Chương 7. Một số kỹ thuật nâng cao

TRẦN VIỆT KHÁNH

Email: tranvietkhanh79@gmail.com

Cập nhật: 04 tháng 09 năm 2016

Nội dung

- Ngoại lệ (Exception)
- Mẫu (Template)

Ngoại lệ (Exception)

- Một Exception (ngoại lệ) là một vấn đề xuất hiện trong khi thực thi một chương trình. Một Exception trong C# là một phản hồi về một tình huống ngoại lệ mà xuất hiện trong khi một chương trình đang chạy, ví dụ như chia cho số 0.
- Exception cung cấp một cách để truyền điều khiển từ một phần của một chương trình tới phần khác. Exception Handling (Xử lý ngoại lệ) trong C# được xây dựng dựa trên 4 từ khóa là: **try**, **catch**, **finally**, và **throw**.
- **try**: Một khối try nhận diện một khối code mà ở đó các exception cụ thể được kích hoạt. Nó được theo sau bởi một hoặc nhiều khối catch.
- **catch**: Một chương trình bắt một Exception với một Exception Handler tại vị trí trong một chương trình nơi bạn muốn xử lý vấn đề đó. Từ khóa **catch** trong C# chỉ dẫn việc bắt một exception.
- **finally**: Một khối finally được sử dụng để thực thi một tập hợp lệnh đã cho, dù có hay không một exception đươc ném hoặc không được ném. Ví dụ, nếu bạn mở một file, nó phải được đóng, nếu không sẽ có một exception được tạo ra.
- **throw**: Một chương trình ném một exception khi có một vấn đề xuất hiện. Điều này được thực hiện bởi sử dụng từ khóa **throw** trong C#.

Ngoại lệ (Exception)

- **Cú pháp**
- Giả sử một khối tạo một Exeption, một phương thức bắt một exception bởi sử dụng kết hợp các từ khóa try và catch. Một khối try/catch được đặt xung quanh code mà có thể tạo một exception. Code bên trong một khối try/catch được xem như là code được bảo vệ, và cú pháp để sử dụng try/catch trong C# như sau:
  - try { // các lệnh có thể gây ra ngoại lệ (exception) }

    catch(tên_ngoại_lệ e1) { // phần code để xử lý lỗi }

    catch(tên_ngoại_lệ e2) { // phần code để xử lý lỗi }
    catch(tên_ngoại_lệ eN) { // phần code để xử lý lỗi }
    finally { // các lệnh được thực thi }

Mẫu (Template)

- Template trong C++ nói cho dễ hiểu là "kiểu dữ liệu" được quyết định tại compile time.
  - Compile time: Lúc chương trình được compiler biên dịch
  - Run time: Lúc chương trình đã được dịch ra executable và có thể chạy
- Template có hai loại chính:
  - Hàm template
  - Lớp template

Ví dụ:

```
template <typename T>
class class_template {
        T data[10];
};
template<typename T>
void function_template( const vector<T>& vec > {
}
```

Mẫu (Template)

- **Cú pháp khi dùng template**

  Có hai định nghĩa cơ bản mà chúng ta cần phải biết về template là:
  1. Tham số template:

```
template<typename T>
T max( T a, T b ) {
}
```

  2. Đối template:

```
max<int>( 3, 4 );
```

- Tham số và đối template cũng giống như tham số và đối trong hàm, số lượng không bị giới hạn, bạn có thể khai báo tùy ý số lượng đối và tham số,
- Ví dụ:

```
template<typename T1, typename T2, typename T3, typename T4>
void func( T1 t1, T2 t2, T3 t3, T4 t4 ) {


}
```

FAQs

XIN CẢM ƠN !