**Anko چیست ؟**

Anko یک کتابخونه قدرتمند توسعه یافته توسط شرکت JetBrains هه. هدف اصلیش تولید صفحات UI با استفاده از Code به جای فایل xml هه .این یک ویژگی خیلی جالبه که بهتون پیشنهاد میکنم امتحانش کنید ولی من توی این پروژه ازش استفاده نمیکنم.برای من استفاده از xml راحت­تره ولی شما میتونین این روش رو پیش بگیرین.

اگرچه این تنها ویژگی نیست که ما از این کتابخونه بدست میاریم.Anko شامل تعداد زیادی از خصیصه ها و توابعه که میتونه کار رو براتون راحت­تر کنه.شما مثال­های زیادی توی این دوره خواهین دید و سریعا متوجه خواهید شد که چه مشکلاتی رو این کتابخونه حل میکنه.

**شروع به استفاده از Anko**

قبل از اینکه جلوتر بریم بهتره از Anko برای خلاصه سازی کدهامون استفاده کنیم.همینطور که خواهید دید، زمانی که از Anko استفاده میکنیم، اون همیشه یک import رو به همراه نام اون خصیصه یا تابع که ازش استفاده میکنیم رو به فایل اضافه میکنه. به این دلیل که Anko از توابع الحاقی استفاده میکنه تا بتونه ویژگیهای جدید رو به محیطتون اضافه کنه.

داخل کلاس MainActivity ما میتونیم از تابع ساده­تری برای پیداکردن RecyclerView استفاده کنیم:

val forecastList: RecyclerView = find(R.id.forecast\_list)

بهتره که با این کتابخونه بیشتر آشنا شین چون که خیلی از توابع این کتابخونه میتونه کارهارو براتون راحت تر کنه.

**توابع الحاقی:**

تابع الحاقی تابعیه که میتونه رفتار جدیدی رو به کلاس اضافه کنه، حتی زمانی که دسترسی به سورس کد اون کلاس نداشته باشیم. مزیت استفاده از این توابع توی کاتلین اینه که لازم نیست اون شی رو به عنوان یک آرگومان ورودی به تابع پاس بدیم.تابع الحاقی دقیقا طوری عمل میکنه انگار جزئی از کلاسه و ما حتی میتونیم خود اون کلاس رو با استفاده از this به تابعمون پاس بدیم!

به عنوان مثال ما میتونیم یک toast درست کنیم که دیگه لازم نباشه context رو بهش پاس بدیم! و این میتونه توی Activity ها و Service ها مورد استفاده قراربگیره!

fun Context.toast(message: CharSequence, duration: Int = *Toast*.*LENGTH\_SHORT*) {  
 *Toast*.makeText(this, message, duration).show()  
}

به عنوان مثال وقتی میخوایم داخل Activity ازش استفاده کنیم :

toast("Hello world!")  
toast("Hello world!", Toast.*LENGTH\_LONG*)

البته که Anko یک تابع Toast به مانند همین تابع درست کرده.Anko توابع مختلفی برای هم CharSequence و هم resource درست کرده که از اونجا رشته رو بهش پاس بدین، همچنین توابعی برای toast های کوتاه و بلند هم درست کرده!

toast("Hello world!")  
longToast(R.*id*.*hello\_world*)

توابع الحاقی حتی میتونن روی خصیصه ها هم انجام بشن!مثال بعدی که میبینین درواقع یک تابع الحاقی پیاده شده برروی یک خصیصه است که به مانند توابع الحاقی معمولیه. اینجا ما getter و setter دلخواه خودمون رو نوشتیم

public var TextView.*text*: CharSequence  
 get() = getText()  
 set(v) = setText(v)

توابع الحاقی در واقع کلاس اصلی رو تغییر نمیدن بلکه تنها یک static import به جاییه که ازش استفاده میشه. توابع الحاقی میتونه در هر کلاسی تعریف بشه، این میتونه تمرین خوبی باشه اگه یک دسته از توابع الحاقی مرتبط رو توی یک فایل ذخیره کنیم.