Mobil megoldások

Kocsis Gergely

2019.10.07.

Activity állapotmentés

Töltsük le a tárgy oldaláról az előző órán készített ShoppingList alkalmazást és nyissuk meg Android Studioban.

Futtassuk és figyeljük meg, mi történik az alábbi esetekben:

- A listához adunk egy új tételt, majd a telefont elforgatjuk 90°-kal.
- A listához adunk egy új tételt, majd egy másik tételért megnyitjuk a tételválasztó ablakot, de választás helyett a ← gombbal térünk vissza
- Az alkalmazásból kilépünk a o gombbal, majd visszatérünk
- Az alkalmazásból kilépünk a □ gombbal, majd visszatérünk
- Az alkalmazásból kilépünk a ← gombbal, majd újranyitjuk

Activity állapotmentés

A Main Activity onCreate metódusát megvizsgáva vegyük észre, hogy lehetőséget ad mentett állapot visszaállítására.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    ...
```

Activity állapotmentés

A Main Activity onCreate metódusát megvizsgáva vegyük észre, hogy lehetőséget ad mentett állapot visszaállítására.

```
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        ...
```

Az onCreate(…) metódusban felhasznált mentett állapotnak az onSaveInstanceState metódus felüldefiniálásával adhatunk meg további saját értékeket.

```
@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putString(SAVESTATE_LIST_TEXT, itemsList.getText().toString());
    outState.putBoolean(SAVESTATE_LISTEMPTY_BOOL, isListEmpty);
}
```

String konstansok a jobb olvashatóságért és a hibák kiküszöbölésére

Activity állapotmentés

Az állapot visszaállításához az onCreate metódust kell módosítani.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    itemsList = findViewById(R.id.listTextView);

    if (savedInstanceState!=null){
        itemsList.setText(savedInstanceState.getString(SAVESTATE_LIST_TEXT));
        isListEmpty=savedInstanceState.getBoolean(SAVESTATE_LISTEMPTY_BOOL);
    }
}


@Override
protected void onSaveInstanceState(Bundle outState) {
    super.onSaveInstanceState(outState);
    outState.putString(SAVESTATE_LIST_TEXT, itemsList.getText().toString());
    outState.putBoolean(SAVESTATE_LISTEMPTY_BOOL, isListEmpty);
}
```

Activity állapotmentés

Töltsük le a tárgy oldaláról az előző órán készített ShoppingList alkalmazást és nyissuk meg Android Studioban.

Futtassuk és figyeljük meg, mi történik az alábbi esetekben:

- ~~A listához adunk egy új tételt, majd a telefont elforgatjuk 90°-kal.~~
- A listához adunk egy új tételt, majd egy másik tételért megnyitjuk a tételválasztó ablakot, de választás helyett a ← gombbal térünk vissza
- Az alkalmazásból kilépünk a o gombbal, majd visszatérünk
- Az alkalmazásból kilépünk a □ gombbal, majd visszatérünk
- Az alkalmazásból kilépünk a ← gombbal, majd újranyitjuk

Activity állapotmentés

Az első probléma megoldódott, de a többi esetben az állapot valamiért nem áll vissza.
Az oka, hogy ezekben az esetekben egy új Activity példány indul, amiben nincs tárolva a másik (eredeti) példány állapota.
Két megoldás van:

1) Alkalmazzunk perzisztens adattárolást az állapot mentésére
2) Állítsuk át az alkalmazást, hogy ezekben az esetekben ne indítson új Activity-t, csak térjen vissza a hívási láncban már megtalálható példányhoz.

Activity állapotmentés

Az első probléma megoldódott, de a többi esetben az állapot valamiért nem áll vissza.
Az oka, hogy ezekben az esetekben egy új Activity példány indul, amiben nincs tárolva a másik (eredeti) példány állapota.
Két megoldás van:

1) Alkalmazzunk perzisztens adattárolást az állapot mentésére
2) Állítsuk át az alkalmazást, hogy ezekben az esetekben ne indítson új Activity-t, csak térjen vissza a hívási láncban már megtalálható példányhoz.

```
<activity android:name=".MainActivity"
    android:launchMode="singleTop">
```

Activity állapotmentés

Töltsük le a tárgy oldaláról az előző órán készített ShoppingList alkalmazást és nyissuk meg Android Studioban.

Futtassuk és figyeljük meg, mi történik az alábbi esetekben:

- ~~A listához adunk egy új tételt, majd a telefont elforgatjuk 90°-kal.~~
- ~~A listához adunk egy új tételt, majd egy másik tételért megnyitjuk a tételválasztó ablakot, de választás helyett a ← gombbal térünk vissza~~
- ~~Az alkalmazásból kilépünk a o gombbal, majd visszatérünk~~
- ~~Az alkalmazásból kilépünk a □ gombbal, majd visszatérünk~~
- ~~Az alkalmazásból kilépünk a ← gombbal, majd újranyitjuk~~

:)

Fragmensek

Készítsünk egy új dinamikus Fragmenst, mely segítségével az Alma gomb lenyomása után lehetőséget adunk a felhasználónak, hogy megadja az alma színét (zöld, piros).

Első lépésként hozzunk létre gy új fragmenst (New → Fragment → Fragment (Blank)
Készítéskor generáljuk le a Layout-ot is, de ne adjuk hozzá a callback metódusokat és factory metódusokat.

Fragmensek

Készítsünk egy új dinamikus Fragmenst, mely segítségével az Alma gomb lenyomása után lehetőséget adunk a felhasználónak, hogy megadja az alma színét (zöld, piros).

Első lépésként hozzunk létre gy új fragmenst (New → Fragment → Fragment (Blank)
Készítéskor generáljuk le a Layout-ot is, de ne adjuk hozzá a callback metódusokat és factory metódusokat.

Adjunk a Fragmenshez egy TextView-t és két RadioButtont egy RadioGroupban. Érdemes ezeket egy ConstraintLayout-ben elhelyezni a könnyebb design kedvéért.

A RadioButton-ok felirata legyen “Red” és “Green” a TextView pedig legyen “Color: “

Fragmensek

Az ItemsListActivity layout fájljában adjuk egy ún. placeholder elemet a gombok elé.

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

Ide fog majd bekerülni a Fragmens.

Fragmensek

Az ItemsListActivity Java kódjához az alábbi metódust adva lehetőséget adunk, hogy megjelenjen a Fragmens a placeholder elemben

```
public void displayColorFragment(){
    BlankFragment colorFragment = BlankFragment.newinstance();
    FragmentManager fragmentManager = getSupportFragmentManager();
    FragmentTransaction fragmentTransaction = fragmentManager
            .beginTransaction();
    fragmentTransaction.add(R.id.frame_containder,
            colorFragment).addToBackStack(null).commit();
}
```

Ahhoz, hogy működjön, a Fragmensben szerepelnie kell a newinstance factory metódusnak.

Fragmensek

Már csak annyi hiányzik, hogy a RadioButton lenyomása után a szín is hozzáadódjon a névhez, ha az alma gomb megnyomása előtt azt a felhasználó kiválasztja. (`onCreateView`)

```
final View rootView =
        inflater.inflate(R.layout.fragment_blank, container, false);
final RadioGroup radioGroup = rootView.findViewById(R.id.radioGroup);
radioGroup.setOnCheckedChangeListener(new
RadioGroup.OnCheckedChangeListener() {
    @Override
    public void onCheckedChanged(RadioGroup radioGroup, int i) {
        View radioButton = radioGroup.findViewById(i);
        int index = radioGroup.indexOfChild(radioButton);
        switch (index) {
            case 0: colorName="Red";   break;
            case 1: colorName="Green"; break;
            default: break;
        }
    }
});

return rootView;
```