Guia - Ferramenta PMD

SUMÁRIO

1. Objetivo

Guiar o usuário na instalação, configuração e utilização da ferramenta PMD para verificação estática automatizada de código.

2. Referências

Guia - PMD.pdf

3. Pré-Requisitos

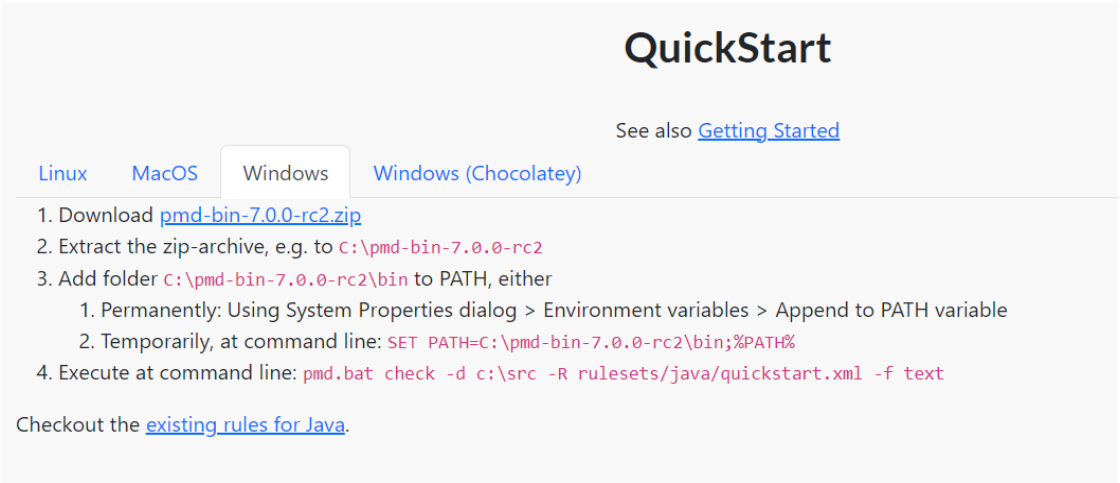
Pré-requisitos necessários para a instalação e funcionamento da ferramenta em sua máquina:

- Java 8 ou superior: Java JRE
- Compactador de zip:
  - Para windows: 7-zip ou Winzip
  - Para Linux: InfoZip

4. Instalação da ferramenta

Informações de como baixar e instalar a ferramenta

1. Acesse o site oficial do PMD em: https://pmd.github.io/
2. Siga as instruções na seção “QuickStart”, de acordo com seu sistema operacional



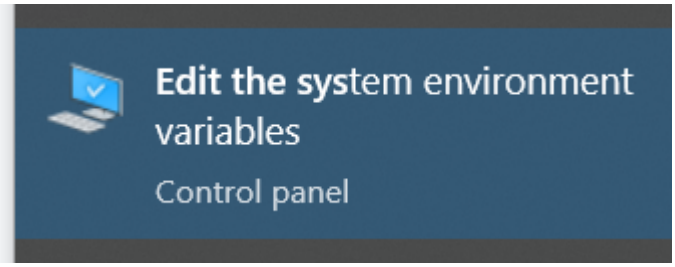
3. Para windows:

3.1. Faça download do PMD na seção "Downloads", preferencialmente a versão mais recente.

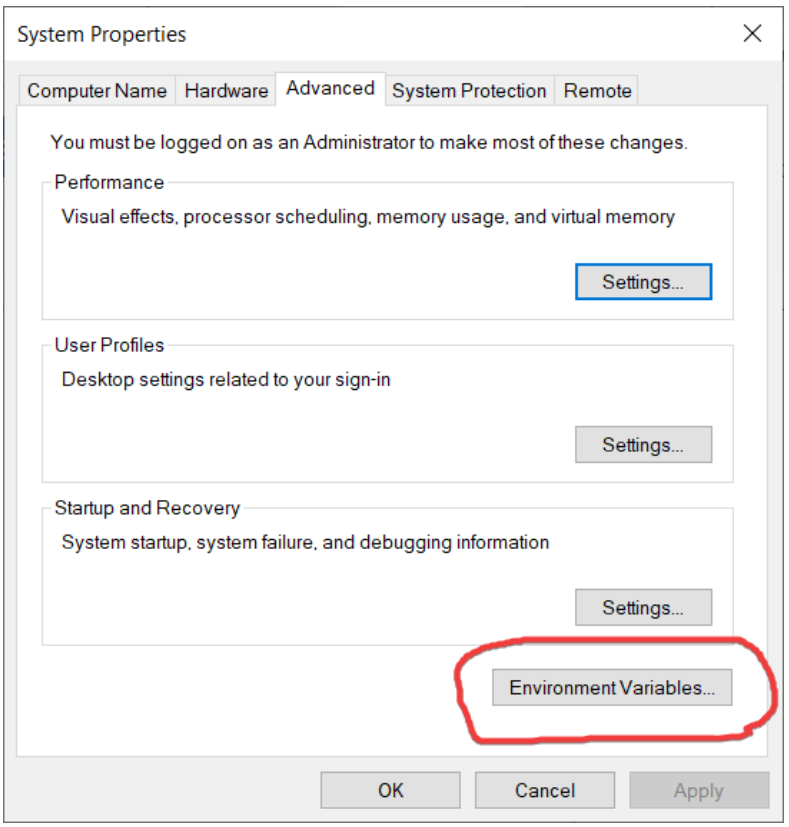
3.2. Extraia o conteúdo do zip no seu disco local “C:\”

3.3. Adicione o endereço da pasta “bin” para a variável PATH nas variáveis de ambiente do sistema. Para tal faça o seguinte:

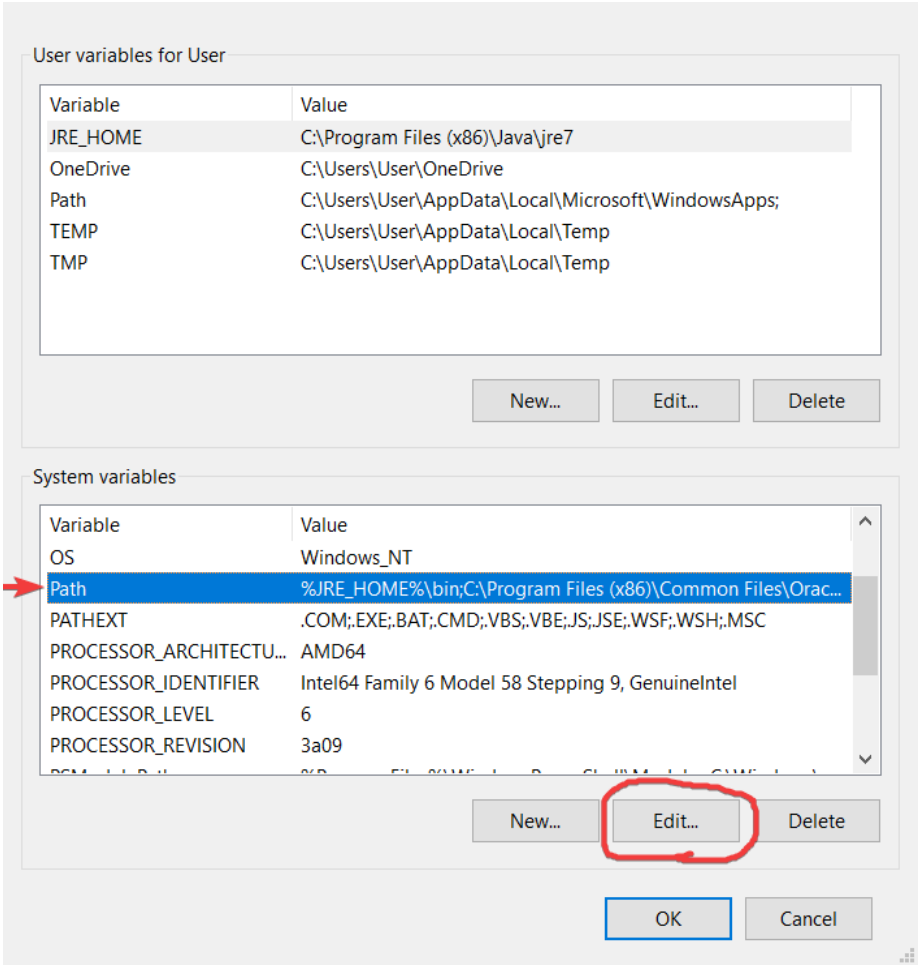
3.3.1. Pesquise por “Editar as variáveis de ambiente” na barra de pesquisa do windows



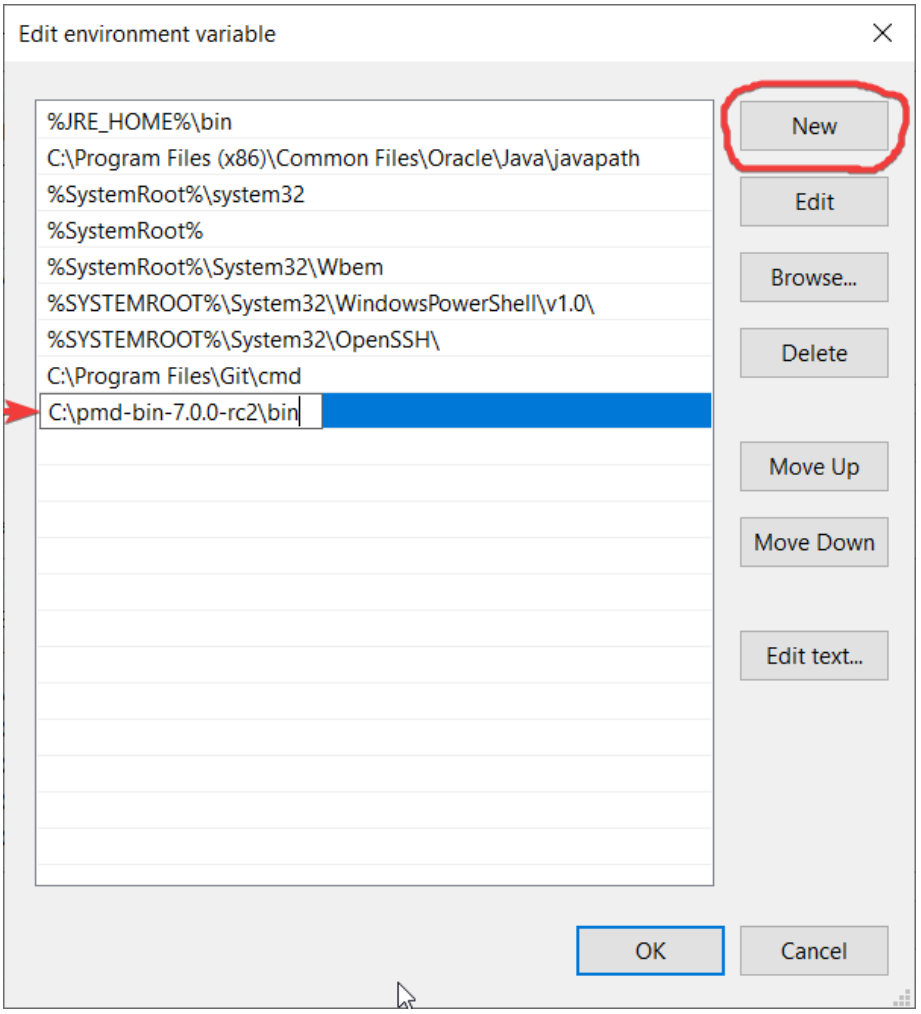
3.3.2. Na janela de “Propriedades do Sistema” e dentro da aba “Avançado”, selecione a opção "Variáveis de Ambiente”



3.3.3. Dentro da janela "Variáveis de Ambiente”, selecione a variável “Path” dentro de "Variáveis de Sistema”, em seguida clique na opção “Editar”



3.3.4. Clique na opção “novo” para adicionar uma nova variável, e digite o endereço da pasta “bin” de acordo com a versão do PMD instalada no disco. Exemplo: para a versão “7.0.0-rc2” o endereço é “C:\pmd-bin-7.0.0-rc2\bin”



3.3.5. Após digitar o endereço, clique em “OK” nas duas últimas janelas, e em "Aplicar" na janela de “Propriedades do Sistema”.

3.4. Por fim, execute a seguinte linha de comando: `pmd.bat check -d c:\src -R rulesets/java/quickstart.xml -f text.` Para tal, você deve utilizar o Prompt de Comando.

5. Utilização da Ferramenta

Após instalada, a ferramenta pode ser executada de diversos modos:

- Como um plugin do Maven
- Como um “Task” do Ant.
- Como um "Task” do Gradle.

- Por linha de comando

Detalharemos a execução por linha de comando, por ser a mais simples e de uso imediato, sem necessitar de downloads ou instalações adicionais.

Para realizar a análise de código via linha de comando no Windows, você deve executar o comando "`pmd.bat check`" juntamente de no mínimo uma opção e listas de endereços:

1. As opções mais usadas são:
   1.1. `-f <format>`: especifica o formato do relatório a ser gerado, com os mais básicos sendo `text` e `xml`. Outras opções incluem `csv`, `textpad`, `textcolor`, `json`, `html`, dentre outros. Para uma lista completa de formatos e seus usos, acesse: https://docs.pmd-code.org/latest/pmd_userdocs_report_formats.html

   1.2. `--aux-classpath <classpath>`: especifica o classpath das bibliotecas utilizadas pelo código analisado. O uso correto dessa opção permite ao PMD realizar uma análise muito mais profunda.

   1.3. `-R <path>`: especifica o arquivo com o conjunto de regras (ou ruleset) a ser usado na verificação. O PMD utiliza arquivos de configuração xml, chamados de rulesets, que especificam quais regras devem ser usadas na verificação. Também é possível executar uma única regra especificando sua categoria e nome.

   1.4. `<source>`: especifica a origem a ser analisada, podendo ser um arquivo, diretório, jar ou zip contendo o código. As opções `-d <path>` e `--dir <path>` são equivalentes a essa opção.

   Lista completa das opções disponíveis para serem usadas: https://docs.pmd-code.org/latest/pmd_userdocs_cli_reference.html

Exemplo de execução válida do PMD via linha de comando:

```
pmd.bat check -f text -R rulesets/java/quickstart.xml ..\..\src\main\java
```

Analisando a execução do comando anterior: "`-f text`" especifica que o relatório deve ser gerado em formato de texto, "`-R rulesets/java/quickstart.xml`" especifica

o arquivo xml com o ruleset a ser usado na verificação, e “`..\..\src\main\java`” especifica o arquivo a ser analisado. Perceba que o ruleset “quickstart.xml” é um arquivo de regras padrão para a linguagem Java fornecido pelo PMD, é fortemente recomendado a criação do seu próprio conjunto de regras.

A execução do comando deve ocorrer da seguinte forma:

```
C:\> pmd.bat check -f text -R rulesets/java/quickstart.xml ..\..\src\main\java

.../src/main/java/com/me/RuleSet.java:123  These nested if statements could be combined
.../src/main/java/com/me/RuleSet.java:231  Useless parentheses.
.../src/main/java/com/me/RuleSet.java:232  Useless parentheses.
.../src/main/java/com/me/RuleSet.java:357  These nested if statements could be combined
.../src/main/java/com/me/RuleSetWriter.java:66     Avoid empty catch blocks
```

6. Criação de Rulesets

A criação de arquivos xml com os conjuntos de regras personalizados a serem usados na verificação, adaptados às necessidades do projeto, é crucial para a utilização efetiva da ferramenta.

6.1. Criação de Rulesets com regras do PMD

Uma maneira mais fácil e rápida de criar e configurar rulesets é por meio da utilização das regras já fornecidas pelo PMD. Para criar seu próprio ruleset você deve criar um arquivo xml, e em seguida pode utilizar o seguinte template:

```
<?xml version="1.0"?>
<ruleset name="Custom Rules"
    xmlns="http://pmd.sourceforge.net/ruleset/2.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://pmd.sourceforge.net/ruleset/2.0.0
https://pmd.sourceforge.io/ruleset_2_0_0.xsd">
    <description>
        My custom rules
    </description>
    <!-- Your rules will come here -->
</ruleset>
```

A seguir, é possível referenciar as regras a serem utilizadas no processo de verificação individualmente, dentro do elemento "`<ruleset>`". Observe o seguinte exemplo para a adição da regra `EmptyCatchBlock` para o ruleset criado:

```
<rule ref="category/java/errorprone.xml/EmptyCatchBlock" />
```

As referências dentro do elemento "`<rule>`" apontarão para uma categoria de regras, e possivelmente para uma regra específica, como é o caso. Neste exemplo, estão sendo referenciadas as categorias de regras para a linguagem `java`, a categoria `errorprone` e por fim a regra `EmptyCatchBlock` dentro dessas categorias.

Também é possível incluir múltiplas regras em um único elemento "`<rule>`", para tal referenciando uma categoria inteira, e fazendo uso dos elementos "`<exclude>`" e "`<include>`" para excluir e incluir regras, respectivamente:

```
<rule ref="category/java/codestyle.xml">
    <exclude name="WhileLoopsMustUseBraces"/>
    <exclude name="IfElseStmtsMustUseBraces"/>
</rule>
```

Neste exemplo foi referenciada a categoria (ou ruleset) `codestyle.xml` inteira, incluindo assim todas as suas regras. Consecutivamente foram utilizados dois elementos "`<exclude>`" para remover as regras `WhileLoopsMustUseBraces` e `IfElseStmtsMustUseBraces` do ruleset a ser utilizado

Finalmente, é possível excluir e re-incluir arquivos de serem verificados pelo ruleset criado, utilizando os elementos "`<exclude-pattern>`" e "`<include-pattern>`", respectivamente:

```
<ruleset name="myruleset"
            ...
    <exclude-pattern>.*/some/package/.*</exclude-pattern>
  <exclude-pattern>.*/some/other/package/FunkyClassName.*</exclude-pattern>
    <include-pattern>.*/some/package/ButNotThisClass.*</include-pattern>
</ruleset>
```

Para uma lista completa de todas as regras disponíveis, bem como informações sobre configuração de propriedades próprias das regras, acesse: https://docs.pmd-code.org/latest/tag_rule_references.html.

6.2. Criação de Regras Customizadas

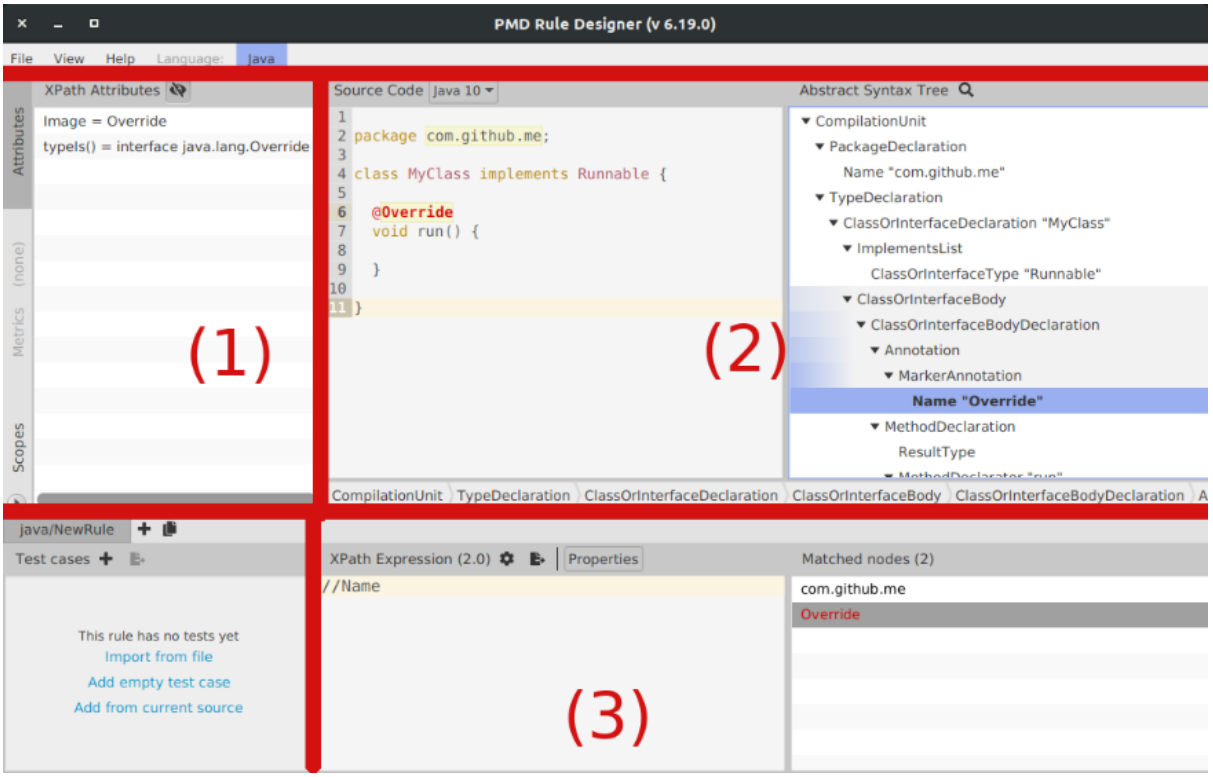
Outra possibilidade é a criação de regras customizadas. Existem duas maneiras para criar regras através do PMD:

- Usando XPath
- Usando Java

Abordaremos a criação através do XPath, dado que na data de criação deste guia o método via Java ainda estava sendo desenvolvido.
Para criar uma regra usando XPath, utilizaremos a ferramenta Rule Designer, já inclusa na instalação do PMD. Para abrir a ferramenta use o comando:

```
C:\> pmd.bat designer
```

A aplicação deve executar em seguida, mostrando o seguinte Layout:



A região (2) é o editor principal, você deve escrever o trecho de código a ser verificado na seção à esquerda, assim que escrever, a estrutura de dados chamada

de "Abstract Syntax Tree” (ou AST) relacionada ao código irá aparecer à direita. Essa estrutura é composta por nodos (ou “nodes”) em cadeia.
A região (1) mostra os atributos XPath referentes ao “node” selecionado na estrutura AST do código.
A região (3) é o editor XPath, ao inserir uma consulta XPath nessa área os resultados compatíveis na estrutura AST serão exibidos na seção “Matched Nodes”, no canto inferior direito.

O processo para criação de regras é o seguinte:

1. Escreva no editor principal o trecho de código com os erros que se procuram tratar.
2. Examine a tabela AST e identifique em qual “node” a violação deve ser reportada.
3. Escreva uma expressão XPath compatível ao “node” no editor XPath.
4. Refine a expressão XPath iterativamente usando diferentes trechos de código, de modo que ela trate vários casos de violação sem se referir a outros “nodes”
5. Exporte sua expressão XPath para um elemento “`<rule>`” dentro de seu arquivo XML, colocando-o em seu ruleset.

Suponhamos que você trabalhou com o seguinte trecho de código:

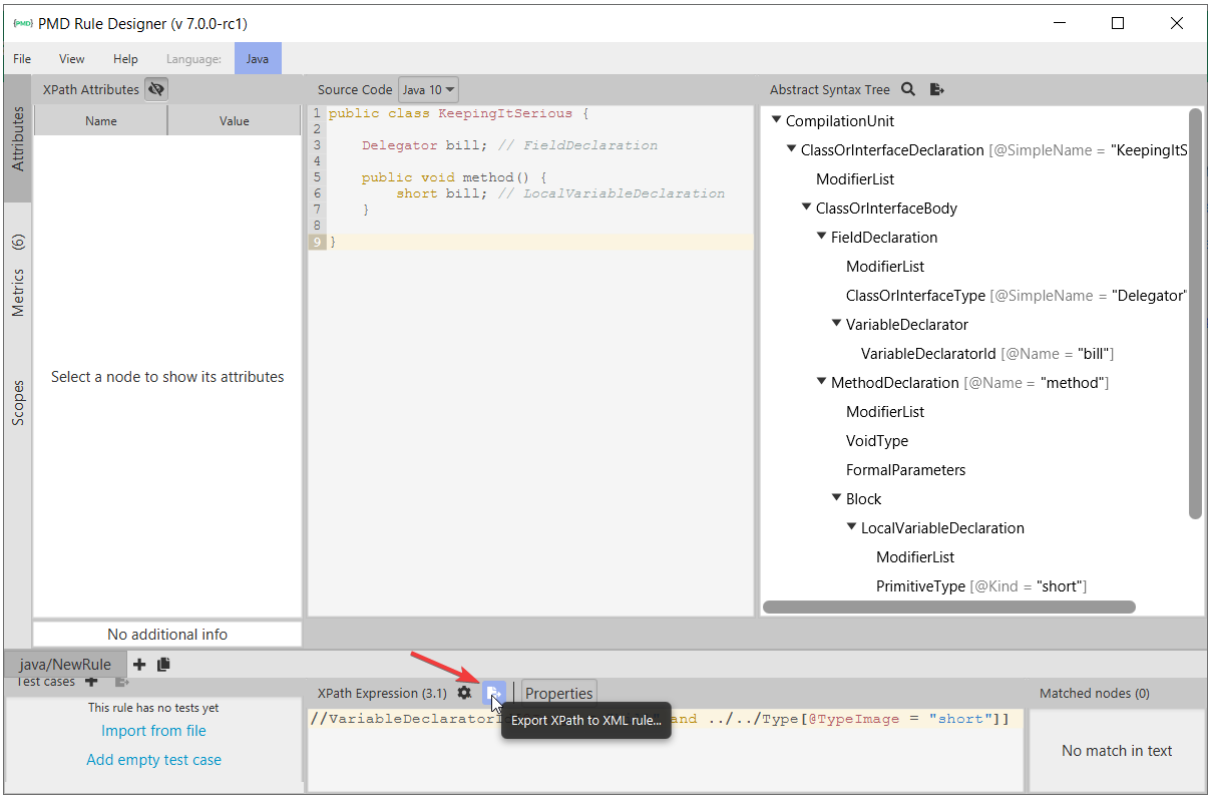
```
public class KeepingItSerious {
    Delegator bill; // FieldDeclaration

    public void method() {
        short bill; // LocalVariableDeclaration
    }
}
```

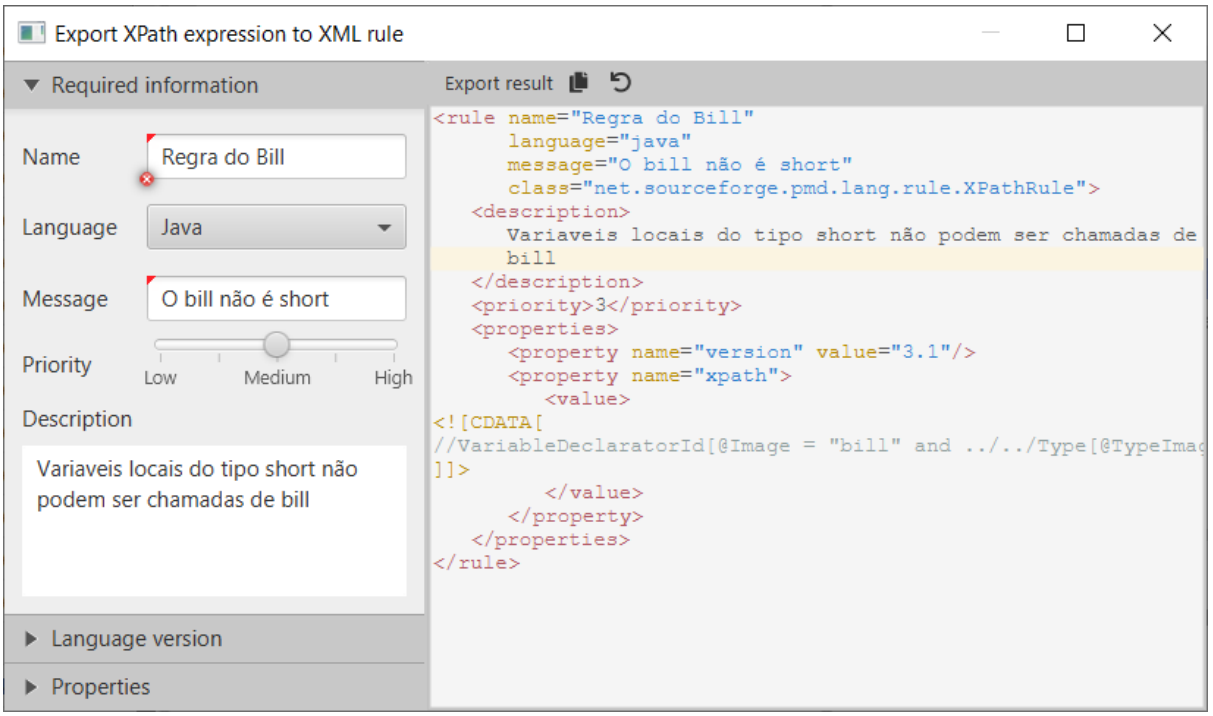
E gerou a seguinte expressão XPath para impedir que variáveis locais do tipo “short” sejam chamadas de “bill”:

```
//VariableDeclaratorId[@Image = "bill" and ../../Type[@TypeImage ="short"]]
```

Para exportar essa expressão para seu ruleset XML, você deve:



Na região do editor XPath, selecione a opção de “Export XPath to XML rule”



Na próxima janela, customize as informações da sua regra, o código do elemento "`<rule>`" a ser adicionado em seu "`<ruleset>`" aparecerá à direita, apenas copie e cole esse código dentro do arquivo XML que você irá usar.

Para mais informações sobre o uso do Rule Designer e criação de regras, acesse: https://docs.pmd-code.org/latest/pmd_userdocs_extending_your_first_rule.html

7. Métricas Utilizadas

O PMD usa um conjunto de métricas e regras para realizar a verificação estática de código, dentre elas destacamos:

1. Complexidade Ciclomática (Cyclomatic Complexity): Essa métrica mede a complexidade de um método ou função com base no número de caminhos independentes que podem ser percorridos durante a execução do código. Quanto maior a complexidade ciclomática, maior a chance de existirem problemas no código.

2. Linhas de Código (Lines of Code - LOC): Essa métrica simplesmente conta o número de linhas de código em um método, função ou arquivo. O PMD pode usar essa métrica para identificar métodos ou funções muito longos que podem ser difíceis de entender ou manter.

3. Número de Parâmetros (Number of Parameters): Essa métrica mede o número de parâmetros passados para um método ou função. Muitos parâmetros podem indicar uma complexidade maior e podem ser difíceis de gerenciar. O PMD pode sugerir refatorações para reduzir o número de parâmetros.

4. Número de Métodos (Number of Methods): Essa métrica conta o número de métodos ou funções em uma classe ou arquivo. Um grande número de métodos pode indicar uma classe muito complexa e pode ser difícil de entender e manter.

5. Duplicação de Código (Code Duplication): Essa métrica identifica trechos de código duplicados. O PMD pode apontar essas duplicações e sugerir refatorações para reduzir a redundância.

8. Documentação Online

Caso ocorra algum problema ou alguma dúvida persista, acesse a documentação oficial online em: https://docs.pmd-code.org/latest/index.html