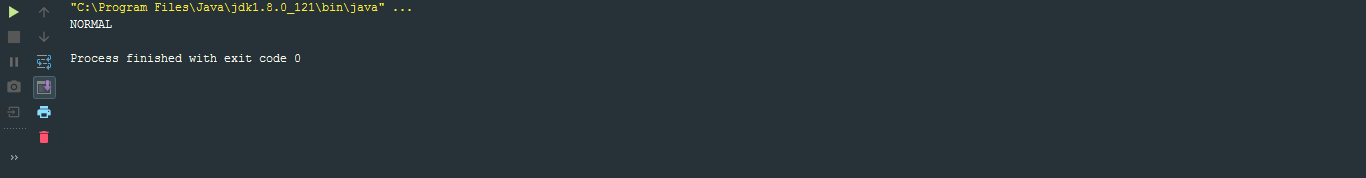
رسیدم به بخش Enum که ساختشون راحت و استفاده ازشون راحت تر. به عنوان مثال کد زیر رو نگاه کنید.

enum class Priority {  
 MINOR,  
 NORMAL,  
 MAJOR,  
 CRITICAL  
}  
  
fun main**(**args: Array<String>**)** {  
 val priority = Priority.NORMAL

println**(**priority**)**  
}

خب حالا اگه من بیام و priority رو پرینت کنم چی میشه؟ ( با فشار دادن دکمه اجرا میتونین نتیجه رو ببینین)

در اینجا کاتلین به صورت دیفالت میاد و مقدار اون enum رو به String تبدیل و چاپ میکنه.

خب حالا فرض کنین میخوایم هر کدوم از enum هامون یک مقدار داشته باشن، یعنی یک خصیصه داشته باشیم به نام value و مثلا وقتی به NORMAL دسترسی پیدا کردیم، مقدار اونو بخونیم.

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**,  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}  
  
fun main**(**args: Array<String>**)** {  
 val priority = Priority.NORMAL  
println**(**priority.value**)**}

همچنین ما میتونیم از کامپایلر بخوایم که بهمون بگه این enum ای که ما میخوایم بهش دسترسی داشته باشیم چندمین enum توی کلاس خودشه.

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**,  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}

fun main**(**args: Array<String>**)** {  
 val priority = Priority.NORMAL  
  
 println**(**priority**)** println**(**priority.value**)** println**(**priority.ordinal**)**}

همچنین ما خصیصه name رو داریم

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**,  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}

fun main**(**args: Array<String>**)** {  
 val priority = Priority.NORMAL  
  
 println**(**Priority.CRITICAL.name**)**

}

که اگه برنامه ران بشه، بهمون CRITICAL رو برمیگردونه.

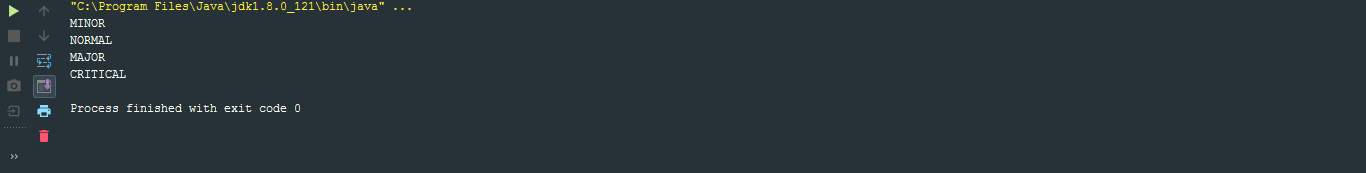
فرض کنین که میخوایم تموم enum های داخل کلاس رو لیست کنیم

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**,  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}

fun main**(**args: Array<String>**)** {

for**(**priorityInList in Priority.values**())** println**(**priorityInList**)**}

درواقع تابع values() میاد و یک لیست از تمومی مقدار های داخل کلاس Priority رو بهمون میده.



اگه فرض کنیم بخوایم با استفاده از کلاس به مقدار یک enum دسترسی داشته باشیم از تابع valueOf() استفاده میکنیم

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**,  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}

fun main**(**args: Array<String>**)** {

val p = Priority.valueOf**(**"MAJOR"**)** println**(**p**)**

}

خب همونطور که دیدید وقتی یکی از enum هارو پرینت میکنیم در واقع نام اون enum رو برامون پرینت میکنه، همون کاری که با متغیر name انجام میدیم، ولی من میتونم بیام و تابع toString رو override کنم

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**{  
 override fun toString**()**: String {  
 return "Minor priority"  
 }  
 },  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**}

fun main**(**args: Array<String>**)** {

val priority = Priority. MINOR

println**(**priority**)**

}

خب، تا اینجا یاد گرفتیم چگونه از توابع و خصیصه های enum استفاده کنیم، حالا بیایم و یک تابع داخل enum بنویسیم

ابتدا بیایم یک abstract داخل enum درست کنیم.

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**{  
 override fun toString**()**: String {  
 return "Minor priority"  
 }  
 },  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)** abstract fun text**()**: String  
}

fun main**(**args: Array<String>**)** {

val priority = Priority. MINOR

priority.text()

}

خب اگه این کد رو بزنید متوجه میشید که کامپایلر بهتون ارور میده. یادتونه که گفتم یک جا هست که شما نیاز به نقطه ویرگول دارین ؟ اون اینجاست، بعد از اخرین enum یک نقطه ویرگول بذارین.

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**{  
 override fun toString**()**: String {  
 return "Minor priority"  
 }  
 },  
 NORMAL**(**0**)**,  
 MAJOR**(**1**)**,  
 CRITICAL**(**10**)**;  
  
 abstract fun text**()**: String  
}

درواقع وقتی شما حتی این نقطه ویرگول رو هم بذارید بازم کامپایلر بهتون ارور میده، البته نه این که ما کار اشتباهی کردیم، بلکه خب اینجا ما دوتا ارور داشتیم. توجه کنین که ما متد abstract رو درست کردیم، پس باید برای همه ی enum های داخل این کلاس این متد رو override کنیم.

enum class Priority**(**val value : Int **)** {  
 MINOR**(**-1**)**{  
 override fun text**()**: String {  
 TODO**(**"not implemented"**)** //To change body of created functions use File | Settings | File Templates.  
 }  
  
 override fun toString**()**: String {  
 return "Minor priority"  
 }  
 },  
 NORMAL**(**0**)** {  
 override fun text**()**: String {  
 TODO**(**"not implemented"**)** //To change body of created functions use File | Settings | File Templates.  
 }  
 },  
 MAJOR**(**1**)** {  
 override fun text**()**: String {  
 TODO**(**"not implemented"**)** //To change body of created functions use File | Settings | File Templates.  
 }  
 },  
 CRITICAL**(**10**)** {  
 override fun text**()**: String {  
 TODO**(**"not implemented"**)** //To change body of created functions use File | Settings | File Templates.  
 }  
 };  
  
 abstract fun text**()**: String  
}

خب حالا تنها کاری که باید بکنید اینه که برای هر enum بگید تابع مورد نظرتون میخواد چیکار انجام بده و تمام