KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
227 Nguyễn Văn Cừ, Phường 4, Quận 5, TP.HCM
Điện thoại: (08) 38.354.266 – Fax: (08) 38.350.096

THIẾT KẾ PHẦN MỀM

BÀI TẬP NHÓM LẦN 3

The Broken Window Theory & The Boy Scout Rule

Group: Fullstackoverflowrestling

TÊN	MSSV	EMAIL
Vòng Sao Hùng	22120118	22120118@student.hcmus.edu.vn
Lê Nguyễn Hồng Ngọc	22120232	22120232@student.hcmus.edu.vn
Phan Tấn Phát	22120264	22120264@student.hcmus.edu.vn

Mục lục

I. The Broken Window Theory

1. Định nghĩa

- Broken window theory (lý thuyết của sổ vỡ) là một giả thuyết trong tội phạm học và xã hội học cho rằng nếu một khu vực có dấu hiệu xuống cấp hoặc mất trật tự mà không được khắc phục, nó sẽ tạo ra môi trường khuyến khích các hành vi tội phạm nghiêm trọng hơn
- Cốt lõi của lý thuyết này là ý tưởng rằng "tội phạm là kết quả của môi trường hơn là chỉ do cá nhân". Nếu mọi người nhìn thấy những dấu hiệu của sự vô tổ chức, chẳng hạn như cửa sổ vỡ, vỉa hè bẩn, graffiti trên tường, hoặc những hành vi vi phạm nhỏ như xả rác, uống rượu nơi công cộng, thì họ sẽ có xu hướng tin rằng luật pháp và trật tự không được thực thi, từ đó khiến các hành vi phạm pháp nghiêm trọng hơn có cơ hội phát triển.
- Lý thuyết này được sử dụng để giải thích cách một cộng đồng có thể dần dần suy thoái nếu những hành vi mất trật tự không được kiểm soát, và ngược lại, nếu môi trường được duy trì gọn gàng, trật tự, thì nó có thể giúp ngăn chặn tội phạm ngay từ đầu.

2. Nguồn gốc

- **Nghiên cứu của Philip Zimbardo (1969)**
  - Trước khi thuật ngữ "Broken Windows" ra đời, một trong những thí nghiệm quan trọng nhất đặt nền móng cho lý thuyết này được thực hiện bởi Philip Zimbardo, một nhà tâm lý học xã hội nổi tiếng (người đã thực hiện thí nghiệm nhà tù Stanford).
  - Năm 1969, Zimbardo tiến hành một thí nghiệm với hai chiếc xe hơi giống hệt nhau nhưng được đặt tại hai khu vực rất khác nhau:
    - Chiếc xe thứ nhất đặt ở một khu vực có tỷ lệ tội phạm cao, thu nhập thấp (Bronx, New York).
    - Chiếc xe thứ hai đặt ở một khu vực thượng lưu, ít tội phạm hơn (Palo Alto, California).
  - Kết quả cho thấy:
    - Chiếc xe ở Bronx nhanh chóng bị phá hoại và cướp bóc chỉ sau vài giờ.
    - Chiếc xe ở Palo Alto không bị động đến trong nhiều ngày.
  - Tuy nhiên, Zimbardo đã cố tình đập vỡ một cửa sổ của chiếc xe ở Palo Alto, và điều bất ngờ xảy ra: ngay sau đó, những người qua đường bắt đầu phá hoại nó. Tình trạng xuống cấp nhanh chóng lan rộng, và chỉ trong một thời gian ngắn, chiếc xe bị hư hỏng hoàn toàn, giống như ở Bronx.
  - Thí nghiệm này chỉ ra rằng chính dấu hiệu của sự hư hỏng (như cửa sổ vỡ) tạo ra cảm giác rằng không ai quan tâm, từ đó kích thích hành vi phá hoại lan rộng hơn.
- **Bài báo "Broken Windows" của James Q. Wilson và George L. Kelling (1982)**
  - Dựa trên nghiên cứu của Zimbardo, hai nhà nghiên cứu tội phạm học James Q. Wilson và George L. Kelling đã chính thức đưa ra lý thuyết

"Broken Windows" trong bài báo cùng tên được xuất bản trên tạp chí *The Atlantic* năm 1982.

- Họ lập luận rằng nếu một cửa sổ bị vỡ mà không được sửa chữa, những người khác sẽ cho rằng không ai quan tâm đến việc duy trì trật tự, và điều này có thể khuyến khích thêm những hành vi vi phạm khác. Điều tương tự áp dụng cho những hành vi vi phạm nhỏ như xả rác, vẽ bậy, hoặc say xỉn nơi công cộng – nếu không bị xử lý, chúng có thể dẫn đến tình trạng vô luật pháp nghiêm trọng hơn.

3. Cơ chế hoạt động

- Lý thuyết Broken Windows dựa trên ba nguyên tắc chính:
  - Môi trường tác động đến hành vi con người
    - Khi một khu vực có dấu hiệu bỏ hoang hoặc không được chăm sóc, người dân sẽ có xu hướng tin rằng các quy tắc và luật lệ không được thực thi.
    - Ngược lại, nếu khu vực sạch sẽ, trật tự, mọi người sẽ có xu hướng tuân thủ quy tắc hơn.
  - Tội phạm và mất trật tự có tính lan truyền
    - Một hành vi phạm pháp nhỏ có thể dẫn đến nhiều hành vi lớn hơn nếu không được kiểm soát.
    - Nếu một cộng đồng không có biện pháp xử lý ngay từ đầu, tội phạm sẽ có xu hướng gia tăng theo thời gian.
  - Cảnh sát và cộng đồng có thể ngăn chặn tội phạm bằng cách duy trì trật tự
    - Nếu cảnh sát và chính quyền tập trung vào việc duy trì trật tự, như sửa chữa các cửa sổ vỡ, dọn dẹp đường phố, xóa bỏ graffiti, thì họ có thể ngăn chặn tội phạm nghiêm trọng ngay từ đầu.

4. Ứng dụng trong thực tế

- Quản lý đô thị và cộng đồng
  - Các thành phố áp dụng nguyên tắc này bằng cách duy trì không gian công cộng sạch đẹp, như dọn dẹp graffiti, sửa chữa đường phố, và xử lý các vi phạm nhỏ để ngăn chặn tình trạng xuống cấp của môi trường sống.
  - Ví dụ: Singapore là một trong những quốc gia áp dụng nguyên tắc này rất hiệu quả. Chính phủ duy trì các quy định nghiêm ngặt về vệ sinh công cộng, quản lý giao thông, và xử phạt các hành vi vi phạm nhỏ như xả rác hay hút thuốc lá nơi công cộng.
- Doanh nghiệp và môi trường làm việc
  - Nếu một doanh nghiệp không kiểm soát kỷ luật từ những vi phạm nhỏ, nhân viên sẽ mất dần ý thức trách nhiệm, dẫn đến năng suất giảm và môi trường làm việc thiếu chuyên nghiệp.
  - Một số công ty lớn như Google, Microsoft duy trì văn hóa "clean desk policy" (bàn làm việc sạch sẽ) để tạo cảm giác gọn gàng, chuyên nghiệp và trách nhiệm trong công việc.
- Giáo dục và đào tạo

- Trong giáo dục, nếu học sinh được nhắc nhở và rèn luyện tính kỷ luật ngay từ những điều nhỏ nhặt, như đến lớp đúng giờ, giữ gìn sách vở, thì họ sẽ có ý thức trách nhiệm cao hơn.
- Nếu giáo viên và nhà trường bỏ qua những vi phạm nhỏ, học sinh có thể sẽ dần mất kiểm soát, dẫn đến những hành vi nghiêm trọng hơn như gian lận hoặc vi phạm nội quy nghiêm trọng.

5. Ứng dụng trong môi trường lập trình

- Trong ngành lập trình, lý thuyết Broken Windows cũng rất phù hợp để duy trì chất lượng code và quản lý dự án phần mềm. Nếu những "cửa sổ vỡ" trong code không được sửa chữa kịp thời, hệ thống có thể dần xuống cấp và trở nên khó bảo trì.
- Code Quality & Technical Debt (Chất lượng code và nợ kỹ thuật)
  - Khi một lập trình viên viết code kém chất lượng nhưng không sửa ngay, các lập trình viên khác có thể nghĩ rằng đó là tiêu chuẩn chấp nhận được và tiếp tục viết code tương tự.
  - Kết quả là codebase sẽ trở nên lộn xộn, khó bảo trì, và gia tăng nợ kỹ thuật (technical debt).
  - Giải pháp: Duy trì code sạch (clean code), review code thường xuyên, và tuân thủ coding convention.
- Bug Fixing & Code Maintenance (Sửa lỗi và bảo trì hệ thống)
  - Nếu một bug nhỏ không được sửa ngay, nó có thể dẫn đến nhiều bug lớn hơn hoặc tạo ra sự bất cẩn trong đội ngũ lập trình viên.
  - Các lập trình viên có thể nghĩ rằng "chỉ cần code chạy được là đủ", dẫn đến việc bỏ qua các tiêu chuẩn về chất lượng và hiệu suất.
  - Giải pháp: Thực hiện code review, viết test cases đầy đủ, và refactor code thường xuyên.
- Clean Code và Refactoring (Lập trình sạch và tái cấu trúc code)
  - Khi một lập trình viên viết code xấu mà không sửa ngay, những người khác có thể tiếp tục viết code theo kiểu tương tự.
  - Điều này làm cho dự án ngày càng khó bảo trì và dễ gặp lỗi.
  - Giải pháp: Duy trì nguyên tắc Clean Code, áp dụng SOLID principles, DRY (Don't Repeat Yourself), KISS (Keep It Simple, Stupid).
- Code Review và Team Culture (Văn hóa code và review code)
  - Nếu trong một nhóm lập trình, mọi người không quan tâm đến code review, chất lượng code sẽ dần suy giảm.
  - Các "cửa sổ vỡ" trong code như naming convention không nhất quán, lỗi logic nhỏ, code khó đọc sẽ tích tụ, làm cho dự án trở nên khó bảo trì.
  - Giải pháp: Duy trì code review nghiêm túc, hướng dẫn các lập trình viên junior, tổ chức technical sharing trong team.

6. Tranh cãi và hạn chế của lý thuyết Broken Windows

- Mặc dù lý thuyết Broken Windows có tác động mạnh mẽ, nhưng nó cũng vấp phải nhiều chỉ trích:
  - Không phải lúc nào cũng đúng: Một số nghiên cứu sau này cho rằng tỷ lệ tội phạm giảm ở New York có thể do các yếu tố khác, như kinh tế phát triển, dân số thay đổi, chứ không chỉ do chính sách "zero tolerance".

- Có thể dẫn đến phân biệt đối xử: Một số cảnh sát lợi dụng lý thuyết này để mạnh tay với những vi phạm nhỏ, đặc biệt là ở các khu vực nghèo, gây ra tranh cãi về công bằng xã hội.
- Chưa chắc có hiệu quả lâu dài: Việc xử lý vi phạm nhỏ có thể giúp duy trì trật tự, nhưng nếu không có biện pháp cải thiện kinh tế, giáo dục, thì tội phạm vẫn có thể quay lại.

II. The Boy Scount Rule

1. Định nghĩa

- The Boy Scout Rule là một nguyên tắc quan trọng trong lập trình phần mềm, khuyến khích lập trình viên luôn để lại code tốt hơn so với khi họ tìm thấy nó**.**
- Câu nói nổi tiếng mô tả nguyên tắc này là:
  - **"Always leave the campground cleaner than you found it." (*Luôn để lại khu cắm trại sạch hơn so với khi bạn tìm thấy nó.*)**
- Nguyên tắc này nhấn mạnh rằng bất kỳ khi nào bạn chỉnh sửa hoặc làm việc trên một đoạn code, hãy để nó tốt hơn trước đó. Điều này giúp giảm nợ kỹ thuật (technical debt), cải thiện chất lượng code, và giữ cho hệ thống dễ bảo trì hơn trong dài hạn.

2. Nguồn gốc

- The Boy Scout Rule lấy cảm hứng từ luật của tổ chức Hướng đạo sinh (Boy Scouts) – một tổ chức giáo dục thanh thiếu niên nổi tiếng trên thế giới.
- Tổ chức này dạy các thành viên rằng:
  - Khi đến một khu cắm trại, họ nên dọn dẹp sạch sẽ trước khi rời đi, thậm chí ngay cả khi đống rác đó không phải do họ tạo ra.
  - Điều này giúp bảo vệ môi trường và duy trì ý thức trách nhiệm của cộng đồng.
- Lập trình viên nổi tiếng Robert C. Martin (Uncle Bob) đã áp dụng triết lý này vào lập trình và đặt ra nguyên tắc The Boy Scout Rule trong cuốn sách "Clean Code" (2008).

3. Tại sao The Boy Scout Rule lại quan trọng trong lập trình?

- Giảm dần "nợ kỹ thuật" (Technical Debt)
  - Nếu mỗi lập trình viên cải thiện một chút khi họ làm việc trên code, hệ thống sẽ ngày càng tốt hơn.
  - Điều này giúp tránh tình trạng "code spaghetti" và giúp code dễ bảo trì hơn.
- Cải thiện chất lượng code từng bước một
  - Bạn không cần phải refactor toàn bộ dự án cùng một lúc.
  - Chỉ cần cải thiện một chút mỗi khi bạn chạm vào code, dần dần, codebase sẽ trở nên sạch hơn và tốt hơn.
- Tránh hiệu ứng "Broken Windows" trong code
  - Nếu lập trình viên thấy code xấu và không sửa, họ có thể nghĩ rằng viết code xấu là bình thường, dẫn đến codebase ngày càng rối.
  - Sửa những lỗi nhỏ giúp duy trì tiêu chuẩn code cao.
- Giúp nhóm phát triển dễ làm việc hơn
  - Khi code dễ đọc, dễ hiểu, lập trình viên mới có thể nắm bắt nhanh hơn mà không mất thời gian tìm hiểu code lộn xộn.

- Debugging, testing, và mở rộng chức năng cũng dễ dàng hơn.

4. Ứng dụng của The Boy Scout Rule trong lập trình

- Viết code sạch hơn (Clean Code)
  - Cải thiện tên biến, tên hàm để dễ hiểu hơn
  - Xóa code không cần thiết.
  - Tối ưu hóa logic để code ngắn gọn hơn.
- Refactor Code (Tái cấu trúc code)
  - Nếu bạn phát hiện một đoạn code dài dòng, khó hiểu, hãy tối ưu nó ngay cả khi bạn không phải là người viết ban đầu.
- Xóa Code Không Dùng (Dead Code)
  - Mỗi lần làm việc trên một file, hãy kiểm tra và xóa những dòng code không còn cần thiết để giảm sự lộn xộn.
- Cải thiện Logging & Debugging
  - Nếu bạn thấy hệ thống có log quá nhiều thông tin không cần thiết, hãy dọn dẹp để giúp debugging dễ hơn.

III. Tham khảo

- The Broken Window Theory
- The Boy Scout Rule