Xử lý ngoại lệ & Debug

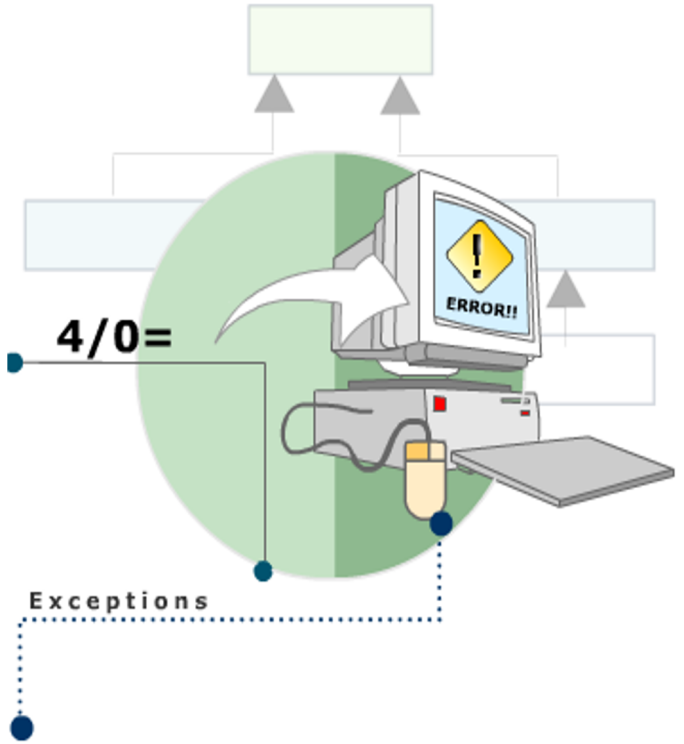
Module: Advanced Programming with JavaScript

Mục tiêu

- Trình bày được ý nghĩa của việc xử lí ngoại lệ
- Sử dụng được try-catch để bắt ngoại lệ
- Tung được ngoại lệ phù hợp
- Sử dụng được các lớp ngoại lệ thông dụng có sẵn
- Định nghĩa được ác lớp ngoại lệ tuỳ biến
- Trình bày được ý nghĩa của debug
- Triển khai được debug ứng dụng javascript

Ngoại lệ

- Ngoại lệ là các lỗi phát sinh trong quá trình thực thi
- Ví dụ:
  - Lỗi khi lái xe
  - Lỗi khi máy tính xử lý phép chia cho 0



Các kiểu lỗi thường xảy ra

- Lỗi cú pháp
- Lỗi khi chạy chương trình
- Lỗi về tính logic cấu trúc của chương trình

Lỗi cú pháp

- Lỗi cú pháp (Syntax Error) xảy ra tại thời gian biên dịch trong các ngôn ngữ chương trình truyền thống và tại thời gian phiên dịch trong Javascript.
- Ví dụ, dòng sau gây ra một lỗi cú pháp bởi vì nó thiếu dấu ngoặc đơn đóng:

```
function isNum(number) {
  if(number>1
     return true;
}

isNum(2);
```

Lỗi khi chạy chương trình

- Lỗi trong khi chạy chương trình (Runtime Error) xảy ra trong suốt thời gian thực thi
- Ví dụ, dòng sau tạo một Runtime Error bởi vì ở đây cú pháp là đúng, nhưng trong khi chạy, nó cố gắng gọi một phương thức mà không tồn tại.

```
function isNum(number) {
  if(number>1) {
    throw new Error("Value must be 1 or below");
  }
  return true;
}


checkNumber(2);
```

Lỗi logic

- Lỗi về tính logic của cấu trúc chương trình (Logic Error) là kiểu lỗi khó để có thể tìm dấu vết.
- Xảy ra khi bạn tạo một lỗi về tính logic mà điều khiển script của bạn và không nhận được kết quả như mong đợi.
- Khó nắm bắt được các lỗi này, bởi vì nó phụ thuộc vào yêu cầu và kiểu logic mà bạn đặt vào chương trình.

Xử lý ngoại lệ trong JavaScript

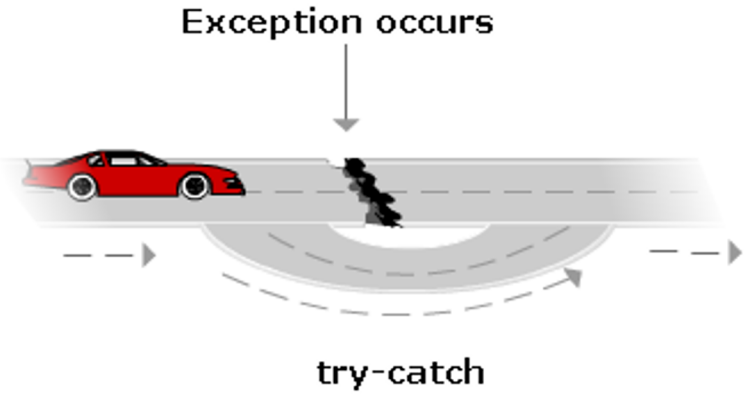
- Khả năng bắt và giải quyết ngoại lệ sử dụng catch.
- Khả năng tách logic xử lý ngoại lệ trong một hàm ra khỏi phần còn lại của hàm sử dụng **try**.
- Khả năng tạo và ném ngoại lệ sử dụng **throw**.

Sử dụng khối try-catch

- Khối try-catch dùng để
  - Tách phần giải quyết lỗi ra khỏi phần có thể sinh lỗi
  - Quy định các loại ngoại lệ được bắt tại mức thực thi hiện hành
- Cú pháp

```
try {
        Statement_1;
        Statement_2;
}
catch(ErrorObject){
        Statement_1;
}
```

  - Mã liên quan đến thuật toán nằm trong khối
  - Mã giải quyết lỗi đặt trong các khối catch



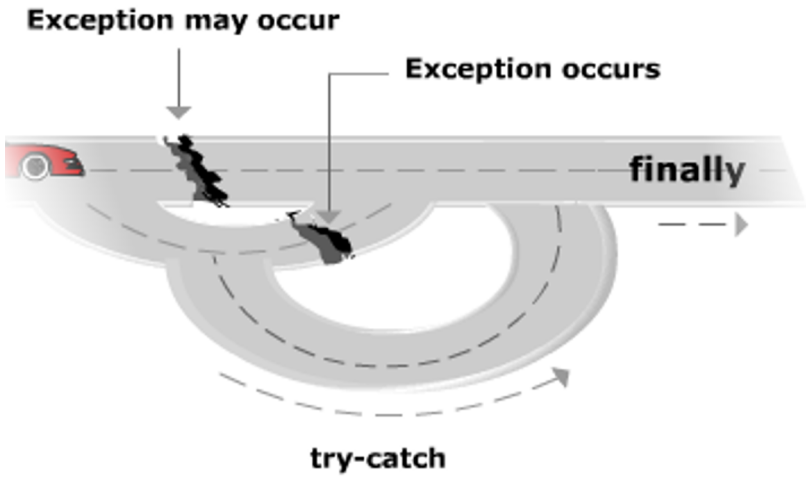
Demo

Sử dụng khối try-catch

Khối finally

- Khối **finally** mà sẽ luôn luôn thực thi vô điều kiện sau **try/catch**.

```
try {
        Statement_1;
        Statement_2;
}
catch(ErrorObject){
        Statement_1;
}
finally{
        //Clean up code
        Statement_1;
}
```

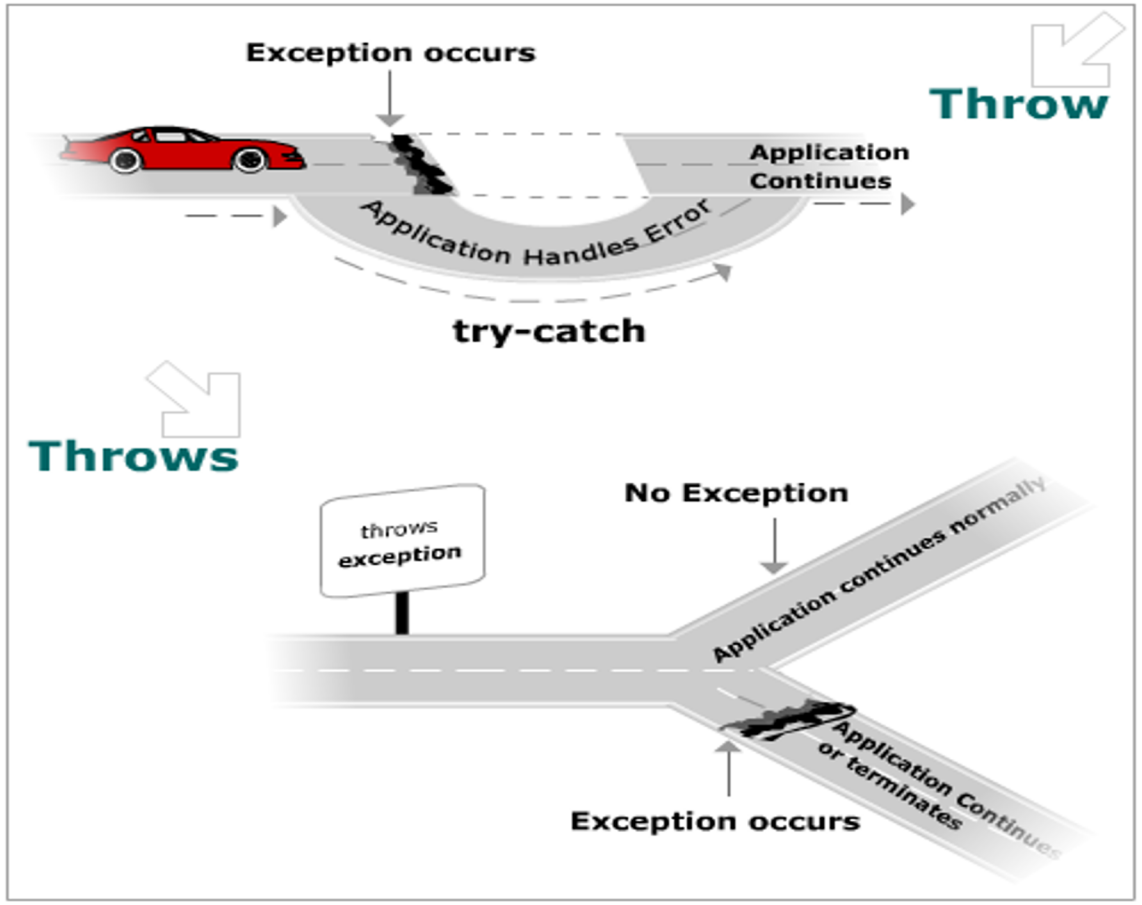


Demo

Sử dụng khối finally

Ném ngoại lệ

- Sử dụng lệnh **throw** để ném các lỗi Error có sẵn hoặc các lỗi do người dùng tự định nghĩa. Sau đó các lỗi này có thể được bắt và chúng ta có thể thực hiện một hành động hợp lý.



Ném ngoại lệ

- Để ném một ngoại lệ, ta dùng từ khóa throw kèm theo đối tượng mà ta định ném
- Ta có thể dùng mọi thứ làm ngoại lệ, kể cả giá trị thuộc kiểu có sẵn

Ngoại lệ do người dùng tự tạo

- Để tạo một ngoại lệ của người dùng, bạn phải tạo một lớp là mở rộng của lớp Error có sẵn.
- Lớp ngoại lệ do người dùng tự định nghĩa kế thừa từ lớp Error và chúng ta có thể thêm hàm xử lý.

Debug

Debug

- Debug là quá trình tìm kiếm lỗi hay nguyên nhân gây ra lỗi (bug) để có hướng sửa lỗi (fix bug).
- Mục đích của Debug không chỉ là để loại bỏ lỗi (bug/error) khỏi chương trình mà còn để giúp lập trình viên hiểu rõ hơn quá trình thực thi của chương trình.

Ví dụ debug sử dụng console.log

```
let a = 1;
let b = 2;
let total = a + b;
console.log (total);
document.getElementById("total").innerHTML = total;
```

Ví dụ debug sử dụng debugger & break point

```
let a = 1;
let b = 2;
let total = a + b;
document.getElementById("total").innerHTML = total;
```

Tóm tắt bài học

- Ngoại lệ là các lỗi phát sinh trong quá trình thực thi
- Xử lý ngoại lệ với khối try-catch-finally
- Sử dụng lệnh throw để ném các lỗi
- Debug là quá trình tìm kiếm lỗi hay nguyên nhân gây ra lỗi (bug) để có hướng sửa lỗi (fix bug).
- Có thể sử dụng console.log hoặc debugger để debug ứng dụng JavaScript

Hướng dẫn

- Hướng dẫn làm bài thực hành và bài tập
- Chuẩn bị bài tiếp: *Chuỗi vs Regex*