**Functional Interface trong Java 8**

*Đăng vào 30/04/2018 Được đăng bởi*[**GP Coder**](https://gpcoder.com/author/gpadmin/)***18761****Lượt xem*

Trong bài trước, tôi đã giới thiệu với các bạn Interface trong Java 8 là Default method và Static method. Trong bài này, chúng ta tiếp tục tìm hiểu một khái niệm mới về Interface trong Java 8 là Functional Interface.

Trước khi tìm hiểu Functional Interface, chúng ta hãy cùng so sánh thế nào Lập trình hướng đối tượng (Object Oriented Programming – OOP) và Lập trình hướng chức năng (Functional Programming – FP) trong Java.

**Nội dung**[[Ẩn](https://gpcoder.com/3869-functional-interface-trong-java-8/)]

* [1 So sánh Object-Oriented Programming vs Functional Programming](https://gpcoder.com/3869-functional-interface-trong-java-8/#So_sanh_Object-Oriented_Programming_vs_Functional_Programming)
* [2 Functional Interface là gì?](https://gpcoder.com/3869-functional-interface-trong-java-8/#Functional_Interface_la_gi)
* [3 Quy tắc khai báo Functional Interface](https://gpcoder.com/3869-functional-interface-trong-java-8/#Quy_tac_khai_baoFunctional_Interface)
  + [3.1 Một Functional Interface hợp lệ chỉ có duy nhất một method trừu tượng.](https://gpcoder.com/3869-functional-interface-trong-java-8/#Mot_Functional_Interfacehop_le_chi_co_duy_nhat_mot_method_truu_tuong)
  + [3.2 Một Functional Interface có thể có các phương thức của lớp java.lang.Object.](https://gpcoder.com/3869-functional-interface-trong-java-8/#Mot_Functional_Interface_co_the_co_cac_phuong_thuc_cua_lop_javalangObject)
  + [3.3 Phương thức default and static không phá vỡ quy tắc của functional interface.](https://gpcoder.com/3869-functional-interface-trong-java-8/#Phuong_thuc_default_and_static_khong_pha_vo_quy_tac_cua_functional_interface)
  + [3.4 Một Functional Interface có thể mở rộng một Interface khác chỉ khi nó không có bất kỳ phương thức trừu tượng nào.](https://gpcoder.com/3869-functional-interface-trong-java-8/#Mot_Functional_Interface_co_the_mo_rong_mot_Interface_khac_chi_khi_no_khong_co_bat_ky_phuong_thuc_truu_tuong_nao)
* [4 Các Functional Interface đã được định nghĩa trong Java 8 – Java Predefined-Functional Interfaces](https://gpcoder.com/3869-functional-interface-trong-java-8/#Cac_Functional_Interface_da_duoc_dinh_nghia_trong_Java_8_8211Java_Predefined-Functional_Interfaces)

**So sánh Object-Oriented Programming vs Functional Programming**

[**Object-Oriented Programming (OOP)**](https://en.wikipedia.org/wiki/Object-oriented_programming) là một mô hình lập trình dựa trên khái niệm **đối tượng (object)**, là kết hợp giữa **data** thể hiện dưới dạng các trường (field) hoặc thuộc tính (attribute) và **code** thể hiện dưới dạng thủ tục (procedure) hoặc phương thức (method).

[**Functional programming (FP)**](https://en.wikipedia.org/wiki/Functional_programming) là một mô hình lập trình, một kiểu xây dựng cấu trúc và các phần tử của các chương trình, xử lý tính toán dựa trên các function và nó tránh thay đổi trạng thái, dữ liệu có thể thay đổi được (mutable data).

Trong một chương trình, có hai thành phần chính: dữ liệu và hành vi (behavior hoặc code). **OOP** nói rằng tập hợp dữ liệu và hành vi liên quan của nó thành một “đối tượng” giúp dễ hiểu hơn về cách thức hoạt động của một chương trình. **FP** nói rằng dữ liệu và hành vi là khác biệt rõ rệt và nên được giữ riêng biệt để rõ ràng.

Chi tiết các bạn có thể tham khảo thêm tại link sau:

* <https://www.ntu.edu.sg/home/ehchua/programming/java/JDK8_NewFeatures.html#zz-1.>

**Functional Interface là gì?**

Java 8 gọi các Interface có duy nhất một method trừu tượng là các **Functional Interface**, nó cũng có thể được gọi là **Single Abstract Method**(SAM) một cụm từ đôi khi chúng ta bắt gặp.

Java 8 cũng giới thiệu một annotation mới là **@FunctionalInterface**. Nó có thể sử dụng cho mục đích bắt lỗi ở thời điểm biên dịch nếu vô tình thêm một method trừu tượng khác nữa vào interface có đánh dấu bởi annotation này mà vi phạm quy tắc của Functional Interface.

Lợi ích chính của functional interface là chúng ta có thể sử dụng **Lambda Expression** để tạo ra thể hiện (instance) cho interface đó.

**Quy tắc khai báo Functional Interface**

**Một Functional Interface hợp lệ chỉ có duy nhất một method trừu tượng.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8 | **package** com.gpcoder.functional\_interface;    @FunctionalInterface  **public** **interface** DemoFunctionalInterface {    **void** doSomething();    } |

**Functional Interface bên dưới không hợp lệ vì không có bất kỳ phương thức trừu tượng nào.**

[Graphical user interface, text, application, email

Description automatically generated](https://gpcoder.com/wp-content/uploads/2018/04/Invalid-Functional-Interface-2.png)

**Functional Interface bên dưới cũng không hợp lệ vì có nhiều hơn một method trừu tượng.**

[Graphical user interface, text, application

Description automatically generated](https://gpcoder.com/wp-content/uploads/2018/04/Invalid-Functional-Interface-1.png)

**Một Functional Interface có thể có các phương thức của lớp java.lang.Object.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13 | **package** com.gpcoder.functional\_interface;    @FunctionalInterface  **public** **interface** DemoFunctionalInterface2 {    **void** doSomething();    **int** hashCode();        String toString();    **boolean** equals(Object obj);  } |

Functional Interface bên dưới cũng không hợp lệ, bởi vì nó chỉ khai báo các phương thức của lớp Java.lang.Object. Không có bất kỳ phương thức trừu tượng nào của riêng nó.

[Graphical user interface, text

Description automatically generated](https://gpcoder.com/wp-content/uploads/2018/04/Invalid-Functional-Interface-3.png)

Functional Interface bên dưới cũng không hợp lệ, bởi vì phương thức clone() của lớp java.lang.Object có phạm truy cập là **protected**, trong khi tất cả các phương thức trừu tượng của interface đều là **public abstract** nên phưng thức clone() này không phải là override lại phương thức clone() của lớp Object. Do đó interface này được xem như khai báo 2 phương thức doSomething() và clone(), nên vi phạm nguyên tắc của Functional Interface.

[Graphical user interface, text, application

Description automatically generated](https://gpcoder.com/wp-content/uploads/2018/04/Invalid-Functional-Interface-5.png)

**Phương thức default and static không phá vỡ quy tắc của functional interface.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20 | **package** com.gpcoder.functional\_interface;    @FunctionalInterface  **public** **interface** DemoFunctionalInterface3 {    **void** doSomething();    **default** **void** defaultMethod1() {        }    **default** **void** defaultMethod2() {        }    **static** **void** staticMethod() {        }    } |

**Một Functional Interface có thể mở rộng một Interface khác chỉ khi nó không có bất kỳ phương thức trừu tượng nào.**

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14 | **package** com.gpcoder.functional\_interface;    **interface** BaseInterface1 {  **void** base();  }    **interface** BaseInterface2 {  **void** base();  }    @FunctionalInterface  **public** **interface** DemoFunctionalInterface4 **extends** BaseInterface1, BaseInterface2 {    } |

Functional Interface bên dưới không hợp lệ vì có Interface con có một phương thức trừu tượng doSomething() của riêng nó và kế thừa 1 phương thức base() từ Interface Base.

[Graphical user interface, text, application, email

Description automatically generated](https://gpcoder.com/wp-content/uploads/2018/04/Invalid-Functional-Interface-4.png)

**Các Functional Interface đã được định nghĩa trong Java 8 – Java Predefined-Functional Interfaces**

Java cung cấp các Functional Interface được định nghĩa trước để xử lý lập trình functional bằng cách sử dụng các lambda expression và method references. Bạn cũng có thể định nghĩa Functional Interface tùy chỉnh của riêng bạn.

Danh sách các Functional Interface được đặt trong gói **java.util.function**. Các bạn có thể tham khảo theo link sau: <https://docs.oracle.com/javase/8/docs/api/java/util/function/package-summary.html>

Như vậy là chúng ta đã tìm hiểu xong một khái niệm mới trong Java 8 là **Functional Interface**. Trong các bài viết tiếp theo chúng ta sẽ cùng tìm hiểu một tính năng mới khác rất là hay trong Java 8 là**Lambda Expression**. Các bạn sẽ thấy được lợi ích và cách sử dụng của Lamda Expression và Functional Interface trong Java 8 như: **Function**, **Predicate**, **Consumer**, **Consumer**, …

Hy vọng bài viết này giúp ích cho các bạn. Cảm ơn các bạn đã quan tâm và theo dõi bài viết. Hẹn gặp lại ở bài viết tiếp theo.

**4.4**

25

**Nếu bạn thấy hay thì hãy chia sẻ bài viết cho mọi người nhé!**

SHARES

*Chuyên mục:*[**Java 8**](https://gpcoder.com/category/java-core/java-8/)*Được gắn thẻ:*[**Java 8**](https://gpcoder.com/tag/java-8/)

[Interface trong Java 8 – Default method và Static method](https://gpcoder.com/3854-interface-trong-java-8-default-method-va-static-method/)

[Phương thức tham chiếu trong Java 8 – Method References](https://gpcoder.com/3882-phuong-thuc-tham-chieu-trong-java-8-method-references/)

**Có thể bạn muốn xem:**

* [Biểu thức Lambda trong Java 8 – Lambda Expressions](https://gpcoder.com/3898-bieu-thuc-lambda-trong-java-8-lambda-expressions/) **(07/05/2018)**
* [String Joiner trong Java 8](https://gpcoder.com/4139-string-joiner-trong-java-8/) **(23/07/2018)**
* [Phương thức forEach() trong java 8](https://gpcoder.com/3919-phuong-thuc-foreach-trong-java-8/) **(10/05/2018)**
* [Lớp Collectors trong Java 8](https://gpcoder.com/3973-lop-collectors-trong-java-8/) **(18/06/2018)**
* [Optional trong Java 8](https://gpcoder.com/3945-optional-trong-java-8/) **(17/05/2018)**

**Bình luận**

4 bình luận