Laboratório de Experimentação de Software

# **Relatório Final:** Análise Comparativa de Repositórios Python e Java

21/09/2020 | Bruno Marini - 634883

## Introdução

Com mais de 40 milhões de usuários e a criação de mais de 44 milhões novos repositórios em 2019 [1], o GitHub acaba sendo a plataforma de armazenamento remoto de código versionado mais popular dentre seus concorrentes, principalmente quando falamos de projetos *open source*. Se atentando a isso, a plataforma também disponibiliza diversas funcionalidades que fomentam o desenvolvimento de software de maneira colaborativa ao mesmo tempo que organizada. Através de *issues*, *pull requests*, *releases*, *forks* e *wikis*, milhares de usuários podem identificar, contribuir, evoluir e documentar o seu software, contanto que seja público.

Porém, prévio a esse processo de colaboração entre desenvolvedores, está a escolha da linguagem em o que o código será escrito. Esta é uma escolha que pondera diversos critérios, como o domínio da aplicação, o ambiente de execução, o suporte à falhas, experiências prévias dos desenvolvedores, curva de aprendizado para desenvolvedores inexperientes e principalmente o equilíbrio entre agilidade e qualidade. Portanto, isto ressalta a importância na escolha certeira de uma linguagem que siga esses critérios.

Tido esse contexto, este trabalho - proposto na disciplina de Laboratório de Experimentação de Software do curso de Engenharia de Software da PUC Minas no 2º semestre de 2020 - tem o objetivo de, através de uma análise quantitativa, responder uma série de questões que expõem as principais características dos 200 repositórios públicos mais populares (de acordo com o número de estrelas) do GitHub, escritos ou em Java e ou em Python. Com isso, será possível observar os principais critérios que compõem os repositórios escritos nessas linguagens.

### Questões de Pesquisa

As questões que fomentam este trabalho são:

1. Quais as características dos *top-100* repositórios Java mais populares?
2. Quais as características dos *top-100* repositórios Python mais populares?

3. Repositórios Java e Python populares possuem características de “boa qualidade” semelhantes?
4. A popularidade influencia nas características dos repositórios Java e Python?

## Hipóteses

Com o objetivo de conjecturar em cima das questões de pesquisa propostas, para cada uma, foi elaborado uma hipótese - e sua respectiva justificativa - a ser validada a partir da análise dos resultados obtidos. Dessa forma, podemos validar cada hipótese a partir da resposta quantitativa de cada cada questão.

Dito isso, as hipóteses são:

1. Os top-100 repositórios Java são mais velhos, têm mais linhas de código fonte (SLOC), têm uma uma frequência menor de *releases* e possuem grande quantidade de estrelas, *releases* e *forks*.

   **Justificativa:** sendo uma linguagem presente desde 1996 e com popularidade constante por ser uma das primeiras a dissipar a Programação Orientada a Objetos, é esperado que a mediana da idade dos repositórios escritos em Java seja mais elevada e, por consequência disso, uma frequência menor de *releases*, mas com maior acumulação de estrelas, *releases* e *forks* ao longo do tempo. Além disso, devido a sua sintaxe mais verbosa, é esperável que tenha uma quantidade maior de SLOC.

2. Os top-100 repositórios Phyton são mais novos, têm menos linhas de código fonte (SLOC), têm uma uma frequência maior de *releases* e possuem grande quantidade de estrelas, *watchers* e *forks*.

   **Justificativa:** por mais Python que esteja presente há mais tempo - desde 1992 - em comparação com o Java, a sua popularidade só veio a crescer e estar famosa entre os os desenvolvedores mais recentemente. Por isto, é esperado que a mediana da idade dos repositórios escritos em Python seja menor. Porém, devido a sua popularidade, repositórios escritos em Python tem uma atenção maior dos desenvolvedores e, por isto, possuem uma frequência maior de *releases* e uma quantidade maior de estrelas, *watchers* e *forks*. Além disso, devido a sua sintaxe menos verbosa, é esperável que tenha uma quantidade menor de SLOC.

3. “Definindo a ‘qualidade’ de um código pelo seu nível de documentação, ou seja, quantidade de comentários, repositórios Java e Python possuem um nível de qualidade muito próximo, ou seja, repositórios escritos em uma das linguagens não são de

qualidade superior a aqueles escritos em outra. Portanto, pouco importa a escolha entre Java e Python para o critério de qualidade.

**Justificativa:** por esta hipótese ser baseada na métrica da quantidade de linhas de comentários (CLOC) de um repositório, não tem diferenciação de sintaxe como nas hipóteses 1 e 2. Por isto, tanto aqueles escritos em Java quanto em Python têm a mesma chance de serem de qualidade, portanto possuem características semelhantes de "boa qualidade".

4. Quanto mais popular, mais velho e maior a quantidade de *releases*, *watchers* e *forks*, mas não há correlação com frequência de *releases* e LOC, SLOC, CLOC.

   **Justificativa:** como foi possível concluir no relatório final do trabalho anterior "Características de Repositórios GitHub Populares", existe uma forte correlação entre a popularidade de um repositório com a sua idade e quantidade de *releases*, mas nenhuma correlação com a frequência de *releases*. Portanto isto não deve ser diferente para repositórios escritos ou em Java ou em Python, e deve existir a mesma tendência para *watchers* e *forks*, mas não para LOC, SLOC e CLOC.

### Métricas

Visando testar a validade das hipóteses e, com isso também, responder as questões de pesquisa, a análise será baseada nas seguintes métricas:

1. Nº de estrelas;
2. Nº de *watchers*;
3. Nº de *forks*;
4. Linhas de código total (LOC);
5. Linhas de código fonte (SLOC);
6. Linhas de comentários (CLOC);
7. Nº de *releases*;
8. Frequência de *releases* (nº de *releases* / idade do repositório);
9. Idade do repositório (a partir da data da sua criação).

## Metodologia

Foi desenvolvido um *script* em Node.js que, a partir de um *token* da API do GitHub, realiza uma busca paginada da query GraphQL a seguir - alternando somente o atributo language entre "Python" e "Java", conforme desejado - e calcula o LOC, SLOC e CLOC de cada repositório

enquanto, paralelamente, os resultados são salvos em seus respectivos arquivos CSV. Tanto o código quanto a sua documentação pode ser encontrada em: https://github.com/TheMarini/python-vs-java-metrics/tree/v0.2.0.

```
{
    search(type: REPOSITORY, query: "stars:>100 language:python", first: 100) {
      repositoryCount
      pageInfo {
        endCursor
      }
      nodes {
        ... on Repository {
          nameWithOwner
          stargazerCount
          createdAt
          forkCount
          watchers {
            totalCount
          }
          releases {
            totalCount
          }
        }
      }
    }
  }
```

Enquanto a obtenção das métricas 1, 2, 3, 7, 8 e 9 foi feita através da *query* acima com a API do Github, os valores de LOC, SLOC e CLOC (métricas 4, 5 e 6 respectivamente) foram obtidos através da ferramenta de análise estática de código sloc, pelo comando "sloc <diretório> --keys total,source,comment". Desta forma, a cada resultado da busca paginada, automaticamente foi clonado um repositório, calculado com a ferramenta, armazenado os valores no respectivo CSV (python.csv ou java.csv), deletado do disco de armazenamento e feito o mesmo processo para o repositório seguinte.

Esta busca e análise estática de código foi realizada às 23h30 do dia 07/10/2020 para os 100 primeiros repositórios públicos Phython e para os 100 primeiros repositórios públicos Java, ordenados pela sua quantidade de estrelas, e almejava todos os seus atributos necessários para cumprir as métricas de cada questão de pesquisa e, consequentemente, de cada hipótese.

## Resultados Obtidos

Como proposto pelo próprio enunciado deste trabalho e para ser justo nos casos que há um repositório com o valor de uma métrica muito superior do que os demais na mesma categoria - como pelo nº de estrelas, nº de *watchers*, nº de *forks* e etc. -, tantos os gráficos quanto as

análises foram baseadas na **mediana** [2] dos resultados obtidos para questão de pesquisa e hipótese.

Dito isso, segue os resultados obtidos.

## **Questão 1:** "Quais as características dos top-100 repositórios Java mais populares?"

Como é possível observar nas tabelas a seguir, os 100 repositórios mais populares escritos em Java têm como destaque a mediana elevada de estrelas, *forks* e *watchers*, mas principalmente de LOC, SLOC e CLOC. Além disso, é notável a disparidade entre os valores máximos e mínimos, o que resulta em um desvio padrão alto, afetando os valores de média. Um último detalhe a se perceber, através dos valores mínimos, é a existência de repositórios com 0 *releases* e 0 linhas de código.

Tabela 1.1

| Java | | Métricas | | | | | |
|---|---|---|---|---|---|---|---|
| | | **Estrelas** | **Idade (anos)** | **Forks** | **Watchers** | **Releases** | **Frequência de Releases** |
| **Medidas** | **Máximo** | 112129 | 10,67 | 37024 | 5231 | 191 | 4,90% |
| | **Mediana** | 17225 | 4,89 | 4711 | 920 | 8 | 0,45% |
| | **Mínimo** | 12035 | 0,36 | 1010 | 258 | 0 | 0,00% |
| | **Média** | 23910 | 5,34 | 7429 | 1257 | 24 | 1,24% |
| | **Desvio padrão** | 16931 | 2,55 | 7674 | 986 | 38 | 4,08% |

Tabela 1.2

| Java | | Métricas | | | |
|---|---|---|---|---|---|
| | | **LOC** | **SLOC** | **CLOC** | **% de CLOC** |
| **Medidas** | **Máximo** | 2490968 | 1810262 | 535772 | 21,51% |
| | **Mediana** | 77329 | 47525 | 8733 | 11,29% |
| | **Mínimo** | 0 | 0 | 0 | 0 |
| | **Média** | 283768 | 193599 | 55567 | 19,58% |
| | **Desvio padrão** | 541234 | 367241 | 113568 | 20,98% |

**Hipótese 1:** "Os top-100 repositórios Java são mais velhos, têm mais linhas de código fonte (SLOC), têm uma uma frequência menor de releases e possuem grande quantidade de estrelas, releases e forks."

Falsa. Os top-100 Java, quando comparados com os top-100 Python, possuem uma mediana inferior de idade (portanto são mais jovens), uma frequência superior de *releases* e menor quantidade de estrelas e *forks*. Porém, possuem sim mais linhas de código fonte (SLOC) e mais *releases*.

## **Questão 2:** "Quais as características dos top-100 repositórios Python mais populares?"

Como é possível observar nas tabelas a seguir, os 100 repositórios mais populares escritos em Python têm como destaque a mediana elevada de estrelas, idade, *forks* e *watchers*, semelhante aos repositórios Java, porém o inverso acontece no número de LOC, SLOC e CLOC, com medianas muito inferiores. Além disso, é notável a disparidade entre os valores máximos e mínimos, o que resulta em um desvio padrão alto, afetando os valores de média. Um último detalhe a se perceber, através dos valores mínimos, é a existência de repositórios com 0 *releases* e 0 linhas de código.

Tabela 2.1

| Python | | Métricas | | | | | |
|---|---|---|---|---|---|---|---|
| | | Estrelas | Idade (anos) | Forks | Watchers | Releases | Frequência de Releases |
| Medidas | Máximo | 108679 | 12,3 | 42423 | 5893 | 658 | 14,66% |
| | Mediana | 21759 | 5,01 | 4904 | 881 | 4 | 0,19% |
| | Mínimo | 14156 | 1,03 | 709 | 181 | 0 | 0,00% |
| | Média | 29631 | 5,51 | 7125 | 1278 | 32 | 1,59% |
| | Desvio padrão | 20137 | 2,81 | 7388 | 1111 | 88 | 8,61% |

Tabela 2.2

| Python | | Métricas | | | |
|---|---|---|---|---|---|
| | | LOC | SLOC | CLOC | % de CLOC |
| Medidas | Máximo | 1521520 | 1120258 | 280891 | 18,46% |
| | Mediana | 16320 | 10690 | 2011 | 12,32% |

| | | | | | |
|---|---|---|---|---|---|
| | **Mínimo** | 0 | 0 | 0 | 0,00% |
| | **Média** | 103723 | 73126 | 19017 | 18,33% |
| | **Desvio padrão** | 230687 | 167196 | 44349 | 19,22% |

**Hipótese 2:** "Os top-100 repositórios Phyton são mais novos, têm menos linhas de código fonte (SLOC), têm uma uma frequência maior de releases e possuem grande quantidade de estrelas, watchers e forks."

Falsa. Os top-100 Python, quando comparados com os top-100 Python, possuem uma mediana superior de idade (portanto são mais velhos), uma frequência inferior de *releases* e menor quantidade de *watchers*. Porém, possuem sim menos linhas de código fonte (SLOC) e mais estrelas e *forks*.

## **Questão 3:** "Repositórios Java e Python populares possuem características de 'boa qualidade' semelhantes?"

Definindo a "qualidade" de um código pelo seu nível de documentação, ou seja, quantidade de comentários (assim como foi feito para a hipótese associada à esta questão), sim os top-100 repositórios Python e os top-100 repositórios Java possuem semelhança na característica de boa qualidade: % de CLOC. Isto porque os valores máximos, medianos, mínimos, médios e o próprio desvio padrão são muito próximos comparando-os entre as duas linguagens.

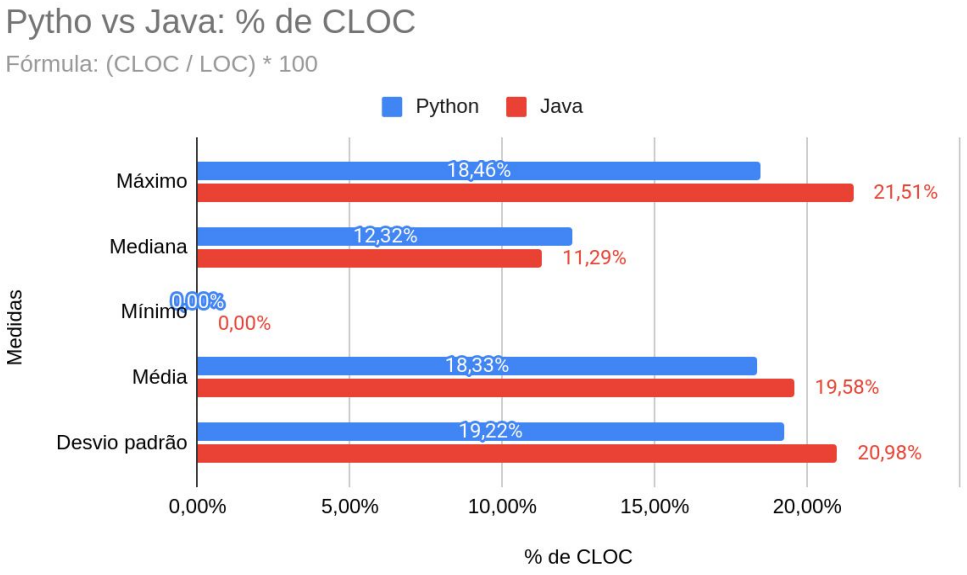


Gráfico 3.1

**Hipótese 3:** "Definindo a 'qualidade' de um código pelo seu nível de documentação, ou seja, quantidade de comentários, repositórios Java e Python possuem um nível de qualidade muito próximo, ou seja, repositórios escritos em uma das linguagens não são de qualidade superior a aqueles escritos em outra. Portanto, pouco importa a escolha entre Java e Python para o critério de qualidade."

Verdadeiro. Semelhante ao que foi descrito na questão 3, associada a esta hipótese, e observando o gráfico 3.1, os repositórios Java e Python uma diferença no nível de qualidade de apenas 1,03% (de acordo com os valores medianos). Ou seja, pensando em Python e Java, a escolha da linguagem não define superioridade no nível qualidade, portanto pouco importa em qual será escrito.

## **Questão 4:** "A popularidade influencia nas características dos repositórios Java e Python?"

Sim. Analisando as tabelas 1.1 e 2.1, Python, a linguagem mais popular entre as duas - de acordo com a mediana do nº de estrelas -. também possui outras métricas superiores quando comparado com Java. Como as medianas de: idade (5,01 contra 4,89), *forks* (4904 contra 4711), *watchers* (não superior, mas com uma diferença de apenas 39 *watchers*) e *releases* (não superior, mas com uma diferença de apenas 4 *releases*). Porém isto não é visto com a frequência das *releases* (com 0,19% contra 0,45%) e nem nas métricas que não têm nenhuma associação com a popularidade, sendo estas LOC (com 16320 contra 77329), SLOC (com 10690 contra 47525) e CLOC (com 2011 contra 8733).

**Hipótese 4:** "Quanto mais popular, mais velho e maior a quantidade de releases, watchers e forks, mas não há correlação com frequência de releases e LOC, SLOC, CLOC."

Verdadeira. Conforme descrito na resposta da questão 4, associada à esta hipótese.

# Conclusão

Com a associação de algumas métricas do GitHub - como estrelas, *watchers*, *forks*, e *releases* - com algumas métricas de análise estática de código - como LOC, SLOC e CLOC - dos 100 repositórios públicos mais populares escritos em Python e dos 100 repositórios públicos mais populares escritos em Java, foi possível responder todas as questões propostas e validar

algumas hipóteses que respondam constantes dúvidas no ambiente de projetos *open source*, como a escolha da linguagem e critérios que definem uma boa qualidade de código.

## Referências


1. The expanding Octoverse. Disponível em: https://octoverse.github.com (acessado em 23/09/2020);
2. Measures of Central Tendency: Mean, Median, and Mode. Disponível em: https://www.sciencebuddies.org/science-fair-projects/science-fair/summarizing-your-data#meanmedianandmode (acessado em 25/09/2020).