Live de Python

Introdução à Qualidade de Código

Apresentação

Flávio Meira

flaviomeira10_1

flaviomeira10@gmail.com

https://www.linkedin.com/in/flaviomeira10

@Flaviomeira10

Roteiro

- Necessidades do desenvolvimento de software
- Métricas de qualidade de código (análise estática)
- Ferramentas

Necessidades do desenvolvimento de software

- Entregas frequentes
- Garantir funcionamento básico do software (regressão)
- Manter a qualidade da aplicação e do código

Necessidades do desenvolvimento de software

Entregas frequentes

- Chrome: Release entre 45 - 50 dias - Android, IOS, Linux, MacOS, Windows
- Firefox: Releases a cada aprox 60 dias - Windows 64-bit, Windows 32-bit, MacOS, Linux 64-bit, Linux 32-bit

Necessidades do desenvolvimento de software

Garantir funcionamento básico do software

- Regressão
- Garantir que tudo o que estava funcionando, permanece funcionando nas novas versões

Necessidades do desenvolvimento de software

Manter a qualidade do código

- Fácil manutenção
- Fácil de testar
- Integração contínua

Métricas de qualidade de código

- Débito técnico
- Cobertura de código (code coverage)
- Complexidade ciclomática
- Acoplamento e Coesão
- Manutenibilidade de software

Métricas de qualidade de código

Débito técnico

É um conceito do desenvolvimento de software que reflete a custo de retrabalho por ter escolhido uma solução mais fácil ao invés de usar a melhor abordagem, por questão de tempo

Métricas de qualidade de código

Débito técnico

Métrica utilizada para medir a capacidade de realizar futuras alterações, correções de problemas ou entregas de novas funcionalidades, com esforço aceitável

Métricas de qualidade de código

Débito técnico

Métricas de qualidade de código

Cobertura de código (code coverage)

É a medida usada para descrever o quanto o código fonte de um programa é exercitado, quando uma suíte de testes é executada

Métricas de qualidade de código

Cobertura de código (code coverage)

- Métrica polêmica
- Statement Coverage
- Branch Coverage

Métricas de qualidade de código

Cobertura de código (code coverage)

- Statement Coverage
  - Também conhecido como cobertura de linhas
  - Não considera loopings ou if’s
  - Uma linha pode conter mais de uma instrução

Métricas de qualidade de código

Cobertura de código (code coverage)

- Statement Coverage
  - Cobertura:

$$\frac{\text{N° de linhas exercitadas}}{\text{N° de linhas}} \text{ X } 100\%$$

Métricas de qualidade de código

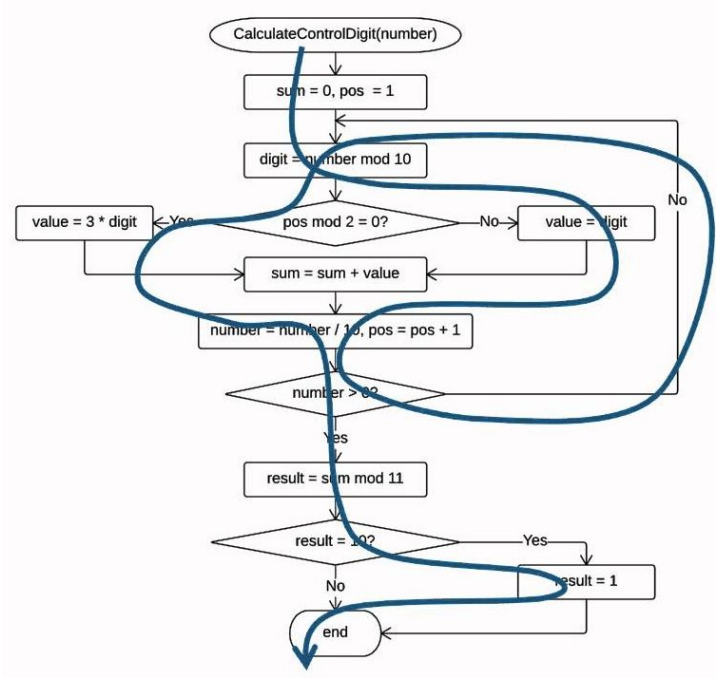
Cobertura de código (code coverage)

- Branch Coverage
  - Exige que todos os caminhos sejam testados
  - Em if’s exige que sejam testadas as condições true e false

Métricas de qualidade de código

Cobertura de código (code coverage)

- Branch Coverage



Métricas de qualidade de código

Cobertura de código (code coverage)

- Branch Coverage
  - Cobertura:

  (Número das decisões testadas / Número total de decisões) x 100 %

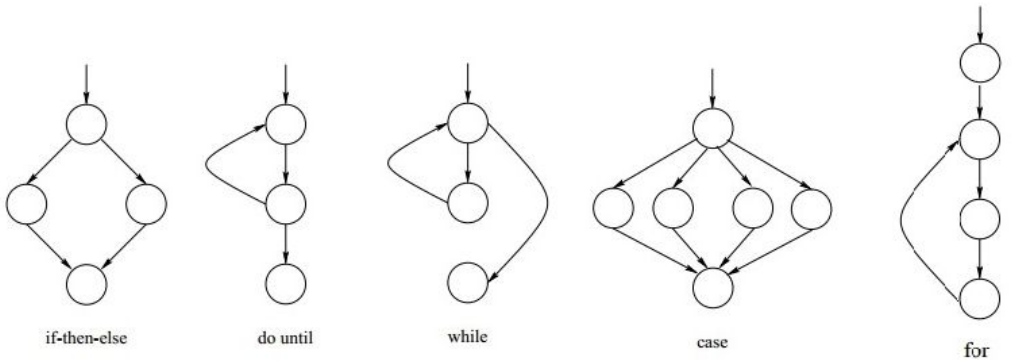
Métricas de qualidade de código

Complexidade Ciclomática

- Métrica utilizada para quantificar a complexidade do software
- Mede a quantidade de caminhos de execuções diferentes de um bloco pode ter
- [Radon] Complexidade ciclomática é o número de decisões que um bloco pode conter mais 1

Métricas de qualidade de código

Complexidade Ciclomática



Métricas de qualidade de código

Complexidade Ciclomática

Construct	Effect on CC	Reasoning
if	+1	An `if` statement is a single decision.
elif	+1	The `elif` statement adds another decision.
else	+0	The `else` statement does not cause a new decision. The decision is at the `if`.
for	+1	There is a decision at the start of the loop.
while	+1	There is a decision at the `while` statement.
except	+1	Each `except` branch adds a new conditional path of execution.
finally	+0	The finally block is unconditionally executed.
with	+1	The `with` statement roughly corresponds to a try/except block (see PEP 343 for details).
assert	+1	The `assert` statement internally roughly equals a conditional statement.
Comprehension	+1	A list/set/dict comprehension of generator expression is equivalent to a for loop.
Boolean Operator	+1	Every boolean operator (and, or) adds a decision point.

http://radon.readthedocs.io/en/latest/intro.html

Métricas de qualidade de código

Complexidade Ciclomática

```
def normalize(word):
    result = ""
    for letter in word:
        if ((ord(letter) >= 65 and ord(letter) <= 90)
            or (ord(letter) >= 97 and ord(letter) <= 172)):
            result += letter
        else:
            result += ""
    return result
```

Este if representa uma complexidade = 3 (por conta dos operadores), porém a função inteira possui complexidade = 6

Métricas de qualidade de código

Complexidade Ciclomática

```
def normalize(word):
    allowed_letters = list(range(65, 90)) + list(range(97, 172))
    result = ""
    for letter in word:
        if ord(letter) in allowed_letters:
            result += letter
        else:
            result += ""
    return result
```

Este if representa uma complexidade = 1, porém a função inteira possui complexidade = 3

Métricas de qualidade de código

Acoplamento

- Grau de dependência em que um artefato (clase, objeto, framework etc) se relaciona com outro
- Acoplamento fraco ou baixo acoplamento
- Acoplamento forte ou alto acoplamento

Métricas de qualidade de código

Acoplamento

- Alto acoplamento torna o código difícil de entender
- Dificulta a criação de testes
- Aumenta o tempo de desenvolvimento de manutenção e novas funcionalidades

Métricas de qualidade de código

Coesão

- Está ligado ao princípio da responsabilidade única
- Códigos coesos são aqueles onde seus membros estão intimamente ligados e estão ali por um objetivo comum

Métricas de qualidade de código

Coesão

- Baixa coesão significa que o componente possui responsabilidades além das suas
- Dificuldade em dar manutenção
- Dificulta o reuso de componentes

Métricas de qualidade de código

Manutenibilidade de software

- Refere-se à facilidade e segurança em efetuar correções ou novas implementações no software
- Baseia-se na relação entre complexidade ciclomática, linhas de código, custo de correção vs custo de desenvolvimento

Métricas de qualidade de código

Manutenibilidade de software

- PEP 8 Style Guide for Python Code
- PEP 257 Docstring Conventions
- Baixa complexidade ciclomática
- Baixo acoplamento
- Alta coesão
- Cobertura de código

Ferramentas

- Radon - http://radon.readthedocs.io/en/latest/index.html
- Pylint - https://www.pylint.org/
- PyFlakes - https://pypi.python.org/pypi/pyflakes
- Sonar - https://docs.sonarqube.org/display/PLUG/SonarPython
- Coveralls - https://coveralls.io/

Referências

- https://en.wikipedia.org/wiki/Google_Chrome_version_history
- https://wiki.mozilla.org/RapidRelease/Calendar
- https://martinfowler.com/bliki/TechnicalDebt.html
- https://docs.sonarqube.org/display/SONAR/Concepts
- https://www.fullstackpython.com/code-metrics.html
- https://www.python.org/dev/peps/pep-0008/
- https://www.python.org/dev/peps/pep-0257/

Conclusão

Qualidade de software é a capacidade de atender/resolver o problema do cliente de forma a otimizar o trabalho (lucro).

Conclusão

Muito obrigado!