# شرح بالأمثلة العملية لل concrete class, abstract class and interface

**haythem.kmal** **في ٢٣ فبراير**

For English readers
**https://macdiscussions.udacity.com/t/explanation-with-practical-examples-concrete-class-abstract-class-and-interface/111722?u=haythem.kmal**

السلام عليكم مبرمجي المستقبل
اليوم سأقوم بزيادة ايضاح المفاهيم الاتية `concrete class` , `abstract class` and `interface` مع العلم أن الأمثلة المذكورة هنا قد تم ذكرها في فيديوهات الشرح ولكن رأيت أن المحاضر مر عليها مرورا سريعا.

بداية أود أن أوضح بعض الكلمات التي سأقوم بذكرها `abstract method` تعنيmethod بدون body كالاتي
هذه *method لها body*

```
Public String sayHi(String name){
// this is the body of the method
return "Hi " + name ;
}
```

هذه *method ليس لها body* ويطلق عليها abstract method

```
abstract void sayHi(String name);
```

ثانيا قمت باستبعاد كلمة implementation ولكن يمكنك وضعها كلما ذكرت كلمة ملىء اواستكمال ...باختصار أثناء الشرح ستجد أننا يجب أن نقوم بعمل implementation لل abstract class او لل interface بمعنى اكمال وملىء ال abstract methods اي التي ليس لها body وبالتالي عند عمل Implementation لل abstract method فاننا نقوم بكتابة body لها

ثالثا سأقوم بادراج بعض ال methods من بعض ال classes فلا تشغل بالك في كيفية عمل هذه ال methods حتى لا تتشتت .أنا فقط قمت بوضعها لكي أوصل لك نقطة ستفهمها من السياق 😉

حسنا، مافائدة معرفة هذه المفاهيم ... هذه المفاهيم وهذه الأنواع من ال classes او ال interfaces وعلى حسب علمي تفيد مطوري البرامج المحترفين -هذا لايمنع من اضطرارنا لاستخدام بعضها كمطوريين مبتدئين- ومأأقصده بالاحتراف هنا أن مطوري لغة الجافا خصوصا ذلك الجزء الذي يتعامل مع بيئة الاندرويد قد وضعوا في حسبانهم أن هناك بعض المطورين قد يرغبوا في تغيير و تعديل طريقة تنفيذ بعض الأكواد وبالطبع هناك بعض مطوري برامج الاندرويد من النوع المبتدىء والذي يرغب فقط في استخدام ال classes الجاهزة والكاملة ... ستضح لك هذه النقطة أثناء الشرح .

حسنا لنبدأ بتعريف ال concrete class هذا النوع من ال classes عبارة عن class جاهز تماما وكامل تماما للاستخدام ولكي تضح لك الرؤية سأعطيك مثالا... مثلا ال TextView هذا في الأصل عبارة عن class وكلنا يعلم ذلك وأنت تستخدم ال TextView بكل أريحية ولا تقوم بأي تعديل على الأكواد الجاهزة بل أنت كمطور راض تماما بحدود وامكانيات ال TextView حيث مثلا تقوم بتحديد نوع الخط وحجمه وماهو النص المكتوب و الخ الخ وقد رأى منشئى هذا ال class أي ال TextView class أن هناك الكثير من مطوري البرامج سيستخدمون هذا ال class كما هو ولا يوجد مبرر لاتاحة الفرصة لهم لتشكيل methods معينة بطريقة معينة ( هذه الجملة ستضح أكثر فيما بعد ...فقط تابع الشرح) بل هم يعتقدون انهم أنشأووا class كامل متكامل صالح للاستخدام المباشر وهذا بالضرورة يعني أن جميع ال methods لها bodyتقوم بتنفيذه .

***الخلاصة*** 🎁
عندما تجد class به هذه المميزات أي أنه صالح للاستخدام المباشر وجميع ال methods كاملة ولها body فاننا نطلق عليه concrete class .

حسنا اذن ماهو ال abstract class وكيف يختلف عن سابقه ؟! في الواقع سأعطيك مثالا يوضح لك هذه النقطة وأهميتها .يعتبر ال ViewGroup نوع من ال abstract class .أظنك تعلم نوعين من ال ViewGroup فقد علمنا أن ال LinearLayout و RelativeLayout هما ViewGroup ولكن مهلا لا تتسرع فقط ال ViewGroup class يعتبر abstract class لماذا ؟؟ حسنا عليك أن تعلم أولا مامعنى abstract class.هذه الكلمة تعني أن ال class غير كامل وغير جاهز للاستخدام المباشر بمعنى اخر أن هناك بعض ال methods تُركت فارغة بدون body ويطلق على ال method في هذه الحالة abstract method و لاعجب فكما قلنا كلمة abstract توحي بعدم الكمال

*من ضمن هذه ال method الموجودة في ال ViewGroup abstract class*

```
@Override
protected abstract void onLayout(boolean changed,
        int l, int t, int r, int b);
```

... على اية حال، هذه ال methods تركت فارغة لسبب فقد رأي مطوري اللغة أن هناك بعض المطورين ذو كفاءة عالية قد تنتابهم الرغبة بالتعديل على كيفية عمل بعض ال methods و لكي تتضح لك الرؤية أكثر لنرى المثال الاتي ...بالنسبة لل ViewGroup وكما علمنا ولأنه abstract class فأنه لابد لك من انشاء concrete class من هذا ال abstract class لكي تستفيد منه وتستطيع استخدامه مباشرة . حسنا قام مطوري الاندرويد باستخدام هذا ال ViewGroup abstract class و قاموا بانشاء كلا من LinearLayout و ال RelativeLayout وقد تتذكر أنك تستخدم هذين النوعين مباشرة بدون عناء ولكن لكي تعلم مايحدث في الخلفية فقد قام المطورون بعمل extends لل ViewGroup class اي أنهم استخدموا ال ViewGroup لانشاء مثلا ال LinearLayout ولكن قاموا بملىء بعض ال methods التي تعطي ال LinearLayout خاصيتها المميزة في ترتيب ال views من ضمن هذه ال methods التي قاموا باستكمال طريقة تنفيذها method تسمى ()onlayout

هذه هي ال *method* افي ال *LinearLayout class*

```
@Override
protected void onLayout(boolean changed, int l, int t, int r, int b) {
    if (mOrientation == VERTICAL) {
        layoutVertical(l, t, r, b);
    } else {
        layoutHorizontal(l, t, r, b);
    }
}
```

أيضا عند انشاء ال RelativeLayout class قد قاموا بتغغير نفس هذه ال method والتي تسمى ()onlayout

هذه هي ال *method* في *RelativeLayout class*

```
@Override
protected void onLayout(boolean changed, int l, int t, int r, int b) {
    //  The layout has actually already been performed and the positions
    //  cached.  Apply the cached values to the children.
    final int count = getChildCount();
    for (int i = 0; i < count; i++) {
        View child = getChildAt(i);
        if (child.getVisibility() != GONE) {
            RelativeLayout.LayoutParams st =
                    (RelativeLayout.LayoutParams) child.getLayoutParams();
            child.layout(st.mLeft, st.mTop, st.mRight, st.mBottom);
        }
    }
}
```

بحيث تظهر ال views بالطريقة التي نعتادها مع ال RelativeLayout حسنا أتمنى أن تكون قد توصلت أن الان ال LinearLayout و ال RelativeLayout هما الان concrete classes ناشئة من abstract class يدعى ViewGroup.سنرجع للسؤال الفلسفي ؟! لماذا لم يتم انشاء ال LinearLayout و ال RelativeLayout مباشرة بحيث يستخدمها المطور في ترتيب ال views ما أعنيه هو لم كل هذا العناء؟؟

حسنا ياصديقي ، بداية عليك أن تعلم أن مطوري اللغة لا يضعوا في حسبانهم المستخدم النهائي فقط ولكن هم يصنعوا لك اداة قادرة على استيعاب أفكارك وقدراتك البرمجية وهم أنفسهم قد يستفيدوا من تلك الخطوة وهذا مايحدث فعليا فمثلا ستجد أن هناك أكثر من class يتبع ال ViewGroup فهم يستفيدوا من ال methods الكاملة والتي من طبيعة حالتها لاتحتاج الى تغيير ومتشابهة الى حد كبير بين أنواع ال ViewGroups ولكن يقوموا بتعديل وملىء بعض ال methods التي تعطى كل نوع من ال ViewGroup خصائصه المميزة ومأأقصده من أنواع ال ViewGroup هو مثلا LinearLayout, RelativeLayour , GridLayout,FrameLayout ,...etc. كما أنهم يضعوا في حسبانهم شخصا ذكيا مثلك يرغب في تطوير ViewGroup خاص به يقوم بوضع ال views بطريقة معينة قد يكون ابتكرها مثلا يعني 🙂

**الخلاصة** 🎁

ان ال abstract class هو عبارة عن class به بعض ال methods الكاملة والتي يمكن استخدامها مباشرة ولكن أيضا يحتوى على methods أخرى لابد لك من اكمال ال body الخاص بها ولا يمكن استخدام ال abstract class مباشرة الا بعد اكمال وملىء هذه ال methods ولا يتم هذا الا من خلال انشاء concrete class لهذا ال abstract class و في ال concrete class نقوم بملىء واكمال هذه ال methods .

حسنا اذن ماهو ال interface ?? ال interface يشبه الى حد كبير ال abstract class الا ان مايميزه ان جمييييييع ال methods التي يحتويها ال interface عبارة عن abstract method أي ليس لها body وفي ذلك حكمة فمثلا عندنا ال View.OnClickListener هوا عبارة عن Interface وكما درسنا فان هذا ال interface يستخدم في تحديد مايحدث عند الضغط على view معين هذا ال interface يحتوي على abstract method واحدة وهي `onClick(View v)` وكما تعلم لاستخدام هذا ال Interface لابد لك من ملىء واكمال ال body الخاص بهذه ال method. وكلنا يعلم الطريقة ولكن مايهمني أن تعلمه هو لماذا تم بناء View.OnClickListener على أساس أنه Interface ؟! حسنا لو تفكرت في الامر قليلا لوجدت أن هذا هو عين الصواب فقد أتاح لك مطوري الاندرويد هذا ال interface لأنه بطبيعة الحال يريدوا أن يفسحوا لك المجال بحيث تكتب داخل هذا ال body اي أكواد تريد أن تنفذها عند الضغط على ال view المناسب حيث أنه من غير المنطقي أن يحددوا مثلا أنه عند الضغط فان النص يتغير للنص الفلاني مهما فعلوا لن يستطيعوا أن يجمعوا مايمكن فعله عند الضغط ولو فعلوا لأحجموا تفكيرك ولكن لانها Interface فقد أتاحوا الفرصة لك ولغيرك لوضع الاكواد التي يريدونها فمثلا قد تريد عند الضغط انشاء array غيرك قد يرغب في تغيير لون النص وغيركما قد يرغب في انشاء متغير من نوع String ستجد ان ال interfaces تتمحور حول هذه النقاط

**الخلاصة** 🎁

ال interface يعتبر نوع من العقد بين مطور اللغة وبينك حيث يقدم لك أسماء ال methods وتكون abstract ويقول لك اذا قمت بعمل كذا وكذا فانه يمكنك ملىء واستكمال هذه ال methods وتنفيذ مابها.

باذن الله سأقوم بعمل موضوع اخر ولكن بأمثلة أخرى واقعية وسترى كيف ينطبق ماقلت أيضاعلى ال ArrayList و List interface .أتمنى أن يكون قد اتضح هذا الامر بعض الشيء وأرجو من لديه استفسار ان يتفضل بالسؤال ... تحياتي وبالتوفيق لكم جميعا 🙂

---

- **تجميع لكل تلخيصات دروس الاندرويد**
- **مفاهيم حول الـ Concrete و الـ Abstract**
- **شرح ال ArrayAdapter كمقدمة لل custom ArrayAdapter**
- **شرح الOnclickListener وربطه بموضوع ال interface**
- **شرح ال context بالتفصيل .. هتفهمه باذن الله**

**4 أخرى**