Génie logiciel

Notes du cours de 16/12

L3 Informatique appliquée 2022-2023

MABROUK Fayez

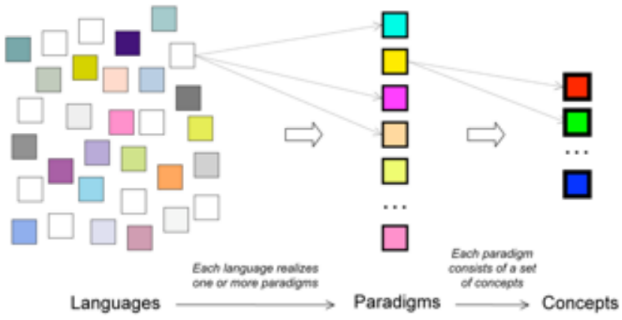
1er janvier 2023

1 Programmation

2 Paradigmes de programmation

2.1 Paradigmes de programmation

* Paradigmes de programmation Un paradigme de programmation est un style de programmation d'un ordinateur défini par un ensemble spécifique de concepts et de techniques de programmation. , incarné par son langage noyau, le petit langage de base dans lequel toutes les abstractions du paradigme peuvent être définies.



* Une langue peut réaliser un paradigme : les langues "pures.
* La plupart des langages permettent plusieurs paradigmes.
* Certains langages sont conçus pour réaliser plusieurs paradigmes : les langages multi-paradigmes.
* Un langage peut évoluer et réaliser de nouveaux paradigmes.

2.2 Ceux qu'il faut connaître

* Impératif : spécifie les instructions au programme (pour arriver à un résultat)
  * Programmation structurée : utilise un flux de contrôle structuré (conditions, boucles).
    * Programmation procédurale : utilise des procédures pour structurer le programme.
  * Programmation orientée objet : concept d'objets (données + code)
* Déclarative : spécifie le résultat au programme (pas les instructions)
  * Programmation fonctionnelle : un programme est une composition de fonctions.
  * Logique : expression en termes de formules logiques.

2.3 Règles générales pour un code propre

* Nécessité de normes de codage :
  * Uniformisation des codes écrits par différents développeurs.
  * Améliore la lisibilité
  * Améliore la maintenabilité
  * Améliore la réutilisation
  * (Peut) réduire la complexité

2.4 Développement piloté par les tests

* 3 lois (de Robert Martin) :
  * Vous ne pouvez pas écrire de code de production avant d'avoir écrit un test unitaire défaillant (test unitaire d'abord)
  * Vous ne pouvez pas écrire plus d'un test unitaire que ce qui est suffisant pour échouer, et ne pas compiler est un échec (un seul test unitaire échoué à la fois).
  * Vous ne pouvez pas écrire plus de code de production qu'il n'est suffisant pour passer le test en cours d'échec (le code ne doit résoudre que le test unitaire)
* Un concept par test.
* Répétable.
* 5 étapes pour ajouter une nouvelle fonctionnalité :

  * Ajoutez un test qui passera si et seulement si l'exigence donnée est satisfaite.
  * Exécuter tous les tests. Le nouveau test devrait échouer car la fonctionnalité n'a pas encore été implémentée.
  * Ajoutez la quantité minimale de code pour réussir le test.
  * Vérifier que tous les tests (y compris les précédents) passent.
  * Refactoriser pour améliorer la lisibilité et la maintenabilité.