[Polymorphisin]

① ← إذا حاولنا نحط subclass object وبستخدمه بمكان محجوز لـ superclass object

بزبط معاها عادي ✓ ← لأنه تابع إله

② ← بقدر أخلي متغير من نوع الـ superclass يأشر على object من نوع الـ sub

→ assigning the address of subclass object to a variable of superclass object

Dynamic Binding ⇒ which specific method to be invoked by the JVM for a specific object determined by an actual type ✓

One Object might have two types

declared type — actual type

→ that means the instance may be created using the constructor of the declared type or its subtype

→ But when it is not created a variable of the reference type can hold a null

▸ Subject : .....

JVM → decides which method to invoke at runtime, and starts searching from the actual type then it goes higher, once it finds the method, it will be implemented.

This is called Binding

Compiler → getting sure that the method is existing in the hierarchy, and starts searching from the higher superclass, and once it finds the method, a syntax error won't be appearing.

called Matching signature

The Job of JVM and compiler are two separate issues

JVM	compiler
depends on actual type	depends on declared type.
→ it does the work at runtime	→ it does the work at compile time.

[casting objects]

Hierarchy of objects

Declared Type ← classes in between ← Actual Type

suppose you declared the object, you can down cast it into any type between the the declared type and the actual type in the hierarchy

Upcasting → done in polymorphism implicitly

DownCasting → done to compromise, and done explicitly.

← بنعمل ال casting عشان نوصل ال compiler ← access على
ال class الي بين ال actual type ، وأي class في ال hierarchy
بين ال actual وال declared

Subject : .... / /

when you want to down cast, you should ensure that the object you want to cast is an instance of the subclass.

⇒ The error occurring if the Object is not an instance of the subclass is a Runtime error

↳ ClassCast Exception

to avoid errors use the operator instanceOf

subtype عرف سواء كان supertype or ال subtype ال

→ casting a primitive type value is different from casting an object reference

→ casting a primitive type value returns a new value, but in casting object it does not.

Subject :

polymorphism is a powerful tool for generic programming.

بتمكني أحطي النوع العام بيستقبل داخله النوع الخاص

← يعني حكى إني أستخدمها لتطبيق عمومي النوع، إني أعمل method
داخل class عام، ال parameter فيه بيحتوي declared type هو النوع
العام نفسه، ولكن أقدر أمرقله أي object ال actual type تبعها
يتبع لنوع خاص، لحتمية اشتراك جميع الأنواع الخاصة بنفس الميثود، وإنه رح
احتاج اطبق الميثود عليهم كلهم.

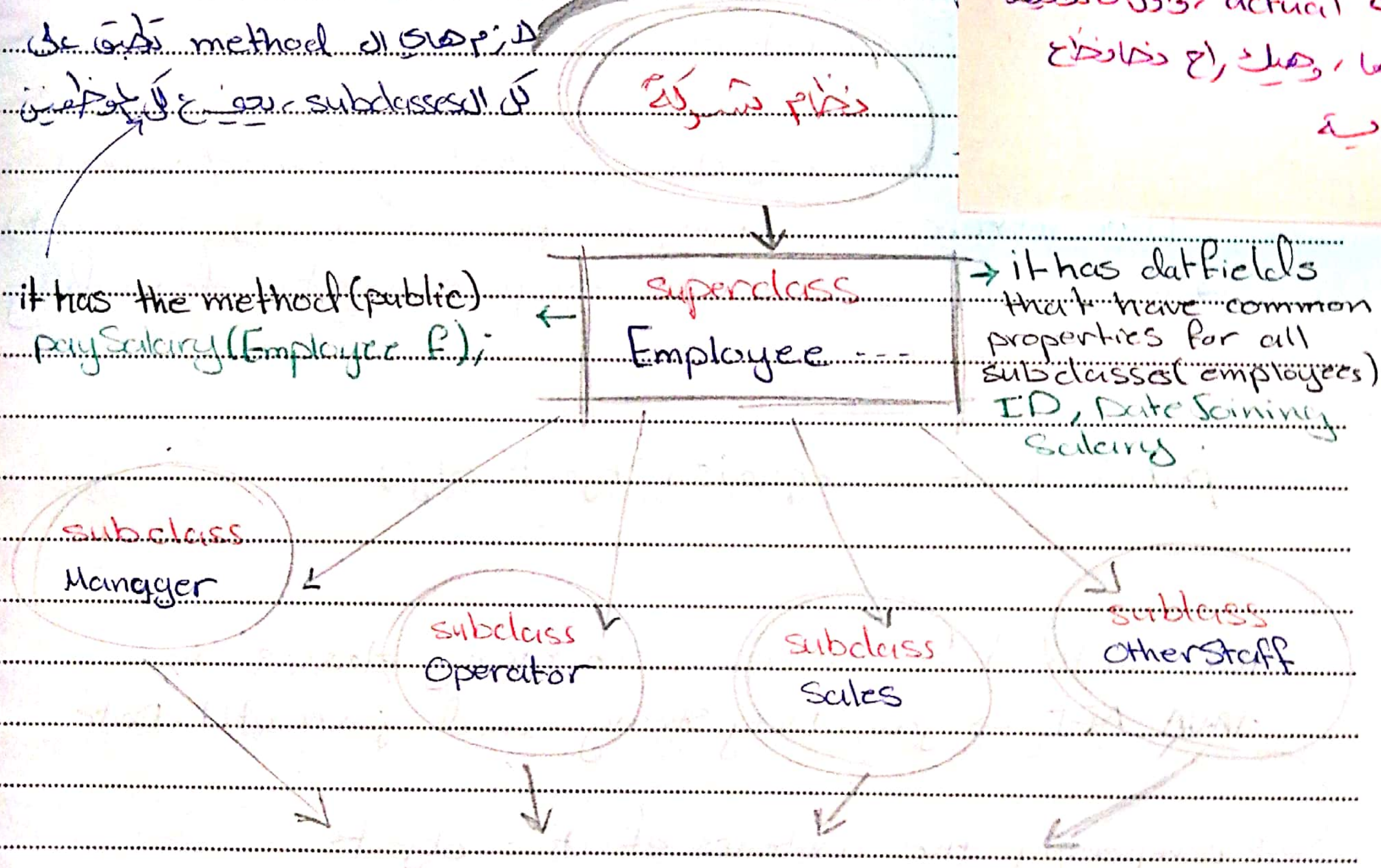
Basically, casting is important because the declared type decides which method to match at compile time.

To Enable generic programming, it is good practice to define a variable with a supertype, which can accept an object of any subtype.

Subject :

ممكن أضطر أعمل overriding لهاي الميثود الرئيسية في بعض ال subtypes عشان طريقة التعامل الراتب تكون مختلفة. لذلك ال JVM لما تبلش تبحث عن الميثود الي بدها تنفذها بتبلش من ال actual type، وأول ما تلاقيها راح تنفذها، وهيك راح نحافظ على الأولوية

Why Casting is Necessary?



← راح نعمل مصفوفة للنظام Datastructure in an Array

← راح يكون نوع ال Array ← ال declared type ← Employee class
لأنه كلهم موظفين

← كل element داخل ال Array، ال actual type بتبعه نوع subclass مختلف، يعني كل نوع موظف يختلف حسب كل واحد

وهاد هو احد خصائص ال polymorphism

وراح نضطر نعمل DownCasting عشان نخلي ال compiler قادر يوصل لأي شي جوا ال subtype.

▸ Subject : ..... / /

[Equals method]

→ defined in object class

→ compares two objects depending on references (two references pointing at the same object or not)

→ it can be overriden to compare objects depending on else things

public boolean equals (Object obj)

This method is overriden in many classes in JAVA API → java.lang.String and java.util.Date

↳ comparing the contents of two objects

→ we should use downcasting as the example in slide 50

[Protected keyword]

any thing declared protected is accessible in:

→ same class
→ any class in the same package or any where in the same package → وهاي لسّة مابتأثر، لأنه البكج نفسه بمثل الـ owner للكود، وبكون بعرف شو بستخدمه
→ any subclass outside the package

Modifying visibility

private	public	protected
members are not intended for use from outside the class	members are intended for all users of the class	methods are intended for the extenders of the class but not for the users of it

▶ Subject : ..... / /

[Notes]

(1) A subclass cannot weaken the accessibility of a method defined in the superclass when overriding it, but you are allowed to widen it

(2) you can prevent extending some class, or overriding some method use final method

Math, String, StringBuilder, StringBuffer are final classes