La classe et l’objet

Java est un langage orienté objet

Un langage orienté objet permet d’offrir une vision plus proche de la réalité physique

La différence entre classe et objet :

La classe définie les caractéristique du concept en générale par exemple du voiture

Et un objet qu’on parle d’une voiture en particulier avec cette caractéristique

Une classe dispose d’un nom qui commence par un majuscule suivi par des caractère minuscule et sans accent lord ce que la classe ce compose de plusieurs mot chaque mot commence par un majiscule Exemple classe Voiture java

public class Planete {  
 String nomP ;  
 String mariereP ;  
 long diametreP ;  
}

**NB :** String est une classe

A la création d’une classe il faut indiquer qu’elles sont ces propriétés

Sin on donne pas des valeur par default pour les propiété crées il prendron des valeur par default ca dépend le type du propriété

EXP de valeur par default

Int 🡪 0 ; float 🡪0,0 ; String 🡪Null ; boolean 🡪 false

char 🡪 un caractère sans valeur

L’objet ou l’instance

C’est un exemplaire de la classe on l’appel aussi instance de la classe

A la création d’un objet on dit instancier la classe par exemple a la classe Voiture on va crée un objet avec la commande new voiture() ici il vas crée l’objet et le mettre dans la mémoire

**NB :** l’association de l’objet a une variable qui va référencer l’emplacement de cette objet dans la mémoire

public class helloUniverse {  
  
 public static void main(String... args){  
  
 Planete soleil = new Planete();  
 soleil.nomP="solei";  
 soleil.mariereP="gazeuse";  
 soleil.diametreP=1391000l;

System.*out*.println("le nom du planete 1 est : " + soleil.nomP + "matire du planete est ; " + soleil.mariereP + " le diametre du planete est : " + soleil.diametreP);

}

}

Dans cette exemple on instacier l’objet soleil de la classe planete

NULL signifie que le variable n’a aucune valeur

Les méthodes

Un classe est definie par des attribu egalement par des méthodes …

Une méthode represente une operation ou une activité qui caractèrise la classe

Void NomMethode() : ici a la déclaration d’une méthode Void(vide) signifie la methode ne retourne pa de résulta

Declaration d’une methode dans la classe planette

void revolution(){  
 System.*out*.println("je suis la planete " + nomP + "je tourne autour de mon etoile");  
}

l’execution du methode dans la classe main pour la planete(objet) mars déjà crées dans la classe mail

mars.revolution();

methode avec return

int accelerer(){  
 System.*out*.println("j'accelère");  
 return 5;  
}

voitureR.accelerer();

Résulta : j'accelère ne donne pas ce qui est en return  
System.*out*.println(voitureR.accelerer());

Resulta : j'accelère

5 il donne aussi resulta du return

Ou

int nvv = voitureR.accelerer();  
System.*out*.println("la nouvelle vitesse est : " + nvv + "KM/h");

qu’on declare un objet java le declare dans un emplacement

a la declaration d’une string java stcké en mémoire un emplacement partuclier la chaine de caractere et a la declaration de la meme manier un dexieme string java va utiliser le prmier.

String Chaine1= ‘phrase 1 ‘ ; String chaine 2=’phrase 1’ ;

Boolean b = chaine1=chaine2 🡪 true ;

String Chaine1= new String(‘phrase 1 ‘) ; String chaine 2=new String(’phrase 1’) ;

Boolean b = chaine1=chaine2 🡪 false ;

La fonction equals compare la chaine caractère par caractère ;

equalsEgnorecase ne fait pas le difference entre le majuscule et le minuscule

Boolean b= chaine1.equals(chaine2) ;

Boolean b= chaine1.equalsEgnorecase(chaine2) ;

La surcharge du méthodes c’est de crée deux méthodes avec le meme nom une avec des parametre l’autre sans parametre.

Le mot clé this

Permet de faire reference a un objet en coure d’utilisation

Qu’on un parametre qui a exactement le même nom que l’attribue de la classe on ecri this.attribue\_Classe pour que java puisse faire la différence

Exemple

int vitesse ;  
int accelerer (int vitesse){  
 this.vitesse=this.vitesse+vitesse;  
  
 return this.vitesse ;  
}

**Proprieté sous forme d’objet**

Encapsuler un objet c’est de faire d’eclarer un variable de type classe\_B dans classe\_A

Exp : e crée la classe moteur et on déclare un variable dans classe voiture de type moteur

public class Moteur {  
 String carbuant ;  
 int cylindre ;  
}

public class Voiture {  
 Moteur moteur ;

]

public static void main(String[] args){

Moteur moteur1= new Moteur();

mohcine.moteur=moteur1; on peut affecter les attribue de la classe moteur a plusieur objet voiture  
reda.moteur=moteur;

NB : les methode peuvent prendre un argument de type objet

**Exemple**

public class Passager {  
  
 String nomPassager;  
}

création d’une methode qui prend un argument de type Passager crée dans la classe precedente

public class Voiture {

void transporter(Passager passagera){  
 System.*out*.println("je transporte un passager qui s'appel : "+pass.nomPassager);  
}

L’utilisation de la methode crée dans la classe main

public static void main(String[] args){

Voiture reda = new Voiture();  
reda.nom="Reda";

reda.nbrPort= 2 ;

Passager pass1=new Passager();  
pass1.nomPassager="yazid";  
reda.pass=pass1;  
reda.transporter(pass1);

Resulta

je transporte un passager qui s'appel : yazid

**Exemple 2**

Création dun classe Ville + les 2 autre Passager et Voiture

public class Ville {  
 String nomVille;  
}

public class Voiture {

Passager pass ;  
Ville vil;

Ville transporter(Passager passagera,Ville villeDeppart){  
 System.*out*.println("je transporte un passager qui s'appel : "+pass.nomPassager);  
 System.*out*.println("le passager est parti de la ville de : "+vil.nomVille);  
  
 Ville villeDest=new Ville();  
 villeDest.nomVille="rabat";  
 return villeDest ;  
}

public static void main(String[] args){

Voiture reda = new Voiture(); Création d’un objet voiture  
reda.nom="Reda";

Passager pass1=new Passager(); création d’un objet passager  
pass1.nomPassager="yazid"; affectation des donnée aux attribues de l’objet passager crée  
reda.pass=pass1; affectation des valeurs de l’objet passager (passe1) a la voiture reda

Ville rabat=new Ville();   
rabat.nomVille="casablanca";  
reda.vil=rabat;

Ville destination= reda.transporter(pass1,rabat); création un variable de type Ville pour recevoire le return de la methode transpport.

System.*out*.println("le passager arrive à :"+destination.nomVille);

}

**Résulta**

je transporte un passager qui s'appel : yazid

le passager est parti de la ville de : casablanca

le passager arrive à :rabat

**Les Proprieté / attribue static ou variable de classe**

Une proprieté ou un attribue static ca veut dire que sa valeur ou la référence à un objet et la meme quelque soit l’instance(l’objet) sauf si on l’a mofidié

Exemple : crées une propriete static forme sui sera fixe pour tous les planette crée et de la tester avec la planette mars

public class Planete {  
 public static String *forme*="SPHERE";

}

public static void main(String... args){

Planete mars = new Planete();  
mars.nomP="mars";  
mars.mariereP="tellurique";

System.*out*.println("la forme de tous les planetes est : "+Planete.*forme*);  
System.*out*.println("la forme de mars est :"+ mars.*forme*);

}

**Résulta**

la forme de tous les planetes est : SPHERE

la forme de mars est :SPHERE

**Les méthodes static**

Une methode static c’est une méthode qui n’applique aucun attribue d’instance

Exemple pour la methodes qu’on utilisé

static void klaxonner(){ ici on a ajouté static cette methode ne depend pas des attribue de l’objet  
 System.*out*.println("tuuuuuuutuuuuuutuuuuu!!!!!");  
}

Exp 2

static tourner(boolean droite,int anglet){  
 String direction = null;  
 if(droite){  
 direction="droite"; cette modification est accepter car on n’implique pas les attribue de l’instance  
 }  
 else {  
 direction="gauche";  
 }  
 System.*out*.println("la voiture est tourné à " + direction + " d'une anglet de : " + anglet);  
}

exp 3 NB cette methode ne peut pas etre static car il modifie une proprieté de l’instance(vitesse)

int rapportC(boolean b){  
 if (b){  
 vitesse++ ;  
 }  
 else {  
 vitesse-- ;  
 }  
 return vitesse;  
}

NB probleme de compilation j’ai changé au code ajouter un 2 eme return dans la methode pour pouvoire executer le code

**Constructeur sans parametre et par défaut**

On peut instancier la classe pour donner des instruction a chaque création d’un objet de cette classe

Exemple la classe voiture

Voiture(){  
 System.*out*.println("la voiture est encoure de construction");  
}

Ce message sera affecter atous les voitures crée

public class Planete {  
 static int *nombreplanette*; le type static pour pouvoire utilister directement dans la classe main  
 Planete(){  
 *nombreplanette*++;  
 }

public static void main(String... args){  
  
 System.*out*.println(Planete.*expansion*(10.2d));  
 System.*out*.println(Planete.*expansion*(14.2d));  
 Planete soleil = new Planete();

Planete venus = new Planete();  
 soleil.nomP="venus";

Planete mars = new Planete();  
 soleil.nomP="mars";

Planete uranu = new Planete();  
 soleil.nomP="uranus";

System.*out*.println("le nombre des planette est Planete.nombreplanette);

Ajouter constructeur avec parametre

Voiture(){  
 System.*out*.println("la voiture est encoure de construction");  
}  
Voiture(String coleur){  
 this.coleur=coleur;  
 System.*out*.println("la vcouleur de la voiture est :"+this.coleur);  
}  
Voiture(int nbrou2){  
 this.nbrou2=nbrou2;  
 System.*out*.println("le nombre de roues est : "+nbrou2);  
}

Voiture voitureR5 = new Voiture();  
Voiture voitureR6 = new Voiture("jaune");  
Voiture voitureR7 = new Voiture(5);

Dans cette exemple java choisi le constructeur adéquat au type de la déclaration de l’objet voiture

On peut ordonné meme autre classe tel que la classe moteur dans les arguments du constructeur

Voiture(Moteur moteur){  
 this.moteur=moteur;  
 }