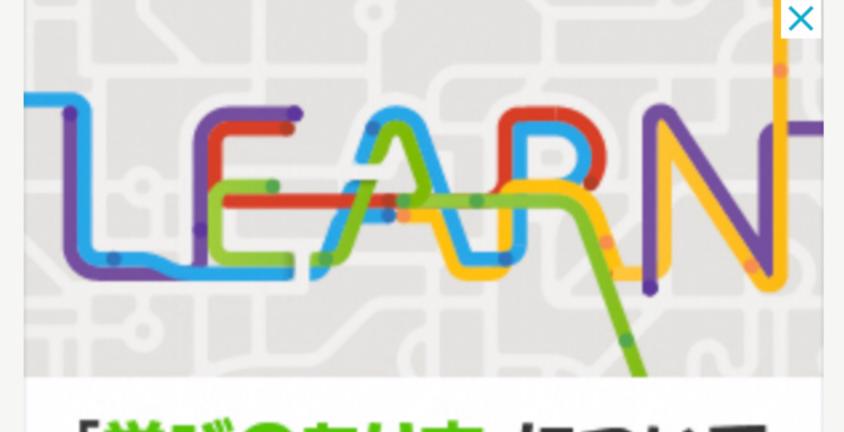
初めての参加でAmazonギフト券1万円ゲットのチャンス!Qiita Advent Calendar 2020

Smarter 9

technology

ノートPCの常識を変える



「学びのあり方」について マイクロソフト、NTTデータ、suinさんに 聞いてみた Qiita Zine try-finallyでのリソースクローズ

try-finallyでのリソースクローズの問題点 try-with-resourcesでのリソースクローズ 教訓 参考





15

LG TM

tryブロックの中で何らかのリソース(InputStream、OutputStream、BufferedReader等)を扱う場合、Java7以 前はfinallyブロックでcloseメソッドを呼び出すことで、tryブロック内の処理が正常終了したか異常終了したかに

関わらずリソースが確実に閉じられることを保証していました。以下のような形です。

protected void copy(String src, String dest) throws IOException { InputStream in = new FileInputStream(src); try { OutputStream out = new FileOutputStream(dest); try { byte[] buf = new byte[100]; int n; while ((n = in.read(buf)) >= 0) { out.write(buf, 0, n); } finally { out.close(); } finally { in.close();

また、finallyブロックで例外が発生しないよう更にチェックしたり等もありますよね。

```
} finally {
   if (br != null)
       try {
          br.close();
        } catch (IOException e) {
           // 例外処理
```

ただし、これらの書き方にはいくつか問題点があります。

## try-finallyでのリソースクローズの問題点

• tryブロックの中とfinallyブロックの中の両方で例外が発生した場合、finallyブロックの中で発生した例外がス

• close処理を色々な箇所で行ったりリソースのnullチェックを行ったり等コードが冗長となる

ローされる(本来tryブロックの中で発生した例外が知りたいはずだが、その例外は抑制される)

こういった問題に対して、Java7でtry-with-resources構文が導入されました。try-with-resources構文によりこれ らの問題は一挙に解決されます。

## tryのすぐ後ろにクローズの対象となるリソースの生成処理を記述します。クローズすべきリソースが複数ある場

protected void copy(String src, String dest) throws IOException {

try-with-resourcesでのリソースクローズ

合はセミコロンで区切ります。

```
try (InputStream inputStream = new FileInputStream(src); OutputStream outputStream = new
       byte[] buf = new byte[100];
       int n;
       while ((n = in.read(buf)) >= 0) {
           out.write(buf, 0, n);
このように書くことで以下のような挙動になります。
```

• finallyブロックでcloseメソッドを呼び出さなくても自動でcloseをしてくれるようになる(ソースが簡潔にな り可読性が向上する)

- close処理内で例外が発生してもtry文の中の例外がスローされる(本来知りたいはずの例外がスローされる)
- これにより「try-finallyでのリソースクローズの問題点」で挙げた問題点が解決されます。

## クローズしなければならないリソースを扱う場合は、try-finallyよりもtry-with-resourcesを使おう

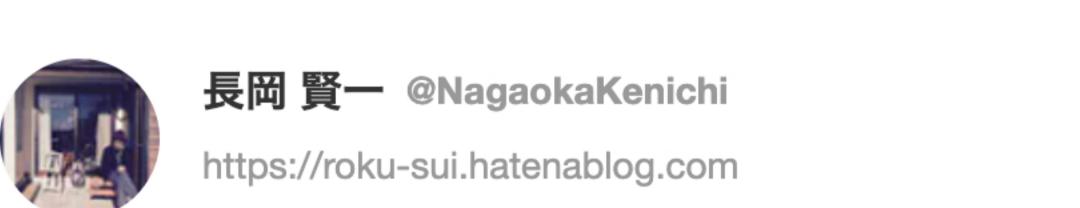
教訓

参考

## Oracle Technology Network Effective Java

□ ストック 編集リクエスト

フォロー





記事を読んでも解決しない…そんな疑問も

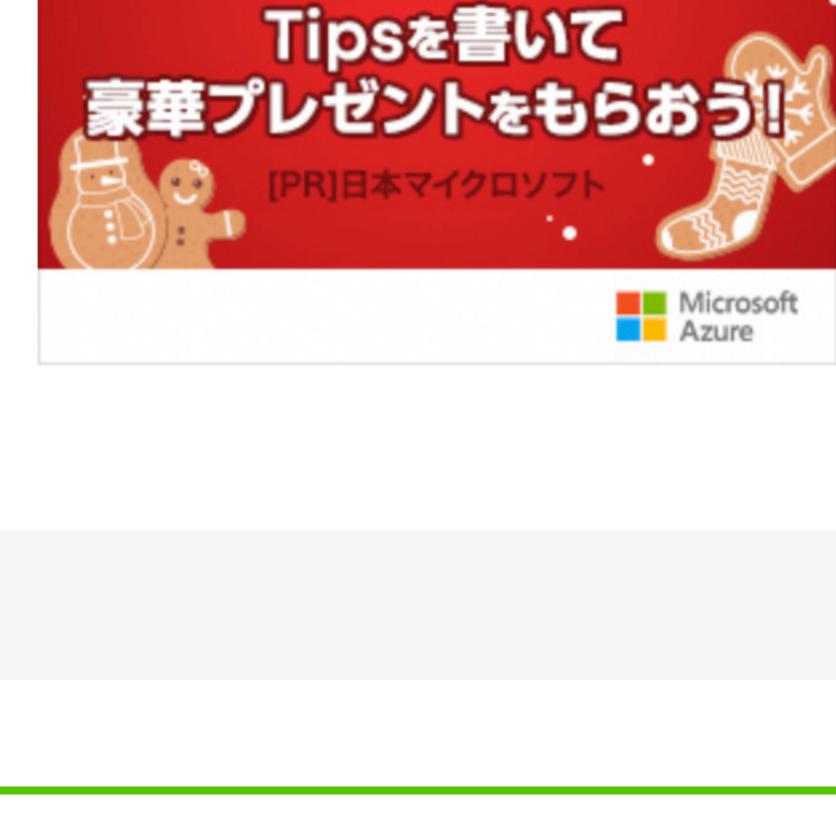
Q&A

プレビュー

How developers code is here.

© 2011-2020 Increments Inc.

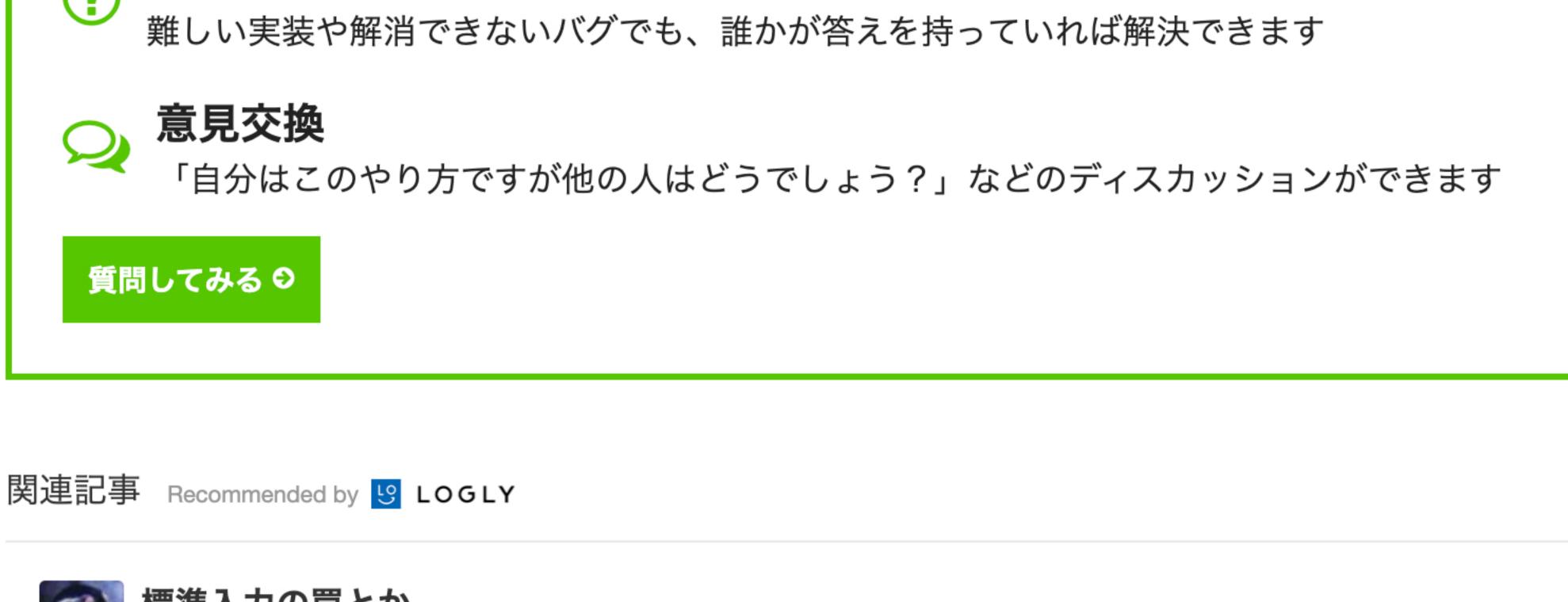
¥



Advent Calendar 2020

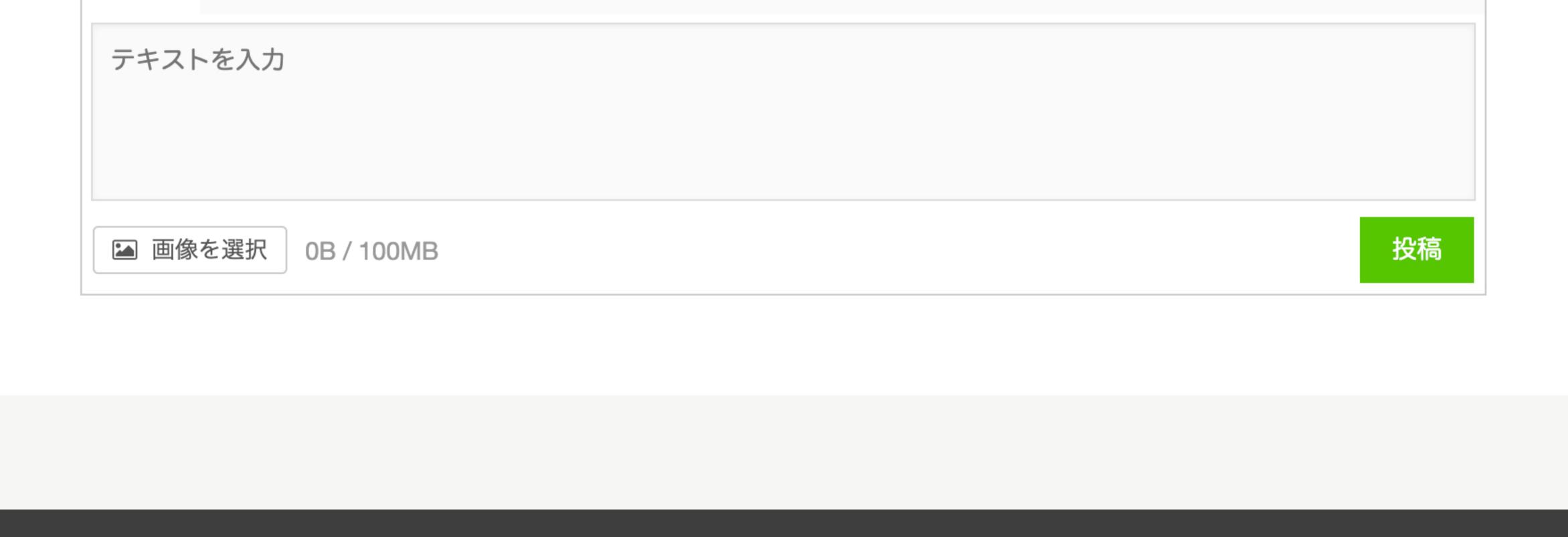
Cloud Native

アプリケーション開発の



質問でスッキリ解決するかもしれません!





Increments

Qiita

About

利用規約

リリース

プライバシー ヘルプ

ガイドライン 広告掲載

API

ご意見