Java OOP ရဲ့အျခခံသေဘာတရားေလးေတြကုိအဓိကထားေရးထားပါတယ္။

**💻The Basic Knowledge of Java OOP in One Place**

အရင္တုန္းက Java ကိုစသင္တုန္းက Java နဲ့ပက္သက္ျပီး program ေသးေသးေလးေတြအမ်ားၾကီးေရးျဖစ္တယ္။

Object ကုိသုံးျပီးေရးတဲ့ program ေတြလည္းပါတယ္။ Java ရဲ့ class တစ္ခုကုိ object ေဆာက္မယ္၊ ေဆာက္ထားတဲ့ object ကေန အဲ့ဒီ class မွာရွိတဲ့အရာေတြကုိေခၚသုံးတက္ခဲ့တယ္။ ျပီးေတာ့ object ဆုိတာဘာလဲလုိ့နည္းနည္းနားလည္ခဲ့တယ္။ မွတ္မွတ္ရရ ဆို KMD ကဆရာတေယာက္ေျပာတာပါ၊ Java programming ရဲ့ object ဆိုတာ real world မွာရွိတဲ့အရာအားလုံးကုိ ေျပာတာလုိ့သူကသင္ေပးခဲ့တယ္။

အဲ့ဒီတုန္းကေတာ့ ကုိယ္ေတြက ေက်ာင္းေတြမွာပဲျဖစ္ျဖစ္ တျခားသင္တန္းေတြမွာပဲျဖစ္ျဖစ္ object ဆိုတာဘာလဲလုိ့ေမးရင္ထထေျဖျပီး အေပ်ာ္ၾကီးေပ်ာ္ခဲ့ဖူးတယ္။ ☺

ဒါေပမယ့္ Java ဟာ Object Oriented Programming Language တခုျဖစ္တယ္၊ Java ဟာ 100% OOP ျဖစ္တယ္။

အဲ့လုိေျပာတာေတြကုိသိပ္လဲနားမလည္သလုိ အေရးလည္းမထားခဲ့ဘူး။ ကုိယ္ဘာကုိယ္ real world မွာရွိတဲ့အရာမွန္သမ်ွကုိ object လုိ့ေခၚတယ္လုိ့သိျပီးပဲေက်နပ္ေနခဲ့တယ္။

**►So why they call Java is Object Oriented Programming Language and Java is 100% OOP?**

ဘာလုိ့မသိလည္းဆုိေတာ့ တကယ္ေတာ့ဒီေနရာမွာ က်ြန္ေတာ္တုိ့ဟာ object ကုိသုံးတတ္ယုံ နားလည္ယံုထက္ပုိျပီးသိဖုိ့လုိအပ္ေနတာပါ။

အဲတာကဘာနဲ့တူလဲဆုိေတာ့ ဥပမာ၊

(စကားခ်ပ္။ ကြန္ပ်ဴတာတကၠသုိလ္ကအေၾကာင္းေလးလဲေရးခ်င္တာနဲ့စာရွည္သြားရင္သီးခံပါရန္)

က်ြန္ေတာ္ငယ္ငယ္တုန္းက chess ထိုးတာေတာ္ေတာ္၀ါသနာပါတယ္၊ ေတာ္ေတာ္လည္းထိုးတတ္တယ္လုိ့ကုိယ့္ကိုယ္ထင္ဖူးတယ္။ သူငယ္ခ်င္းေတြထဲမွာ ျပီးေတာ့လမ္းထဲက chess ထုိးတဲ့ဦးေလးၾကီးေတြထဲမွာေတာ့ကုိယ္က ဆရာၾကီးပဲ။

ဒီလုိနဲ့တေန့ေတာ့က်ြန္ေတာ္တုိ့ ကြန္ပ်ဴတာတကၠသုိလ္က က်ြန္ေတာ္ေက်ာင္းေနာက္ဆုံးနွစ္မွာ အားကစားျပိဳင္ပြဲေလးေတြစျပီးလုပ္လာတယ္။ အဲ့ဒီနွစ္ကေနစျပီး အခုထိနွစ္တုိင္းေက်ာင္းမွာျပဳိင္ပြဲေတြလုပ္ျဖစ္တယ္ထင္တယ္။ အဲဲ့ေတာ့လုိရင္းက ျပိင္ပြဲမွာ chess ျပိဳင္ပြဲလည္းပါလာတယ္ဆုိပါေတာ့။ အဲ့ေတာ့က်ြန္ေတာ္လည္းဘာရမလဲ ပုိင္ျပီဆုိျပီး၀င္ျပိဳင္ေတာ့တာေပါ့။

ေတာက္ေလ်ွာက္နုိင္လာျပီး semi-final ေရာက္ေတာ့မွရူွံးေတာ့တာပဲ။ best of 3 မွာတစ္ပြဲမွမနုိင္ပဲ လက္ေရအကြာၾကီးနဲ့ကုိရွုူံးတာ။ အဲ့မွာအားေပးတဲ့သူငယ္ခ်င္းေတြေရာကုိယ္ေရာအံံ့ေတြေအာျပီး အၾကီးအက်ယ္ကြဲေတာ့တာပဲ၊ အရွက္။ ☺

ေနာက္မွသိတာကသူက chess စာအုပ္ေတြဖတ္ျပီးထိုးတာကုိး။ အဲ့ေတာ့မွကုိယ္လဲ chess စာအုပ္ေတြဖတ္ျပီးေလ့လာၾကည့္ေတာ့မွ၊ chess opening ေတြ defensive ေတြ၊ chess နယ္ရုပ္ေလးေတြကုိဘလုိေနရာမ်ိဴးမွာထားသင့္လဲဆုိတာေတြသေဘာေပါက္လာတယ္။

အဲ့ေတာ့ Java ရဲ့ OOP မွာလဲ အဲ့အတုိင္းပါပဲ။ က်ြန္ေတာ္တုိ့က object ကုိ သံုးတတ္ယံု နားလည္ယံုထက္ပုိျပီးသိဖုိ့လုိေနတာပါ။ အဲလုိမ်ဴိးပုိသိဖုိ့အတြက္ Java OOP ရဲ့အဓိက features ေတြျဖစ္တဲ့

✔Encapsulation

✔Inheritance

✔Polymorphism

✔Abstraction

စတာေတြကုိေကာင္းေကာင္းနားလည္ျပီး သံုးတတ္ရမွာပါ။ အဲ့လိုနားလည္မွ Java is 100% OOP, Java is OOP Language စတာေတြကုိဘာလုိ့ေျပာေနၾကလည္းနားလည္မွာပါ။

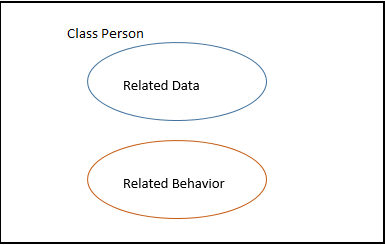
ဒီေနရာမွာေတာ့က်ြန္ေတာ္အၾကမ္းဖ်င္းနားလည္ယုံေလာက္ေျပာသြားျပီး ေနာက္မွတခုခ်င္းဆီကုိ အေသးစိတ္သက္သက္ဆီေရးသြားပါမယ္။

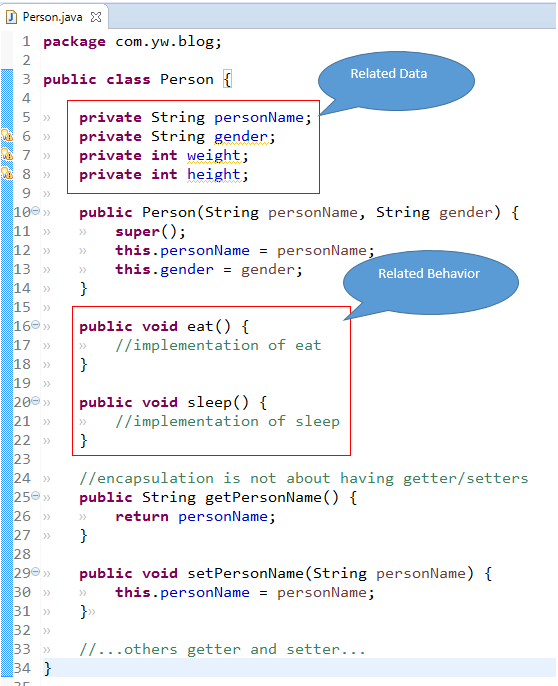
**1. Encapsulation**

Encapsulation ဆုိတာ object oriented programming (OOP) ရဲ့ concept တစ္ခုလုိ့ေျပာလုိ့ရပါတယ္။

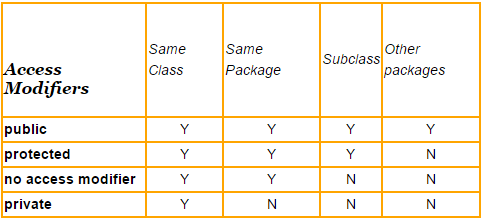
အဲ့တာေၾကာင့္မုိ့လုိ့ Abstraction နဲ့ေရာသြားနုိင္ေပမယ့္ Abstraction ကအဲ့တာထက္ပုိပါတယ္။ ဒါဆုိက်ြန္ေတာ္တုိ့ Java Encapsulation ဆုိတာဘာလဲ၊ OOP ရဲ့ဘယ္လုိ concept မ်ိဴးလဲ၊ Java မွာဘယ္လုိသံုးလုိ့ရတာလဲ ဆုိတာ သိေအာင္လုပ္ၾကရေအာင္။

ဒီေအာက္ကပုံ၂ ပုံကုုိအရင္စဥ္းစားၾကည့္ေပးပါ။





အေပၚကပုံေတြက Encapsulation ရဲ့ဥပမာေတြပါ။ သင္လဲသတိထားမိမယ္ထင္ပါတယ္။ ပုံအရဆုိ Encapsulation ဆုိတာ Java Class ေတြနဲ့သက္ဆိုင္တဲ့ (စကားခ်ပ္၊ package ေတြလည္းပါပါတယ္၊ ေနာက္ topic တစ္ခုေနနက္ဆက္ေရးပါမယ္။) data ေတြ၊ သူ့ရဲ့ behavior ေတြကုိ အသုံးျပဳေနၾကတဲ့ object ေတြရဲ့အသုံးျပဳနုိင္ခြင့္ေတြကုိထိန္းခ်ဴပ္တာပါ။ Data hiding လုိ့လည္းေခၚၾကပါတယ္။ ဒါေပမယ့္Encapsulation ဆုိတာ သက္ဆုိင္တဲ့အရာေတြကုိစုထားျပီးေတာ့ သူ့ကုိအသုံးျပဳေနတဲ့သူေတြကုိ hide လုပ္လုိက္တာပါ၊ data ေတြကုိ hide လုပ္ျပီးေတာ့ behavior ေတြကို အကန့္အသတ္နဲ့ေပးသုံးလုိက္တာပါ။ အဲ့ဒီ concept ကုိအေျခခံျပီးေတာ့ programmer ေတြစိတ္ၾကိဳက္ design ခ်ရမွာပါ။ Well defined Java Class ေတြအကုန္လုံး Java Encapsulation ရဲ့ဥပမာေတြပါပဲ။ Encapsulation ကုိမ်ားေသာအားျဖင့္ Java ရဲ့ Access Modifier ေတြသုံးျပီး design ခ်ပါတယ္။



**2. Inheritance**

ဒီေနရာမွာ Inheritance ကုိအေျခခံေလာက္သိဖုိ့ကေတာ့လြယ္ကူပါတယ္။ Java မွာ Inheritance ကုုိသံုးမယ္ဆုိရင္ extends keyword ကုိ သံုးရပါတယ္။ အေမြဆက္ခံရမယ့္ class ကုိ base class or super class လုိ့ေခၚျပီး၊ အေမြဆက္ခံမယ့္ class ကုိ derived class or sub class လုိ့ေခၚပါတယ္။

အဲ့ေတာ့ Inheritance ဆုိတာ sub class က super class ကိုအေမြဆက္ခံျပီး super class မွာရွိတဲ့ data ေတြ behavior ေတြကုိပုိင္ဆုိင္ခြင့္ရတာပဲျဖစ္တယ္။ အကုန္လုံးပုိင္ဆိုင္ခြင့္ရမရေတာ့ သင္က အဲ့ဒီ super class ကုိ Encapsulation ကုိသုံးျပီး design ဘယ္လုိခ်ထားမလဲဆုိတာနဲ့သက္ဆိုင္ပါတယ္။

Java Inheritance ရဲ့တစ္ျခား rule ေတြကေတာ့ super class က sub class ေတြအမ်ားၾကီးရွိလုိ့ရျပီး၊ sub class ကေတာ့ super class တစ္ခုတည္းရွိရပါမယ္။ ဘာလုိ့လည္းဆုိေတာ့ Java မွာ multiple inheritance ကုိလက္မခံလုိ့ပါ။

**3. Polymorphism**

Poly ဆုိတည္းကတစ္ခုထက္မကပါ။ Polymorphism ကုိ Inheritance structure ေတြအမ်ားၾကီးရွိတဲ့ေနရာေတြမွာျမင္ေတြ့နုိင္ပါတယ္။ အဲ့လုိ Inheritance structure အဆင့္ဆင့္ရွိတဲ့ super class ေတြကို object ေဆာက္တဲ့အခါမွာ ဘယ္ super class ေတြကုိ object ေဆာက္တာလဲ၊ ဘယ္ super class ရဲ့ data ေတြ behavior ေတြကုိေခၚသုံးမွာလည္းဆုိတာဆုံးျဖတ္ရပါတယ္။ Java မွာ compile time and run time ဆုိျပီးနွစ္မ်ဴိးရွိပါတယ္။ အဲ့လုိပဲ Polymorphism ကလဲ ဒီ object ကဘယ္ super class ကုိ object ေဆာက္ျပီးေခၚသုံးေနလည္းဆုိတာ compile time and run time မွာဆုံးျဖတ္ပါတယ္။ အဲ့တာကို Polymorphism လုိ့ေခၚလုိ့ရပါတယ္။ ဒီေနရာမွာ Polymorphism ကိုေျခခံေလာက္နားလည္ထားရင္လုံေလာက္ပါတယ္။ ေနာက္သက္သက္ topic တစ္ခုေနနဲ့ example program ေတြထဲ့ျပီးဆက္ေရးသြားပါမယ္။

**►My Conclusion**

ဒီေနရာမွာက်ြန္ေတာ္ Java Abstraction ကုိခ်န္ခဲ့ပါတယ္။ Abstraction ဆုိတာ abstract class ေလးလို့ပဲလုိ့မွားတတ္ပါတယ္။ အဲ့ထက္ကုိပုိပါတယ္၊ OOP ရဲ့အဓိက concept ျဖစ္သုိ Java တစ္ခုလုံးနဲ့သက္ဆိုင္တာပါ။ topic အသစ္နဲ့ေရးသြားပါမယ္။

အဲ့ေတာ့က်ြန္ေတာ္ေျပာတဲ့ Java OOP ရဲ့အဓိက features ေတြဟာ object ကုိအေျခခံျပီးေတာ့ Java ရဲ့ေနရာတုိင္းမွာအသုံးျပဳေနတာပါ၊ အရမ္းလည္းအသံုး၀င္ပါတယ္။ မပါမျဖစ္ပါပဲ။ အဲ့ဒါေၾကာင့္မို့မျဖစ္မေနနားလည္ဖုိ့လုိျပီးေတာ့၊ Java is Object Oriented Programming Language, Java is 100% OOP ျဖစ္ေနတာပါ။ အေသးစိတ္အဆက္ေတြကုိဆက္ေရးသြားပါဦးမယ္။