Programmation multi-paradigme

Université Bretagne Sud
G.Ménier : gildas.menier@univ-ubs.fr

INF2162 Programmation multi-paradigme (Gildas Ménier)

En suivant ce cours, vous vous **engagez** à :

- Ramasser et éteindre votre téléphone avant le début du cours ou du TD. A ne pas l'utiliser en cours/TD.
- Participer à tous les cours et TDs (sauf urgence ou rendez-vous important : vous vous engagez à me prévenir avant).
- Rendre les TD sur la plateforme moddle AVANT l'heure et la date limite. Vous comprenez que 10mn avant l'heure limite, vous n'êtes pas assuré(e) de pouvoir le faire : il faut donc prévoir 20mn avant la date limite.
- Ne pas m'envoyer les TDs par email.
- Ne pas troubler cours et TD, ne pas apporter de nourriture ou de boissons visibles pendant cours et TD.
- Pas d'utilisation tablette ou PC sauf certificat médical / médecine universitaire.

En suivant ce cours/TD, vous vous engagez inconditionnellement à appliquer les règles indiquées. Tout manquement peut aller de l'annulation de la note d'une TD à l'exclusion des cours/TD de l'UE - voire à l'annulation de la note de l'UE.

Espace de cours

Vous trouverez ci joint les supports (incomplets) de cours et les espaces TD / rendu de TD.
Les supports de cours sont complétés au fur et à mesure des cours.

Il est fortement conseillé de prendre des notes : toutes les notions ne sont (volontairement) pas expliquées dans les supports de cours. Par contre, toutes les notions abordées (qu'elles soient ou pas dans les supports) sont à l'examen.

Scala_2022

The Importance of Cursive Handwriting Over Typewriting for Learning in the Classroom: A High-Density EEG Study of 12-Year-Old Children and Young Adults
Eva Ose Askvik 1 , F R Ruud van der Weel 1 , Audrey L H van der Meer 1

The Importance of Cursive Handwriting Over Typewriting for Learning in the Classroom: A High-Density EEG Study of 12-Year-Old Children and Young Adults

Eva Ose Askvik [1], F R Ruud van der Weel [1], Audrey L H van der Meer [1]

Affiliations + expand

PMID: 32849069 PMCID: PMC7399101 DOI: 10.3389/fpsyg.2020.01810

Free PMC article

Abstract

To write by hand, to type, or to draw - which of these strategies is the most efficient for optimal learning in the classroom? As digital devices are increasingly replacing traditional writing by hand, it is crucial to examine the long-term implications of this practice. High-density electroencephalogram (HD EEG) was used in 12 young adults and 12, 12-year-old children to study brain electrical activity as they were writing in cursive by hand, typewriting, or drawing visually presented words that were varying in difficulty. Analyses of temporal spectral evolution (TSE, i.e., time-dependent amplitude

› Front Psychol. 2017 May 9;8:706. doi: 10.3389/fpsyg.2017.00706. eCollection 2017.

Only Three Fingers Write, but the Whole Brain Works: A High-Density EEG Study Showing Advantages of Drawing Over Typing for Learning

Audrey L H van der Meer [1], F R Ruud van der Weel [1]

Affiliations + expand

PMID: 28536546 PMCID: PMC5422512 DOI: 10.3389/fpsyg.2017.00706

Free PMC article

Abstract

Are different parts of the brain active when we type on a keyboard as opposed to when we draw visual images on a tablet? Electroencephalogram (EEG) was used in young adults to study brain electrical activity as they were typing or describing in words visually presented Pictionary™ words using a keyboard, or as they were drawing pictures of the same words on a tablet using a stylus. Analyses of temporal spectral evolution (time-dependent amplitude changes) were performed on EEG data recorded with a 256-channel sensor array. We found that when drawing, brain areas in the parietal and occipital regions showed event related desynchronization activity in the theta/alpha

Motor control of handwriting in the developing brain: A review

Sarah Palmis [1], Jeremy Danna [1], Jean-Luc Velay [1], Marieke Longcamp [1]

Affiliations + expand

PMID: 28891745 DOI: 10.1080/02643294.2017.1367654

Abstract

This review focuses on the acquisition of writing motor aspects in adults, and in 5-to 12-year-old children without learning disabilities. We first describe the behavioural aspects of adult writing and dominant models based on the notion of motor programs. We show that handwriting acquisition is characterized by the transition from reactive movements programmed stroke-by-stroke in younger children, to an automatic control of the whole trajectory when the motor programs are memorized at about 10 years old. Then, we describe the neural correlates of adult writing, and the changes that could occur with learning during childhood. The acquisition of a new skill is characterized by the involvement of a network more restricted in space and where neural specificity is increased in key regions. The cerebellum and the left dorsal premotor cortex are of fundamental importance in motor learning, and could be at the core of the acquisition of handwriting.

Keywords: Cerebellum; children; handwriting; motor learning; premotor cortex.

Programmation multi-paradigme

Université Bretagne Sud
G.Ménier : gildas.menier@univ-ubs.fr

Prérequis

- Java 5 (+ jar et écosystème Java)
- Algorithmique
- Sans Google et Sans ordinateur
- GIT

- IDE : Eclipse, Idea, etc **SAUF** si c'est un handicap (!)
- Notepad + javac etc..

Prérequis

```
List<Integer> numbers = Arrays.asList(2, 3, 6, 19, 120);
System.out.println(
  numbers.stream()
    .peek(e -> System.out:println)
    .filter(e -> e > 10)
    .filter(e -> e % 2 == 0)
    .map(e -> e * 2)
    .findFirst()
    .map(e -> "La valeur est " + e)
    .orElse("No value found"));
```

Java de base

(Java 8 -> Java 18)

Présentation des cours

- Manière de considérer l'exécution du programme
- Contrainte imposée par le langage de programmation
- Manière de penser la programmation
- Expressivité du langage
  - Écrire moins pour faire plus
  - Manière de penser adaptée au problème à résoudre
- Certains langages sont plus adaptés que d'autre à la résolution de certains problèmes
- Base de données : locale, distribuée, tolérante aux fautes
- Logistique : ET / optimisation
- Interaction homme / machine

Présentation

- Multi-paradigme
- Le langage contient ce qu'il faut pour aborder le problème de manières différentes
- Maîtriser ces manières
- Java < 8: une manière de faire
  - Java 8 introduction (timide) de fonctionnel
- Algorithmique

Présentation

- En fait, du bon sens
- Pas (***plus***) de bricolage
- Planification et étude du problème
- Faire évoluer le logiciel

- L'objectif n'est pas seulement de faire un programme qui 'tourne'
- Extensible
- Compréhensible
- Confiance

- Être capable d'expliquer ce que vous avez fait

Présentation

- Langages
- **Java**
- Haskell
- **Scala**
- **Javascript**
- Autres
  - Inventor
  - Prolog
  - Erlang / Elixir / Ruby

Paradigmes de programmation

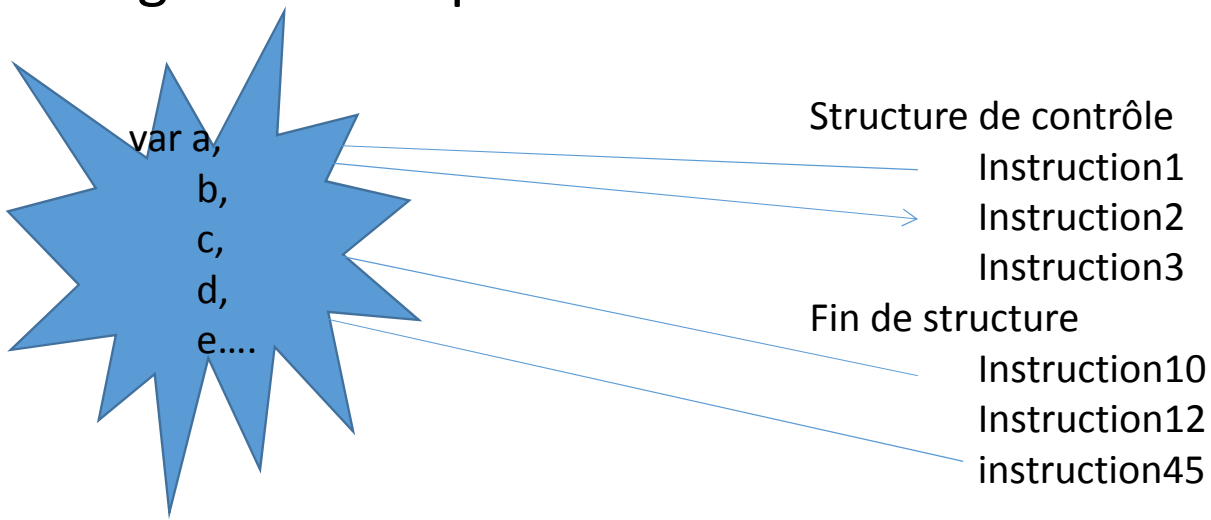
- Programmation impérative
  - Une séquence d'instruction change un état général
  - Détailler une suite d'instruction qui indique comment modifier des variables
  - Les variables contiennent les conditions de départ et leur évolution donne le résultat du programme
  - Démarche : trouver les structures de données et trouver une séquence d'instruction qui modifie l'état
  - Pas à pas -> diagnostique
  - Essai / erreur

Paradigmes de programmation

- A = A+1
- A <- A+1
- A+1 -> A
- A // A+1
- A+1 % A
- A,A+1

Paradigmes de programmation

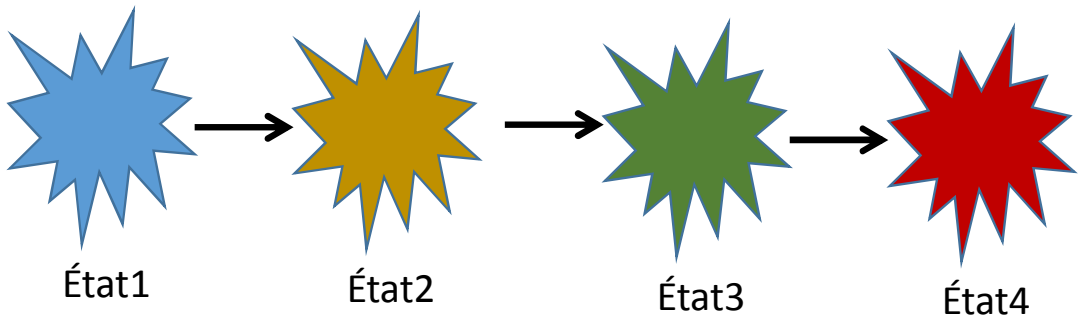
Cadre général : impératif



État : ensemble des variables

Paradigmes de programmation

Cadre général : impératif



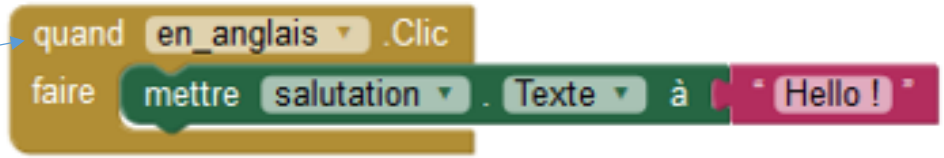
Chaque instruction modifie l'état de la mémoire

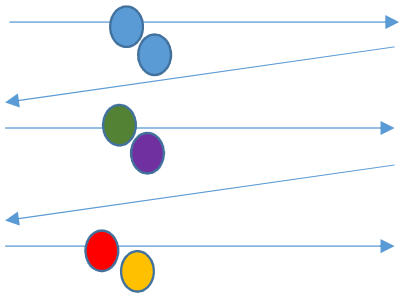
Paradigmes de programmation

- Programmation structurée
  - Instructions de contrôle
  - Blocs de code
- Programmation procédurale
  - Procédures
- Programmation modulaire
- C++ / C / Java / PHP / Python / Ruby

Paradigmes de programmation

- Programmation évènementielle
  - Quand il se produit… Faire
    - fin
- Programmation séquentielle
  - Notion de bloc d'exécution
  - Région spécifique
  - Le sens de l'exécution est important
  - De gauche à droite et de haut en bas





  - Trouver une erreur revient à faire des hypothèses
  - À les vérifier

Paradigmes de programmation

- Programmation orientée objet
  - Une collection d'objets en interaction via méthodes
  - Chaque objet est responsable de code
  - Chaque objet possède un état

  - Programmation objet / orientée objet
    - Java / C++ : orienté objet
    - Smalltalk : programmation objet
      - Méthode = message envoyé à un objet
      - Paradigme : programmation par messages

Organisation
Responsabilité
(délégation)

Paradigmes de programmation

- Programmation par prototypes
  - Self / Javascript
  - Objet sans (nécessairement) classes

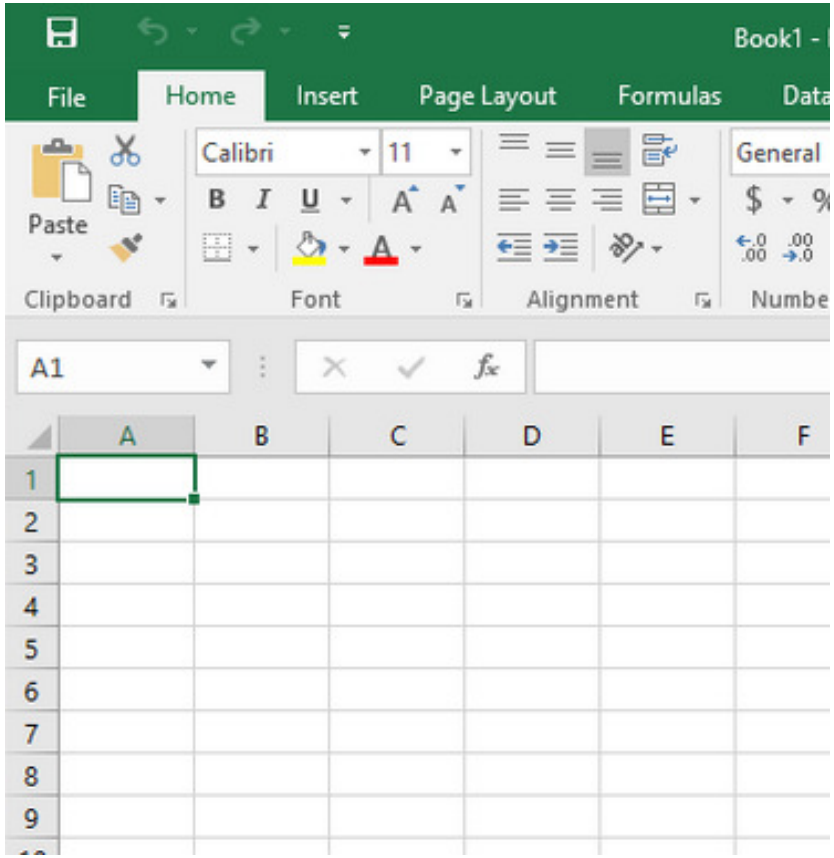
- Programmation chimique
  - Gamma
  - Transformation de multi ensembles
  - Exemples
  - Map/reduce
  - On ne s'intéresse pas au *comment*, mais au *quoi faire* !

Paradigmes de programmation

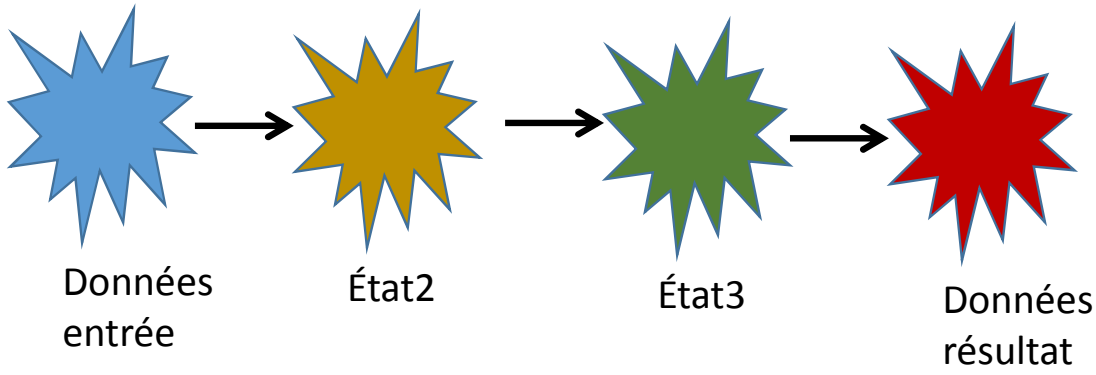
- Programmation déclarative
  - Prolog
    - exemple
  - SQL
  - OWL
  - SPARQL
  - Expressions régulières

Paradigmes de programmation

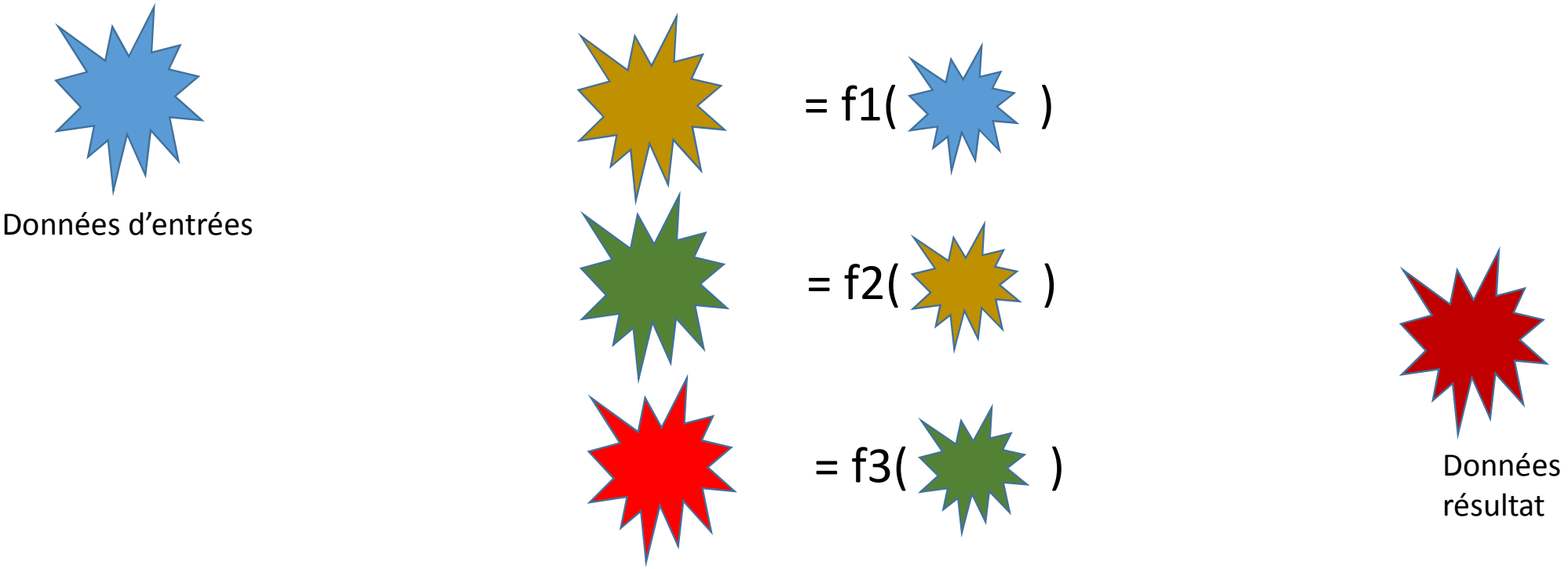
- Programmation par flots de données
  - Définition de dépendances entre données
  - A = B+1
    - Programmation réactive
  - Excel
  - Angular (cf cours second semestre)
    - Modèle Vue Contrôleur
    - Reactif



Fonctionnel

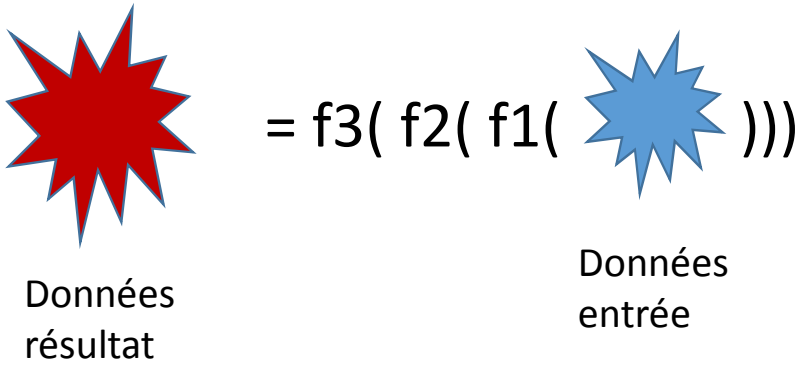


- Pas d'état ?!
  - Le résultat d'un programme c'est la transformation des données de départ



Fonctionnel

- Pas d'état ?!
  - Le résultat d'un programme c'est la transformation des données de départ



  - Pas besoin de variables
  - Pas besoin de mémoire
  - Distribution
  - Fiabilité

Paradigme récursif

- Boucle impérative
  - Parcourir explicitement chaque portion
  - Pas besoin de fonction

```
for( int i =0; i< 10; i++) {
   System.out.println(i);
}
```

- Boucle récursive
  - Utiliser une fonction
  - Faire une petite partie, puis recommencer avec le reste

```
void compterJqa10APartirDe(int i) {
  if (i<10) {
    System.out.println(i);
    compterJqa10APartirDe(i+1);
  }
}
```

```
compterJqa10APartirDe(0);
```

Paradigme récursif

- Récursivité

- Un exemple simple :

```
Fonction résoudreLeProblèmeSurEspace( espace ) {
    siEncorePossible(espace) {
        on utilise une partie U;
        resoudreLeProblèmeSurEspace(espace –U)
    }
}
```

Résolution partielle
Et on relance une résolution sur ce qui reste
(on est certain de l'arrêt)

```
Fonction distribuerLeGateau( gateau ) {
    siIlResteDu(gateau) {
        on distribue une part
        distribuerLeGateau(gateau-part)
    }
}
```

Paradigmes de programmation

- Programmation récursive
- Programmation par multi agents
- Programmation par acteurs
- Programmation par contrainte
- Programmation non déterministe
- Programmation réflexive
- Programmation scalaire
- Programmation systolique
- Programmation par contrats
- Programmation orientée composants
- Programmation génétique
- Etc..

Programmation multi-paradigme

- Développer
  - préoccupations

méthode Pas de méthode

- 1. ca marche ?
  - tests
  - complexité

- 2. Extensibilité
  - si je veux rajouter une fonctionnalité, est-ce que je dois tout modifier ?
    - proportion ?
    - Erreurs introduites
  - si le code est bien écrit, alors on peut rajouter des fonctionnalité en modifiant un minimum

Programmation multi-paradigme

- Développer
  - préoccupations
    - 3. Modularité
      - si je modifie un fichier, est-ce que ca a une conséquence pour les autres fichiers ?
        - réutilisation
        - travail en équipe
        - limiter la diffusion des erreurs
        - cascades

Programmation multi-paradigme

- Développer
  - préoccupations
    - 4. Réutilisabilité
      - Beaucoup de duplication de code ?
      - Documentation ?
      - Qualité ?

Programmation multi-paradigme

- Développer
  - préoccupations
    - 5. Testabilité
      - Facile à tester ?
      - Certitude ?
      - Couverture de test ?

Programmation multi-paradigme

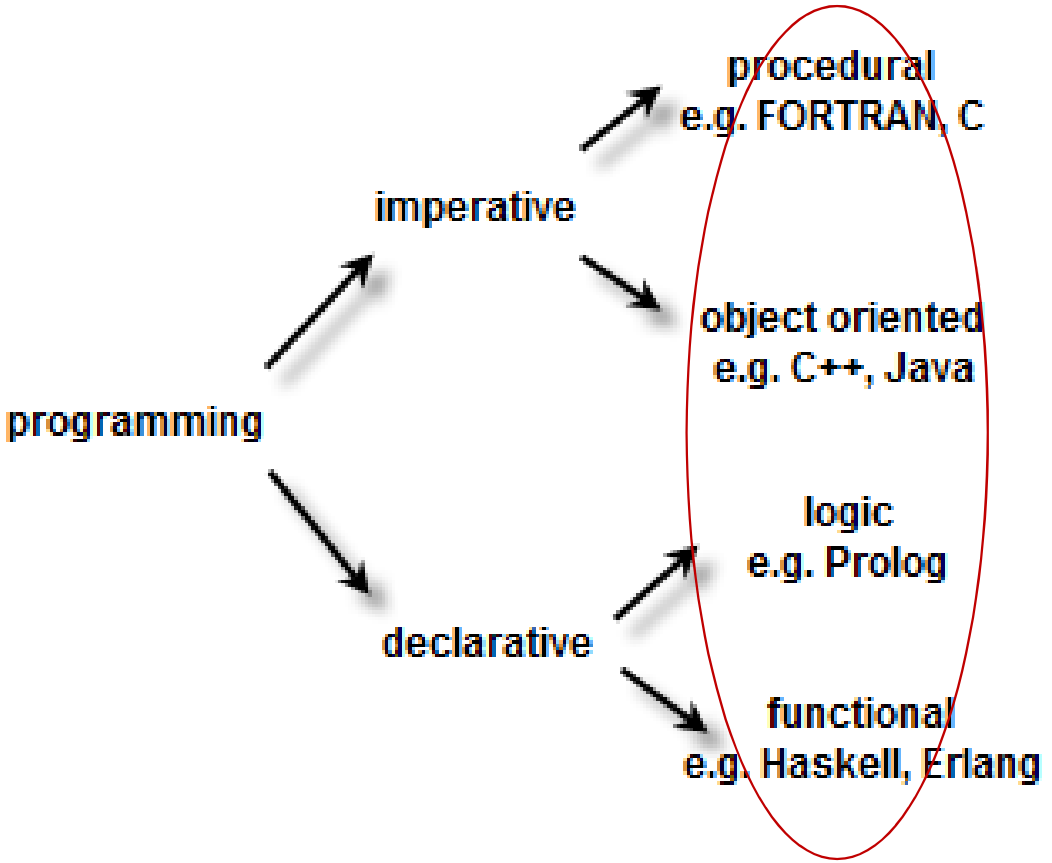
- Développer
  - préoccupations
    - 6. Clarté
      - La structure est facile à comprendre ?
      - facile à expliquer ?
      - simple à documenter ?

Programmation multi-paradigme

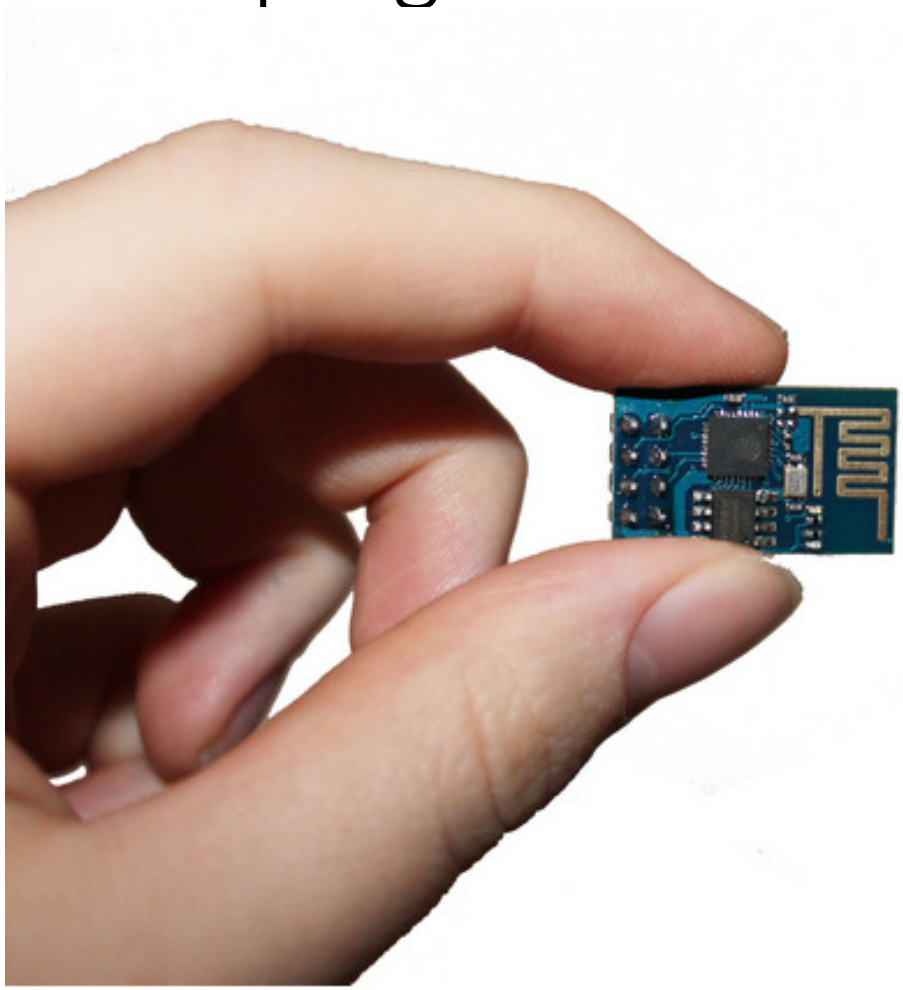
- Développer
  - préoccupations
    - Extensible ?
    - Modulaire ?
    - Réutilisable ?
    - Testable ?
    - Clair ?

    - Programmation fonctionnelle
    - Programmation impérative
    - Programmation déclarative

    … multiparadigme



Paradigmes de programmation

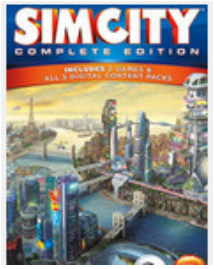




```
cfg={}
cfg.ssid="ESP_MENIER"; cfg.pwd="12345678";
wifi.ap.config(cfg)

cfg={}
cfg.ip="192.168.1.1"; cfg.netmask="255.255.255.0"; cfg.gateway="192.168.1.1";
wifi.ap.setip(cfg);
wifi.setmode(wifi.SOFTAP)

srv=net.createServer(net.TCP)

srv:listen(80,
 function(conn)
   conn:on("receive",
                function(conn)
                    local res="<h1>Salut les M1</h1><br/><H2>Cette page est envoyée par l'ESP8266</H2>"
                    conn:send(res) end)
   conn:on("sent",
                 function(conn) conn:close() end)
  end)
```

Langage multi-paradigme

- Problème de conception



- Java
- JVM
- Réutilisation du code
- … formation des développeurs