4 PEP 8 : BONNES PRATIQUES POUR CODER EN PYTHON

En suivant les directives de PEP 8, les développeurs peuvent produire un code plus lisible et plus facile à comprendre pour les autres développeurs. Cela permet de faciliter la maintenance du code et de minimiser les erreurs, ce qui peut conduire à des programmes plus robustes et plus faciles à déboguer.

1 MISE EN FORME DU CODE

• **Utiliser des indentations de 4 espaces** pour aligner le code (ou régler la tabulation sur 4 caractères)

• **Limiter la longueur des lignes à 79 caractères**, sauf dans certains cas où cela est justifié. Exemple : pour une longue URL

• **Utiliser des lignes vides pour séparer les fonctions, les classes et les blocs de code logiques** afin de rendre le code plus lisible

• **Placer les parenthèses autour des expressions logiques plutôt que les opérateurs logiques** eux-mêmes pour éviter toute confusion avec les priorités d'opérateurs. Exemple : *if (x > 0) and (y < 0)* plutôt que *if x > 0 and y < 0:*

• **Utiliser des espaces autour des opérateurs** pour améliorer la lisibilité du code. Ex : *x = 5 + 2* plutôt que *x=5+2*

2 CONVENTIONS DE NOMMAGE

• **Utiliser des noms en minuscules pour les variables et les fonctions, séparés par des underscores**. Exemple : *ma_variable*, *ma_fonction*

• **Utiliser des majuscules pour les noms de classes, en utilisant la convention CamelCase** où chaque mot commence par une majuscule. Exemple : *MaClasse*

• **Utiliser des noms en majuscules pour les constantes, séparés par des underscores**. Exemple : *MA_CONSTANTE*

• **Éviter d'utiliser des caractères spéciaux comme les tirets bas (_) pour les noms de variables** sauf s'ils ont une signification particulière dans votre code

• **Éviter d'utiliser des noms de variables qui ont une signification particulière dans Python** (Exemples : *list*, *dict*, *str*) car cela pourrait provoquer des conflits avec les noms de fonctions et de classes intégrées

• **Utiliser des verbes pour les noms de fonctions** afin d'indiquer clairement leur action. Exemple : *calculer_prix()* plutôt que *prix()*

• **Éviter d'utiliser des abréviations sauf si elles sont couramment comprises** dans le domaine d'application de votre code

3 COMMENTAIRES

• **Utiliser des commentaires pour expliquer ce que fait le code, à quoi il sert** et non pas comment il le fait

• **Utiliser des phrases complètes, ponctuées et compréhensibles**

• **Éviter les commentaires redondants ou évidents**. Exemple : *x = x + 1 # Incrémenter x.*

• **Placer les commentaires sur une ligne séparée** plutôt que sur la même ligne que le code

• **Éviter d'utiliser des commentaires pour désactiver du code**. Utiliser plutôt des outils de contrôle de version pour gérer le code mort

• **Utiliser une longueur de ligne maximale de 72 caractères** pour les commentaires en ligne, et de 79 caractères pour les commentaires de documentation multilignes

• **Utiliser de la documentation au format Docstrings** pour chaque fonction, chaque classe et chaque module

• **Utiliser de préférence l'anglais :** PEP 8 n'impose pas de règle concernant la langue de la documentation, mais, étant donné que l'anglais est la langue de facto de l'industrie informatique, il est recommandé de rédiger la documentation en anglais pour que le projet soit accessible à une audience plus large et pour faciliter la collaboration avec d'autres développeurs et contributeurs