ListView w Androidzie

Anna Gogolińska

Tablice i listy

- Tworzenie tablicy: **typ[] tab = new typ[wielkość];**
  - Na przykład: int[] tab = new int[n];
    double[][] tab2 = new double[10][10];
  - Odwołanie się do elementu np. tab[i] (jak w C++).
- Lista – wygodniejsze rozwiązanie niż tablica.
  - Tworzenie: **List<Typ> lista = new ArrayList<>();**
  - Za Typ trzeba podać klasę, więc może być String, ale nie może być int, double. Trzeba podać Integer, Double, Boolean, itd. Np.
    List<Integer> lista = new ArrayList<>();
  - Potem jak wpisze się „lista.” pojawi się lista metod.
    - lista.add(element); - dodanie elementu.
    - lista.remove(index); - usunięcie elementu o danym numerze.
    - lista.get(index); - pobranie elementu o danym numerze.

Separator listy (przypomnienie)

- Jeśli używa się ListView to można kontrolować przestrzeń (wysokość) i kolor linii między elementami.

Cecha	Atrybut XML	Element
Wysokość Separatora	android:dividerHeight	ListView
Kolor Separatora	android:divider	ListView

- <ListView
  android:layout_width="match_parent"
  android:layout_height="wrap_content"
  android:divider="#F1C40F"
  android:dividerHeight="4dp" />

ListView

- Element do wyświetlania listy.
- Jego działanie polega na tym, że w kodzie aktywności tworzy się listę z danymi i łączy się tą listę z danymi z ListView.
  - Wtedy ListView wyświetla elementy z listy z danymi.
  - Jeśli chce się coś zmienić w wyświetlanych elementach (dodać nowy element do wyświetlania, usunąć, zmienić treść) robi się to zmieniając dane w liście z danymi.
  - Połączenie listy z widoku (ListView) i listy z danymi odbywa się poprzez adapter.

ListView – ustawianie danych

- W pliku xml należy dodać ListView.
- Aby określić jakie dane mają być wyświetlane przez listę należy stworzyć **ArrayAdapter**, który łączy listę (widok) z listą danych.
  - ListView, listę z danymi i adapter definiuje się przed onCreate().
  - Łączenie listy z danymi robi się w onCreate().
    - Należy dodać pierwsze elementy do listy z danymi.
  - Jeśli zmieni się listę z danymi (doda element, usunie) to aby zmiana ta była widoczna należy wywołać: **adapter.notifyDataSetChanged();**
    - Ma to miejsce zazwyczaj w metodzie będąca reakcją na jakieś zdarzenie.
    - W ten sposób dodajemy czy usuwany elementy z widoku.

Przykład

```
public class MainActivity extends AppCompatActivity {

    private ListView myListView;
    private List<String> daneListy = new ArrayList<>();
    private ArrayAdapter<String> adapter;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        … //domyślny kod
        myListView = findViewById(R.id.listViewId);
        daneListy.add("Pierwszy element");
        daneListy.add("Drugi element");
        daneListy.add("Trzeci element");
        adapter = new ArrayAdapter<>(this, android.R.layout.simple_list_item_1,  daneListy);
        myListView.setAdapter(adapter);
    }

//metoda w reakcji na naciśnięcie przycisku
    public void onDodaj(View view) {
        EditText myEditText = findViewById(R.id.editTextId);
        String nowyElement = myEditText.getText().toString();
        daneListy.add(nowyElement);
        adapter.notifyDataSetChanged();
        myEditText.setText("");
    }
}
```

ListView – reakcja na zaznaczenie elementu

- Należy zdefiniować dla listy **onItemClick(),** która ma następujące argumenty: **position** (indeks kliknięty elementu) oraz referencję do **parent** (samego ListView). Używając tych dwóch, możesz pobrać obiekt danych. Robi się to w onCreate().

```
// Zakładam, że 'myListView' jest już skonfigurowany za pomocą ArrayAdapter
myListView.setOnItemClickListener(new AdapterView.OnItemClickListener() {

    @Override
    public void onItemClick(AdapterView<?> parent, View view, int position, long id) {

        // 1. Pobieramy kliknięty element z adaptera, używając jego pozycji (position)
        // Wymagane jest rzutowanie (String), ponieważ wiemy, że Adapter przechowuje napisy.
        String wybranyElement = (String) parent.getItemAtPosition(position);

        // 2. Wyświetlamy pobraną wartość i jej indeks
        String komunikat = "Kliknięto pozycję nr " + position + ". Wartość: " + wybranyElement;
        }
});
```

Pobranie zaznaczone elementu

- Jeśli w reakcji na inne zdarzenie (np. naciśnięcie przycisku) ma być pobierany zaznaczony element z listy:
  - W xml dla listy ustawić aby można było na raz zaznaczyć jeden element:
    android:choiceMode="singleChoice"
  - W kodzie pobieramy numer zaznaczonego elementu, a potem z listy z danymi pobieramy element o tym numerze:
    int pozycja = myListView.getCheckedItemPosition();
    String wybranyElement = daneListy.get(pozycja);

Ilość wyświetlanych elementów

- Domyślnie ListView wyświetla wszystkie swoje elementy jeden pod drugim.
  - Jeśli elementy się nie mieszczą (**android:layout_height** jest mniejsza niż ilość elementów) automatycznie pojawia się pasek przewijania.
  - Można zmieniać ilość widocznych elementów poprzez zmianę ustawienia wysokości ListView.

Ustawienie layout_height	Efekt działania ListView	Wpływ na wydajność
match_parent	Zajmuje całą dostępną przestrzeń. Jeśli elementów jest więcej, staje się przewijalny.	Wydajny (korzysta z recyklingu widoków).
200dp (lub inna stała wartość)	Zajmuje stałą wysokość 200dp. Jeśli elementów jest więcej, staje się przewijalny.	Wydajny.
wrap_content	Rozciąga się, aby pomieścić wszystkie elementy.	NIEWYDAJNY przy dużej liczbie danych