{پیشنهاد میشه این قسمت رو بر روی IntellJ IDEA امتحان کنید}

خب به قسمتی رسیدیم که دقیقا به بخش خاصی تعلق ندارن ولی خب نکات مهمی رو پوشش میده.

ممکنه شما متوجه یک چیز توی Null Checks ها شده باشین و من با همون شروع میکنم.

تا اینجا یادتون میاد که این کد رو داشتیم

fun main(args: Array<String>) {  
 var nullMessage :String? = null  
 println(nullMessage.length)  
}

و خب گفتیم اگه متغیری بتونه مقدار null رو بگیره، باید قبل استفاده ازش چک بشه، و چک کردنش رو هم با "?" انجام میدادیم. خب حالا اگه بیاین قبل استفاده از nullMessage یک مقدار بهش بدین

fun main(args: Array<String>) {  
 var nullMessage :String? = null  
 nullMessage = "Some Value"  
 println**(**nullMessage?.length**)**  
}

کامپایلر یه اخطار کوچولو بهمون میده و میگه : "نیازی نیست متغیر غیر null رو چک کنیم".یعنی میتونیم "?" رو برداریم.

fun main(args: Array<String>) {  
 var nullMessage :String? = null  
 nullMessage = "Some Value"  
 println**(**nullMessage.length**)**  
}

ولی خب تا اون رو برمیداریم اون متغیر یک بک گراند سبز میگیره به خودش و اگه بریم و موس رو روش نگه داریم متوجه میشیم که نوشته "Smart cast to kotlin.String". این به چه معنیه؟ بهتون نشون میدم.

بیاین یک فایل به نام Casting درست کنیم و کد زیر رو توش بزنیم

open class Person  
  
class Employee: Person**()**{  
 fun *vaction*Days**(**days: Int**)**{  
 if**(**days < 60**)** println**(**"You need more vacation"**)** }  
}  
  
class Contractor: Person**()**fun hasVacations**(**obj: Person**)**{  
 if**(**obj is Employee**)**{  
   
 }  
}  
fun main**(**args: Array<String>**)** {  
  
}

خب تا اینجا کار خاصی نکردیم.دوتا کلاس ساختیم به نام های Employee و Contactor که از کلاس Person ارث بری میشن و همچنین داخل Employee یک متد ساختیم. خب حالا بعدش اومدیم و یک تابع داخل فایلمون ساختیم به نام hasVacations که توش یک شرط داره که البته بدنه شرط رو هنوز ننوشتیم، ولی خب میدونیم که اگر obj یک شی از نوع Employee بود داخل if میره. خب اگه بریم داخل if و بخوایم بنویسم obj.vacationDays متوجه میشیم که IntelliJ برامون ادامه اش رو مینویسه! توجه کنین که obj از نوع کلاس Person بود که تابعی به اون اسم نداره! اگه کد رو کامل بنویسیم متوجه میشیم که دوباره همون حالت پیش میاد، دوباره مینویسه "Smart casting to Employee" .

open class Person  
  
class Employee: Person**()**{  
 fun *vaction*Days**(**days: Int**)**{  
 if**(**days < 60**)** println**(**"You need more vacation"**)** }  
}  
  
class Contractor: Person**()**fun hasVacations**(**obj: Person**)**{  
 if**(**obj is Employee**)**{  
 obj.vactionDays**(**20**)** }  
}

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

یعنی کامپایلر میاد میگه " من متوجه میشم که زمانی این کد اجرا میشه که obj یک Employee باشه و نیازی نیست که مثل جاوا اینجوری (Employee)obj.vacationDays(20) بنویسیمش

درواقع خود کامپایلر برامون انجامش میده و به اندازه کافی هوشمند هست که بتونه این کار رو انجام بده.

حالا اگه برگردیم به قسمت Nulls و کدمون رو نگاه کنیم

fun main(args: Array<String>) {  
 var nullMessage :String? = null  
 nullMessage = "Some Value"  
 println**(**nullMessage.length**)**  
}

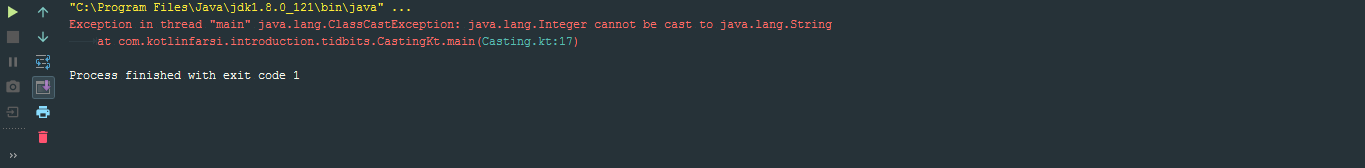
درواقع کامپایلر اینجا میگه "من میدونم این متغیر رو جوری تعریف کردی که میتونه null باشه، ولی خب قبل چک کردنش بهش مقدار دادی، پس من خودم به یک رشته(غیر قابل null شدن) تبدیلش میکنم که نیازی نداشته باشه که چک بشه!"

خب حالا فهمیدیم که کامپایلر هوشمنده و بعضی جاها خودش cast رو انجام میده، ولی اگه بعضی جاها ما خودمون بخوایم این casting رو انجام بدیم چیکار باید بکنیم؟

درواقع با کلیدواژه "as" این کار رو انجام میدیم.

var input : Any = 10  
fun main**(**args: Array<String>**)** {  
 val str = input as String  
 println**(**str.length**)**}

اینجا گفتیم که میدونیم input از جنس Any هستش، ولی اینجا میخوایم به String کستش کنیم و طولش رو برام بگیریم. تنها کاری که کردیم اون وسط این بود که از کلیدواژه as استفاده کردیم.

البته خب قبل اجرا هیچ مشکلی نداشتیم و هیچ اروری هم نبود. ولی خب وقتی کد رو اجرا میکنیم به ارور میخوریم

و اونم به این دلیله که اگه یادتون باشه گفتیم هیچ گونه کستینگ به صورت لفظی در کاتلین نداریم و اگه بخوایم یک متغیر رو کست کنیم باید از توابعش استفاده کنیم.

خب حالا ما چطوری میتونیم این کارو به یک روش امن انجام بدم؟

کاری که اینجا انجام میدیم اینجوریه

var input : Any = 10  
fun main**(**args: Array<String>**)** {  
 val str = input as? String  
 println**(**str?.length**)**}

به این معنی که من این دفعه "تلاش" میکنم که input رو به String تبدیل کنم.