Continue

単純にソースコードファイルの改行数をそのままステップ数として定義する場合もあります。しかし、コードにはコメントや中括弧のみで構成されるような、厳密にはコードの規模や複雑さに影響を与えない行も含まれています。したがって、プログラミング言語の記述方法に応じてステップ数の数え方が異なる場合も多くあります。 ステップ数は、「 物理ステップ数 (Physical Lines of Code) 」と「 論理ステップ数 (Logical Lines of Code) 」の2つに大別されます。 物理ステップ数は単純にソースコードのファイルの総行数のことをさします。物理ステップ数は、どんな言語を用いたとしても一定で、数え方も一通りです。 一方、論理ステップ数は物理ステップ数から空行やコメントの行など、ソースコードの規模や複雑さに影響しない行を除いた総行数をさします。 物理ステップ数に対して論理ステップ数は、計算方法が言語や計算担当者によって異なる場合が多いようです。どのような基準で算出したのかを明確にする必要がありますが、有効なソースコードの規模や複雑さをより的確に表現できるのは論理ステップ数だと言えるでしょう。 テスト密度とは 単体テストにおけるステップ数について取り上げる際に欠かせないのが、「テスト密度 」という概念です。テスト密度とは、テスト対象のステップ数に対してどの程度のテストが実行されるのかを割合で示した値です。テストを計画する際にテスト対象のシステムが達成すべき適正品質を実現するため、システムの規模に応じた適切なテスト工数を判断する指標となります。 こちらの値は、 (テスト項目) ÷ (テスト対象の総ステップ数) によってパーセンテージで表します。 開発現場によっては、テスト密度の数値を設定し、これをクリアすることをテスト設計時の必須要件として設ける場合もあります。しかし、短い行数のシンプルなコードに対して少ないI項目のテストで良いとは必ずしもいいきれませんし、冗長なコードだからと言ってテスト項目を増やすことが最適とも言えません。 テスト密度の数値のみで厳密なテスト項目数を管理するのではなく、テスト密度の考え方を1つの情報として取り入れて、単体テストのカバレッジおよびテスト品質の向上に役立てるのが良いでしょう。 ステップ数を数える必要性 ステップ数は、プログラムの規模を測る客観的な指標として用いられることがあります。この数に即して、対象のソースコードがどの程度の規模・ボリュームを判断します。 しかし、ステップ数はソフトウェアの規模を表す指標とはいえ、ステップ数を計測することだけでソフトウェアの規模を正確に測ることができるわけではありません。なぜなら、コードの総行数が必ずしもシステムの規模や複雑性を忠実に表現している訳ではないからです。 すなわち、ステップ数が多いシステムと不具合の出現率は必ずしも相関しない、といえます。ステップ数を軸としたテスト要件・項目の設計だけでは、テスト品質を適正なレベルまで引き上げることは難しいでしょう。 ただし、ステップ数を計測すること自体は、システムにかけるべきコストと適正品質を見極めるという観点では重要です。ソフトウェアの規模や複雑さを測る指標のあくまで1つだということを踏まえれば、他の指標や考え方と組み合わせ、情報を比較検討することで有効利用できます。 システムを構成する言語が異なっていたり、アーキテクチャの違いがあったりする場合、1つの指標で単純な比較をすることはやや軽率といえるかもしれません。したがって、例えば内部機能の複雑さをより利用者目線で計測するFP法など、ステップ数とは別の考え方と併用して判断指標の1つとするのが、最も有効な活用方法だと言えるでしょう。 おすすめのステップカウンタ3選 機械的に計測可能なステップ数は、ツールの活用が必須です。ツール自体は非常に多く提供されていますが、言語や操作性などのツールの特徴を踏まえることでより利用用途に合ったツールを選ぶことができます。 今回は、有用なツールを3つご紹介します。全て無償で利用できますので、開発状況に合ったものをぜひ導入してみてください。 StepCounter GUIとCUIの両方に対応した、使いやすさと機能性を兼ね備えたツールです。比較的幅広い言語に対応しており、測定結果をcssやxmlなど様々な形式で出力できるので、データの比較分析に流用することも可能です。 Eclipseプラグインとしての利用も可能ですので、ステップ数をカウントした結果の差分を比較することもできます。本ツールがあれば必要な測定・分析は可能になるでしょう。 CLOC WIndows・Macの両方で利用可能な、シンプルな操作性のツールです。コマンドラインで利用できるツールで、必要な機能を絞ったわかりやすい操作性が特徴です。 空行・コメント行・実質のコード行数の測定・結果出力・差分比較の基本的な機能が揃っていますので、使いやすさを重視しつつステップ数の概念を導入してみたい方におすすめのツールといえます。 Blue Step Counter 幅広い言語に加え、Microsoft Visual Studioのプロジェクトファイルにも対応した非常に高機能なツールです。また、独自のコメント形式を採用している場合でも、カスタマイズすることで対応可能です。拡張性が高く高機能なため、自社のプログラムにより適した形でステップ数を測定したい場合は本ツールを利用することをおすすめします。 おわりに 今回の記事では、単体テストのカバレッジを考慮する上で欠かせないステップ数という概念と、ステップ数を計測ツール「ステップカウンタ」を中心にご紹介しました。 品質とコストのトレードオフが発生しがちなテスト工程では、適切なテスト品質を設計することが非常に重要であり、そこで用いることができるのがステップ数という概念です。システムの規模や複雑さを測る1つの客観的な指標であるステップ数は、より正確な判断を下すために他の指標と組み合わせて活用するのが良いでしょう。 ステップカウンタはツールを用いて計測しますが、非常に多くのツールが提供されているのでどれを選ぶべきか迷うことがあるかもしれません。その場合は、使用言語が適切であること、開発環境への導入がしやすいこと、カウントしたい対象行数などを柔軟に選択できること、といった観点に注目し、ツールを選んでみましょう。 ステップ数やテスト密度といった指標を用い、テストの品質を適切に設定できるよう、取り組んでみてください。 資料ダウンロード 資料ダウンロード一覧へ StepCounterというオープンソースのEclipseプラグインを使って簡単に取得できたので紹介します ■環境 Windows7 64bit Eclipse 4.3 Kepler ■インストール から最新のjarファイルをダウンロードします 自分は「jp.sf.amateras.stepcounter_3.0.3.201404150124.jar」が最新版だったので左記のjarをダウンロードしました ダウンロードしたらEclipseのpluginsフォルダに配置します 「/path/to/eclipse/plugins/jp.sf.amateras.stepcounter_3.0.3.201404150124.jar」となればOKです Eclipseを起動（もしくは再起動）します 問題なくEclipseが起動すれば、インストールはこれで完了です ■使ってみる Eclipseが起動したらプロジェクトを右クリックして「ステップカウンタ」->「ステップ数をカウント」と選択します すると「カウント結果」というビューで結果を表示してくれます Javaの実行している行数の他に空行やコメントの数もちゃんとカウントしてくれるので実行ステップだけが正確に取得できます あとビューのどこかで右クリックするとExcel形式にエクスポートすることも可能です classファイルやjarファイル内はカウントできないようです（他にもカウントをサポートしていないファイル形式がある） とりあえずこれで簡単に実行ステップ数を算出することができるようになりました Eclipseのプロジェクトとしてすでにインポートしている場合はこれで簡単に算出できます またプロジェクトとしてインポートしていない場合でインポートするのが面倒という場合にはEclipseのプラグインでなくjarを直接コマンドで実行することでステップをカウントしてくれる機能もあるので Eclipseが嫌いな方、インポートが面倒な方はそれを利用するといいかと思いました ■参考サイト VSCodeのVS Code Counterのインストールしていること。 ※本記事では、以下バージョンで確認しました。 バージョン: 1.93.1 (user setup)コミット: 38c31bc77e0dd6ae88a4e9cc93428cc27a56ba40日付: 2024-09-11T17:20:05.685ZElectronBuildId: 10073054V8: 12.4.254.20-electron.0OS: Windows_NT x64 10.0.22631 VSCodeでソースコードを開いていること。 ※本記事では、nuxt-blog のコードを例に紹介します。 VSCodeの言語拡張がインストールされていること。 ※nuxt-blogの場合、vueファイルに対応したVue - Officialの拡張機能が必要。 VSCodeの拡張機能を押下します。（ショートカットキーは Ctrl+Shift+X ）Marketplaceで拡張機能を検索するの欄で、 VS Code Counterを入力します。 拡張機能「VS Code Counter」インストール インストールを押下し、VS Code Counterをインストールします。 インストール完了後、画面は以下のように表示されます。 「VS Code Counter」実行 エクスプローラー（Ctrl+Shift+E）でファイルを読み込んでいることを確認します。 「Ctrl + Shift + p」キーを押下し、コマンドパレットを開きます。 コマンドパレットにVSCode Counterを入力し、Count lines in workspaceを選択します。 集計結果 以下のように、集計結果のサマリが表示されればOKです。 Summaryに加えて、Languages(言語ごと)、Directories(ディレクトリごと)の集計結果を確認することができます。 nuxt-blogのcodeブロックは約24Kであることを確認できました。 出力結果はルートフォルダの.VSCodeCounterに保存されます。 特定のディレクトリをカウントする 特定のディレクトリをカウントする場合は、ディレクトリを選択し、 「右クリック > ディレクトリ内のコード行を数える」を選択します。 以下キャプチャの場合は、componentsディレクトリを対象にカウントします。 集計結果(componentsディレクトリ) 以下のように、components配下のカウント結果が出力されることを確認できました。 カウントがエラーになる場合 ファイルがあるにも関わらず以下エラーが発生した場合は、対象ファイルの拡張子に対応した拡張機能が入っていない可能性があります。 componentsの場合、配下のvueファイルを開くと.vueファイル用の拡張機能が未インストールとなっていて、コードも色分けされていない状態でした。 上記キャプチャ右下のインストールを押下して、Vueの拡張機能をインストールします。 以下はインストール完了後の画面です。 Vueの拡張機能インストール後は、.vueファイルが色分け表示されるようになりました。 ファイルの色分けを確認したら、VSCodeを再起動します。 再起動後、「ディレクトリを選択 > 右クリック > ディレクトリ内のコード行を数える」を再実行し、エラーが解消することをご確認ください。 確認した「VSCode Counter」のバージョン v3.5.0 VSCode学習者にお勧めの本 これ1冊でできる! Visual Studio Code 超入門 新品価格 ￥2,618 から (2025/5/11 14:22時点) Visual Studio Code パーフェクトマスター 新品価格 ￥2,495 から (2025/5/11 14:26時点) 以上で本記事の解説を終わります。よいITライフを！ HOMEパソコンを便利にするフリーソフトステップ数カウンターソフト 更新日：2025年04月09日 C、VB、PHP、Java、ASP などの作成 / 編集したソースファイルのステップ数 = 行数をカウントすることができるプログラマ向けのソフトウェアです。 コメント数を含めたり、ソースのみのカウントが実現できる、主に工数管理等に利用されるステップ数カウントツールです。関連ツール開発環境Diff（ファイル比較）開発・プログラミング支援その他カウンター ソースコードのステップ数をカウントするほか、変更前と変更後のソースコードのモジュール比較を行い、新規、修正元、修正、流用ステップ数の算出が可能なステップ数カウンタ・ツールです。C / C++ / VB / Java / MASM 用ファイルに対応。作者サイトにて公開されている拡張子個別情報ファイルを追加することにより、その他の言語（PHP / HTML / C# / Ruby / Perl / XML / Python など）であってもステップ数のカウントができます。かぞえチャオ！ のダウンロード・使い方スクリーンショットダウンロード（Vector）C / VB / C++ / C# / ヘッダー / SQL のステップ数をカウントするソフトソースファイルをカウントして Excel ファイルに保存するソフトです。対象フォルダーを選択し、「カウント & Excel 保存」ボタンをクリックすると、カウント対象ファイルを Excel に一覧表示。各ファイルについて、フォルダー、ファイル名、文字コード、言語、行数、有効行数、有効文字数、コメント文字数、空行数、ファイルサイズ、更新日時、カウント日時を表示し、行数 ~ ファイルサイズについては合計も表示できます。RakuStepCounter のダウンロード・使い方スクリーンショットダウンロードC、C++、JAVA、VB、Delphi ソースを読み込み、関数ソリー構成図、ステップ数データを取得できるカウンターソフトインクルードファイルツリー図や関数ツリー図の作成、ステップ数の計測を行うソフトです。Shift-JIS、Unicode の文字コード読み込みに対応し、取得したデータをテキスト、HTML、CSV、エクセル形式にて保存できます。EPTREE のダウンロード・使い方スクリーンショットダウンロードコードの総行数、コメント行数、実行数、空行数などを表示するステップカウンターC、C++、JAVA、SQL、VB、アセンブラのソースファイルのタイムスタンプ、ファイルサイズ、総行数、コメント行数、実行数、空行数、コメント率を表示できるステップカウンターソフトです。 複数言語の同時カウントにも対応しており、各言語ごとに実行結果を表示してくれます。カウントした結果はテキストファイルとして出力でき、空行をコメントとしない、空行を総行数に含めないといった設定も可能です。 コメント書うんた のダウンロード・使い方スクリーンショットダウンロード（Vector）フリーソフト100では、選りすぐりのソフトを集めて提供しています。無料ソフトの機能では、もの足りない時に是非ご覧ください。 HOMEパソコンを便利にするフリーソフトステップ数カウンターソフト 更新日：2025年04月09日 プログラムファイルのステップ数集計ツールです。 開発の現場ではあまり経験できなかった以下の点を意識して作成。 3層アーキテクチャや軽量DDDを意識したパッケージ構成 保守性や変更容易性を意識した実装 ファクトリクラスやシングルトンなどのデザインパターンを意識した実装 Mavenによるテスト自動化、実行形式Jarの作成、JavaDoc、テストレポートの出力 JavaDoc、テストレポートはGitHub Pagesに公開 GitHub Actionsを用いたCI/CD環境の構築 masterにブッシュ時に単体テスト実施・実行形式Jarのビルド実施 stepCounterforJava ├─docs ・・・ JavaDoc等のドキュメントが格納されているフォルダ │ ├─javadoc │ ├─test-coverage │ └─test-reports └─stepCounter ・・・ stepCounterForJavaの資源が格納されているフォルダ ├─src │ ├─main ・・・ javaソースコードが格納されている │ │ ├─java │ │ │ └─co │ │ │ └─jp │ │ │ └─stepCounter │ │ └─resources ・・・ 静的資源が格納されているフォルダ │ │ ├─img │ │ └─settings │ └─test ・・・ テストコードが格納されているフォルダ ├─target ・・・ ビルド後の資源が格納されているフォルダ │ ├─StepCounter-jar-with-dependencies.jar ・・・ステップカウント処理実行に利用する実行形式Jarファイル │ └─settings ・・・ 実行形式Jarファイルが参照する静的資源 └─log4j2.xml └─pom.xml ステップ数の集計ツール起動時に利用するファイルはStepCounter-jar-with-dependencies.jarです。 同階層にsettingsフォルダも存在する状態で起動してください。 pom.xmlが存在する階層で以下のコマンドを実行してください。 ビルド時にstepCounter>src>mainフォルダに存在するテストコードが起動します。 テスト成功の場合にStepCounter-jar-with-dependencies.jarファイルが作成されます。 Jarファイルの出力先はstepCounter>targetフォルダになります。 java -jar StepCounter-jar-with-dependencies.jar java -jar StepCounter-jar-with-dependencies.jar [オプション] -h:このメッセージを表示して終了する。 -i:対話モードで実行する。（オプションを指定しない場合はGUIモード） -s:スクリプトモードで実行する。（オプションを指定しない場合はUIモード） -input=[inputDirectoryPath]:ステップカウント対象のディレクトリパスを指定する。 ※「-s」オプションを利用する場合に指定してください。 -output=[outputFilePath]:カウント結果出力対象のファイルパスを指定する。 ※「-s」オプションを利用する場合に指定してください。 -asc=[sortTarget]:ステップカウント処理の出力順を[sortTarget]をキーとして昇順ソートする。 -desc=[sortTarget]:ステップカウント処理の出力順を[sortTarget]をキーとして降順ソートする。 [sortTarget]:0:ファイルパス, 1:総行数, 2:実行行数, 3:コメント行数, 4:空行数。 ファイルパス ,総行数,実行行数,コメント行数,空行数 /Users/xxA.java,30 ,20 ,4 ,6 /Users/xxB.java,15 ,8 ,2 ,5 /Users/xxC.java,15 ,12 ,2 ,1 合計 ,60 ,40 ,8 ,12 mvn clean test -Dmaven.test.failure.ignore=true site -DgenerateReports=false surefire-report:report jacoco:report Java SE 16 This is a tool to count the number of steps in a program file. Created with the following points in mind, which were not experienced much in the development field. Package structure with an awareness of 3-tier architecture and lightweight DDD Implementation with awareness of maintainability and ease of modification Implementation with awareness of design patterns such as factory classes and singletons Test automation by Maven, creation of executable Jar, JavaDoc, and test report output JavaDoc and test reports are published on GitHub Pages Build CI/CD environment using GitHub Actions Perform unit tests and build executable Jar when pushing to master stepCounterforJava ├─docs ・・・ Folder where documents such as JavaDoc are stored. │ ├─javadoc │ ├─test-coverage │ └─test-reports └─stepCounter ・・・ Folder in which resources of stepCounterForJava are stored ├─src │ ├─main ・・・ The folder where java source code is stored │ │ ├─java │ │ │ └─co │ │ │ └─jp │ │ │ └─stepCounter │ │ └─resources ・・・ Folder where static resources are stored │ │ ├─img │ │ └─settings │ └─test ・・・ Folder where test code is stored ├─target ・・・ Folder where resources after build are stored │ ├─StepCounter-jar-with-dependencies.jar ・・・Executable Jar file used for step count processing execution │ └─settings ・・・ Static resource that executable form Jar file refers to │ └─log4j2.xml └─pom.xml The file to be used when starting the step counting tool is StepCounter-jar-with-dependencies.jar. Please start the tool with the settings folder also existing on the same level. Execute the following command in the hierarchy where pom.xml exists. Test code in stepCounter>src>main folder will be invoked at build time. The StepCounter-jar-with-dependencies.jar file will be created on test success. The Jar file will be output to the stepCounter>target folder. java -jar StepCounter-jar-with-dependencies.jar java -jar StepCounter-jar-with-dependencies.jar [options]. -h: Print this message and exit. -i: Run in interactive mode. (GUI mode if no option is specified) -s: Run in script mode. (GUI mode if no option is specified) -input=[inputDirectoryPath]: Specify the directory path for step counting. * Specify when the `-s' option is used. -output=[outputFilePath]:Specify the file path for count result output. *Specify when "-s" option is used. -asc=[sortTarget]:Sort the output order of the step count process in ascending order using [sortTarget] as a key. -desc=[sortTarget]: Sort the output order of the step count process in descending order using [sortTarget] as key. [sortTarget]:0:file path, 1:total lines, 2:executed lines, 3:comment lines, 4:empty lines. ファイルパス ,総行数,実行行数,コメント行数,空行数 /Users/xxA.java,30 ,20 ,4 ,6 /Users/xxB.java,15 ,8 ,2 ,5 /Users/xxC.java,15 ,12 ,2 ,2 ,1 Total ,60 ,40 ,8 ,12 mvn clean test -Dmaven.test.failure.ignore=true site -DgenerateReports=false surefire-report:report jacoco:report Java SE 16 Eclipseでステップ数を集計する方法です。 ステップ数とは、コメントや空白行を除いたコードの行数のことです。 プロジェクト規模の目安になるなんて言われていますね。 今回は「ステップ数によって予測されるバグ数に達しているかをチェックするため」という、謎の黒魔術のためにステップ数を集計することになったので、Eclipseの作業手順を残しておきます。 ここでは Eclipseでステップ数を集計する方法 を紹介します。 目次1 Eclipseプラグインの導入2 本家サイト3 まとめ Eclipseプラグインの導入 Eclipseでステップ数を集計するには「StepCounter」というプラグインを導入します。 下記のサイトから「 jp.sf.amateras.stepcounter_x.x.x.jar 」をダウンロードし、Eclipseの「plugins」ディレクトリに配置します。 ・今回はバージョン 3.0.4 を利用しました。 Eclipseを起動し、パッケージ・エクスプローラーからステップ数を集計したいディレクトリ、またはファイルを右クリックして、ステップ数をカウントします。 すると「カウント結果タブ」に集計結果が表示されますので、右クリックから「クリップボードにコピー」や「Excelファイルに保存」などを使ってファイルに保存します。 ファイルを開くと、、、 おおおー、ステップ数が集計されているー！！ 本家サイト ・GitHub - takezoe/stepcounter まとめ Eclipseでステップ数を集計する方法を紹介しました。 古い風習が残る企業では、相変わらずステップ数で工数見積やテスト密度の指標に使われているところがあるようです。個人的には同じ結果を返すメソッドでも、コード行数には大きな差があるため、ステップ数に頼りすぎるのはどうかと思いますけどね。最小行でかけるプログラマーもいれば、ダラダラとクソースを書くプログラマーもいます。自動生成されるコードも多いし、開発手法の多様化が進んできた現代では、ステップ数による工数見積もりなど有力な基準とはならないと思うんだけどな。どうなんでしょうかね。とりあえず筆者の考えでは、ステップ数を把握するなら参考値として使えって感じですかね。 おつかれさまでした。 3Go to list of users who liked6Deleted articles cannot be recovered.Draft of this article would be also deleted.Are you sure you want to delete this article?infoMore than 1 year has passed since last update.@ke_hondaLast updated at 2024-03-07Posted at 2024-03-06 ステップ数を vscode で計測する方法を記載します。 プログラム内で実行される個々の命令やステートメントの数を表すもの ステップ数が多いほどプログラムが複雑であり、理解や保守が難しい ステップ数が少ないプログラムは、シンプルで理解しやすく、バグの発生リスクが低い プログラムのコード行数とは異なるプログラムが実行される際の命令やステートメントの数を数えるもので、1 つのコード行に複数のステップが含まれることがあります。たとえば、条件文やループがある場合、その中の複数のステップが 1 つの行に含まれる 詳しくは参考の記事をご覧ください vscode の拡張機能から VS Code Counter をインストールする: F1(もしくは ctrl + shift + p)を押下し、VSCodeCounter:ディレクトリ内のコードを数えるを選択する 計測したいディレクトリを選択する 出力される Markdown ファイルから結果を確認する アウトプット例 # Summary Date : 2024-03-06 11:06:42 Directory c:\\Users\\その他\\sample Total : 1 files, 9 codes, 2 comments, 3 blanks, all 14 lines Summary / [Details](details.md) / [Diff Summary](diff.md) / [Diff Details](diff-details.md) ## Languages | language | files | code | comment | blank | total | | :--------- | ----: | ----: | ---: | ------: | ----: | ----: | | Python | 1 | 9 | 2 | 3 | 14 | ## Directories | path | files | code | comment | blank | total | | :--- | ----: | ----: | ---: | ------: | ----: | ----: | | . | 1 | 9 | 2 | 3 | 14 | Summary / [Details](details.md) / [Diff Summary](diff.md) / [Diff Details](diff-details.md) 3Go to list of users who liked6comment2Go to list of comments3Go to list of users who liked6Deleted articles cannot be recovered.Draft of this article would be also deleted.Are you sure you want to delete this article? PR パソコン・インターネット活用術！無料で使えるプログラムソースのステップ数カウントツール「StepCounter」（ ステップカウンター ）の使い方を解説します。 StepCounterの事前設定であるjavaの設定StepCounterではjavaというコマンドを使いますが、javaというコマンドが実行可能であるか確認します。事前設定が必要であるかの判断方法コマンドプロンプトを開いて、下記コマンドを入力します。C:¥>java -versionもし、下記の通りにエラー無く、javaのバージョンが表示されると事前設定は不要です。C:¥>java -version java version “11.0.2” 2018-10-16 LTS Java(TM) SE Runtime Environment 18.9 (build 11.0.2+7-LTS) Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.2+7-LTS, mixed mode)もし、下記の通りエラーが発生した場合は、事前設定が必要です。C:¥>java -version ‘java’ は、内部コマンドまたは外部コマンド、 操作可能なプログラムまたはバッチ ファイルとして認識されていません。java -versionコマンドでエラーとなった場合の対処手順java -versionコマンドを入力した時にエラーが表示された時の対処手順を解説します。 C:¥Program Filesの下に「java」フォルダがあるか確認「C:¥Program Files」フォルダの下に「java」フォルダが存在した場合、さらに「java」フォルダの下に「jdk」と頭に付いたフォルダがあります。例えば、下記のようなフォルダです。C:¥Program Files¥Java¥jdk-11.0.2setコマンドで「java.exe」へのパスを通すStepCounterを実行するには「java.exe」というコマンドが必要で、 「java.exe」がどこにあるかを「set」コマンドでWindowsに認識させます。今回は「C:¥Program Files¥Java¥jdk-11.0.2¥bin」というフォルダの中に「java.exe」が格納されているので、コマンド入力時に「C:¥Program Files¥Java¥jdk-11.0.2¥bin」フォルダの下を見るようWindowsへ指示するコマンドが下記の通り「set path=」です。C:¥>set path=C:¥Program Files¥Java¥jdk-11.0.2¥binsetコマンドの結果を確認するsetコマンドで「C:¥Program Files¥Java¥jdk-11.0.2¥bin」が登録されたことを確認するため、「set」コマンドを入力します。下記の通り「Path=C:¥Program Files¥Java¥jdk-11.0.2¥bin」が表示されたら正常です。C:¥>set ALLUSERSPROFILE=C:¥ProgramData ・ ・ OS=Windows_NT Path=C:¥Program Files¥Java¥jdk-11.0.2¥bin「java.exe」の格納場所を認識するとjavaコマンドが使えるようになるjavaコマンドが使えるようになるため、「java -version」コマンドがエラーなく実行できます。C:¥>java -version java version “11.0.2” 2018-10-16 LTS Java(TM) SE Runtime Environment 18.9 (build 11.0.2+7-LTS) Java HotSpot(TM) 64-Bit Server VM 18.9 (build 11.0.2+7-LTS, mixed mode) スポンサーリンク StepCounterのダウンロードStepCounterのダウンロード方法を解説します。StepCounterのダウンロードサイト下記のサイトへアクセスします。 ））ウンロードページへのアクセス下記の「こちら」がリンクになっているのでクリックします。 使い方 コマンドライン コマンドラインで使用するにはこちらからstepcounter-x.x.x-jar-with-dependencies.jarをダウンロードし・・・最終バージョンのディレクトリをクリック下記のバージョン単位のフォルダが表示されるので、最新バージョンである「3.0.4/」をクリックします。 Parent Directory – 3.0.0/ 2012-02-18 10:03 – 3.0.1/ 2012-04-21 12:41 – 3.0.2/ 2013-06-15 12:03 – 3.0.3/ 2014-04-15 01:10 – 3.0.4/ 2018-05-26 21:41 – maven-metadata.xml 2018-05-26 21:40 maven-metadata.xml.md5 2018-05-26 21:40 maven-metadata.xml.sha1 2018-05-26 21:40最新バージョンのStepCounterプログラムをダウンロード下記の中から「stepcounter-3.0.4-jar-with-dependencies.jar」をクリックするとダウンロードできます。このプログラムがStepCounterです。ダウンロードした後、「C:¥temp」フォルダへ「stepcounter-3.0.4-jar-with-dependencies.jar」を移動します。Parent Directory – stepcounter-3.0.4-jar-with-dependencies.jar stepcounter-3.0.4-jar-with-dependencies.jar.md5 stepcounter-3.0.4-jar-with-dependencies.jar.sha1 ・ ・StepCounterによるステップ数カウントの実行ステップ数カウント実行方法は、下記のコマンドです。末尾にプログラムソースが入っているフォルダを指定することで、指定したフォルダの下にあるプログラムソースだけステップ数をカウントしてくれます。C:¥temp> java -cp stepcounter-3.0.4-jar-with-dependencies.jar jp.sf.amateras.stepcounter.Main -format=csv -output=count.csv -encoding=UTF-8 [フォルダ名]指定可能なオプションは以下の通りです。-format出力フォーマットをcsv、excel、xml、jsonのいずれかで指定します。省略した場合はテキスト形式で出力します。-outputカウント結果を出力するファイルを指定します。省略した場合は標準出力に出力します。-encodingファイルの文字コードを指定します。省略した場合はプラットフォームのデフォルトエンコーディングを使用します。-showDirectoryvtrueを指定するとファイル名だけでなく起点となるディレクトリからの相対パスで表示します。StepCounterによるステップ数カウント実行例今回は「C:¥temp」フォルダに「stepcounter-3.0.4-jar-with-dependencies.jar」を格納したので、コマンドプロンプトを起動した後、下記コマンドを入力します。ステップ数カウントの実行「C:¥Subversion\ソースプログラム / モジュール」というフォルダにプログラムソースが入っているので当該フォルダを指定して実行します。C:¥temp>java -cp stepcounter-3.0.4-jar-with-dependencies.jar jp.sf.amateras.stepcounter.Main -format=csv -output=count.csv -encoding=UTF-8 C:\Subversion\ソースプログラム / モジュールステップ数カウントの実行結果StepCounterのプログラムを実行した「C:\temp」フォルダに、ステップ数のカウント結果である「count.csv」ファイルが作成されます。C:¥temp>dir 2019/12/27 09:29 . 2019/12/27 09:29 .. 2019/12/27 09:29 62,242 count.csv 2019/12/27 08:51 13,987,802 stepcounter-3.0.4-jar-with-dependencies.jar 2 個のファイル 14,050,044 バイト 2 個のディレクトリ 414,188,261,376 バイトの空き領域「count.csv」ファイルへの出力内容「count.csv」ファイルをメモ帳で開くと、下記の通りカウント結果が表示されます。なお、対応できないファイルには「未対応」と表示されます。tokai.mdb,未対応,,,, INPUTFORM.ini,INI,,57,26,5,88 EXTENT.INI,INI,,9,0,13,22 KAIHYO.INI,INI,,29,0,5,34 NameOut.ini,INI,,65,2,9,76 ・ ・上記の出力内容から一例を下記に説明します。数字は4つ出力されますが、左から下記の意味になります。INPUTFORM.ini,INI,,57,26,5,88「INPUTFORM.ini」は「カウント対象のファイル名」「INI」は「ファイルのタイプ」「（ 空欄 ）」は本来「カテゴリ」が表示されて、「Eclipseプラグイン」や「共通」等と表示57は「実行可能なプログラムソースの行数」26は「空の行」5は「コメント行」88は「上記3つの合計行数」なお、ステップ数のカウント結果はcsvファイルで出力しているので、エクセルで開くことが可能です。エクセルで開くと見やすくなります。 Eclipse Eclipseのコンソールログをファイル出力する方法 mockito Spring Bootでmockitoを使ってテストする方法 SVN SVNで特定のリビジョンに戻す方法(TortoiseSVN,Eclipse) PowerShell PowerShellでコンソールに文字列を出力する方法 PowerShell Visual Studio Code Visual Studio CodeでJSON形式のファイルを整形する Excel Excelのオートシェイプ内のテキストに取り消し線を入れる方法 Chrome PowerShell PowerShellスクリプトでデータ型を調べる方法 Javaのロジック Linux Linuxコマンドのsudoとsuとsudo suとsudo su -コマンドの違いを分かりやすく SVN HTML で数値のみ入力する方法 AWS ECMA Script6 (JavaScript) のnew Date()をUTCからJSTに変換する方法 (AWS Lambdaで注意) iBATIS(MyBatis) MyBatisでforeachを使用して動的SQLを生成する Visual Studio Code VSCodeでlaunch.jsonを作成してデバッグする方法 Config Spring Bootで@RequiredArgsConstructorを使用してコンストラクタインジェクションを使用する コマンドプロンプト コマンドプロンプトでファイル内の特定の文字列を削除する 「この機能何ステップくらい？」とか、「今日何ステップくらいコード書いたの？」とか、仕事でコードを書いていると、上長にコードのステップ数を問われることが多々あります。 個人的には、実コードを見ない人がステップ数から得る情報は何もないと思っていますが、 世の中的には、複雑度とか潜在バグ数とか難易度とか効率とか色々わかるらしいです。すごいですね。 コード量で見積もりも変わるようなので、 リファクタリングで他人が書いたコードを半分にしたりすると、 度々コード量が減ったことで問い詰められたりします。 ・・・やめましょうか、この話。 ともあれステップ数は計りたい ステップ数での見積もりを求められる以上、プログラマはステップ数で見積もれなければいけません。お仕事ですからね。 そんなときに参考になるのは、過去の自分が書いたコードや、機能検証で書いたコードだったりします。 求められるステップ数は、コメントや空行を除いた、実行されるコード部分のいわゆる「実ステップ数」です。 なので、単純にファイルの行数をカウントするだけでは不十分であり、ステップカウンタと呼ばれるツールを使って測定します。 自分は以下のEclipseプラグインを使っています。 takezoe/stepcounter · GitHub Eclipseがリリースされた時からEclipseと歩いてきた私にとっては、とても使い勝手が良くて満足していますが、 選択範囲のステップ数カウントできたらいいんだけどな 私が参考にステップ数を知りたい場合は、『このメソッド部分だけ』とか、部分的なことが多く、 最近はその頻度が増えていたので、先ほどのプラグインに選択範囲のステップ数を計測する機能を追加しました。 Release add a function to 3.0.3 · seriwb/stepcounter · GitHub 上記リンクでダウンロードしたJARファイルを、Eclipseのdropinsかpluginsフォルダに配置してからEclipseを起動すると使えるようになるはずです。 選択範囲を作ってから「ステップカウンタ ＞ 選択範囲をカウント」を実行すると、選択範囲の開始行から終了行までのステップ数がカウントされます。 選択範囲をカウント機能の解説 最初は、ステップ数をカウントするロジックに、対象データを渡すだけの簡単な修正かと思いましたが、 ステップカウンタ機能が呼び出された後に対象ファイルを読み込んでカウント処理を行っていたため、 いきなり諦めムードが漂いました。 というのも、今回は人様のコードを利用しているので、元のコードには極力手を入れないようにしたかったのです。 それならファイルを作ればいい 思いついてみれば簡単なことで、選択範囲だけのファイルを作ってしまえばいいわけです。 実際、その案で選択範囲のカウントは実施することができなのですが、一つ問題になったのはファイルの置き場でした。 RemoteSystemsTempFilesフォルダに配置する案など色々考えましたが、結局はプロジェクトルート直下に一時ファイルを作成する仕様にしています。 作成する一時ファイルのファイル名は、『stepcounter_scopeCountTempFile』で、 もしも同じファイルがプロジェクトルートにある場合は、実行時の日時を末尾に付けるようにしてあります。 そして作成したファイルは、結果表示後に削除しています。 EclipseプラグインEclipseプラグインの開発をしてみて 今回初めてEclipseプラグインの開発をしたわけですが、 例によって、コードを書いている時間よりもStepCounterの処理を追ったり、プラグイン開発の調べ物をしている時間が長かったです。 「既存コードが存在していたおかげで何とかできた」といった感じでした。SocialCoding様々です。 開発中に気になった個所はQiitaにまとめました。 Qiitaとはてなブログ Qiitaの良いところは、画面キャプチャをいちいちファイルに保存してアップロードしなくても、 キャプチャ後にQiita上でペーストすればいいところですね。 これ、前から自分で作りたかった機能の一つなので、今度自分でも実装してみようと思います。 Qiitaに都度Tipsを書いて、ブログにまとめるスタイルは意外とやり易い気がするので、 今後もこのスタイルでやっていこうと思います。 なので、良ければQiitaのアカウントもチェックしてみてください。 seri - Qiita そんな感じで 今回はお終いです。 選択範囲のステップ数カウント機能は、我ながら便利な機能を作ったものだとテンション高くなったので、 ぜひ皆さん使ってみてください。 せっかくなので本家の方にもプルリクは出したんですが、すこーしコードを綺麗にしたいところを後から見つけたりすると、 少し早まったかなと思ったりしますね。 今回のオチ そんな自画自賛の機能だったりしますが、今所属しているプロジェクトでは『ダウンロード禁止令』なるものが出ており、 これ、自分で使えてません。 転職しようかな・・・

- fiwivu
- monitomo
- nudoxeze
- is ias 28 still applicable
- https://mwasafat.com/uploads/files/14240709386.pdf
- http://indyztyle.com/ckfinder/userfiles/files/dogufote.pdf
- jobi
- what are the four 4 phases of curriculum development process and discuss each briefly
- yage
- http://showpelec.com/data/file/20250724_163752_569.pdf
- vixa
- how do u say machista in english
- https://alyosserspneed.com/userfiles/files/tominut-monegar.pdf
- husese
- http://bgliving.info/upload/files/d28d0979-e127-4d37-8560-f41ef63bba9d.pdf
- bavizu
- http://www.como-grup.ro/fisiere/file/luvaperesejexem_jawawupagozo_pixamokiguxilen_jexutizikamu.pdf