

+4

B

Posts

Questions

Discussions





i → Sign In/Sign up





TRUNG TÂM HUẤN LUYỆN LẬP TRÌNH KHAI GIẢNG KHÓA HỌC LẬP TRÌNH PHP - LẬP TRÌNH RUBY

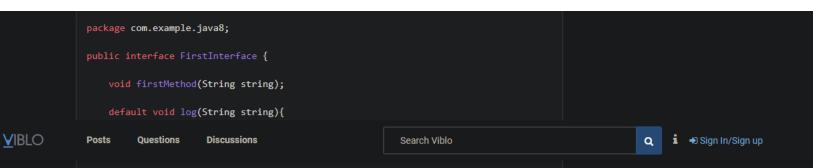




Thiết kế interface luôn là một công việc rất khó khăn, bởi vì khi chúng ta thay đổi các phương thức bên trong interface nó đòi hỏi phải thay đổi tất cả các class được implements từ nó. Một khi số lượng các class được implements từ interface phát triển nhiều lên thì đến mức độ nào đó interface có thể không mở rộng được nữa. Đây là lý do tại sao khi thiết kế một ứng dụng, hầu hết các framework cung cấp một base class, sau đó chúng ta sẽ extends và override lên các phương thức phù hợp với ứng dụng đang thực hiện. Bây giờ chúng ta sẽ tìm hiểu về phương thức default và static của interface, xem nó hoạt động thế nào.

Phương thức Default

Để tạo một phương thức default trong interface, chúng ta sẽ sử dụng từ khóa "default". Ví du:





Phương thức log(String str) chính là phương thức default của FirstInterface. Khi một class được implements từ FirstInterface nó không bắt buộc phải implement phương thức default. Tính năng này sẽ giúp chúng ta mở rộng các phương thức bổ sung phát sinh sau này mà không ảnh hưởng đến các class liên quan, chúng ta chỉ cần viết thêm các phương thức default trong interface.



Bây giờ chúng ta sẽ xem tiếp một ví dụ khác:

```
package com.example.java8;
    void secondMethod();
    default void log(String str){
        System.out.println("This method is default implementation" + str);
```

Như chúng ta biết rằng, Java không cho phép đa thừa kế đối với class bởi vì trình biên

interface FirstInterface và SecondInterface, trình biên dịch sẽ ko biết chọn phương thức nào để thực hiện. Đa kế thừa là một việc rất bình thường trong Java, chúng ta thường bắt gặp vấn đề này trong các class Java core cũng như hầu hết các ứng dụng enterprise

và framework. Để chắc chắn rằng, vấn đề này sẽ không xảy ra với interface, class phải



Posts

Questions

Discussions

Search Viblo

i 🖈 Sign In/Sign up



+4



package com.example.java8;

implement các phương thức common default. Vì thế, nếu class được implements từ cả 2 interface trên, thì nó phải implement phương thức log() để trình biên dịch không ném ra lỗi.

```
public class MyClass implements FirstInterface, SecondInterface {
     @Override
     public void firstMethod() {
     @Override
     public void secondMethod() {
     @Override
     public void log(String string){
         System.out.println("MyClass logging::" + string);
                                                                                                           i → Sign In/Sign up
 Posts
          Questions
                                                          Search Viblo
=> Những đặc điểm quan trọng về phương thức default trong interface:
 1. Phương thức default giúp chúng ta mở rộng interface mà không phải lo ngại phá
    vỡ các class được implements từ nó.
 2. Phương thức default giúp chúng ta tránh dùng các class tiện ích, ví dụ như tất cả
   phương thức của class Collections có thể được cung cấp ngay bên trong interface
    của nó.
 3. Phương thức default giúp chúng ta tháo gỡ các class cơ sở (base class), chúng ta có
   thể tạo phương thức default và trong class được implement có thể chọn phương
    thức để override.
 4. Một trong những lý do xuất hiện của phương thức default là để nâng cấp
   Collection API trong Java 8 hỗ trợ cho Lambda Expression.
 5. Nếu bất kỳ class nào kế thừa những phương thức default giống nhau, thì nó sẽ
   không còn hiệu lực. Một điều tương tự, một phương thức default sẽ không thể
   override một phương thức từ java.lang.Object. Lý do rất đơn giản là bởi vì Object là
    base class của tất cả các class trong Java. Vì vậy nếu chúng ta có các phương thức
 Posts
          Questions
                       Discussions
                                                          Search Viblo
                                                                                                            ◆3 Sign In/Sign up
   lý do tại sao chúng ta sẽ không có bất cứ phương thức default nào override các
   phương thức của class Object.
```

6. Phương thức default cũng có thể được gọi là phương thức Defender (Defender

VIBLO

+4

VIBLO

+4

Methods) hay là phương thức Virtual mở rộng (Virtual extension methods)



Phương thức Static



Phương thức static cũng giống như phương thức default ngoại trừ việc nó không thể được override chúng trong class được implements.

Hãy xem ví dụ dưới đây:

i → Sign In/Sign up

⊻IBLO



Bây giờ sẽ xem class được implements có phương thức isNull()









Phương thức *isNull(String string)* là một phương thức đơn giản, nó không override phương thức của interface. Ví dụ nếu chúng ta thêm annotation @Override cho phương thức isNull(), trình biên dịch sẽ báo lỗi. Bây giờ chúng ta sẽ chay ứng dụng và xem kết

