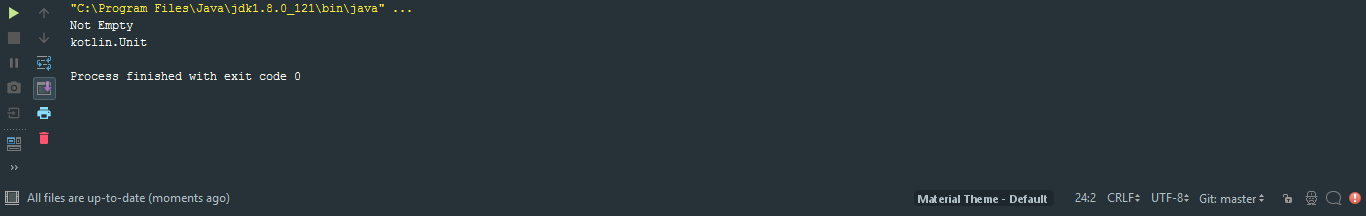
مثل بقیه زبان های برنامه نویسی کاتلین هم از عبارات شرطی نیز استفاده میکنه ولی با مقداری تفاوت که در ادامه میبینیم:

var myString= "Not Empty"  
if**(**myString != ""**)**{  
 println**(**"Not Empty"**)**}else if **(**myString.startsWith**(**"a"**))**{  
 println**(**"Starts With A"**)**}else{  
 println**(**"Is Empty"**)**}

خب تا اینجا که مثل بقیه زبون ها عمل کردیم، ولی یکی از ویژگی های کاتلین اینه که میتونه به عنوان یک عبارت بیان بشه، قطعه کد زیر رو نگاه کنید:

var result = if**(**myString != ""**)**{  
 println**(**"Not Empty"**)**}else {  
 println**(**"Starts With A"**)**}  
  
println**(**result**)**

اگر این رو اجرا کنیم برنامه خروجی زیر رو میده

چرا kotlin.Unit ؟ درواقع Unit د رکاتلین به مانند void میمونه. ولی من گفتم که کاتلین این اجازه رو میده که بتونیم از if به صورت یک عبارت شرطی استفاده کنیم، خب باید توجه کنیم که اگر if بدین صورت استفاده بشه، کاتلین اخرین خط رو به عنوان مقدار برمیگردونه، یعنی اگر بخواهم به هدف دلخواهمون برسیم باید کد رو به صورت زیر تغییر بدیم:

var result = if**(**myString != ""**)**{  
 "Not Empty"  
}else {  
 "Starts With A"  
}  
  
println**(**result**)**

توجه کنین که مقدار اخرین خط بازگردادنده بشه، پس اگر بعد از “Not Empty” مقدار 20 گذاشته شود، جواب 20 برگردانده میشود.

بیاین یک نکته دیگه رو هم بگم. اگه عبارت بالا رو به این صورت بنویسم

var result = if**(**myString != ""**)**{  
 20  
}else {  
 "Starts With A"  
}  
println**(**result**)**

متوجه میشیم که کاپایلر یک اخطار میده و میگه result رو به Any کست میکنه!! در واقع Any یک کلاس بالایی است که دیگر شی ها از اون ارث بری شدند( اگر با مفهوم ارث بری اشنا نیستید، بعدا در مورد اون ها صحبت خواهیم کرد) و در واقع این اخطار به ما اینو میگه : " ایا خبر داری که این عبارت شرطیت یک جا Int بر میگردونه و یک جای دیگه String ؟"

نکته دیگه باید توجه کنید اینه که وقتی داریم مقدار یک if رو به یک متغیر نسبت میدیم حتما باید برای تمامی شرایط مقداری برگردونیم.مثلا اگر به صورت زیر بنویسیم

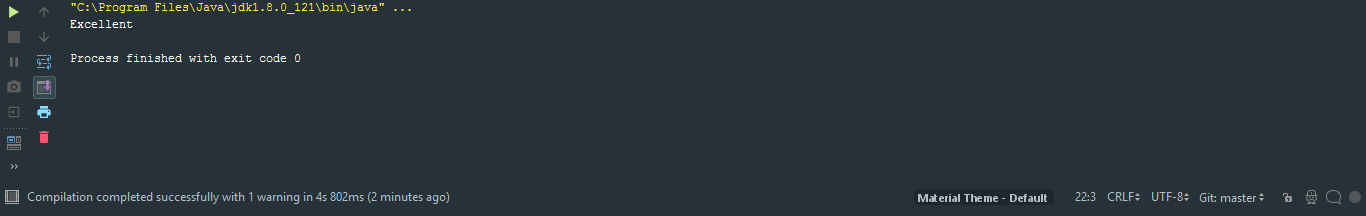
var result = if**(**myString != ""**)**{  
 20  
}  
println**(**result**)**

متوجه میشیم که برناممون با ارور مواجه میشه، و اونم دلیلش اینه که کامپایلر نمیدونه که اگر مثلا myString ==”” بود چه عملی رو انجام بده.

همانند if ما یک عبارت دیگه داریم به نام when که تقریبا به مانند case در زبون های دیگه میمونه.

var result = "Value"  
when**(**result**)**{  
 is String **->** println**(**"Excellent"**)** "Value" **->** println**(**"It's a Value"**)**}

مثلا عبارت بالا زو مد نظر بگیرین، اومدم نوشتم اگه result یک String بود عبارت اول رو چاپ کنه و اگر برابر بود با “Value” عبارت دوم رو. خب به نظرتون اگه این کد رو اجرا کنم خروجی چی میشه ؟

همینطور که میبینین عبارت “Excellent” چاپ شد. در واقع وقتی در داخل when به یکی از جواب ها برسیم کامپایلر دیگه خط های پایین رو نگاه نمیکنه و سریع break میکنه.

همانند if ، when هم میتونه عبارت شرطی باشه و نتیجش رو به یک متغییر پاس بدیم.

var result = "Value"  
val whenValue = when**(**result**)**{  
 "Value" **->** {  
 println**(**"It's a Value"**)** println**(**"Second statement"**)** }  
 is String **->** println**(**"Excellent"**)**}  
println**(**whenValue**)**

عبارت بالا رو مینویسیم، دقیقا به مانند if ، ولی وقتی این کد ر اجرا میکنیم با ارور مواجه میشیم. اینجا هم مثل if کامپایلر برمیگرده میگه " من میدونم اگه برابر “Value” بود چیکار کنم، میدونم اگه String بود چیکار کنم ولی اگه توی وضعیت دیگه ای بود چیکار کنم ؟ مشکل را با استفاده از کلید واژه else رفع میکنم، بدین صورت

var result = "Value"  
val whenValue = when**(**result**)**{  
 "Value" **->** {  
 println**(**"It's a Value"**)** println**(**"Second statement"**)** }  
 is String **->** println**(**"Excellent"**)** else **->** println**(**"It came to this?"**)**}  
println**(**whenValue**)**

البته همینطور ک متوجه شدین، اگر این برنامه رو اجرا کنیم مقدار Unit برگردانده میشود و دلیل ان هم واضح است چرا که ما هنوز چیزی در اخرین خط شرط هامون برنگردوندیم!

var result = "Value"  
val whenValue = when**(**result**)**{  
 "Value" **->** {  
 println**(**"It's a Value"**)** println**(**"Second statement"**)** "Here"  
 }  
 is String **->** {  
 println**(**"Excellent"**)** "And Here"  
 }  
 else **->** {  
 println**(**"It came to this?"**)** "and what about here?"  
 }  
}  
println**(**whenValue**)**