Analyse et Programmation Orientées Objets / C++

La relation de composition

Construction par assemblage (1)

➡ **Nature des attributs d'une classe**

- Attributs de types primitifs
- Attributs de types définis par l'utilisateur
- Attributs scalaires et vectoriels

Extension : attributs de type OBJET

Construction par assemblage (2)

➡ **Exemple**

- Description d'une voiture :
  - Un châssis **Classe Chassis**
  - Un moteur **Classe Moteur**
  - Une carrosserie **Classe Carrosserie**

- Description d'une carrosserie :
  - Modèle
  - Type
  - Couleur
  - ---

Construction par assemblage (3)

➡ **Objectifs de la relation "is_part_of"**

- Favoriser la réutilisation des codes sources

- Réutilisation des données
- Respecter les règles d'encapsulation
- Favoriser la conception par composition formelle

Construction par assemblage (4)

➡ **Portée de la relation**

- Relation exclusivement entre classes

- Classe composée
- Composants objets
- Structure hiérarchique à plusieurs niveaux

Déclaration d'une classe composée

➡ **Langage C++**

```
# include  "CY.h "
# include  "CZ.h "
class CX  {
private :
int    m_xxx;
CY   m_yyy;
CZ   m_zzz;
public :
--- Déclaration des méthodes

};
```

Constructeur d'instances de CX (1)

➡ **Première syntaxe en langage C++**

```
# include  "CX.h"

CX::CX (int v1, CY obj1, CZ obj2 )    {
    m_xxx = v1;
    m_yyy = obj1;
    m_zzz = obj2;
}
```

Constructeur d'instances de CX (2)

➡ **Seconde syntaxe en langage C++**

```
# include  "CX.h "

CX::CX (<list args> ) : m_yyy (<sub_list1 args>) ,
                        m_zzz  (<sub_list2  args>)    {
    --- Affectation des attributs propres à CX
}
```

Syntaxe déconseillée !

Déclarations des instances

➡ **Syntaxe en langage C++**

```
# include  "CX.h "

void main () {
 CX   x1 (<list params> );

}
```