

    public void updateViewer() {
        viewer.setHumanPoints(humanPoints);
        viewer.setComputerPoints(computerPoints);
        viewer.setHumanChoice(humanChoice);
        viewer.setComputerChoice(computerChoice);
    }

    public void changeUI() {
        if(viewer instanceof TextViewerFragment)
             viewer = new ImageViewerFragment();
        else
            viewer = new TextViewerFragment();
        viewer.setController(this);
        activity.setViewer(viewer);
        // motsvarande för input
    }
}
```

Viewer-implementeringarna

```
public class AAAViewerFragment extends Fragment implements Viewer {
    private Controller controller;

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

    public void onResume() {
        super.onResume();
        controller.updateViewer(); // Uppdatering av nytt fragment
    }

    // Viewer-implementering
    public void setController(Controller controller) {…}
    public void setHumanPoints(int points) {…}
    public void setComputerPoints(int points) {…}
    public void setHumanChoice(int choice) {…}
    public void setComputerChoice(int choice) {…}
    public void humanWinner() {…}
    public void computerWinner() {…}
}
```

Input-implementeringarna

```
public class ...InputFragment extends Fragment implements Input {
    private Controller controller;

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

    private void registerListeners() {
        :
        btnChangeUI.setOnClickListener(new OnClickListener() {
            public void onClick(View view) {
                controller.changeUI();
            }
        });
    }
    // Input-implementering
    public void setController(Controller controller) {…}
    public void enableChoiceButtons(boolean enabled) {…}

    private class ChoiceButtonListener implements OnClickListener {
        :
    }
    :
}
```