RecyclerView

OnItemClickListener

RecyclerView OnItemClickListener

La definición y programación del listener «OnItemClickListener» pasa a ser responsabilidad del programador. Su vinculación debe realizarse con cada uno de los Views empleados para visualizar los datos subyacentes del RecyclerView

RecyclerView

OnItemClickListener

```
class CountriesAdapter(val countries: List<Country>,
                       val onItemClickListener : OnItemClickListener
                      ) : RecyclerView.Adapter<CountryViewHolder>()
{

  interface OnItemClickListener
  {
    fun onItemClick(position: Int, country: Country);
  }
```

Se define el interface que tendrá que implementar el listener

Se define la propiedad en la que se almacenará el listener

RecyclerView

OnItemClickListener

```
public class MainActivity extends Activity implements CountriesAdapter.OnItemClickListener
{
  lateinit countries : List<Country>                                  // datos subyacentes
  lateinit rvCountries : RecyclerView                                 // RecyclerView
  lateinit countriesAdapter : CountriesAdapter                       // RecyclerView.Adapter

  override vfun onCreate(savedInstanceState : Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    rvCountries = findViewById(R.id.rv_countries)
    countriesAdapter = CountriesAdapter(countries, this)
    rvCountries.setAdapter(countriesAdapter)
  }

  override fun onItemClick(position: int, county: Country) {
    : : :
  }
```

Se establece el listener en el adapter del RecyclerView

RecyclerView

OnItemClickListener

```
public class MainActivity extends Activity implements CountriesAdapter.OnItemClickListener
{
  lateinit countries : List<Country>                        // datos subyacentes
  lateinit rvCountries : RecyclerView                       // RecyclerView
  lateinit countriesAdapter : CountriesAdapter              // RecyclerView.Adapter

  override vfun onCreate(savedInstanceState : Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)
    rvCountries = findViewById(R.id.rv_countries)
    countriesAdapter = CountriesAdapter(countries, this)

    rvCountries.setAdapter(countriesAdapter)
  }
```

En el Activity se crea el listener

```
  override fun onItemClick(position: int, county: Country) {

    Toast.makeText(this, country.name, Toast.LENGTH_SHORT).show()

  }
```

RecyclerView

OnItemClickListener

```
class CountriesAdapter(val countries: List<Country>,
                       val onItemClickListener : OnItemClickListener
                      ) : RecyclerView.Adapter<CountryViewHolder>()
{

 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int) : CountryViewHolder
  {
    val context : Context = parent.context;
    val view : View = LayoutInflater.from(context).inflate(R.layout.item_view, parent, false);
    val countryViewHolder : CountryViewHolder = CountryViewHolder(view);
    return countryViewHolder;
  }

 inner class CountryViewHolder(view: View) : RecyclerView.ViewHolder(view), View.OnClickListener
  {
    val tvLine1 : TextView = view.findViewById<TextView>(R.id.tv_line1);
    val tvLine2 : TextView = view.findViewById<TextView>(R.id.tv_line2);




  }
```

El ViewHolder se define como una inner class de instancia (no static) a fin de que sus objetos puedan acceder a todo lo definido en el adapter (clase contenedora)

RecyclerView

OnItemClickListener

```
class CountriesAdapter(val countries: List<Country>,
                       val onItemClickListener : OnItemClickListener
                      ) : RecyclerView.Adapter<CountryViewHolder>()
{

 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int) : CountryViewHolder
  {
    val context : Context = parent.context;
    val view : View = LayoutInflater.from(context).inflate(R.layout.item_view, parent, false);
    val countryViewHolder : CountryViewHolder = CountryViewHolder(view)
    return countryViewHolder;
  }

 inner class CountryViewHolder(view: View) : RecyclerView.ViewHolder(view), View.OnClickListener
  {
    val tvLine1 : TextView = view.findViewById<TextView>(R.id.tv_line1)
    val tvLine2 : TextView = view.findViewById<TextView>(R.id.tv_line2)

    init {
             this.itemView.setOnClickListener(this)
         }

    override fun onClick(View view): Unit { : : : }
```

Cuando se crea un ViewHolder se establece que el listener OnClickListener de su view sea el propio ViewHolder

RecyclerView

OnItemClickListener

```
class CountriesAdapter(val countries: List<Country>,
                       val onItemClickListener : OnItemClickListener
                      ) : RecyclerView.Adapter<CountryViewHolder>()
{

 override fun onCreateViewHolder(parent: ViewGroup, viewType: Int) : CountryViewHolder
  {
    val context : Context = parent.context;
    val view : View = LayoutInflater.from(context).inflate(R.layout.item_view, parent, false);
    val countryViewHolder : CountryViewHolder = CountryViewHolder(view)
    return countryViewHolder;
  }

 inner class CountryViewHolder(view: View) : RecyclerView.ViewHolder(view), View.OnClickListener
  {
    val tvLine1 : TextView = view.findViewById<TextView>(R.id.tv_line1)
    val tvLine2 : TextView = view.findViewById<TextView>(R.id.tv_line2)

    override fun onClick(View view): Unit
    {
       val position: Int = this.getAdapterPosition()
       val country: Country = countries.get(position)
       onItemClickListener.onItemClick(position, country);
    }
```

En ViewHolder se define el listener OnClickListener que se asignará al itemView que representa cada ítem. Este listener invocará a su vez al listener onItemClickListener