Bài 61: Sử dụng throw và throws

- ✓ Sử dụng throw
- ✓ Sử dụng throws
- ✓ Sự khác nhau
- ✓ Ví dụ minh họa

Sử dụng throw

- Keyword throw dùng để văng một ngoại lệ một cách tường minh
- Hữu ích khi ta muốn chủ động tạo và ném ngoại lệ trong một ngữ cảnh nào đó
- Ngoại lệ bị ném ra ngoài có thể là ngoại lệ mới hoặc ngoại lệ đã được xử lý trong catch
- Throw được sử dụng bên trong thân của một phương thức

Ví dụ

➢Ví dụ sau văng một ngoại lệ NullPointerException:

```
private static void method1() {
    String str = null;
    if (str == null) {
        var msg = "str là null, vui lòng kiểm tra lại";
        throw new NullPointerException(msg);
    }
    System.out.println("Các lệnh phía sau nơi ném ngoại lệ");
}

private static void method1() {
    String str = null;
    try {
        System.out.println("Số kí tự trong chuỗi str là: "
                + str.length());
    } catch (NullPointerException e) {
        System.out.println("Chuỗi kí tự vô định");
        throw e; // chủ động ném ngoại lệ này ra
    }
    System.out.println("Các lệnh phía sau nơi ném ngoại lệ");
}
```

Sử dụng throws

- Sử dụng throws khi muốn trì hoãn việc xử lý ngoại lệ hoặc là muốn đẩy việc xử lý ngoại lệ cho một phương thức khác
- Keyword này đặt ở cuối khai báo tên phương thức, sau dấu)
- Bạn có thể khai báo 1 hoặc nhiều kiểu ngoại lệ khác nhau với throws
- Nếu khai báo các ngoại lệ checked exception với throws thì nơi gọi phương thức phải:
  - Hoặc là xử lý ngoại lệ được đẩy về
  - Hoặc là tiếp tục throws các checked exception

Ví dụ 1

➢Ví dụ trì hoãn ngoại lệ ParseException:

```
public static void main(String[] args) {
    try {
        method1();
        System.out.println("Xử lý ngoại lệ có thể xảy ra trong main");
    } catch (ParseException e) {
        System.out.println("Đã xảy ra ngoại lệ: " + e.getMessage());
        System.out.println("Vui lòng kiểm tra lại!");
    }
}

private static void method1() throws ParseException {
    method2(); // không xử lý ngoại lệ xảy ra -> tiếp tục throws
}

private static void method2() throws ParseException {
    SimpleDateFormat dateFormat =
            new SimpleDateFormat("dd/MM/yyyy");
    Date myBirthday = null;
    var dateString = "";
    // dòng code dưới đây có thể xảy ra ngoại lệ:
    dateString = "22/10/2020"; // đọc dữ liệu từ file
    myBirthday = dateFormat.parse(dateString);
    System.out.println("Sinh nhật của tôi là: "
              + dateFormat.format(myBirthday));
}
```

Mô tả

➢Ví dụ trì hoãn ngoại lệ ParseException:

```
public static void main(String[] args) {
    try {
        method1();
        System.out.println("Xử lý ngoại lệ có thể xảy ra trong main");
    } catch (ParseException e) {
        System.out.println("Đã xảy ra ngoại lệ: " + e.getMessage());
        System.out.println("Vui lòng kiểm tra lại!");
    }
}

private static void method1() throws ParseException {
    method2(); // không xử lý ngoại lệ xảy ra -> tiếp tục throws
}

private static void method2() throws ParseException {
    SimpleDateFormat dateFormat =
            new SimpleDateFormat( pattern: "dd/MM/yyyy");
    Date myBirthday = null;
    var dateString = "";
    // dòng code dưới đây có thể xảy ra ngoại lệ:
    dateString = "22/10/2020"; // đọc dữ liệu từ file
    myBirthday = dateFormat.parse(dateString);
    System.out.println("Sinh nhật của tôi là: " + dateFormat.format(myBirthday));
}
```

Ví dụ 2

```java
public static void main(String[] args) {
    try {
        method1();
        System.out.println("Xử lý ngoại lệ có thể xảy ra trong main");
    } catch (ParseException | NullPointerException e) {
        System.out.println("Đã xảy ra ngoại lệ: " + e.getMessage());
        System.out.println("Vui lòng kiểm tra lại!");
    }
    // các câu lệnh kế tiếp sau nơi bắt ngoại lệ:
    System.out.println("Thực hiện các câu lệnh kế tiếp...");
}

private static void method1() throws ParseException,
                                     NullPointerException {
    method2(); // không xử lý ngoại lệ xảy ra -> tiếp tục throws
}

private static void method2() throws ParseException,
                                     NullPointerException {
    SimpleDateFormat dateFormat = null;
    Date myBirthday = null;
    var dateString = "";
    // dòng code dưới đây có thể xảy ra ngoại lệ:
    dateString = "22/10/2020"; // đọc dữ liệu từ file
    myBirthday = dateFormat.parse(dateString);
    System.out.println("Sinh nhật của tôi là: "
            + dateFormat.format(myBirthday));
}
```

throw vs throws

- Sau đây là một số điểm khác nhau giữa throw và throws:

Throw	throws
Dùng để trực tiếp/chủ động ném một ngoại lệ ra khỏi khối/phương thức	Dùng để khai báo sự trì hoãn của một hoặc một số ngoại lệ nào đó
Không dùng để ném ngoại lệ thuộc loại checked exception	Có thể khai bảo cả checked exception
Throw sử dụng để ném một đối tượng	Throws dùng để khai báo các kiểu ngoại lệ
Bạn chỉ có thể ném một ngoại lệ với throw	Bạn có thể khai báo nhiều kiểu ngoại lệ với throws
Xuất hiện ở trong thân phương thức	Xuất hiện ở khai báo tên phương thức

Minh họa

- Thực hiện trong công cụ lập trình

Nội dung tiếp theo

Ngoại lệ người dùng tự định nghĩa