Generic

1.

2.

3. Chúng ta cũng có thể viết các hàm chung có thể được gọi với các loại đối số khác nhau dựa trên loại đối số được truyền cho phương thức chung. Trình biên dịch xử lý từng phương thức.

Ưu điểm của Generics:

Các chương trình sử dụng Generics có nhiều lợi ích so với non-generic.

1. Code Reuse: Chúng ta có thể viết một phương thức / lớp / giao diện một lần và sử dụng nó cho bất kỳ kiểu nào chúng ta muốn.

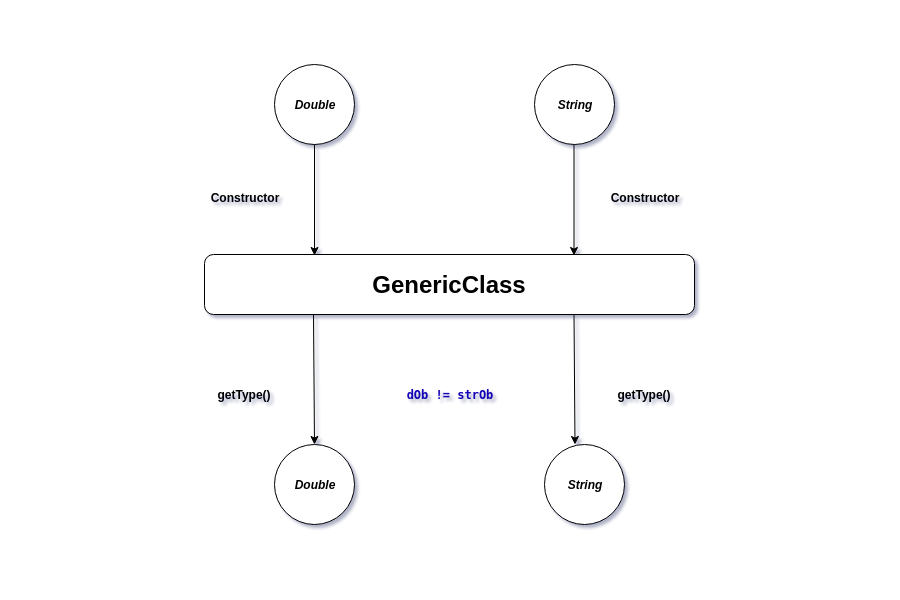
2. Loại An toàn: Generics tạo ra lỗi xuất hiện trong thời gian biên dịch tại thời điểm chạy (). Giả sử muốn tạo một ArrayList lưu trữ tên của các sinh viên và nếu do nhầm lẫn người lập trình thêm một đối tượng số nguyên thay vì một chuỗi ký tự, trình biên dịch sẽ cho phép điều đó. Tuy nhiên, khi truy xuất dữ liệu này từ ArrayList, nó gây ra sự cố trong thời gian chạy.

**Generic**

Generics trong Java là một trong những tính năng quan trọng nhất được giới thiệu kể từ Java 5. Thuật ngữ Generics trong Java đại diện cho một tập hợp các tính năng trong một ngôn ngữ, liên quan đến việc xác định và sử dụng các phương thức và kiểu chung. Generics là một tính năng của Java giúp cho lập trình viên có thể chỉ định rõ kiểu dữ liệu mà họ muốn làm việc với một class, một interface hay một phương thức nào đó Trong Java, các phương thức và kiểu chung chung khác với các phương thức và kiểu thông thường. Chúng khác nhau vì các phương thức chung có tham số kiểu.

**Tại sao lại cần sử dụng Generics trong Java?**

Java Generics cho phép chúng ta viết một phương thức duy nhất có thể thực hiện các hoạt động trong nhiều loại đối tượng hỗ trợ phương thức đó. Sử dụng các lớp và phương thức Java Generic, lập trình viên có thể chỉ định một tập hợp các phương thức có liên quan với một khai báo phương thức chung / chung hoặc với một khai báo một lớp.



***Tại sao lại cần sử dụng Generics trong Java***

Ví dụ, khái niệm Java Generics cho phép chúng ta viết một phương thức chung để sắp xếp một mảng gồm các loại đối tượng khác nhau, như gọi phương thức chung với Mảng ký tự, Mảng số nguyên, Mảng chuỗi, Mảng đôi, ... để sắp xếp các phần tử mảng .

Hơn nữa, Java Generics cung cấp kiểu thời gian biên dịch an toàn (compile-time type safety) cho phép lập trình viên bắt được các kiểu không hợp lệ hoặc lỗi trong quá trình biên dịch.

**Lớp Generics trong Java**

Một lớp Generic trong Java có thể là một  lớp tham chiếu đến bất kỳ kiểu nào. Để tạo một  lớp chung của một kiểu cụ thể, chúng ta đặt tham số kiểu T. Dấu ngoặc nhọn <> được sử dụng để chỉ định các kiểu tham số trong tạo lớp chung của Java.

## ****Quy tắc xác định các phương thức Generics:****

* Nên có một phần tham số kiểu trong tất cả các khai báo phương thức chung, được phân cách bằng dấu ngoặc nhọn <> đứng trước kiểu trả về của phương thức.
* Nếu có nhiều hơn một tham số trong danh sách tham số thì mỗi tham số kiểu phải được phân tách bằng dấu phẩy.
* Chúng ta cũng có thể sử dụng các tham số kiểu để khai báo kiểu trả về và để chúng hoạt động như trình giữ chỗ cho các kiểu đối số được truyền cho phương thức chung, được gọi là đối số kiểu thực.
* Phần thân phương thức của một phương thức chung được khai báo tương tự như bất kỳ phương thức không chung nào khác.
* Tham số kiểu trong một phương thức chỉ có thể đại diện cho các kiểu tham chiếu, các kiểu không nguyên thủy như int, double và char.
* **Collection<?>:** mô tả một tập hợp chấp nhận tất cả các loại đối số (chứa mọi kiểu đối tượng).
* **List<? extends Number>:** mô tả một dsanh sách, nơi mà các phần tử là kiểu Number hoặc kiểu con của Number.
* **Comparator<? super String>:** Mô tả một bộ so sánh (Comparator) mà thông số phải là String hoặc cha của String.