Génie logiciel : règles pour un code propre

Karën Fort

karen.fort@univ-lorraine.fr / https://members.loria.fr/KFort

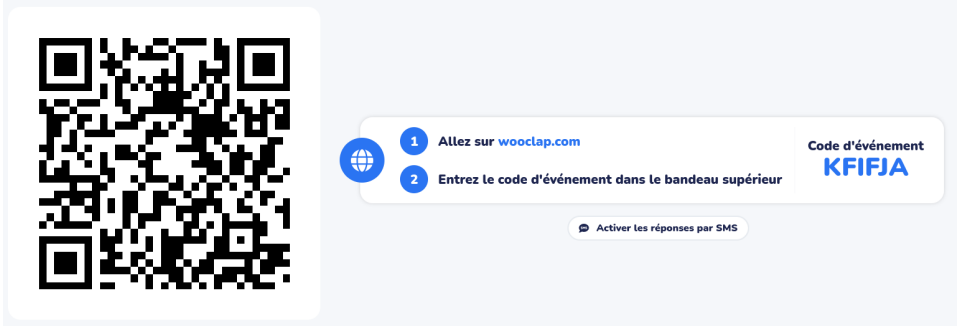
Institut des sciences du Digital Management & Cognition
COMPOSANTE DE L'UNIVERSITÉ DE LORRAINE

UNIVERSITÉ DE LORRAINE

Quelques sources d'inspiration

- Plus que de l'inspiration, j'ai repris le cours de Génie Logiciel – Implementation de Sylvain Lobry (univ. de Paris), avec son accord

Wooclap


1 Allez sur wooclap.com
2 Entrez le code d'événement dans le bandeau supérieur
Code d'événement
KFIFJA
Activer les réponses par SMS

Wooclap

Exercice

Que faut-il faire pour avoir un code propre ?

Motivations

Besoin de normes pour :

- uniformiser le code écrit par différents développeurs
- améliorer la lisibilité
- améliorer la maintenabilité
- (potentiellement) réduire la complexité
- améliorer la réutilisabilité
- aider à détecter les erreurs
- améliorer l'efficacité

→ Soyez cohérents

Voir le Google Python Style Guide

Adapté du cours de Sylvain Lobry

Nommage de variables

- le nom d'une variable doit permettre de comprendre à quoi elle correspond (contre-ex : int i)

Adapté du cours de Sylvain Lobry

Nommage de variables

- le nom d'une variable doit permettre de comprendre à quoi elle correspond (contre-ex : int i)
- a minima, ce nom ne doit pas porter à confusion (appeler *list* un vecteur ou *string* un entier)

Adapté du cours de Sylvain Lobry

Nommage de variables

- le nom d'une variable doit permettre de comprendre à quoi elle correspond (contre-ex : int i)
- a minima, ce nom ne doit pas porter à confusion (appeler *list* un vecteur ou *string* un entier)
- lorsque 2 variables ont des points communs, adapter le nom de manière qui fasse sens (contre-ex : i1, i2, ...)

Adapté du cours de Sylvain Lobry

Nommage de variables

- le nom d'une variable doit permettre de comprendre à quoi elle correspond (contre-ex : int i)
- a minima, ce nom ne doit pas porter à confusion (appeler *list* un vecteur ou *string* un entier)
- lorsque 2 variables ont des points communs, adapter le nom de manière qui fasse sens (contre-ex : i1, i2, ...)
- vous vous adressez à des gens, utilisez des noms prononçables

Adapté du cours de Sylvain Lobry

Nommage de variables

- le nom d'une variable doit permettre de comprendre à quoi elle correspond (contre-ex : int i)
- a minima, ce nom ne doit pas porter à confusion (appeler *list* un vecteur ou *string* un entier)
- lorsque 2 variables ont des points communs, adapter le nom de manière qui fasse sens (contre-ex : i1, i2, ...)
- vous vous adressez à des gens, utilisez des noms prononçables
- le nom de la variable doit pouvoir être cherché (contre-ex : i)

Adapté du cours de Sylvain Lobry

Wooclap

Exercice

Quel est le pire nom de variable que vous ayez vu/écrit ?

Concernant les fonctions

Une fonction :

- doit rester de taille modeste

Adapté du cours de Sylvain Lobry
Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche

Adapté du cours de Sylvain Lobry
Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche
- doit avoir un nom qui la décrit

Adapté du cours de Sylvain Lobry
Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche
- doit avoir un nom qui la décrit
- ne doit avoir que peu d'arguments : la solution consiste souvent à regrouper les arguments dans des objets

Adapté du cours de Sylvain Lobry

Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche
- doit avoir un nom qui la décrit
- ne doit avoir que peu d'arguments : la solution consiste souvent à regrouper les arguments dans des objets
- ne doit pas renvoyer de codes d'erreurs (utilisez les exceptions)

Adapté du cours de Sylvain Lobry
Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche
- doit avoir un nom qui la décrit
- ne doit avoir que peu d'arguments : la solution consiste souvent à regrouper les arguments dans des objets
- ne doit pas renvoyer de codes d'erreurs (utilisez les exceptions)
- ne doit se situer qu'à un seul niveau d'abstraction (par ex. : récupérer des données et les normaliser ne doit pas être fait dans la même fonction) $\rightarrow$ SLA

Adapté du cours de Sylvain Lobry

Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Concernant les fonctions

Une fonction :

- doit rester de taille modeste
- ne doit réaliser qu'une seule tâche
- doit avoir un nom qui la décrit
- ne doit avoir que peu d'arguments : la solution consiste souvent à regrouper les arguments dans des objets
- ne doit pas renvoyer de codes d'erreurs (utilisez les exceptions)
- ne doit se situer qu'à un seul niveau d'abstraction (par ex. : récupérer des données et les normaliser ne doit pas être fait dans la même fonction) $\rightarrow$ SLA
- ne doit pas générer d'effets de bord (inattendus du point de vue de l'appelant)

Adapté du cours de Sylvain Lobry
Pour plus de détails sur SLA : https://medium.com/@outfunnel/the-single-level-of-abstraction-principle-52b5d56ef54b

Wooclap

Exercice

Qu'est-ce qu'un effet de bord ?

"It's harder to read code than to write it."
[Joel Spolsky]

https://quotefancy.com/programming-quotes

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)
  - ou à mentionner un TODO

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)
  - ou à mentionner un TODO
- un commentaire ne doit pas être :

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)
  - ou à mentionner un TODO
- un commentaire ne doit pas être :
  - redondant avec le code

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)
  - ou à mentionner un TODO
- un commentaire ne doit pas être :
  - redondant avec le code
  - obligatoire (il serait alors probablement redondant)

Adapté du cours de Sylvain Lobry

Concernant les commentaires

- si le code n'est pas clair, ne l'expliquez pas dans les commentaires, changez le code
- un bon commentaire doit servir soit :
  - à informer
  - ou à expliquer une intention
  - ou à clarifier (s'il n'y a pas d'autres possibilités)
  - ou à mentionner un TODO
- un commentaire ne doit pas être :
  - redondant avec le code
  - obligatoire (il serait alors probablement redondant)
  - du code mort (à supprimer)

Adapté du cours de Sylvain Lobry

"Code is like humor. When you have to explain it, it's bad."
[Cory House]

https://quotefancy.com/programming-quotes

Concernant le formatage

- un bon code est bien formaté

Adapté du cours de Sylvain Lobry

Concernant le formatage

- un bon code est bien formaté
- ne pas oublier d'indenter (même pour un *if* très court ou pour les petites boucles)

Adapté du cours de Sylvain Lobry

Concernant le formatage

- un bon code est bien formaté
- ne pas oublier d'indenter (même pour un *if* très court ou pour les petites boucles)
- utiliser les espaces verticaux pour séparer les concepts

Adapté du cours de Sylvain Lobry

Concernant le formatage

- un bon code est bien formaté
- ne pas oublier d'indenter (même pour un *if* très court ou pour les petites boucles)
- utiliser les espaces verticaux pour séparer les concepts
- les fonctions doivent avoir une taille limitée

Adapté du cours de Sylvain Lobry

Concernant le formatage

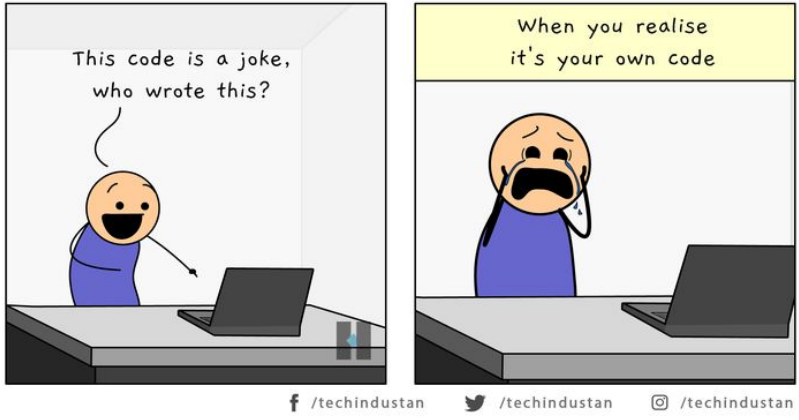
- un bon code est bien formaté
- ne pas oublier d'indenter (même pour un *if* très court ou pour les petites boucles)
- utiliser les espaces verticaux pour séparer les concepts
- les fonctions doivent avoir une taille limitée
- les fonctions doivent être ordonnées du plus haut niveau en haut, au plus bas niveau en bas (peut parfois poser problème)

Adapté du cours de Sylvain Lobry

"Any fool can write code that a computer can understand. Good programmers write code that humans can understand."

[Martin Fowler]

https://quotefancy.com/programming-quotes


This code is a joke,
who wrote this?
When you realise
it's your own code
/techindustan
/techindustan
/techindustan

Le code et vous

Pour un code propre

Pour finir

CQFR : Ce Qu'il Faut Retenir

- ne mettez pas votre ego dans votre code
- ne gardez pas le vieux code au cas où
- KISS