Top 10 TIPs giúp Coder Debug hiệu quả và đơn giản hơn

Bởi **Lý Quốc Siêu** - 24 Tháng Hai, 2020

Contents

Đối với mỗi một lập trình viên thì việc gặp lỗi, bug trong quá trình xây dựng, phát triển một phần mềm, chương trình là điều rất khó tránh khỏi. Thực chất Bug là một trong những chủ đề lớn nhất trong ngành lập trình. Vậy làm thế nào để các lập trình viên Debug hiệu quả và tiết kiệm được nhiều thời gian hơn? Sau đây

là **Top 10 TIPs Debug** giúp các lập trình viên giải quyết vấn đề hiệu quả hơn.

Top 10 TIPs Debug đơn giản và dễ thực hiện

1. Hiển thị thật nhiều biến

Hiển thị nhiều biến

Từng dòng code bạn nên nắm rõ giá trị hay tầm giá trị từng biến trong chương trình. Hãy hiển thị nó ra console hoặc text box để nắm rõ hơn.

Để biết giá trị nó thay đổi như thế nào hay ra một giá trị mà bạn không thể đoán trước được thì khi chạy chương trình bạn nhìn vào màn hình console nhé.

Đôi khi bạn nên in một chuỗi cố định trước khi bạn in giá trị biến làm như thế sẽ khiến cho bạn đỡ rối hơn. Các dòng lệnh chạy theo thứ tự lần lượt nên các bạn chú thích trước khi giá biến để biết nó là giá trị của biến nào.

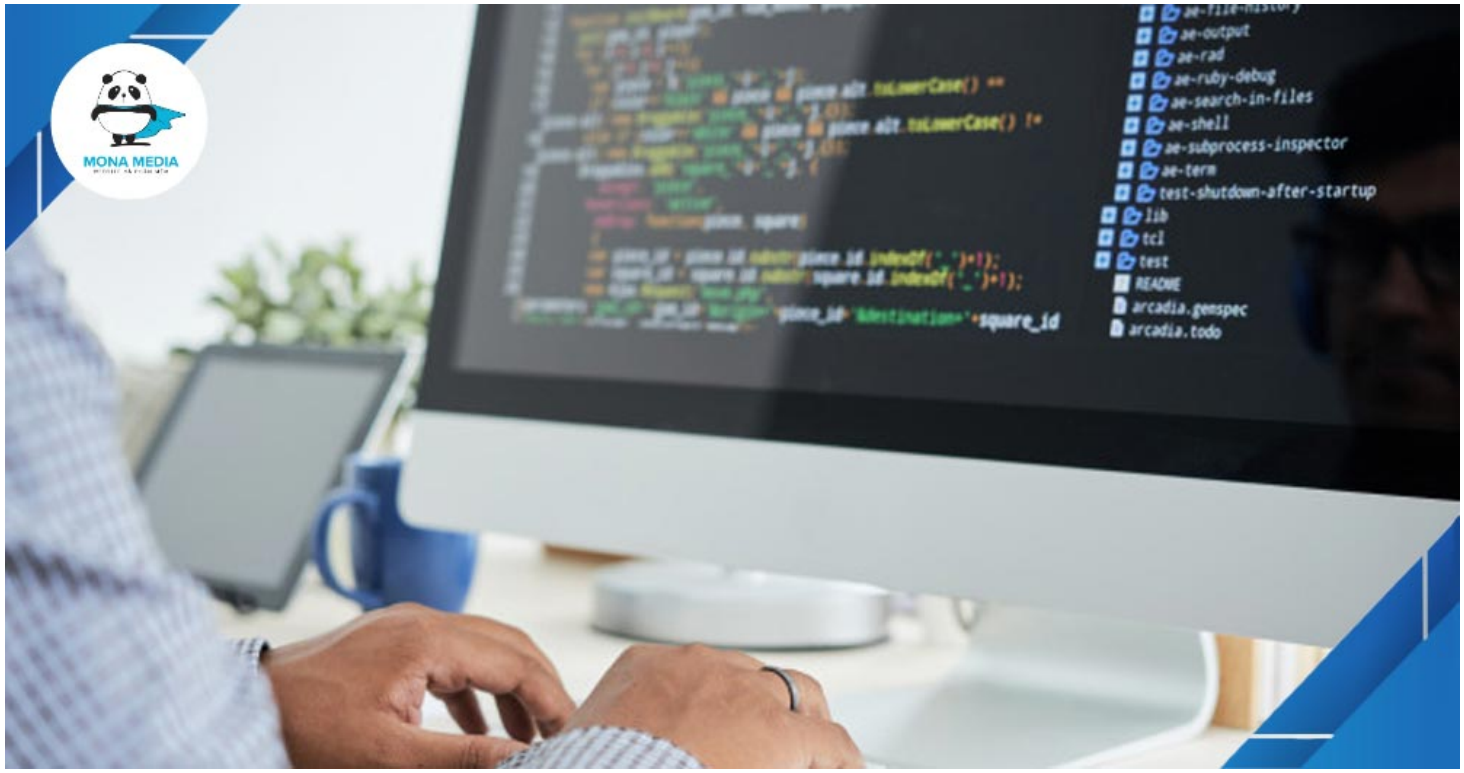
Có những lúc bạn cũng không thể biết được rằng một đoạn code có thực hiện hay không, hãy in ra "got here" trong đoạn code để biết là chương trình đó có thực thi đoạn code trên hay không. Ví dụ: một mệnh đề if hay một vòng lặp for.

2. Hãy bắt đầu với những mẫu code đã chạy được

Bạn vẫn còn những băn khoăn hãy tìm đến mẫu code của người khác đã chạy được và chạy thử. Bạn là một người mới làm tốt nhất bạn nên làm với những code có sẵn, nếu như bạn muốn thay đổi theo ý của mình làm bạn hãy chỉnh sửa để phù hợp.

Bạn không phải đối mặt với deadline, bạn nên google trước một vài đoạn code mẫu về hàm mà bạn dự định dùng nó. Chạy thử đoạn code mẫu trước khi điều chỉnh để kiểm tra xem nó có thực sự chạy không và chạy đúng mục đích không. Tiếp theo bạn hãy thay đổi từng cái nhỏ lần lượt kết hợp với chạy thử xem sự thay đổi này có gây ra lỗi và có ảnh hưởng gì không.

3. Chạy thử chương trình lần nữa mỗi khi bạn debug code



Chạy thử chương trình

Bạn không nên mất quá nhiều thời gian để viết và chạy chương trình lần đầu tiên code của bạn, cũng đừng sử dụng một file rỗng ngay lần đầu. Khi đó bạn sẽ bị vướng mắt vào rất nhiều lỗi nhỏ chi chít khiến bạn bị rối.

Để không gặp phải vấn đề như ở trên bạn nên chạy qua một đoạn mẫu và thay đổi một số chi tiết nhỏ ở trong đoạn code đó, sau đó bạn chạy lại chương trình xem có kết quả bị thay đổi như thế nào. Đừng ngại thực hiện lại nhiều lần vì khi làm nhiều lần khả năng cao là bạn tìm ra lỗi cách dễ dàng hơn đẻ sửa chúng, khi đó cũng có thêm nhiều kinh nghiệm để bạn làm việc sau này. Mỗi lần chạy lại chương trình như thế bạn nắm được liệu rằng chương trình đó có chạy đúng theo mong muốn của mình đưa ra hay không.

4. Đọc chi tiết ERRORS

Trong quá trình thực hiện chắc chắn bạn sẽ gặp " Chương trình có lỗi" nó sẽ khiến bạn bị mắc kẹt vào hàng loạt lỗi xếp chồng lên nhau nó không xa lạ đối với code. Có đến 2/3 các lỗi đó mô tả chính xác và cụ thể nên bạn có thể hiểu và debug. Khi các lỗi xảy ra sẽ kéo theo nhiều thêm các thông báo lỗi khác ở dưới nên khi bạn debug đúng và biên dịch lại, rất nhiều thông báo lỗi sẽ biến mất.

Cũng có những lỗi nhỏ như bạn viết thiếu một ký tự hay viết sai cú pháp hay bạn bỏ bước nào đó, thì thông báo lỗi có ích giúp bạn xác định lỗi sai ở đâu hoặc ít nhất, nó sẽ chỉ ra số thứ tự dòng bị lỗi trong chương trình.

Trong khi đang chạy chương trình thông báo lỗi sẽ chỉ ra dòng code trước khi bạn thoát và từ đó bạn có thể truy ngược lại code và bắt đầu quá trình debug.

5. Google mã lỗi

Google tìm thông tin về mã lỗi

Bạn gặp phải nỗi mà đã tốn thời gian nhưng chưa xác định được lỗi và tìm ra cách sửa lỗi, biện pháp lúc này là copy lỗi đó và paste trên google. Trên đây bạn sẽ tìm thấy nhiều người có thắc mắc với bạn và được giải thích, sửa lỗi cụ thể.

Khi bạn tìm kiếm trên google nó có tác dụng hay không còn phụ thuộc vào lỗi của bạn gặp phải. Tìm hiểu thật kỹ để chắc chắn rằng đoạn code trong câu hỏi gần giống với của bạn, quan trọng nhất là cùng một ngôn ngữ. Hãy đọc các comment và các câu trả lời để xem có khắc phục được lỗi của bạn không.

6. Thử và kiểm chứng

Tìm ra cách sửa lỗi nhưng bạn chưa chắc chắn được 100% cách sửa lỗi thì bạn hãy thử 2 đến 3 cách khác để kiểm chứng. Nếu như chương trình biên dịch thường xuyên sẽ giúp bạn kiểm soát tốt hơn. Khi sửa lỗi bạn sẽ sinh ra một số lỗi khác và khó có thể được biết liệu bạn có sửa đúng hay chưa. Nhưng đừng đi quá xa với nguyên bản cũng có thể không khôi phục được code ban đầu.

Bằng kinh nghiệm của mình trước tiên bạn hãy cố sửa lỗi và tìm tài liệu trước khi nhờ người giúp đỡ.

7. Chú thích trong code

Ngôn ngữ lập trình nào cũng có chức năng chú thích- để giúp các developer ghi chú ý nghĩa của câu lệnh hay thuật toán của chương trình, mà không gây ảnh hưởng gì đến chương trình.

Bạn sử dụng điều này để "ghi chú" các đoạn code mà bạn muốn giữ lại. Công việc này rất dễ làm bạn chỉ cần đánh dấu ghi chú ở đầu đoạn và cuối đoạn code mà bạn ghi chú.

Chương trình dài thì bạn hãy ghi chú các đoạn chương trình không liên quan đến chức năng đang thử nghiệm. Việc làm này làm cho việc biên dịch và chạy chương trình nhanh hơn, bạn có thể tìm được lỗi và cách khắc phục dễ dàng hơn. Điểm đáng lưu ý là bạn không được ghi chú các biến mà đoạn chương trình sau có sử dụng. Nó khiến cho chương trình sau của bạn chạy không đúng theo ý bạn khó tìm ra nguyên nhân. Những đánh dấu ghi chú bạn hãy xóa đi để các đoạn code hoạt động lại trong chương trình khi bạn đã kiểm tra xong thử nghiệm.

8. Nếu bạn không biết vấn đề ở đâu, hãy tìm kiếm nhị phân – binary search (chia để trị)

Có càng nhiều dòng code thì càng nhiều lỗi xảy ra, khi project bạn với rất nhiều dòng code, khi đó rất khó để phát hiện nguyên nhân gây ra lỗi. Những thông báo giúp bạn tìm ra thêm thông tin về lỗi, cũng có các trường hợp mà bạn không thể tìm ra.

Đối với trường hợp này bạn nên dùng binary seach để đỡ mất thời gian

Về lý thuyết, nó chia chương trình ra 2 phần phân nửa. Khi đó bạn tìm xem lỗi đang nằm ở đâu rồi tiếp tục chia làm hai phân nửa, làm như thế cho đến khi phát hiện ra lỗi.

Về áp dụng thực tế, bạn chạy phân nửa đầu tiên của đoạn code và "ghi chú" phân nửa đoạn code còn lại. Gía trị biến khi thực hiện xong nửa đầu thì in ra, xem có hợp lý không. Nếu như ok thì lỗi sai nằm ở phân nửa thứ 2. Gía trị không ổn thì ngược lại, lỗi nằm ở phân nửa đầu tiên. Thực hiện liên tục đến khi tìm ra được lỗi.

9. Hít một hơi và đi quanh, thả lỏng trí óc

Muốn giải quyết vấn đề cách nhanh chóng và hiệu quả thì bạn thả lỏng đầu óc, làm cho tinh thần thoải mái. Tạm gác công việc lập trình sang một bên làm việc khác, bạn có thể nghe đoạn nhạc mà bạn yêu thích. Trong đầu bạn lúc nào cũng suy nghĩ là bắt buộc mình phải khắc phục được lỗi điều này sẽ làm bạn càng căng thẳng hơn, rất khó có thể nghĩ ra giải pháp trong tâm trạng như thế. Bạn đừng có hàng loạt suy nghĩ như không thể bỏ đi trong khi chương trình đang đầy lỗi như thế này. Hay sẽ thế nào khi mình bao giờ mới sửa được lỗi. Chinh điều này tạo thêm cho bạn thêm áp lực thôi nó cũng không thể giúp bạn debug thành công được.

10. Hỏi giúp đỡ

Học hỏi từ người có kinh nghiệm hơn

Bạn đã tìm hiểu từ các trang mạng và cũng đã thực hiện mọi cách nhưng vẫn không hiệu quả bây giờ bạn hãy tìm đến sự trợ giúp từ người có kinh nghiệm. Để bạn đi vào vấn đề cần hỏi một cách nhanh chóng thì đảm bảo đầy đủ nội dung này:

1. Nói ra điều bạn đang muốn làm
2. Post đoạn code bị lỗi
3. Đưa ra các lỗi mà chương trình thông báo
4. Đưa ra những cách bạn đã debug nhưng chưa đem lại hiệu quả

Kết luận

Trên đây là **Top 10 TIPs Debug** mà một lập trình viên cần nắm được để việc giải quyết các vấn đề được nhanh hơn, đọc để tham khảo nhé. Chúc các bạn thành công.

Lý Quốc Siêu