

Algorithmique

Programmation Objet

Python



Andrea G. B. Tettamanzi

Université de Nice Sophia Antipolis

Département Informatique

andrea.tettamanzi@unice.fr

CM - Séance 5

Programmation orientée objet en Python, UML et patrons

Plan

- Programmation OO en langage Python
- Éléments d'UML
- Patrons de conception orientés objet

Programmation OO en Python

- Les objets sont l'abstraction des données en Python.
- Toutes les données dans un programme Python sont représentés par des objets, y compris le code.
- Un objet possède :
 - Une identité (= son adresse en mémoire) : `id()`
 - Un type (opérations supportées + valeurs possibles)
 - Une valeur (mutable ou immutable, selon le type)
- Chaque classe et instance de classe possède un espace de noms (*namespace*), réalisé par un dictionnaire.
- Python n'adhère pas au fait de protéger le code vis-à-vis du programmeur. Python encapsule les objets comme un *namespace* unique mais c'est une encapsulation transparente

Définition d'une classe

nom de la classe (identificateur)

superclasses desquelles la classe hérite

```
class NomCls (c11, c12, c13) :  
    """documentation"""  
    # bloc instructions, définition des méthodes, etc.  
  
    def __init__(self, arg1, arg2) :  
        """documentation"""  
        # initialisation d'une instance.  
        self.arg1 = arg1  
        self.arg2 = arg2
```

Pas de définition explicite d'interface, on utilise les classes pour cela

Création et utilisation d'une instance

```
instance = NomCls (arg1, arg2)
```

```
instance.méthode (x, y)
```

```
instance.attribut
```

```
NomCls.attribut_de_classe
```

UML

Le langage universel de modélisation (Unified Modeling Language, UML) est une famille de notations graphiques qui aide à décrire et concevoir des logiciels, notamment ceux construits en utilisant un style orienté objet

Ingrédients de UML

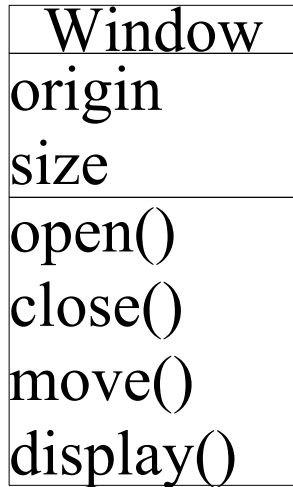
- Le “vocabulaire” de UML inclut trois types d'ingrédients :
 - Entités (choses, *things*): abstractions pertinentes
 - Relations : lient des entités entre elles
 - Diagrammes : regroupent ensembles intéressants d'entités

Types d'entités

- Structurales :
 - Classes, interfaces, collaborations, cas d'utilisation, composantes, nœuds
- Comportementales :
 - Messages, états
- De regroupement :
 - Paquets
- Annotations :
 - Notes

Entités structurales

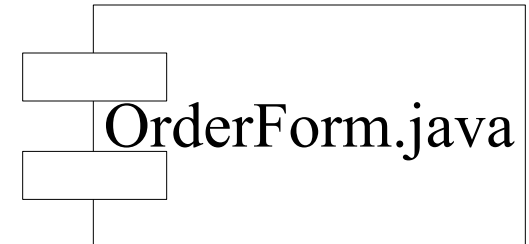
Classes



Collaborations



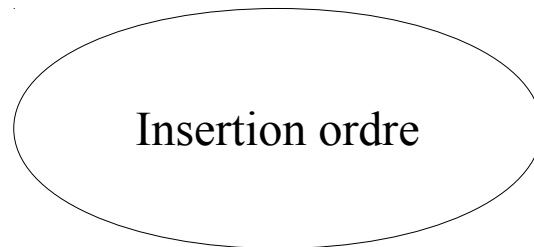
Composantes



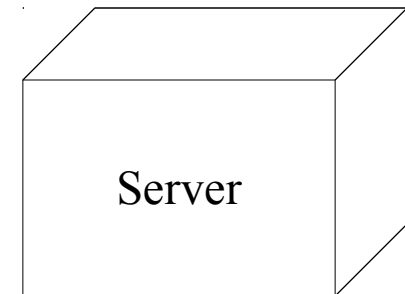
Interfaces



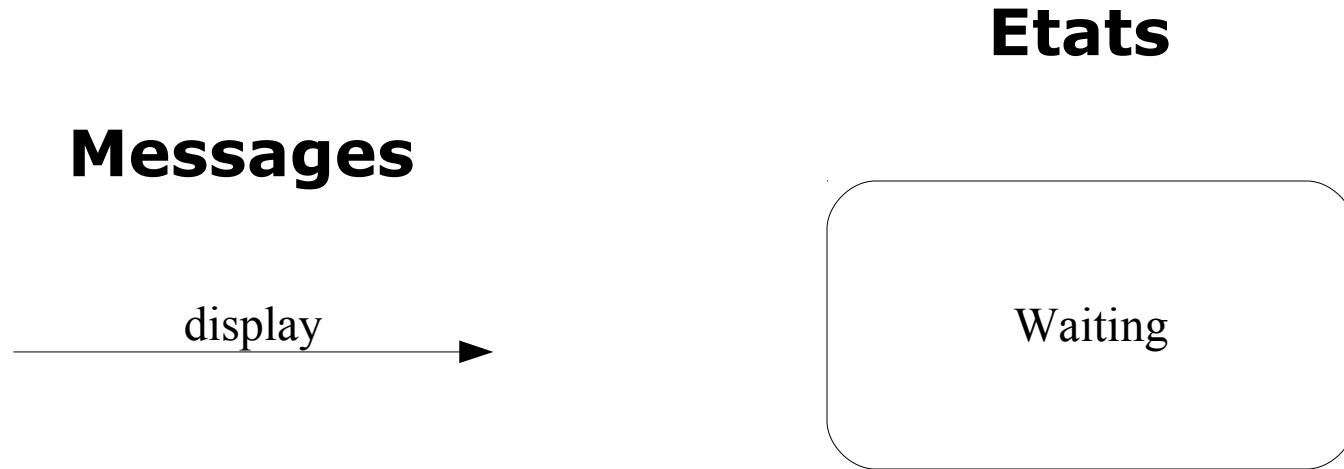
Cas d'utilistion



Noeuds



Entités comportementales



Entités de regroupement

Paquets



Entités d'annotation

Note

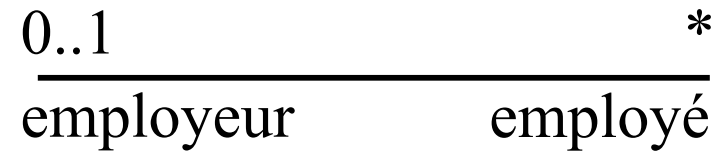
Revient avec une référence au client
stub de l'objet distant

Types de Relations

Dépendences



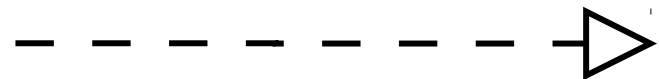
Associations



Généralisations



Réalisations



Diagrammes

- Il existe 13 types distingués de diagrammes UML :
 - Activité
 - Cas d'utilisation
 - Classes
 - Communication
 - Composantes
 - Déploiement
 - Interaction
 - Machines à états
 - Objets
 - Paquets
 - Séquence
 - Structure des composantes
 - Temporisation

Patrons de conception

La programmation orientée objet toute seule ne suffit pas.

Idée de “patron de conception” = schema.

Aggrégations de plusieurs classes logiquement liées.

Conception orientée objet (1)

Concevoir un programme orienté objet réutilisable signifie :

- Identifier des objets pertinents ;
- Les regrouper dans des classes de la juste granularité ;
- Définir leurs interfaces ;
- Définir la hiérarchie des classes et des interfaces ;
- Établir des relations significantes entre elles ;
- Répondre aux spécificités du problème ;
- Se maintenir assez généraux pour pouvoir traiter d'autres problèmes du même type.

Conception orientée objet (2)

Difficilement un projet « réussit bien » au premier essai :

- Il va être recyclé dans des contextes différents ;
- Il va être modifié à chaque fois pour répondre à des besoins différents ;
- Il pourra être amélioré ;
- Finalement, la meilleure solution sera trouvée.

Expertise de conception

La différence entre un concepteur débutant et un expert :

- Parcours par essais et erreurs ;
- répertoire de solutions déjà conçues ;
- Aptitude à trouver la solution la plus appropriée.

Autrement dit, le concepteur expert possède un répertoire de **schémas de conception** prouvés qu'il peut adapter à des nouvelles situations.

Ceci est le sens de **patron de conception**.

Patron de conception

Un patron de conception décrit :

- Un problème que l'on retrouve souvent en écrivant des programmes ;
- Le noyau de sa solution.

Structure d'un patron

Un patron de conception est constitué par :

- Son nom;
- La description du problème, du contexte d'application ;
- La solution :
 - Éléments (classes et objets) constitutifs ;
 - Les relations entre les éléments ;
- Les conséquences de son application :
 - résultats;
 - Avantages/désavantages.

Exemples de patrons

Des patrons sont la réédition orientée objet de techniques de programmation très anciennes :

ex. **interprète**.

D'autres patrons peuvent servir comme des briques de base pour construire des architectures logicielles versatiles et performantes :

ex. **adaptateur**, **itérateur**, etc.

Pour approfondir

Il existe des ouvrages qui examinent les patrons de conception les plus importants. Par exemple :

Gamma, Helm, Johnson, Vlissides

Design Patterns: Elements of reusable object-oriented software

Addison-Wesley, 1995

Merci de votre attention

