* Lowercase: Là tên mà tất cả các chữ cái trong một từ được viết mà không Viết Hoa (ví dụ: while, if, mypackage).

* Uppercase: Là tên trong đó tất cả các chữ cái trong một từ được viết bằng chữ HOA. Khi có nhiều hơn hai từ trong tên, hãy sử dụng dấu gạch dưới để phân tách chúng (ví dụ: MAX\_HOURS, FIRST\_DAY\_OF\_WEEK).

* CamelCase: (còn được gọi là Upper CamelCase) là tên mà mỗi từ mới bắt đầu bằng chữ in hoa (ví dụ: CamelCase, CustomerAccount, PlayingCard).

* Mixed case: (còn được gọi là Lower CamelCase) giống như CamelCase ngoại trừ chữ cái đầu tiên của tên được viết bằng chữ thường (ví dụ: hasChildren, customerFirstName, customerLastName).

**1 Tên packages nên được viết thường.**

Với các dự án nhỏ chỉ có một vài package, bạn chỉ cần đặt cho chúng những cái tên đơn giản (nhưng có ý nghĩa!)

package pokeranalyzer package mycalculator

Nhưng trong các công ty phần mềm và các dự án lớn nơi các package có thể được nhập vào các package khác, các tên thường sẽ được chia nhỏ.  
  
  
Thông thường, điều này sẽ bắt đầu với domain công ty trước khi được chia thành các lớp hoặc tính năng:

package com.mycompany.utilities package org.bobscompany.application.userinterface

**2 Tên class** phải áp dụng **CamelCase**

Cố gắng sử dụng danh từ vì một class thường đại diện cho một cái gì đó trong thế giới thực:

class Customer class Account

**3 .Tên Method trong Java** nên được áp dụng quy tắc **Mixed case**.

Và nên sử dụng các động từ để mô tả những gì phương thức sẽ làm:

void calculateTax() string getSurname()

**4 Tên biến**  nên tuân theo quy tắc **Mixed case**.

 Các tên biến sẽ đại diện cho những gì giá trị của biến đại diện:

string firstName int orderNumber

Chỉ sử dụng tên biến ngắn khi các biến tồn tại ngắn, chẳng hạn như trong vòng lặp for:

for (int i=0; i<20;i++) {   //i only lives in here }

**5 Tên hằng số tiêu chuẩn trong Java nên được Viết Hoa (quy tắc uppercase)**

static final int DEFAULT\_WIDTH static final int MAX\_HEIGHT

## 6  Quy ước viết tên class: sử dụng dấu gạch ngang "-"

Ví dụ thay vì chúng ta viết class như thế này

.innerBox {

border: 1px solid green;

}

thì chúng ra sẽ đổi thành như này cho đúng

.inner-box {

border: 1px solid green;

}

#### **7 Đặt tên cho biến**

#### **Đặt tên cho các số nguyên (Integer)**

Có một số biến luôn luôn là số nguyên, ví dụ như age hoặc year. Ai cũng có thể hiểu được kiểu dữ liệu cho age và year là số, cụ thể là số nguyên. Cho nên bạn không cần làm gì với chúng, bản thân chúng đã tự nói cho bạn biết kiểu của nó là gì.

Một trong những trường hợp sử dụng kiểu số nguyên nhiều nhất là đếm số lượng của một thứ gì đó.

Bạn có thể tìm thấy được các loại biến kiểu như thế này trong bất cứ source code nào. Tôi hay sử dụng các quy ước đặt tên như sau cho các biến kiểu này: numberOfSomething, somethingCount...

Ví dụ: numberOfFailures hoặc failureCount.

Một số người thì hay đặt tên kiểu failures, nhưng vấn đề ở đây là tên biến này không hề chỉ định rõ ràng ý nghĩa và kiểu của biến, điều đó khiến những người khác khi đọc vào đoạn mã này sẽ tốn thêm một chút thời gian để đọc hiểu. Theo ý kiến của tôi, biến failures mang ý nghĩa chỉ định nó là một kiểu collection (một array hoặc list của failures).

Nếu đơn vị của biến không rõ ràng, tôi luôn thêm một chút mô tả đơn vị của biến vào phía sau. Ví dụ, thay vì tooltipShowDelay, tôi sẽ sử dụng tooltipShowDelayInMillis hay thậm chí tooltipShowDelayInMillisecs.

#### 

#### **Đặt tên cho các kiểu số thập phân (Float)**

Số thập phân không quá phổ biến như số nguyên, nhưng thỉnh thoảng bạn sẽ cần tới chúng cho một số giá trị đo lường không nên được làm tròn (ví dụ như giá cả - price, chiều cao - height, chiều rộng - width, trọng lượng - weight).

Khi cần sử dụng kiểu số thập phân, tôi thường đặt theo quy tắc somethingAmount, hoặc amountOfSomething. Bất cứ lúc nào tôi nhìn vào tên biến kiểu này, tôi có thể chắc chắn chúng là số thập phân.

Nếu đơn vị của biến không rõ ràng, tôi sẽ cần thêm mô tả vào cuối tên biến. Ví dụ như widthInCentimeters, rainFallAmountInInchesPerHour, angleInDegrees, failurePercent, failureRatio.

#### 

#### **Đặt tên cho kiểu dữ liệu đúng/sai (Boolean)**

Kiểu dữ liệu boolean chỉ chấp nhận một trong hai giá trị: đúng (true) hoặc sai (false). Tôi thường áp dụng quy tắc isSomething, hasSomething, doesSomething, didSomething, shouldDoSomething, willDoSomething...

Một vài ví dụ cụ thể: isDisabled, hasErrors, allowsWhitespace, didUpdate, shouldUpdate, willUpdate.

Một biến boolean đúng đắn sẽ là câu trả lời cho câu hỏi dạng đúng sai (Yes/No question).

#### 

#### **Đặt tên cho kiểu dữ liệu chuỗi (String)**

Kiểu dữ liệu chuỗi là cực kỳ phổ biến, kể cả đối với các thực thể thực tế, ví dụ như tên - **name**, tiêu đề - **title**...

Đôi lúc tôi cần lưu trữ một kiểu dữ liệu số như là một chuỗi, tôi sẽ tuân theo quy ước sau: someValueString hoặc là someValueAsString. Đoạn code sẽ trở nên dễ đọc và dễ hiểu hơn nhiều.

Ví dụ:

//Javascript

const year = parseInt(yearAsString);

#### 

#### **Mảng, danh sách, bộ dữ liệu (Array, Collection, Set)**

Khi đặt tên cho những biến kiểu này, tôi luôn dùng danh từ số nhiều, ví dụ như customers, errors, tasks,... Trong đa số trường hợp, điều này đủ, bởi bạn không cần biết rõ cách hiện thực khi chạy qua vòng lặp.

Trong một vài trường hợp khác, tôi có thể chỉ định cách hiện thực trong biến, ví dụ: queueOfTasks, stackOfCards, orderedSetOfTimestamps..., nhưng tôi chưa bao giờ đặt tên biến kiểu customerList, handledOperationSet... Nó hết sức dư thừa và dài dòng khi chỉ cần đặt customers, hoặc handledOperations... là người ta đã hiểu rõ.

#### 

#### **Biến mapping (hay key/value)**

Biến mapping thường được sử dụng cho việc get giá trị (value) từ một từ khóa (key) cụ thể. Trong trường hợp này, tôi tuân theo nguyên tắc keyToValueMap. Chằng hạn như chúng ta có một mapping value chứa tổng các đơn hàng của một khách hàng, chúng ta có thể đặt biến customerToOrderCountMap.

#### 

#### **Đặt tên theo cặp (Pair) và bộ (Tuple)**

Trường hợp này khá đơn giản.

Tôi đặt tên biến cặp theo chuẩn variable1AndVariable2, biến bộ theo chuẩn variable1Variable2Variable3AndVariableN.

Ví dụ: heightAndWidth, heightWidthAndDepth...

#### 

#### **Đặt tên cho các đối tượng (Object)**

Các objects được xem như là một hiện thực của lớp (instance of class). Các class là một danh từ với ký tự đầu tiên được viết hoa, ví dụ như Person, Account, User, Order, Product... Tên biến cần phải liên kết với tên class, ví dụ Person $person, Account $account, Order $order... Bạn cũng có thể mô tả tính chất của biến bằng một tính từ đằng trước, ví dụ như Account $deletedAccounts, Order $completedOrders... Điểm mấu chốt ở đây là chúng ta cần giữ tên class cho phần cuối cùng của tên đối tượng.

#### 

#### **Đặt tên cho các tham số của function**

Phần này không khác với cách đặt tên ở những phần trên, nhưng bạn có thể thêm tiền tố "\_" vào đằng trước của biến trong một số trường hợp.

//PHP

function getProductName (Product $product)

{

...

}

//SCSS

@function calculate-rem($\_target, $\_context: 16) {

@return ($\_target / $\_context) \* 1rem;

}

#### **Đặt tên cho các hằng số (Constant)**

Các hằng số là một trường hợp đặc biệt. Tôi luôn luôn đặt tên hằng số theo chuẩn **snake\_case** với toàn bộ các ký tự viết hoa.

Ví dụ: TOOLTIP\_SHOW\_DELAY\_IN\_MILLISECS.

#### 

#### **Các quy tắc chung**

- Tránh viết tên biến tắt một cách tối nghĩa.

Cố gắng tránh viết tắt hết sức có thể. Tuy nhiên có rất nhiều tên viết tắt có thể sử dụng, ví dụ như Str thay cho String, Num thay cho Number, Prop thay cho Property... Các tên biến kiểu này khá phổ biến giữa các lập trình viên nên việc viết chúng hầu như không gây khó khăn cho người khác khi đọc. Nhưng bạn nên tránh việc viết tắt cho các trường hợp không phổ biến. Tôi hầu như chỉ viết tắt để tránh việc kéo dài lê thê tên biến của mình.

- Tránh những tên biến quá ngắn hoặc vô nghĩa.

Tên quá ngắn hầu như không thực sự chỉ rõ ý nghĩa của chúng. Tôi không sử dụng biến $i hay $k, thay vào đó tôi dùng $postIndex hoặc $postKey. Tôi hầu như không đặt các tên biến kiểu $num, $tmp, hay $val... Tôi thích mô tả chúng rõ ràng hơn là đánh đố người khác như vậy.