An overview of python's super( ) Function

super( ) gives you access to methods in a superclass from the subclass that iherits from it.

- وقتی super() را صدا می‌زنیم . اول method مورد نظرمون هم روش صدا می‌کنیم .

این باعث می‌شه بتونیم اول methodو فقط درجایی که بخوایم override کنیم بدون این که تغییر کنه در Parent class .

super can also take two parameters

the first is the Subclass

the second is an object that is an instance of that subclass

Ex) super (square, self) ⟶ square بعدی super داریم می‌گیم برو را صدا کن **این‌کار به چه دردی می‌خورد؟** باعث می‌شه methodهایی که اسم مثل هم دارن باهم قاطی نشن

و دقیقاً اون متدی که در super یک کلاس مشخص وجود داشته رو صدا بزنیم .

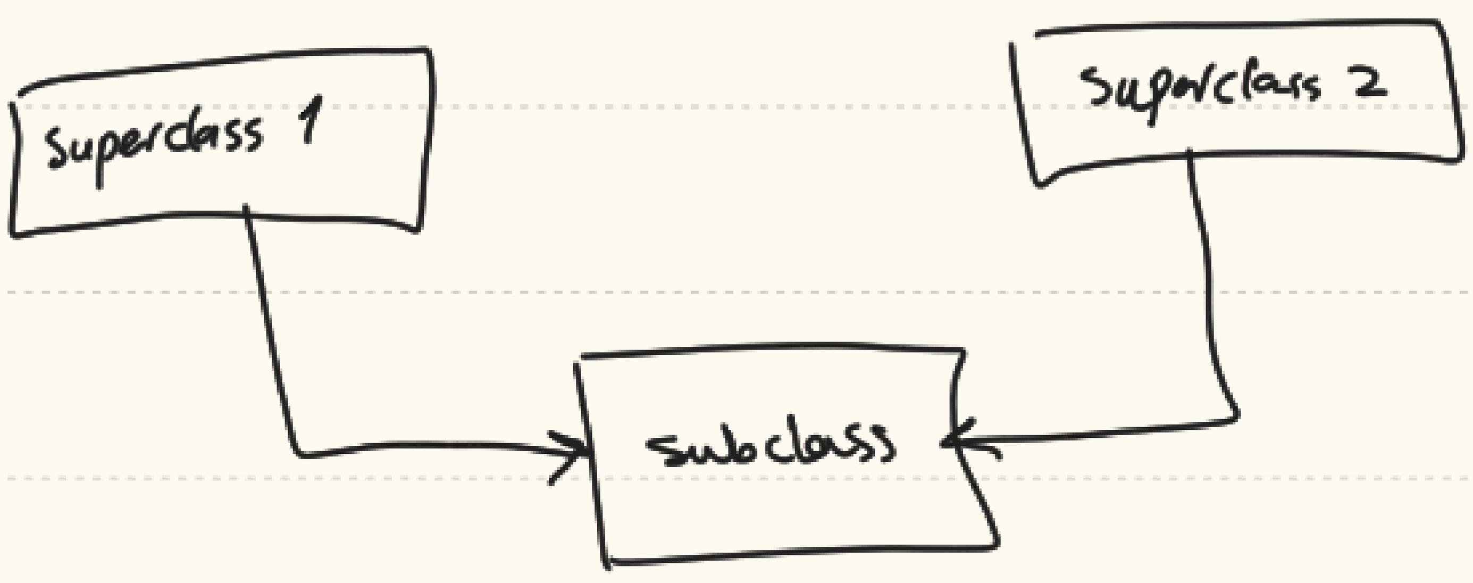
اما اگر نیاز باشه از این روش بطور مداوم استفاده کنیم در واقع design issue وجود دارد .

اکثراً ما باید از super( ) بصورت parameterless استفاده کنیم .

Super( ) doesn't return a method, it returns a proxy object

← یک objectی که class method مورد نظر را صدا می‌کند بدون این که نیاز داشته باشیم دوباره آن را تعریف کنیم

Super( ) in Multipule Inheritence



Method Resolution Order

MRO tells python how to search for inherited methods.

every class has an .__mro__ attribute that allows us to inspect the order

we have some control over how MRO is constructed. hard to explain

وقتی از multiple inheritence استفاده می‌کنیم همه اسم methodها باید unique باشه که باهم قاطی نشن موقعی که یک method را صدا می‌زنیم و در MRO به درستی ordered بشه .

*args **kwargs → وقتی یک function داریم که تعداد ورودی مشخص ندارد

represents non-keyword positional arguments

*args ⇒ ورودی را گرفته و به صورت tuple برمی‌گرداند و در متغیری به نام args ذخیره می‌کند.

**kwargs ⇒ ورودی را گرفته، به صورت dict برمی‌گرداند و در متغیری به نام kwargs ذخیره می‌کند.

represents keyword arguments (as key, value pair)