**ĐÁNH GIÁ VÀ LỰA CHỌN GIẢI PHÁP DEPLOY GAME TRÊN PORTAL**

**Zingplay Portal**

**1-Giới thiệu**

***1.1-Mục đích tài liệu***

Kỹ thuật deploy game trên portal là một kỹ thuật quan trọng và có tính chất nút thắt với dự án ZingPlay Portal. Vì vậy, cần thiết phải có một sự nghiên cứu kỹ lưỡng về kỹ thuật này. Tài liệu này tổng hợp lại các giải pháp đã được thử nghiệm, đánh giá và lựa chọn một giải pháp phù hợp để sử dụng cho dự án Zingplay Portal.

***1.2-Phạm vi tài liệu***

Tài liệu sử dụng cho dự án Zingplay Portal

***1.3-Lịch sử phiên bản***

|  |  |  |  |
| --- | --- | --- | --- |
| **Tên phiên bản** | **Ngày thực hiện** | **Người thực hiện** | **Ghi chú** |
| 1.0 | 06/04/2016 | Nguyễn Tuấn Trí | Bản đầu tiên |

**2-Đánh giá và lựa chọn giải pháp**

***2.1-Tiêu chí lựa chọn***

Có 5 tiêu chí để đánh giá và lựa chọn giải pháp, được sắp xếp theo thứ tự độ quan trọng giảm dần:

* Việc deploy game trên portal không làm thay đổi behavior của game theo cách không mong muốn, không làm cho game phát sinh thêm bug.
* Sau khi thoát game, có thể quay lại được màn hình cũ
* Game và Portal có khả năng giao tiếp với nhau

Trong tương lai có thể có tính năng như: đăng nhập một tài khoản chơi được nhiều game, nạp tiền một lần sử dụng được trong tất cả các game. Để làm được như vậy, game và portal cần giao tiếp được với nhau trong quá trình người chơi chơi game. Vì vậy, đây cũng là một tiêu chí quan trọng.

* Khả năng update resource sau khi release
* Hiệu quả sử dụng bộ nhớ

***2.2-Đánh giá và lựa chọn giải pháp***

**Giải pháp 1**: Thay đổi đường dẫn thư mục của game và restart lại VM

**Ưu điểm:**

* Game deploy trên portal sẽ chạy giống trường hợp được cài như một ứng dụng riêng lẻ, không phát sinh thêm bug. Bởi lẽ, không bị conflict tên hàm, tên biến với portal cũng như với các game khác. (cần test thêm liệu các thao tác với searchPaths có hoạt động đúng theo mong muốn không?)
* Hiệu quả sử dụng bộ nhớ tốt: Vì khi restart VM, cocos sẽ release tất cả các tài nguyên, ví dụ texture cache, sprite frame cache...
* Không ảnh hưởng đến khả năng update resource.

**Nhược điểm:**

* Không quay lại được màn hình cũ.

🡪Hướng giải quyết là: Tìm cách lưu lại trạng thái của portal (tên scene hiện tại, các dữ liệu đã được khởi tạo, ...) trước khi start game và khôi phục lại trạng thái đó sau khi back game.

* Bị hạn chế khả năng giao tiếp giữa game với portal. Cách duy nhất để thực hiện giao tiếp là thông qua file system. Có nghĩa là khi game cần giao tiếp điều gì đó với portal, nó sẽ lưu ra một file. Sau khi người chơi back game, quyền điều khiển trả lại cho portal, portal sẽ đọc file này để biết được thông điệp game gửi đến cho mình. Cách thức giao tiếp này khá phức tạp và có tính chất gián đoạn.

**Giải pháp 2**: Load trực tiếp mã nguồn của game vào portal, khi cần chạy game, thực hiện việc gọi hàm để thực hiện việc chuyển scene, start game bình thường.

**Ưu điểm:**

* Game và portal có thể giao tiếp với nhau dễ dàng, liên tục, không bị gián đoạn.
* Sau khi back game, có thể quay lại màn hình cũ dễ dàng.

**Nhược điểm:**

* Dễ bị conflict tên hàm, tên biến, giữa game với portal và giữa các game với nhau.

🡪 Giải pháp: Đóng gói toàn bộ mã nguồn của game vào một closure, giúp tránh được hoàn toàn việc conflict tên hàm tên biến. Đây cũng chính là cách mà hầu hết các thư viện javascript sử dụng. Tuy nhiên có một lưu ý khi sử dụng giải pháp này đó là: tất cả biến (kể cả khai báo lớp trong cocos) đều cần khai báo với từ khóa var. Vì nếu không khai báo với từ khóa var thì biến sẽ được coi như một biến toàn cục 🡪 closure mất tác dụng. Và cũng không nên quên dấu ; sau mỗi khai báo.

* Nếu giải quyết được vấn đề conflict, thì ta lại gặp phải một vấn đề khác là: Ảnh hưởng đến hiệu năng của việc update resource. Bình thường, chúng ta sửa file trong mã nguồn thì khi update cũng chỉ phải update lại file đó. Còn khi đóng gói toàn bộ code vào một file lớn, thì sửa bất cứ file nào, cũng phải update toàn bộ code. Với cờ tỷ phú, toàn bộ code là khoảng hơn 800kb, khi nén code lại, chỉ còn gần 300kb, nếu tần suất update code thấp thì điều này có thể chấp nhận được.
* Một vấn đề quan trọng nữa là vấn đề về config, cụ thể là:

Kích thước màn hình thiết kế giữa các game khác nhau

Chiến lược giải quyết vấn đề đa màn hình giữa các game khác nhau

Vấn đề liên quan đến đường dẫn, searchPaths, ...

🡪Giải pháp: Lưu lại các thông số cũ của portal và thay thế chúng bằng các thông số mới của game trước khi start. Khi người chơi back game thì lại khôi phục lại các thông số cũ của portal.

* Vấn đề về hiệu năng: Khi chạy nhiều game trên portal (chạy game này, rồi lại thoát ra chạy game khác), nếu quản lý không tốt thì có thể tất cả các resouce của các game đều bị load vào. Khi không sử dụng nữa cũng sẽ vẫn nằm trong bộ nhớ, không bị release đi 🡪 lãng phí.

🡪Giải pháp: Thực hiện các thao tác release tài nguyên khi back game, chẳng hạn xóa các texture không được sử dụng trong texture cache, ...

**Lựa chọn giải pháp**:

Chúng ta sẽ lựa chọn giải pháp 2. Lý do là: các giải pháp giải quyết các nhược điểm của nó được chứng minh là khả thi bằng các cài đặt thử nghiệm trong thực tế.

**Các vấn đề còn tồn tại:**

* Giải pháp cho vấn đề hiệu năng cũng chưa thực sự rõ ràng (cần làm rõ các tài nguyên được quản lý như thế nào? Những tài nguyên nào cần được release? Sử dụng hàm nào để release chúng? ...)
* Cần thử nghiệm thêm với nhiều game và test để phát hiện sớm các vấn đề.
* Vẫn chưa co smootj giải pháp hiệu quả cho searchPaths và đường dẫn file, mới chỉ là sửa lại thủ công bằng tay.