# LỜI NÓI ĐẦU

Nguồn gốc của video game nằm trong quá trình phát triển với ống phóng tia âm cực -dựa trên hệ thống phòng thủ tên lửa vào cuối những năm 1940. Các chương trình này sau đó được chuyển thể thành trò chơi đơn giản khác trong những năm 1950. Vào cuối những năm 1950 và qua các năm 1960, trò chơi máy tính đã được phát triển nhiều hơn (phần lớn là trên các máy tính lớn), với sự tinh tế và phức tạp ngày càng tăng lên (rất nhiều các video game thời kỳ đầu đã biến mất và hầu như không có một bản ghi lại nào). Trải qua các thời kỳ, video game tách ra trên nhiều hệ máy khác nhau: game arcade (hệ máy thùng), máy tính lớn, game console (hệ máy có tay cầm điều khiển), máy tính cá nhân và sau cùng là hệ máy cầm tay.

Video game đầu tiên được thương mại hóa là Computer Space năm 1971, mà về sau đặt nền móng cho một ngành công nghiệp giải trí mới trong cuối những năm 1970 tại Hoa Kỳ, Nhật Bản, và châu Âu. Sự suy sụp đầu tiên xảy ra vào năm 1977 khi các công ty bị bắt buộc phải bán các hệ thống đã lỗi thời của họ tràn ngập thị trường. Sáu năm sau đó, một cuộc sụp đổ lớn hơn xảy ra. Cơn khủng hoảng này (gây ra bởi việc đổ bộ ồ ạt các video game chất lượng tệ hại vào trong thị trường) là kết quả sự sụp đổ hoàn toàn của ngành công nghiệp game trên hệ máy có tay cầm (game console) ở Mỹ, cuối cùng chuyển sự thống trị của thị trường game từ Bắc Mỹ sang Nhật Bản. Trong khi cơn khủng hoảng này tiêu diệt thị trường hệ máy console, thì game trên hệ máy tính cá nhân gần như không bị ảnh hưởng. Các thế hệ tiếp theo của hệ console vẫn tiếp tục bị thống trị bởi các tập đoàn Nhật Bản. Dù đã có vài nỗ lực bởi các tập đoàn ở Bắc Mỹ và Châu Âu nhằm giành lại chỗ đứng của mình, với thế hệ thứ tư các hệ máy console, nhưng sự liên doanh này thất bại hoàn toàn. Cho tới khi thế hệ máy chơi game thứ sáu ra đời với Xbox là thành công thương mại thực sự được phát triển bởi tập đoàn Microsoft nhưng vẫn còn tỏ ra yếu thế về doanh số với các tập đoàn Nhật Bản như Sony, Nintendo hay Sega. Thị trường game cho máy cầm tay cũng đi theo một con đường giống như hệ console với vài thất bại của các tập đoàn Mỹ, tất cả đều không thành công trừ một số thành công hạn chế trong các hệ máy cầm tay thời kỳ đầu.

Và em, một sinh viên khoa công nghệ thông tin xin chọn đề tài làm một game đơn giản tên Flappy Bird bằng java. Mong Flappy Bird sẽ mang lại cho người sử dụng những giây phút thư giãn sau những giờ làm việc căng thẳng.

# MỤC LỤC

[LỜI NÓI ĐẦU 1](#_Toc10639053)

[MỤC LỤC 2](#_Toc10639054)

[CHƯƠNG I. PHÂN TÍCH YÊU CẦU BÀI TOÁN 4](#_Toc10639055)

[1. Đặt vấn đề 4](#_Toc10639056)

[2. Bài toán 5](#_Toc10639057)

[3. Hướng giải quyết 5](#_Toc10639058)

[CHƯƠNG II. GIỚI THIỆU CHUNG 6](#_Toc10639059)

[1. Khái niệm game nói chung 6](#_Toc10639060)

[2. Khái niệm & lịch sử Flappy Bird 11](#_Toc10639061)

[3. Java Development Kit 15](#_Toc10639062)

[3.1. Khái niệm 15](#_Toc10639063)

[3.2. Cách cài đặt JDK 16](#_Toc10639064)

[4. Integrated Development Environment Là Gì 30](#_Toc10639065)

[4.1. Khái niệm 31](#_Toc10639066)

[4.2. Cách cài đặt IDE 33](#_Toc10639067)

[5. Giới thiệu về ngôn ngữ lập trình. 38](#_Toc10639068)

[5.1. Khái niệm Java core. 38](#_Toc10639069)

[5.2. Khái niệm Java spring boot 40](#_Toc10639071)

[5.3. Khái niệm Maven 40](#_Toc10639072)

[CHƯƠNG III. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG 42](#_Toc10639073)

[1. Khảo sát 42](#_Toc10639074)

[1.1. Khảo sát hiện trạng 42](#_Toc10639075)

[1.2 Các chức năng chính của game 43](#_Toc10639076)

[2. Đặc tả 43](#_Toc10639077)

[2.1 Các class chính dự định thiết kế 44](#_Toc10639078)

[3. Sơ đồ hoạt động hệ thống 44](#_Toc10639079)

[4. Sơ đồ phân luồn chức năng 44](#_Toc10639080)

[5. Phân tích thao tác người dùng 46](#_Toc10639081)

[CHƯƠNG IV. CHI TIẾT HỆ THỐNG 47](#_Toc10639082)

[1. Một số mã nguồn 47](#_Toc10639083)

[1.1. App 47](#_Toc10639084)

[1.2. Bird 47](#_Toc10639085)

[1.3. Game 49](#_Toc10639086)

[1.4. GamePanel 52](#_Toc10639087)

[1.5. Keyboard 53](#_Toc10639088)

[1.6. Pipe 54](#_Toc10639089)

[1.7. Render 55](#_Toc10639090)

[1.8. Util 56](#_Toc10639091)

[CHƯƠNG V. TỔNG KẾT 57](#_Toc10639092)

[1. Đánh giá kết quả thực hiện 57](#_Toc10639093)

[2. Hướng phát triển tiếp theo 57](#_Toc10639094)

[LỜI KẾT 58](#_Toc10639095)

[TÀI LIỆU THAM KHẢO 59](#_Toc10639096)

# CHƯƠNG I. PHÂN TÍCH YÊU CẦU BÀI TOÁN

## 1. Đặt vấn đề

Trong các ngà`nh kỹ thuật hệ thống và kỹ nghệ phần mềm, phân tích yêu cầu là công việc bao gồm các tác vụ xác định các yêu cầu cho một hệ thống mới hoặc được thay đổi, dựa trên cơ sở là các yêu cầu (có thể mâu thuẫn) mà những người có vai trò quan trọng đối với hệ thống, chẳng hạn người sử dụng, đưa ra. Việc phân tích yêu cầu có ý nghĩa quan trọng đối với thành công của một dự án.

Việc phân tích yêu cầu một cách có hệ thống còn được gọi là kỹ nghệ yêu cầu (requirements engineering). Đôi khi nó còn được gọi một cách không thật chính xác bằng những cái tên như thu thập yêu cầu (requirements gathering, requirements capture), hoặc đặc tả yêu cầu (requirements specification). Thuật ngữ "phân tích yêu cầu" còn được áp dụng cụ thể cho công việc thuần túy phân tích (thay vì các việc khác chẳng hạn như làm rõ yêu cầu hay viết tài liệu yêu cầu).

Các yêu cầu phải có tính đo được, kiểm thử được, có liên quan đến các nhu cầu hoặc cơ hội doanh nghiệp đã được xác định, và các yêu cầu phải được định nghĩa ở một mức độ chi tiết đủ cho việc thiết kế hệ thống.

Về khái niệm, việc phân tích yêu cầu bao gồm ba loại hoạt động sau:

* Làm rõ yêu cầu (Eliciting requirements): giao tiếp với khách hàng và người sử dụng để xác định các yêu cầu của họ.
* Xem xét yêu cầu (Analyzing requirements): xác định xem các yêu cầu được đặt ra có ở tình trạng không rõ ràng, không hoàn chỉnh, đa nghĩa, hoặc mâu thuẫn hay không, và giải quyết các vấn đề đó.
* Làm tài liệu yêu cầu (Recording requirements): các yêu cầu có thể được ghi lại theo nhiều hình thức, chẳng hạn các tài liệu ngôn ngữ tự nhiên, các tình huống sử dụng (use case), câu chuyện sử dụng (user story), hoặc các đặc tả tiến trình.

Phân tích yêu cầu có thể là một quá trình dài và khó khăn, cần đến nhiều kĩ năng tâm lý khéo léo. Các hệ thống mới làm thay đổi môi trường và các mối quan hệ giữa con người, do đó điều quan trọng là phải xác định được tất cả những người có vai trò quan trọng, xem xét tất cả các nhu cầu của họ và đảm bảo rằng họ hiểu được các hàm ý của hệ thống mới. Các nhà phân tích có thể sử dụng một số kĩ thuật để làm rõ các yêu cầu của khách hàng. Trong lịch sử, các kỹ thuật này bao gồm các cuộc phỏng vấn, thành lập các nhóm trọng tâm (focus group) với các cuộc họp bàn về yêu cầu (requirements workshops), và tạo ra các danh sách yêu cầu. Các kỹ thuật hiện đại hơn gồm có tạo nguyên mẫu (prototyping), và tình huống sử dụng. Khi cần thiết, nhà phân tích sẽ kết hợp các phương pháp này để thiết lập các yêu cầu chính xác của những người có vai trò quan trọng, nhằm mục đích xây dựng một hệ thống thỏa mãn các yêu cầu doanh nghiệp.

## 2. Bài toán

Vấn đề về người dùng và khách hàng

Trong cuốn Rapid Development, Steve McConnell đã liệt kê một loạt các khả năng người dùng có thể cản trở quá trình thu thập yêu cầu:

* Người dùng không hiểu họ muốn gì
* Người dùng không tuân theo một bộ yêu cầu đã được tài liệu hóa
* Người dùng nhất định đòi hỏi các yêu cầu mới sau khi chi phí và kế hoạch phát triển đã được hoạch định xong.
* Mức độ giao tiếp với người dùng là thấp
* người dùng thường không tham gia các đợt thẩm định hoặc không thể tham gia.
* Người dùng không hiểu kỹ thuật
* Người dùng không hiểu quy trình phát triển.

Những điều này có thể dẫn tới tình huống khi yêu cầu người dùng liên tục thay đổi ngay cả khi việc phát triển hệ thống hay sản phẩm đã được bắt đầu.

Vấn đề về kỹ sư/nhà phát triển

* Trong quá trình phân tích yêu cầu, các vấn đề sau có thể nảy sinh từ phía các kỹ sư và nhà phát triển
* Nhân viên kỹ thuật và người dùng cuối có thể có ngôn từ khác nhau. Kết quả là họ có thể tin rằng họ hoàn toàn đồng thuận cho đến khi sản phẩm hoàn thiện được đưa ra.
* Các kỹ sư và nhà phát triển có thể cố lái cho các yêu cầu khớp với một hệ thống hay mô hình sẵn có, thay vì phát triển một hệ thống theo sát nhu cầu của khách hàng
* Việc phân tích có thể do các kỹ sư hoặc lập trình viên thực hiện, thay vì các nhân viên có kỹ năng và kiến thức miền ứng dụng để có thể hiểu các nhu cầu của khách hàng một cách đúng đắn

## 3. Hướng giải quyết

Một giải pháp đối với các vấn đề về giao tiếp là thuê các chuyên gia về doanh nghiệp hoặc chuyên gia phân tích hệ thống game.

Các kỹ thuật được sử dụng để giải quyết bài toán thường là tạo nguyên mẫu, UML và phát triển phần mềm linh hoạt (Agile software development) cũng đã được dùng làm giải pháp cho các vấn đề trên.

Game được thiết kế ra phải đơn giản dễ chơi nhất, dành cho những người chưa từng chơi game vẫn có thể sử dụng được.

**CHƯƠNG II. GIỚI THIỆU CHUNG**

## 1. Khái niệm game nói chung

Video game (gọi tắt là game) là một dạng trò chơi điện tử liên quan đến tính tương tác với một giao diện người sử dụng để tạo ra một phản hồi hình ảnh trên một thiết bị hiển thị (video) như màn hình TV, kính thực tế ảo, màn hình máy tính. Từ video trong video game là cách gọi truyển thống cho một thiết bị hiển thị raster. Tuy nhiên, với sự phổ biến của thuật ngữ "video game", giờ đây nó ngụ ý tất cả các dạng thiết bị hiển thị. Hệ thống thiết bị điện tử sử dụng để chơi video game được gọi là các hệ máy; ví dụ như máy tính cá nhân hay các hệ máy console. Những hệ máy này có kích thước từ chiếc máy tính đồ sộ cho đến những thiết bị nhỏ gọn cầm tay. Một số hệ máy chuyên để chơi game như máy arcade, tiếng Anh arcade machine (hệ máy sử dụng tiền xu để chơi), trước đây khá phổ biến nhưng nay dần không còn được sử dụng nữa.

Các thiết bị đầu vào sử dụng để thao tác trong game được gọi là thiết bị điều khiển game (hay tay cầm - game controller), và thay đổi tùy theo hệ máy. Ví dụ, một tay cầm chuyên dụng có khi chỉ có một nút và một cần điều khiển, trong khi những loại khác lại có hàng tá nút bấm và có hơn một cần điều khiển. Các game cho máy tính cá nhân thế hệ đầu chỉ cần bàn phím để chơi, hoặc thông thường hơn là yêu cầu người sử dụng mua một tay cầm riêng với ít nhất một nút bấm. Rất nhiều máy tính hiện nay cho phép, thậm chí yêu cầu người chơi phải đồng thời sử dụng cả bàn phím và chuột.

Video game cũng thường sử dụng cách khác để tương tác và cung cấp thông tin cho người chơi. Phổ biến nhất là âm thanh, sử dụng các thiết bị tái tạo âm thanh, như loa và tai nghe. Các phản hồi khác có thể là thông qua thiết bị ngoại vi có chức năng haptic: công nghệ tạo chức năng rung phản hồi/cảnh báo cho tay cầm chơi game hay điện thoại di động.

**Lịch sử**

Các video game thời kỳ đầu sử dụng các thiết bị tương tác điện tử với các định dạng hiển thị khác nhau. Mẫu đầu tiên là từ năm 1947- một "thiết bị giải trí bằng ống bắn tia âm cực (CRT-Cathode ray tube)" được Thomas T. Goldsmith Jr. và Estle Ray Mann xin cấp bằng sáng chế vào 25/1/1947 và đã được duyệt vào 14/12/1948, là sáng chế của Mỹ số 2.455.992 . Lấy cảm hứng từ màn hình radar công nghệ cao, nó bao gồm một thiết bị analog cho phép người dùng kiểm soát một vector vẽ chấm trên màn hình để mô phỏng một tên lửa được bắn vào các mục tiêu, được vẽ cố định trên màn hình.

**Các hệ máy**

Thuật ngữ "hệ máy" đề cập đến sự kết hợp đầy đủ giữa hệ thống điện tử hay phần cứng máy tính với hệ phần mềm cấp thấp (cấp cơ bản) để có thể vận hành một video game. Thuật ngữ "hệ thống" cũng hay được sử dụng. Cũng như bây giờ, các hệ máy chơi game cũng được biết với cái tên next "gen" consoles (thế hệ console tiếp theo).

**PC**

Thông thường, việc dùng từ game PC nghĩa rằng người chơi sử dụng một máy tính cá nhân kết nối với một màn hình có độ phân giải cao. Tính mở cho phép một số tính năng cho các nhà phát triển để giảm chi phí phần mềm, tăng tính linh hoạt, đổi mới, cạnh tranh, tạo các bản chỉnh sửa ("mod"), mở lưu trữ để chơi trực tuyến (trong đó người chơi game với những người ở những nơi khác).

**Console**

Một game console sẽ được chơi trên một hệ máy chuyên dụng kết nối với một tivi thông thường hoặc một màn hình tích hợp các chân cắm chuyên dụng.

**Handheld**

Máy chơi game cầm tay là một thiết bị dùng để chơi game chứa tất cả các linh kiện cần thiết như màn hình hiển thị, phần cứng, phần mềm cốt lõi,... vừa với tay người sử dụng và tiện mang theo. Phần cứng của nó thường yếu hơn phần cứng của PC hoặc console. Một số máy cầm tay từ cuối những năm 1970 và đầu những năm 1980 chỉ có thể chơi được một trò. Trong những năm 1990 và 2000, một số trò chơi trên hệ này đã sử dụng hộp ROM, cho phép máy cầm tay có thể chơi nhiều trò khác nhau.

**Arcade**

Game arcade thường được hiểu là một game chơi trên một hệ máy thậm chí còn chuyên dụng hơn với các thiết bị điện tử thiết kế chỉ để chơi một game duy nhất và được đóng trong một thùng riêng biệt và chỉ được chơi khi bỏ xu vào. Sự chuyên dụng này không phải lúc nào cũng rõ ràng vì có thể có những trò chơi mà dành cho nhiều hệ máy. Hơn nữa, có những hệ máy để chơi trò chơi mà không phải là video game như trong trường hợp của hệ máy arcade (trò bắn bóng Pinball chơi trên chiếc máy thùng cồng kềnh với bóng thật và các nút bấm cơ học).

**Trình duyệt web**

Trình duyệt web cũng đã tự thiết lập hệ riêng vào những năm 2000, cung cấp môi trường đa nền tảng thiết kế các video game để chơi trên nhiều nền tảng. Đổi lại, điều này đã tạo ra các điều khoản mới để đạt điều kiện xếp loại trò chơi trên trình duyệt web. Các trò này có thể được xác định dựa trên trang web mà chúng có mặt, chẳng hạn như với các trò chơi trên "Miniclip". Những trò khác được đặt tên dựa trên nền tảng lập trình được sử dụng để phát triển chúng, chẳng hạn game Java và game Flash.

**Thiết bị di động**

Với sự ra đời của các hệ điều hành tiêu chuẩn cho các thiết bị di động như iOS và Android và các thiết bị có phần cứng mạnh hơn, game trên thiết bị di động đã trở thành một nền tảng quan trọng. Các trò chơi này có thể sử dụng các tính năng độc đáo của thiết bị di động mà không cần thiết có mặt trên các hệ khác, chẳng hạn như GPS và thiết bị camera để hỗ trợ trò chơi. Các game di động cũng dẫn đến sự phát triển của các giao dịch vi mô như một mô hình doanh thu hợp lệ cho các trò chơi thông thường.

**Thực tế ảo**

Các trò chơi thực tế ảo (VR game) thường yêu cầu người chơi sử dụng một thiết bị đặc biệt gắn trên đầu có màn hình lập thể, máy theo dõi chuyển động để đưa người chơi vào môi trường ảo phản ứng với chuyển động đầu của họ. Một số hệ thống VR bao gồm các đơn vị điều khiển cho tay người chơi để cung cấp cách thức tương tác trực tiếp với thế giới ảo. Các hệ thống VR thường yêu cầu một máy tính, console hoặc thiết bị xử lý riêng khác để kết hợp với thiết bị gắn trên đầu.

**Thể loại**

Một video game, như hầu hết các thể loại phương tiện thông tin khác, sẽ được chia thành các thể loại dựa trên nhiều yếu tố như lối chơi, loại mục tiêu, phong cách hình ảnh (hoạt hình hay mô phỏng) và các loại khác... Bởi vì các thể loại còn tùy thuộc vào việc định nghĩa nội dung, các thể loại cũng dần thay đổi và phát triển khi các phong cách mới của video game ra đời. Khi các tiến bộ công nghệ và các sản phẩm có giá trị liên quan đến việc phát triển video game ra đời đã khích lệ các game càng giống thực và phức tạp hơn nên có khả năng các thể loại mới xuất hiện càng nhiều (ví dụ: nuôi thú ảo), thúc đẩy ranh giới trong cách chơi của các game hiện tại hoặc trong một số trường hợp thêm các cách chơi mới (như đã thấy với các thiết bị thiết kế đặc biệt như EyeToy của Sony). Một số thể loại xuất hiện là sự kết hợp của các thể loại khác, như các game nhập vai trực tuyến nhiều người chơi (MMORPG). Cũng thường thấy các thể loại cấp cao hơn mà tổng hợp các đặc tính dựa trên các thể loại khác như hành động kết hợp với âm nhạc, hoặc kết hợp với yếu tố kinh dị trong game.

**Chuỗi khối**

Một nền tảng mới của trò chơi video đã xuất hiện vào cuối năm 2017, trong đó người dùng có thể sở hữu tài sản trong trò chơi (tài sản kỹ thuật số) bằng cách sử dụng công nghệ Blockchain. Một ví dụ đó là Cryptokitties.

**Phân loại**

**Game cốt lõi**

Nhìn chung, thảo luận về game trên báo chí và chính trị chỉ xoay quanh các tựa game của game cốt lõi; bao gồm video game phát triển để chơi trên máy tính cá nhân, hệ console và hệ cầm tay.

Game cốt lõi thường được xác định bởi cường độ của chúng, chiều sâu của lối chơi hay quy mô của sản phẩm liên quan đến quá trình tạo game và có thể bao gồm các game trên một phổ rộng các thể loại. Ví dụ dòng game Bit.Trip trên hệ thống mạng WiiWare, dòng Fallout cho PC và hệ console hay LittleBigPlanet cho hệ PS3, tất cả đều nằm trong phân loại game cốt lõi. Game cốt lõi đôi khi được xem xét ở góc độ lối chơi khá phức tạp và thường không hấp dẫn các người chơi thông thường (người chơi game để giải trí).

**Game casual**

Casual games (game đơn giản) có cái tên này do tính dễ dàng tiếp cận của chúng, lối chơi dễ hiểu và nắm bắt quy luật chơi nhanh chóng. Thêm vào đó, các game casual thường hỗ trợ khả năng tham gia vào và thoát ra nhanh theo yêu cầu. Game casual là một thể loại tồn tại rất lâu trước khi thuật ngữ xuất hiện và bao gồm cả các game như Solitaire hay Minesweeper mà thường thấy cài đặt sẵn trong các hệ điều hành Windows của Microsoft.

Thí dụ cho thể loại này như là tìm vật ẩn, câu đố, quản lý thời gian, xếp hình-tetris hoặc rất nhiều các thể loại game chiến thuật chạy trên nền flash khác. Game casual thường được bán qua các dịch vụ bán lẻ trên mạng như PopCap, Zylom và GameHouse hoặc cung cấp miễn phí qua các trang web như Newgrounds hay AddictingGames.

Trong khi hầu hết các game casual chơi trên máy vi tính, điện thoại động hay PDA, chúng cũng có mặt rất nhiều trên các hệ thống dịch vụ download trực tuyến dành riêng cho các hệ máy console (ví dụ Xbox Live, PlayStation Network, hay WiiWare).

**Game nghiêm túc**

Game nghiêm túc là những game được thiết kế chủ yếu để truyền đạt thông tin hoặc để học hỏi kinh nghiệm nào đó cho người chơi. Một số game nghiêm túc thậm chí không đủ điều kiện là một video game theo ý nghĩa truyền thống của thuật ngữ. Ngoài ra, phần mềm giáo dục không hẳn đã thuộc thể loại game này (ví dụ phần mềm hướng dẫn đánh máy, học ngoại ngữ, v.v...) và sự khác biệt chính sẽ dường như dựa trên mục đích chính của tiêu đề cũng như độ tuổi mà phần mềm hướng tới. Như với các thể loại khác, sự mô tả này giống một hướng dẫn hơn là một quy định.

Game nghiêm túc là các game thường làm ra với các mục đích ngoài giải trí đơn thuần. Và như game nòng cốt và game casual, game nghiêm túc có thể bao gồm hoạt động ở bất kỳ thể loại nào, mặc dù một số thể loại như game luyện tập, game giáo dục, hay game tuyên truyền có thể có mặt nhiều hơn trong nhóm game nghiêm túc do mục đích chuyên môn của chúng. Có những game thường được thiết kế để chơi bởi một chuyên gia như một phần của một công việc cụ thể hoặc để tăng cường kỹ năng. Chúng cũng có thể được tạo ra để truyền đạt nhận thức chính trị xã hội trên một phương diện cụ thể.

Một trong những game nghiêm túc có thương hiệu lâu đời nhất có lẽ là Microsoft Flight Simulator (game mô phỏng lái máy bay) lần đầu phát hành vào năm 1982 với tên này. Quân đội Mỹ sử dụng game mô phỏng thục tế ảo này để làm bài tập huấn luyện, cùng với đó là sự phát triển một số lớn các phần hỗ trợ bay khác như lái máy bay cảnh sát, máy bay chiến đấu, máy bay y tế. Một ví dụ về môi trường của một phần mềm không phải game nhưng lại được sử dụng như một nền tảng để phát triển các game nghiêm túc là thế giới ảo của Second Life, đang được sử dụng bởi một số các đơn vị của chính phủ Mỹ (vd:NOAA, NASA, JPL), các trường đại học (vd:Đại học Ohio, MIT) cho giáo dục và các chương trình học tập từ xa và cả trong kinh doanh (như IBM, Cisco Systems) cho các cuộc họp và huấn luyện.

Chiến thuật truyền thông trong các trò chơi video đóng một vai trò quan trọng trong việc đưa ra một tuyên bố hay truyền đạt một thông điệp về các vấn đề quan trọng liên quan. Điều này cho phép hình thức truyền thông cho một đối tượng rộng lớn hơn để có thể nhận và tiếp cận với thông tin nhất định mà nếu không thì có thể nó đã không phổ biến đến như vậy. Một ví dụ về chiến thuật truyền thông trong video game là thể loại game tin tức (hay game thời sự). Đây là những trò chơi ngắn liên quan đến các sự kiện đương đại được thiết kế để minh họa một quan điểm. Ví dụ, TAKE ACTION là một tập hợp studio làm game mà được đồng sáng lập bởi Susana Ruiz và đã thành công mạnh mẽ với thể loại game nghiêm túc. Một số game thể loại này bao gồm Darfur is Dying, Finding Zoe, và In The Balance. Tất cả các game này mang lại nhận thức về các vấn đề hay sự kiện quan trọng bằng một cách thông minh và sáng tạo .

**Game giáo dục**

Vào ngày 23/9/2009, Tổng thống Mỹ Barack Obama đã phát động chiến dịch "Giáo dục để Đổi mới" (Educate to Innovate) nhằm nâng cao kỹ năng khoa học, công nghệ, kỹ thuật và toán học của sinh viên Mỹ. Chiến dịch này tuyên bố rằng nó có kế hoạch để khai thác sức mạnh của các trò chơi tương tác nhằm tiếp cận hàng triệu sinh viên trong các năm tới, tạo cảm hứng cho họ trở thành các nhà sáng chế và phát minh thế hệ tiếp theo. Chiến dịch này đã tạo ra nhiều cơ hội mới cho lĩnh vực video game và đã đóng góp nhiều cuộc thi mới như Stem National Video Game Competition, Imagine Cup. Các cuộc thi này đều là các sự kiện tập trung vào cùng một vấn đề là có thể sử dụng video game như một phương thức truyền thông mới nhằm giáo dục và truyền bá kiến thức. www.NobelPrize.org sử dụng các trò chơi để lôi kéo người dùng tìm hiểu về thông tin liên quan đến những thành tựu của giải Nobel trong khi đang tận hưởng sự vui vẻ từ các trò chơi.

## 2. Khái niệm & lịch sử Flappy Bird

Flappy Bird (tạm dịch là "chú chim vỗ cánh") là một trò chơi điện tử trên điện thoại do Nguyễn Hà Đông, một developer ở Hà Nội, Việt Nam phát triển, và do dotGEARS, một studio phát triển game quy mô nhỏ, hoạt động độc lập có trụ sở tại Việt Nam phát hành vào năm 2013. Trò chơi được trình bày theo phong cách side-scroller (phong cách game với các đối tượng được nhìn thấy ở mặt bên (side-view) và di chuyển từ cạnh trái sang cạnh phải của màn hình), trong đó người chơi điều khiển một chú chim, cố gắng vượt qua các hàng ống màu xanh lá cây mà không chạm vào chúng. Nguyễn Hà Đông tạo ra Flappy Bird trong vòng một vài ngày, sử dụng một nhân vật chú chim mà anh đã thiết kế cho một dự án trò chơi bị hủy bỏ vào năm 2012.

Ban đầu, Flappy Bird được phát hành vào tháng 5 năm 2013 trên nền iOS5, sau đó nâng cấp cho hệ máy iOS6 và mới hơn vào tháng 9 năm 2013. Vào tháng 1 năm 2014, trò chơi bất ngờ trở nên nổi tiếng, đứng đầu bảng thể loạt miễn phí trên iTunes App Store của Mỹ và Trung Quốc và sau đó là trên UK App Store khi nó được mệnh danh là "trò Angry Birds mới". Vào cuối tháng 1, Flappy Bird là ứng dụng được tải về nhiều nhất trên App Store cũng như trên Google Play

Flappy Bird đã bị chỉ trích về mức độ khó của trò chơi, cũng như các cáo buộc về sao chép hình ảnh, âm thanh cũng như cách vận hành của một số trò chơi khác, cũng như tính gây nghiện của trò chơi này. Việc Flappy Bird bỗng nhiên nhảy vọt từ vị trí 1454 lên số 1 vào đầu năm 2014 chỉ trong vòng 26 ngày cũng gây nghi ngờ.

Flappy Bird đã bị gỡ xuống trên App Store và Google Play bởi tác giả của nó vào ngày 10 tháng 2 năm 2014, do những tội lỗi của nó - mà như tác giả cảm thấy - là gây nghiện và bị lạm dụng quá mức. Độ nổi tiếng và việc gỡ xuống đột ngột của trò chơi khiến cộng đồng sôi sục, thậm chí những chiếc điện thoại đã cài sẵn Flappy Bird được rao bán trên mạng Internet với giá cao Các trò chơi tương tự Flappy Bird trở nên nổi tiếng trên iTunes App Store sau khi trò chơi này bị gỡ bỏ, và cả Apple và Google đã gỡ bỏ những trò chơi từ chợ ứng dụng vì lý do quá giống với nguyên tác. Trò chơi cũng đã được phân phối thông qua các kênh không chính thức trên nhiều nền tảng.

Tháng 8 năm 2014, một phiên bản chỉnh sửa của Flappy Bird mang tên Flappy Birds Family được phát hành độc quyển cho Amazon Fire TV. Bay Tek Games cũng cho ra mắt một phiên bản trò chơi hành động Flappy Bird thanh toán bằng tiền xu.[19]

**Mục tiêu**

Mục tiêu của trò chơi là điều khiển một chú chim bay qua những cái ống. Nếu chú chim chạm vào chướng ngại vật thì trò chơi sẽ kết thúc. Mỗi khi chú chim vượt qua một cặp ống thì người chơi nhận được một điểm. Nó sử dụng đồ họa tương tự như Super Mario Bros., với một hệ thống điều khiển vô cùng đơn giản.

**Đón nhận**

Huffington Post gọi ứng dụng này là "một trò chơi điên rồ, khó chịu, khó nhằn và gây bực bội khi kết hợp một đường cong khó siêu dốc với đồ họa nhàm chán, xấu xí và chuyển động giật". Tuy nhiên, một đánh giá tích cực hơn đến từ Jenifer Whiteside của Amongtech.com, cho rằng nó có thể che mờ đi Candy Crush Saga như là một tựa game phổ biến nhất 2014 vì tính gây nghiện của nó.

Độ khó của trò chơi thường gây ra sự tức giận cho người chơi, một trong số đó tuyên bố anh ta đã mất đến nữa giờ để giành được năm điểm. Phiên bản trên Android dễ chơi hơn một chút so với phiên bản trên iOS, theo người sáng tạo ra trò chơi. Flappy Bird cũng được mệnh danh là "Ma túy trên App Store".

IGN cho trò chơi này 5.4 điểm trên 10, cho rằng tuy nó gây nghiện nhưng cũng nông cạn, cực kỳ đơn điệu, thiếu sáng tạo, không có sự tiến triển, và chỉ là thứ để giết thời gian. Khung cảnh trong trò chơi được cho là giống Super Mario, còn nhân vật chính thì giống con cá một mắt hơn là chim. Người đánh giá Vince Ingenito cho rằng Flappy Bird có thể hút hồn người chơi, hoặc khiến họ nhấn lên màn hình vài cái rồi xóa đi trong vòng 2 phút. Một số người dùng IGN đã lên tiếng phản đối bài đánh giá của Vince Ingenito, bình luận rằng IGN không nên phí thời gian viết bài đánh giá "thứ rác rưởi" này.

Theo đánh giá của Patrick O'Rourke, biên tập viên chuyên về công nghệ và các trò chơi của Canada.com thì Flappy Bird là một trò chơi tồi tệ và thuộc loại dở nhất mà anh ta từng chơi.

**Tranh cãi**

The Daily Telegraph đặt dấu hỏi về sự nổi tiếng bất thường của trò chơi, khi mà nó chẳng được chú ý gì trong nhiều tháng sau khi phát hành vào tháng 5 năm 2013, nhưng đến tháng 1 năm 2014 lại đột ngột nhảy từ vị trí số 1454 trong bảng (thống kê của GB Family) lên số 1 chỉ trong vòng 26 ngày (1 tháng 1 đến 27 tháng 1). Tờ Telegraph nghi ngờ tác giả đã tự tạo những lượt truy cập giả nhằm đẩy trò chơi của mình lên vị trí cao hơn trên bảng xếp hạng của các cửa hàng phần mềm. Nguyễn Hà Đông đã từ chối không bình luận về vấn đề này khi được hỏi.

Kotaku nhấn mạnh rằng chẳng có gì trong trò chơi này là nguyên bản, và cáo buộc Flappy Bird đã nhái Super Mario từ cảnh vật trong trò chơi, những ống cống màu xanh, đến tạo hình con chim.

Biên tập viên Patrick O'Rourke của Canada.com cáo buộc Flappy Bird đã bắt chước gần như giống hệt trò chơi tên Piou Piou vs. Cactus (được phát hành cho iOS và Android vào năm 2011). Một số trang tin của Pháp như 20 Minutes, Metronews cũng nghi ngờ Flappy Bird là "hàng nhái" bởi nó "giống đến mức kỳ lạ" Piou Piou vs. Cactus từ cách chơi (gõ gõ vào màn hình) đến tạo hình con chim (màu vàng với cái mỏ to màu đỏ), và các chướng ngại (màu xanh lá).

Kek, tác giả người Pháp của trò chơi Piou Piou vs. Cactus cũng đã lên tiếng về sự tương đồng quá lớn giữa hai trò chơi (Flappy Bird ra đời sau Piou Piou vs. Cactus khoảng 2 năm). Khi Kek liên hệ với Nguyễn Hà Đông thì Đông nói rằng "không hề biết gì về Piou Piou vs. Cactus". Pocket Gamer cho rằng có lẽ đủ chứng cứ để khẳng định Flappy Bird là game nhái, nhưng Kek nói anh ta sẽ không kiện ra tòa mà sẽ dùng số tiền và thời gian đi thưa kiện để đầu tư làm game mới thì hơn.

**Bị "khai tử"**

Vào lúc 2:09 sáng ngày 9 tháng 2 năm 2014, giờ Việt Nam, Nguyễn Hà Đông tuyên bố trên tài khoản Twitter của mình là "trong 22 tiếng tới, tôi sẽ gỡ trò 'Flappy Bird' xuống. Tôi không thể chịu đựng nổi nữa." Trước đó Đông đã viết "Flappy Bird là một thành công của tôi. Nhưng nó cũng đã phá hoại cuộc sống đơn giản của tôi. Thế nên giờ đây, tôi rất ghét nó."

Ngày 10/2/2014, Dong Nguyen gỡ Flappy Bird.

Sau đó, Kotaku có bài xin lỗi cha đẻ Flappy Bird.

Trong một lá thư gửi cho tờ Wall Street Journal, người phát ngôn Yasuhiro Minagawa của Nintendo nhấn mạnh rằng hãng không có lời than phiền nào liên quan đến game Flappy Bird.

Ông nói: “Mặc dù chúng tôi thường không bình luận gì về tin đồn và những nghi vấn, nhưng chúng tôi đã bác bỏ nghi ngờ này.” Trước đây có những thông tin không xác thực nói rằng Nintendo đang muốn kiện Flappy Bird vì sử dụng những hình ảnh tương tự như trò chơi nổi tiếng Super Mario Bros, và đó cũng là một trong những lý do khiến tác giả Nguyễn Hà Đông quyết định gỡ bỏ Flappy Bird ra khỏi các kho ứng dụng App Store cũng như Google Play.

Trong những dòng tweet, Đông chia sẻ rằng trò chơi này bị gỡ đơn giản là do nó “làm hỏng cuộc đời đơn giản” của anh và những áp lực từ báo chí, không phải vì vấn đề pháp lý hay bản quyền. Tuy nhiên theo dư luận đánh giá anh còn bị áp lực từ chính sự than phiền của phụ huynh game thủ, game khiến người chơi dễ nóng giận, ức chế đập phá máy và đồ đạc - Tạo ra game là giúp người chơi thư giãn bớt căng thẳng nhưng game này lại đem lại tiêu cực cho xã hội và cuộc sống của anh. Tuy nhiên không thể không công nhận trò chơi của anh đã tạo cảm hứng cho rất nhiều studio game độc lập phát triển mạnh mẽ sau khi thấy được sự thành công vang dội và khoản thu nhập khổng lồ do game đem lại. Hà Đông cũng khẳng định không bán game của mình cho bất kì công ty nào.

**Khởi động lại**

Trong một cuộc phỏng vấn mới đây với tờ Rolling Stone, Nguyễn Hà Đông cho biết anh đang cân nhắc đến việc đưa Flappy Bird trở lại.

## 3. Java Development Kit

JDK là một bộ công cụ phát triển Java, nó dành cho những người lập trình Java để phát triển ứng dụng. Về cơ bản nó bao gồm:

JRE (Java Runtime Environment) là một môi trường chạy ứng dụng Java.

Javac: Một chương trình để dịch mã mà bạn viết thành mã bytecode, khi ứng dụng Java chạy nó dịch mã bytecode thành mã máy tính và thực thi, điều đó có nghĩa là bytecode chỉ là một mã trung gian.

Archive (jar): Là một chương trình nén các file thành một file duy nhất có đuôi jar. Thường dùng để đóng gói các file class.

Javadoc: Là một công cụ tạo ra tài liệu hướng dẫn sử dụng API.

Và các công cụ khác cần thiết cho phát triển Java.

### 3.1. Khái niệm

Java Development Kit (JDK) là một trong ba gói công nghệ cốt lõi được sử dụng trong lập trình Java, cùng với JVM (Máy ảo Java - Java Virtual Machine) và JRE (Java Runtime Environment - Môi trường Java Runtime). Việc phân biệt giữa ba công nghệ này, cũng như hiểu được cách chúng kết nối với nhau là rất quan trọng.

* JVM là thành phần Java nền tảng để chạy các chương trình.
* JRE là phần on-disk của Java, phần tạo ra JVM.
* JDK cho phép các developer tạo các chương trình Java, trong đó các chương trình có thể được JVM và JRE xử lý và chạy.

Các developer khi mới sử dụng Java thường dễ nhầm lẫn Java Development Kit với Java Runtime Environment. Để phân biệt, ta cần nhớ rằng JDK là một gói các công cụ phát triển phần mềm dựa trên Java, trong khi JRE là gói công cụ sử dụng để chạy Java code.

JRE có thể được sử dụng như một thành phần độc lập để chạy các chương trình Java, dù vậy, JRE vẫn là một phần của JDK. JDK đòi hỏi bộ công cụ JRE vì chạy các chương trình Java là một phần của việc phát triển chúng.

Chúng ta có thể hiểu về JDK theo 2 cách như sau:

Định nghĩa chuyên ngành: JDK là một hệ tiêu chuẩn trong việc triển khai nền tảng Java, bao gồm các trình thông dịch dịch và thư viện lớp.

Định nghĩa thông thường: JDK là gói phần mềm bạn tải xuống để tạo các ứng dụng dựa trên Java.

**Trình biên dịch JDK & Java**

Ngoài JRE, môi trường được sử dụng để chạy các ứng dụng Java, mỗi JDK đều chứa trình biên dịch Java. Trình biên dịch là chương trình phần mềm có khả năng lấy các tệp .java ở dạng thô - văn bản thuần túy - và hiển thị chúng thành các tệp thực thi có đuôi . Class.

**Làm quen với JDK**

Thiết lập một Java trong môi trường phát triển cũng dễ như việc bạn download 1 JDK và thêm vào classpart. Khi tải xuống JDK, bạn sẽ cần chọn phiên bản Java mình muốn sử dụng. Java 8 là phiên bản được sử dụng phổ biến nhất, tuy nhiên, tính đến thời điểm hiện tại, Java 10 là phiên bản mới nhất. Java duy trì khả năng tương thích ngược, vì vậy chúng ta sẽ tải xuống bản phát hành mới nhất.

**Các gói JDK**

Sau khi chọn phiên bản Java, bạn cũng sẽ cần chọn gói Java muốn sử dụng. Các gói là các Java Development Kit dành cho các kiểu phát triển khác nhau. Các gói có sẵn là Java Enterprise Edition (Java EE), Java Standard Edition (Java SE) và Java Mobile Edition (Java ME).

Thông thường, một phiên bản JDK sẽ chứa Java SE. Nếu bạn tải xuống Java EE hoặc Java ME, bạn sẽ có một phiên bản Java SE tiêu chuẩn. Ví dụ, Java EE là nền tảng tiêu chuẩn với các công cụ bổ trợ tiện ích cho phát triển ứng dụng doanh nghiệp như Enterprise JavaBeans hay hỗ trợ cho Object Relational Mapping.

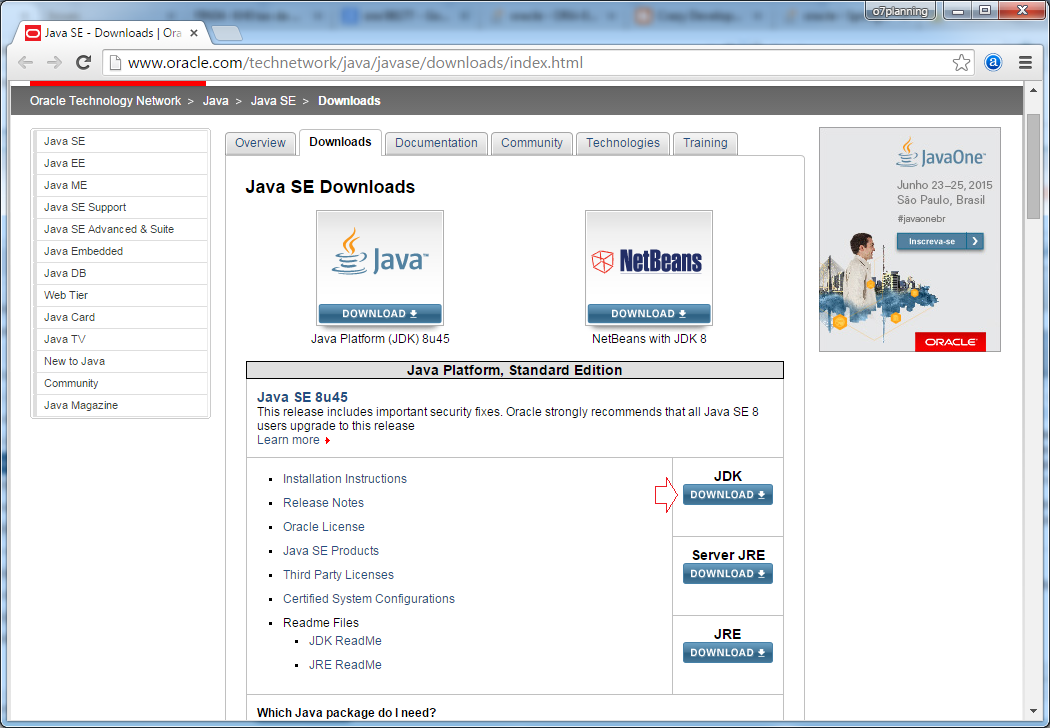
Việc chuyển đổi từ JDK này sang JDK cũng không có gì khó khăn. Vì vậy, bạn không cần phải quá lo lắng về việc chọn đúng phiên bản và gói JDK ngay từ đầu.

**Các phiên bản JDK tương thích**

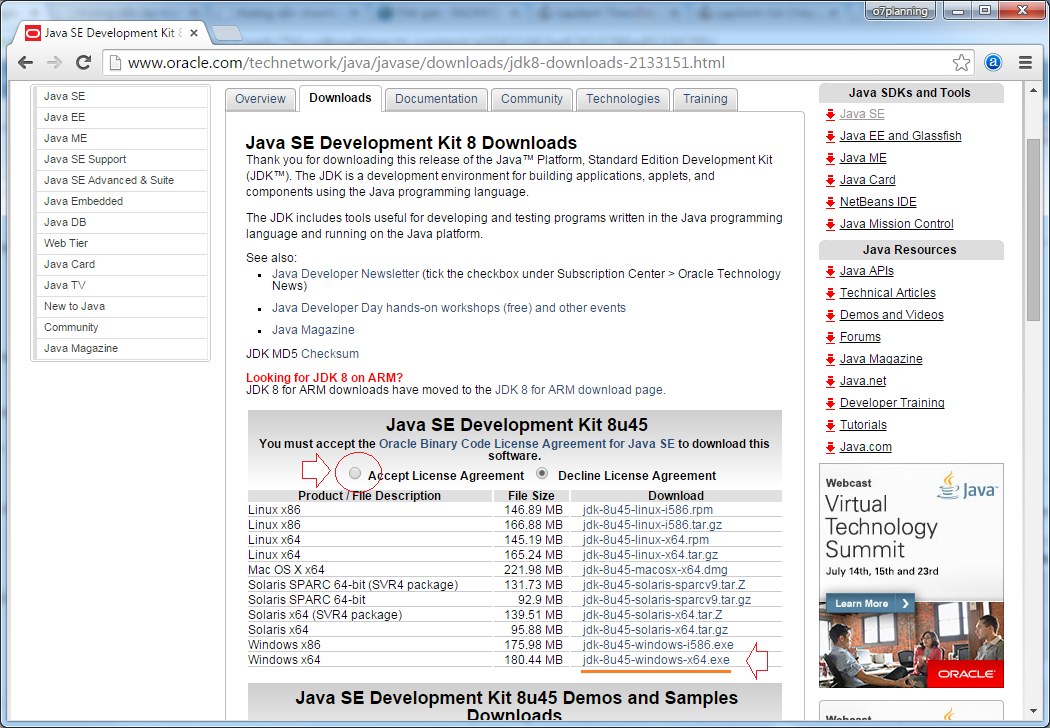
Do JDK cung cấp trình biên dịch cho các chương trình Java của bạn, nên JDK bạn sử dụng sẽ xác định phiên bản Java nào bạn có thể mã hóa. Ví dụ: nếu bạn muốn sử dụng chức năng hỗ trợ mới có trong Java 8, như toán tử Lambda, bạn sẽ cần ít nhất là phiên bản JDK Java 8 để biên dịch. Nếu không, lệnh javac sẽ từ chối mã với lỗi cú pháp.

### 3.2. Cách cài đặt JDK

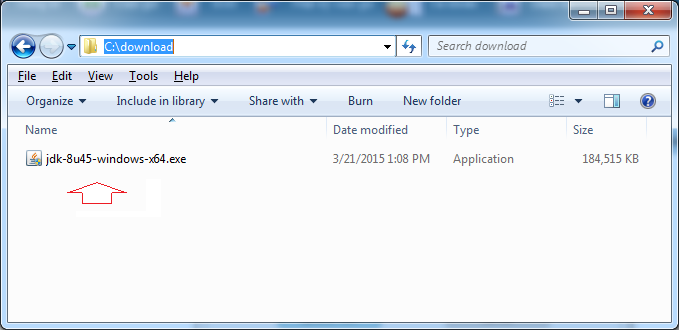
1- Download JDK

* <http://www.oracle.com/technetwork/java/javase/downloads/index.html>

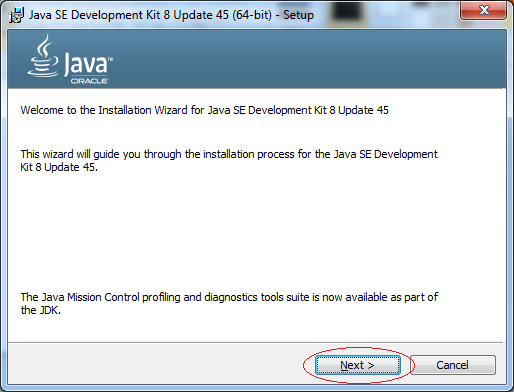
Nên kiểm tra xem hệ điều hành của mình là 32bit hay 64bit để download bộ cài **JDK**phù hợp.



Kết quả download được:

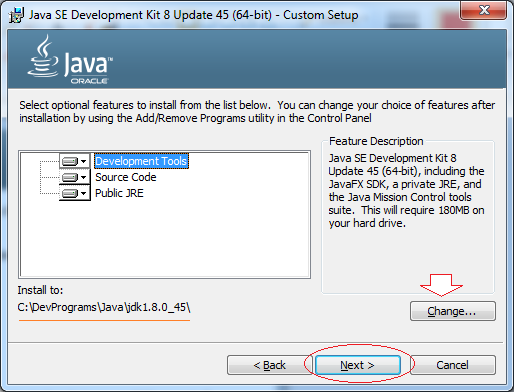


2- Cài đặt Java



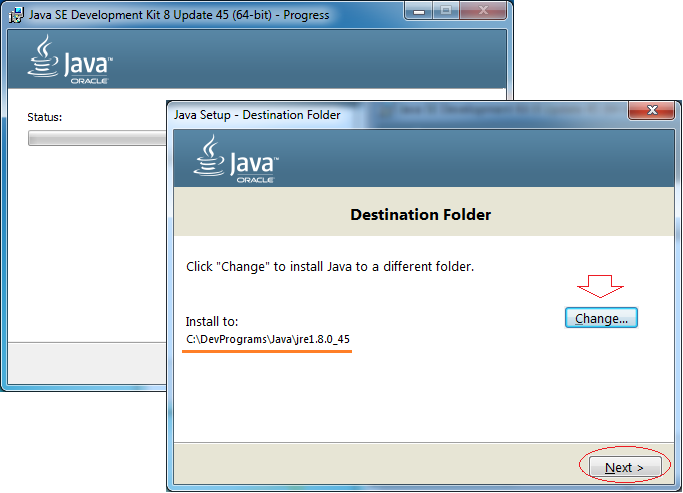
Nhập vào thư mục mà JDK sẽ được cài đặt ra:

* **C:\DevPrograms\Java\jdk1.8.0\_45\**



Ở đây chọn:

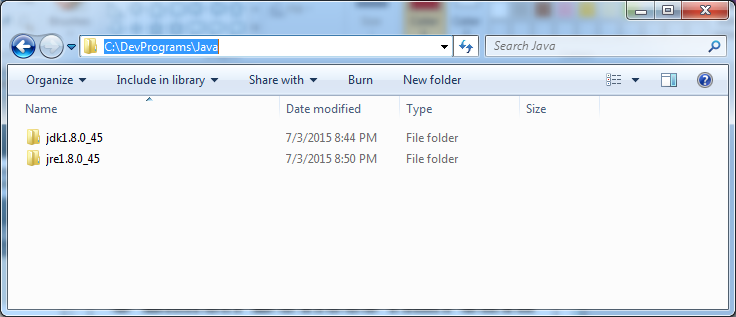
* **C:\DevPrograms\Java\jre.8.0\_45\**



Java đã được cài đặt thành công.



Kết quả bạn có được 2 thư mục:

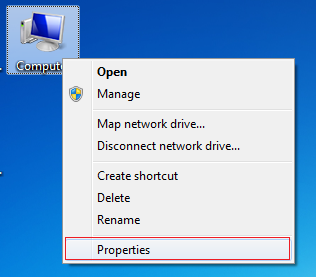


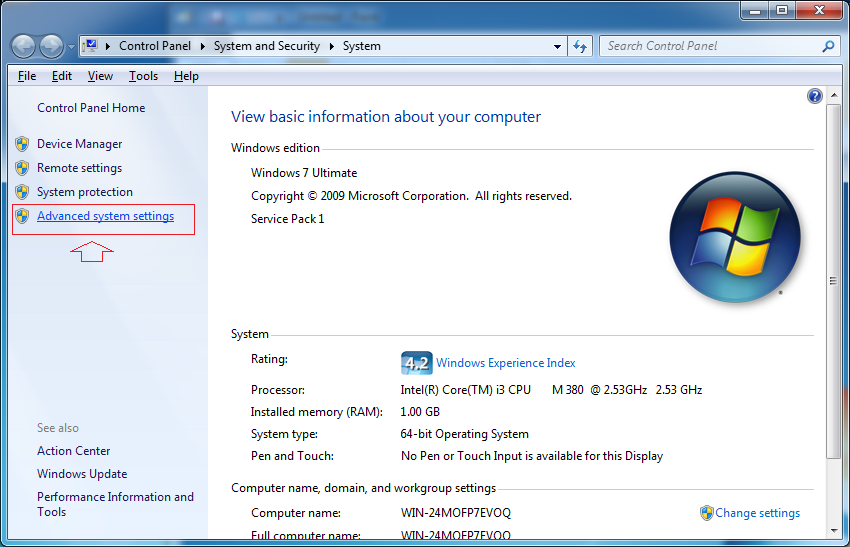
3- Cấu hình biến môi trường cho Java

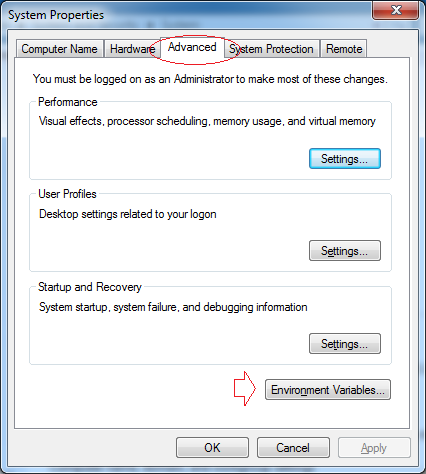
Tiếp theo chúng ta cấu hình biến môi trường cho Java.  Việc này không bắt buộc, nhưng nếu trên máy tính của bạn cài đặt nhiều phiên bản Java, thì việc cấu hình là cần thiết để xác định phiên bản java nào mặc định được sử dụng.

**Sử dụng Window Vista hoặc Window 7:**

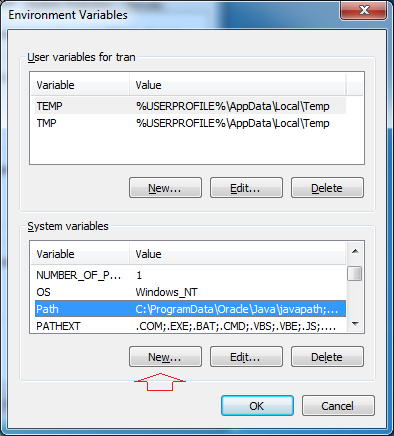
Trên Desktop, nhấn phải chuột vào Computer, chọn **Properties**.





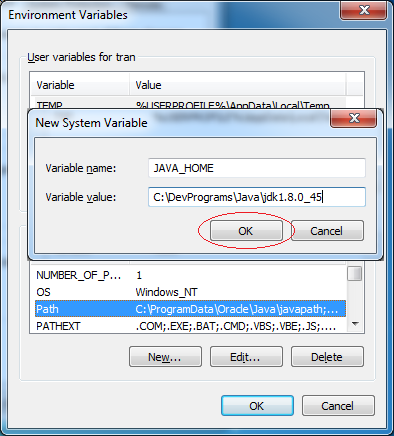


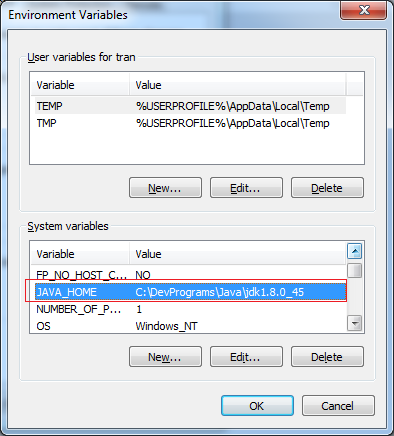
Nhấn ***New*** để tạo mới một biến môi trường có tên **JAVA\_HOME**.



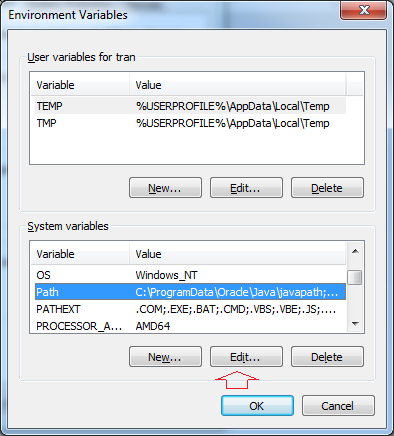
Nhập vào đường dẫn tới thư mục JDK.

* **Variable name:** JAVA\_HOME
* **Variable value:** C:\DevPrograms\Java\jdk1.8.0\_45



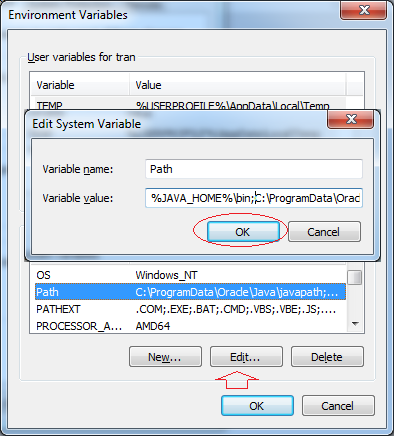


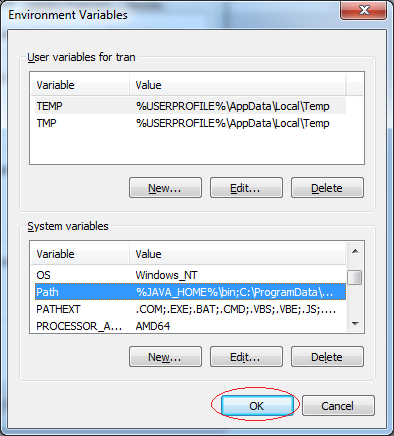
Tiếp theo sửa đổi biến môi trường ***path***



Thêm vào phía trước giá trị của biến môi trường ***path***:

* **%JAVA\_HOME%\bin;**





Cài đặt và cấu hình Java thành công.

## 4. Integrated Development Environment Là Gì

IDE viết tắt của cụm từ Integrated Development Environment là phần mềm cung cấp cho các lập trình viên một môi trường tích hợp bao gồm nhiều công cụ khác nhau như chương trình viết mã lệnh hay code editor, chương trình sửa lỗi hay debugger, chương trình mô phỏng ứng dụng khi chạy thực tế hay simulator.... Nói cách khác thì IDE là một phần mềm bao gồm những gói phần mềm khác giúp phát triển ứng dụng phần mềm.

Các IDE phổ biến đang được sử dụng gồm có Netbeans IDE, Eclipse, PhpStorm, XCode (sử dụng trên hệ điều hành MacOS để phát triển ứng dụng mobile)... Phần tiếp theo chúng ta sẽ cùng tìm hiểu một số IDE phổ biến.

**Netbeans IDE**

Netbeans IDE là phần mềm IDE miễn phí được tạo ra bởi Sun Microsystems và phiên bản mới nhất của Netbeans tính tới thời điểm bài viết này được công bố trên hoclaptrinh.org là phiên bản 8.1. Netbeans IDE được phát triển mới mục đích ban đầu chủ yếu để giúp các lập trình viên xây dựng ứng dụng Java tuy nhiên sau đó nó đã được mở rộng để hỗ trợ sử dụng cho phát triển các ứng dụng PHP, C, C++ và HTML5 (bao gồm Javascript và CSS).

Netbeans có thể được cài đặt trên nhiều hệ điều hành khác nhau bao gồm Windows, MacOS và các bản phân phối phổ biến của Linux khác như Ubuntu hay Mint. Sau khi Oracle mua lại Sun Microsystems thì hiện nay Oracle đang là công ty đứng đằng sau hỗ trợ phát triển và xây dựng Netbeans. Mã nguồn của Netbeans được phát hành theo giấy phép GPL2.

**Eclipse**

Tương tự như Netbeans, Eclipse ban đầu cũng được phát triển nhằm hỗ trợ chủ yếu trong việc xây dựng các phần mềm viết bằng ngôn ngữ Java. Tuy nhiên thông qua việc cài đặt thêm các trình cắm (hay Plugin) thì Eclipse còn có thể hỗ trợ phát triển phầm mềm viết bằng các ngôn ngữ khác như C, C++, Python, Ruby, PHP, R....

Eclipse Foundataion là công ty đứng đằng sau hỗ trợ phát triển Eclipse IDE . Tương tự như Netbeans, Eclipse IDE cũng hỗ trợ nhiều nền tảng hệ điều hành khác nhau như Windows, MacOS là Linux. Eclipse IDE là phần mềm miễn phí và được phát hành theo giấy phép GPL license.

### 4.1. Khái niệm

IDE được sử dụng trong project này sẽ là Intellij IDEA – JetBrain

IntelliJ IDEA là một IDE Java để phát triển các phần mềm máy tính. Nó được phát triển bởi JetBrains (trước đây gọi là IntelliJ), nó được cấp phép Apache 2 cho phiên bản cộng đồng, vfa một phiên bản thương mại độc quyền.Cả hai có thể được sử dụng cho phát triển thương mại.

**Lịch sử**

Phiên bản đầu tiên của IntelliJ IDEA được phát hành vào tháng 1/2001, và là một trong những IDE Java có sẵn đầu tiên với khả năng điều hướng mã và cải tiến mã nguồn được tích hợp.

Năm 2010 Infoworld báo cáo, IntelliJ đã nhận được điểm trung bình kiểm tra cao nhất trong số bốn công cụ lập trình Java hàng đầu: Eclipse, IntelliJ IDEA, NetBeans và JDeveloper.

Tháng 12/2014, Google giới thiêu phiên bản Android Studio 1.0, một IDE nguồn mở cho các ứng dụng Android, dựa trên phiên bản cộng đồng nguồn mở của IntelliJ IDEA. Các môi trường phát triển khác dựa trên framework của IntelliJ bao gồm AppCode, CLion, PhpStorm, PyCharm, RubyMine, WebStorm và MPS.

Yêu cầu hệ thống

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Windows** | **macOS** | **Linux** |
| **Phiên bản OS** | Windows 10/8/7 x64 | macOS 10.8 hoặc mới hơn | GNOME hoặc KDE |
| **RAM** | Tối thiểu 1 GB; đề xuất 4 GB hoặc hơn cho phát triển Android và phần mềm thương mại. | | |
| **Ổ cứng** | 300 MB ổ cứng trống + ít nhất 1 GB cho bộ nhớ cache | | |
| **Phiên bản JDK** | JDK 1.8 từ 2016. | | |
| **Screen resolution** | Độ phân giải màn hình tối thiểu 1024 × 768 | | |

**Tính năng**

Phiên bản 2017.1 bao gồm hỗ trợ cho Java 9 sắp tới, một thiết kế UI cho phát triển ứng dụng Android, Play 2.0 và Scala.

**Hỗ trợ mã hóa**

IDE cung cấp các tính năng nhất định như hoàn thành mã bằng cách phân tích ngữ cảnh, điều hướng mã cho phép nhảy vào một lớp hoặc khai báo trong mã trực tiếp, tái cấu trúc mã và các tùy chọn để sửa các mâu thuẫn thông qua các đề xuất.

**Công cụ tích hợp và kết hợp**

IDE tích hợp sẵn các công cụ build/đóng gói như grunt, bower, gradle, và SBT. Nó hỗ trợ các hệ thống kiểm soát phiên bản như Git, Mercurial, Perforce, và SVN. Các cơ sở dữ liệu như Microsoft SQL Server, ORACLE, PostgreSQL, và MySQL có thể được truy cập trực tiếp từ IDE.

**Hệ sinh thái Plugin**

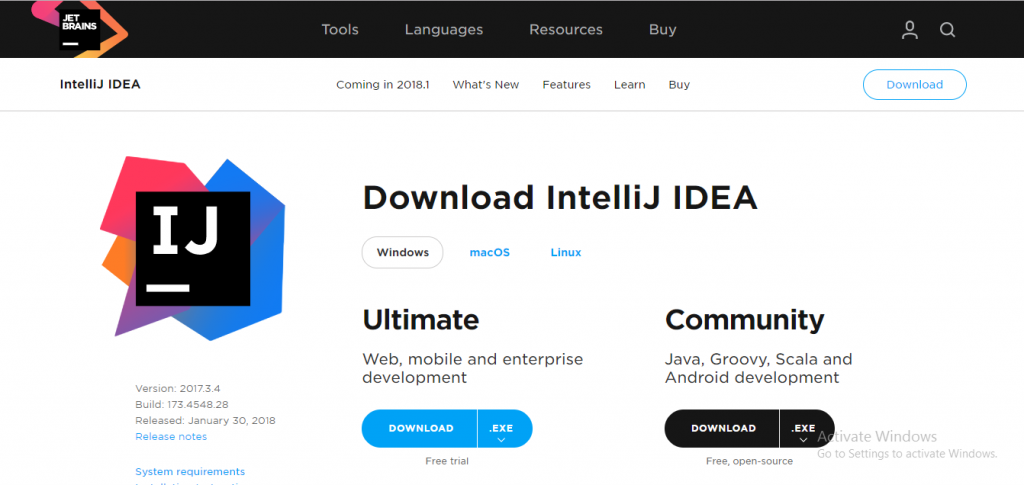
IntelliJ hỗ trợ các plugin thông qua đó người ta có thể thêm chức năng bổ sung cho IDE. Có thể tải xuống và cài đặt các plugin từ trang web của kho lưu trữ plugin của IntelliJ hoặc thông qua tính năng cài đặt và tìm kiếm plugin sẵn có của IDE. Hiện tại, phiên bản IntelliJ IDEA Community có 1495 plugins, trong khi bản Ultimate có 1626. Những con số này nhỏ hơn nhiều so với các trình soạn thảo như Atom, có hơn 7000 packages (về cơ bản là các plugin). Tuy nhiên, một số tính năng được bao gồm theo mặc định trong IntelliJ chỉ có thể được thêm vào Atom bằng cách cài đặt các packages. Ví dụ, linting được tích hợp vào IntelliJ, và nó có thể được cài đặt trong Atom, theo ngôn ngữ lập trình, với nhiều gói khác nhau.

### 4.2. Cách cài đặt IDE

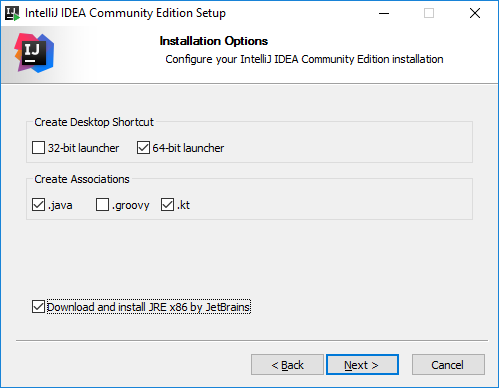
**Cài đặt IntelliJ IDEA**

Download file cài của IntelliJ IDEA tại trang chính thức của Jetbrains. Địa chỉ download:

<https://www.jetbrains.com/idea/download/>

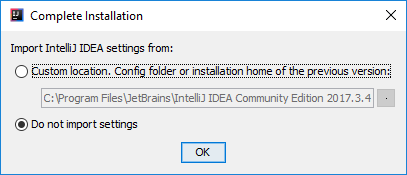
Giao diện trang download IntelliJ IDEA

 Đối với Windows, trong quá trình cài đặt cần lưu ý:



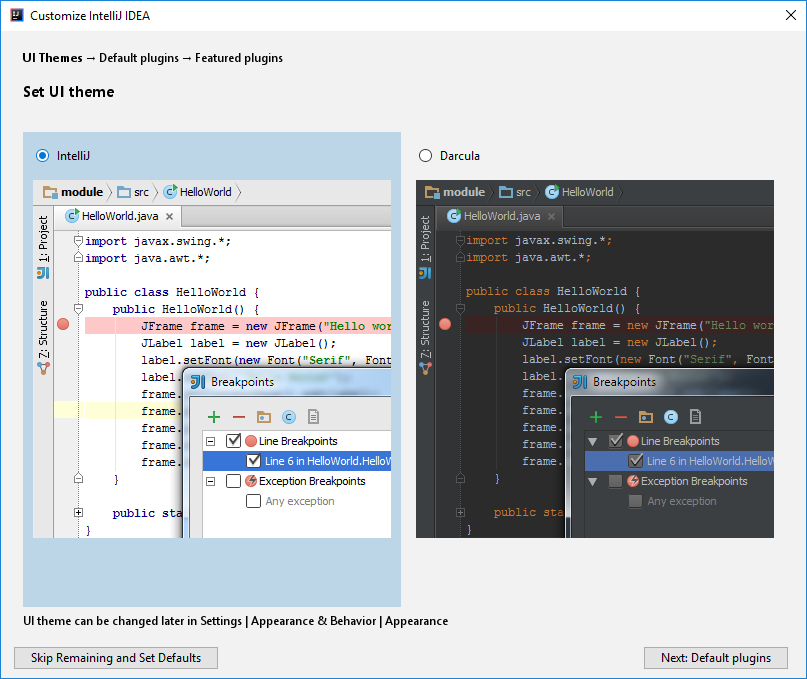
chọn .java .kt và download JRE

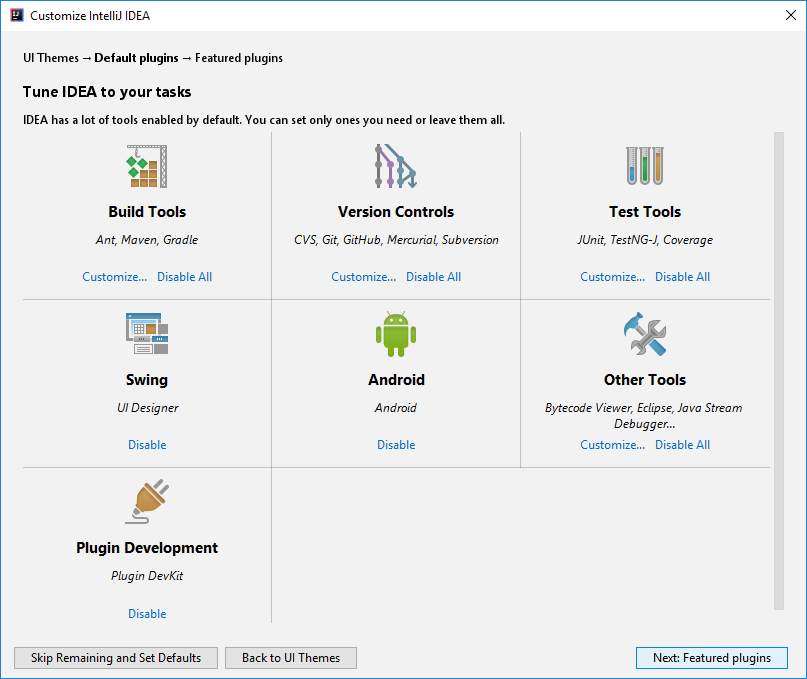
Ở mục Create Desktop Shortcut, dùng win bản bao nhiêu bit thì chọn bản tương ứng. Điều quan trọng ở đây là cần click chọn vào .java và .kt ở trong mục Create Associations. Như vậy ta mới có thể code được Java và Kotlin trên IDE này.

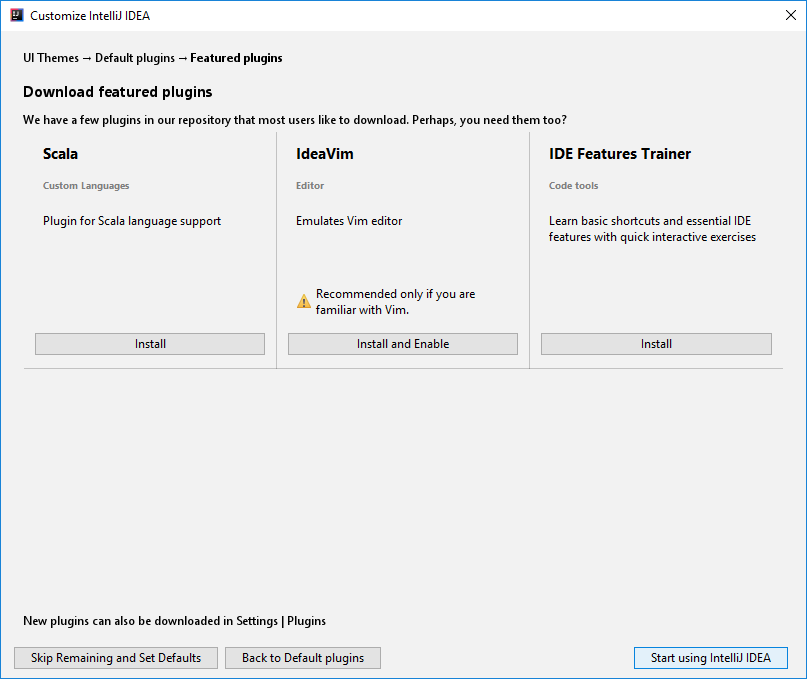


Chọn cấu hình setting cho IDE

Hãy tick chọn Do not import settings để sử dụng cấu hình mặc định của IntelliJ IDEA. Tiếp đó là tới phần chọn theme và các plugin cho IDE.

Chọn theme cho IntelleJ IDEA

Tùy chọn disable 1 số plugin

Ấn nút Start using IntelliJ IDEA để bắt đầu

Cài đặt xong IDE Intellij IDEA – JetBrain.

## 5. Giới thiệu về ngôn ngữ lập trình.

### 5.1. Khái niệm Java core.

Java là một ngôn ngữ lập trình và nền tảng tính toán được phân phối lần đầu tiên bởi Sun Microsystems vào năm 1995. Rất nhiều ứng dụng, trang web đều được viết bằng Java. Java nhanh, bảo mật và đáng tin cậy.

Java là ngôn ngữ [lập trình hướng đối tượng](https://stackjava.com/faq/lap-trinh-huong-doi-tuong-la-gi-uu-nhuoc-diem.html) (OOP).

**Về tốc độ:**

* Trước đây, Java chạy chậm hơn những ngôn ngữ dịch thẳng ra mã máy như C và C++, nhưng sau này nhờ công nghệ “biên dịch tại chỗ” – Just in time compilation, khoảng cách này đã được thu hẹp, và trong một số trường hợp đặc biệt Java có thể chạy nhanh hơn.
* Java chạy nhanh hơn những ngôn ngữ thông dịch như Python, Perl, PHP gấp nhiều lần.

**Về quản lý bộ nhớ:**

* Trong Java, hiện tượng rò rỉ bộ nhớ hầu như không xảy ra do bộ nhớ được quản lý bởi [Java Virtual Machine (JVM)](https://stackjava.com/java-basic/su-khac-nhau-giua-jdk-jre-va-jvm.html) bằng cách tự động “dọn dẹp rác”. Người lập trình không phải quan tâm đến việc cấp phát và xóa bộ nhớ như C, C++. Tuy nhiên khi sử dụng những tài nguyên mạng, file IO, database (nằm ngoài kiểm soát của [JVM](https://stackjava.com/java-basic/su-khac-nhau-giua-jdk-jre-va-jvm.html)) mà người lập trình không đóng các kết nối thì rò rỉ dữ liệu vẫn có thể xảy ra.

**Về cú pháp:**

Cú pháp Java được vay mượn nhiều từ C & C++ nhưng có cú pháp hướng đối tượng đơn giản hơn và ít tính năng xử lý cấp thấp hơn. Do đó việc viết một chương trình bằng Java dễ hơn, đơn giản hơn, đỡ tốn công sửa lỗi hơn.

(Xem thêm: [Các phiên bản của Java](https://stackjava.com/java/cac-version-cua-java.html))

**Ngôn ngữ Java là gì? Đặc trưng của Java**

**Các đặc trưng của Java**

**Hướng đối tượng**

* Mọi thực thể trong chương trình đều là một đối tượng (1 class xác định)
* Các biến, hàm đều nằm trong một class nào đó

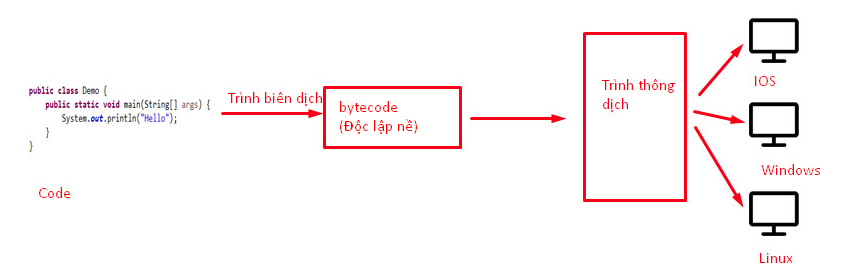
**Đơn giản**

* Loại bỏ con trỏ
* Loại bỏ lệnh goto
* Không cho phép đa kế thừa (chuyển sang sử dụng interface)

**Độc lập phần cứng và hệ điều hành**

Khác với phần lớn ngôn ngữ lập trình thông thường, thay vì biên dịch mã nguồn thành mã máy hoặc thông dịch mã nguồn khi chạy, Java được thiết kế để biên dịch mã nguồn thành bytecode, bytecode sau đó sẽ được môi trường thực thi (runtime environment) chạy.

Do đó một chương trình viết bằng Java có thể chạy trên nhiều thiết bị, nhiều hệ điều hành khác nhau.

[](https://stackjava.com/wp-content/uploads/2018/03/java-la-gi.png)

**Mạnh mẽ**

* Quá trình cấp phát, giải phóng bộ nhớ được thực hiện tự động.
* Yêu cầu chặt chẽ khi khai báo dữ liệu, ép kiểu dữ liệu.
* Tự động phát hiện lỗi lúc biên dịch.
* Không sử dụng con trỏ hoặc các phép toán con trỏ.

**Bảo mật**

**Phân tán**

* Java hỗ trợ lập trình cho các hệ thống phân tán như client-server, RMI… bằng Java web, UDP, TCP…

**Đa luồng**

* Java hỗ trợ lập trình đa luồng (multithreading); việc đồng bộ dữ liệu trong lập trình đa luồng cũng khá đơn giản.

### 5.2. Khái niệm Java spring boot

Spring Boot là một dự án phát triển bởi JAVA (ngôn ngữ java) trong hệ sinh thái Spring framework. Nó giúp cho các lập trình viên chúng ta đơn giản hóa quá trình lập trình một ứng dụng với Spring, chỉ tập trung vào việc phát triển business cho ứng dụng.

Để phát triển một ứng dụng web cơ bản HelloWorld sử dụng Spring framework bạn sẽ cần ít nhất 5 công đoạn sau;

Tạo một project sử dụng Maven với các dependency cần thiết của Spring MVC và Servlet API.

Một tập tin web.xml để khai báo DispatcherServlet của Spring MVC.

Một tập tin cấu hình của Spring MVC.

Một class Controller trả về một trang “Hello World” khi có request đến.

Cuối cùng là phải có một web server dùng để triển khai ứng dụng lên chạy.

Trong các công đoạn này, chỉ có công đoạn tạo một class Controller thì có thể khác cho các ứng dụng khác nhau vì mỗi ứng dụng có một yêu cầu khác nhau. Còn các công đoạn khác thì như nhau.

Giờ đây với Spring Boot, chúng ta có thể tạo dự án Spring một cách nhanh chóng và cấu hình cũng đơn giản dùng Sublime Text để phát triển luôn khỏi cần cài đặt eclipse hay netbean nặng bỏ bà.

Dưới đây là một số tính năng nổi bật của Spring Boot:

* Tạo các ứng dụng Spring độc lập
* Nhúng trực tiếp Tomcat, Jetty hoặc Undertow (không cần phải deploy ra file WAR)
* Các starter dependency giúp việc cấu hình Maven đơn giản hơn
* Tự động cấu hình Spring khi cần thiết
* Không sinh code cấu hình và không yêu cầu phải cấu hình bằng XML …

### 5.4. Khái niệm Maven

Apache maven là một chương trình quản lý dự án cho phép các developers có thể quản lý về version, các dependencies ( các thư viện sử dụng trong dự án ) , quản lý build, tự động download javadoc & source, …

Vì sao phải sử dụng maven? Bạn hãy thử tượng nếu dự án của chúng ta sử dụng rất nhiều thư viên thứ 3 : struts, hibernate, spring, … Việc import thư viện và các dependency ( dịch hiểu là “sự phụ thuộc” ) là rất vất vả, chưa kể đến việc version của các thư viện có thể conflig với nhau, việc import đầy đủ cácthư viện là cả một vấn đề. VD như bạn muốn sử dụng struts, thì điều bắt buộc là chúng ta phải import cả thư viện servlet.

2. Repository : Là nơi để chương trình maven download các dependency. Chi tiết tham khảo ở link :

https://maven.apache.org/guides/introduction/introduction-to-repositories.html 833

3. Install : Apache maven hiện nay đã được tích hợp sẵn vào trong Intellij IDEA.

# 

# CHƯƠNG III. PHÂN TÍCH VÀ THIẾT KẾ HỆ THỐNG

## 1. Khảo sát

Khảo sát là một phương pháp thu thập thông tin từ một số cá nhân (gọi là một mẫu) để tìm hiểu về sự phổ biến lớn hơn mà mẫu đó đưa ra. Mặc dù các cuộc khảo sát có nhiều dạng, phục vụ cho nhiều mục đích khác nhau, họ chia sẻ những đặc trưng nhất định.

Khảo sát trực tuyến là một phương pháp thu thập thông tin từ một số cá nhân (gọi là một mẫu) để tìm hiểu về sự phổ biến lớn hơn mà mẫu đó đưa ra. Mặc dù các cuộc khảo sát trực tuyến có nhiều dạng, phục vụ cho nhiều mục đích khác nhau, họ chia sẻ những đặc trưng nhất định. Để cho các mục tiêu của khảo sát được đáp ứng, các kết quả đưa ra phải đáng tin cậy đối với công chúng. Một mẫu có thể được lựa chọn một cách khoa học để mỗi cá nhân trong một quần thể đều có thể biết đến và lựa chọn. Điều này đảm bảo rằng một mẫu không được chọn tuỳ tiện hoặc chỉ sử dụng những người mong muốn tham gia. Kích cỡ mẫu cho một cuộc khảo sát sẽ phụ thuộc vào mức độ tin cậy cần thiết và các kết quả sẽ được sử dụng như thế nào. Một mẫu được lựa chọn đúng sẽ có thể phản ánh những đặc điểm khác nhau của một quần thể trong một biên độ lỗi rất nhỏ. Có rất nhiều cuộc khảo sát nghiên cứu tổng dân số người lớn nhưng nhiều người khác lại tập trung vào lựa chọn quần thể: nhân viên, các viện nghiên cứu, các trường học, người dùng máy tính, hoặc khách hàng sử dụng một sản phẩm hoặc dịch vụ cụ thể.

### 1.1. Khảo sát hiện trạng

Hook – mồi câu: Điều gì làm cho ý tưởng trò chơi của bạn tuyệt vời? Đối với tôi, đây là điều quan trọng nhất để viết ra. Một khi bạn nắm bắt được điều này, bạn có thể viết xuống ba điểm tiếp theo dễ dàng hơn nhiều. Trò chơi của bạn có phải là thứ kích thích tư duy không? Hay gây tranh cãi? Là nó có một kết cục bất ngờ? Hoặc, nó đang làm điều gì đó chưa bao giờ được thực hiện trước đây?

Mechanic – Cách chơi: Người chơi của bạn cần phải làm gì? Và vì mục đích gì? Đây chính là phần gameplay của bạn. Nó có thể đơn giản như cách nhấn QWOP để di chuyển trong trò chơi QWOP, hay bấm các nút để trò chuyện trong Mystic Messenger, tới hàng tấn combo wombo nút bấm trong Dwarf Fortress.

Story – Cốt chuyện: Người chơi nên nhớ về cốt chuyện của trò chơi như thế nào? Những cảm xúc nào họ nên có khi hoàn thành trò chơi của bạn? Mỗi trò chơi đều có một câu chuyện. Có thể là những con số trong 2048, hay xây dựng một quốc gia trong Civilization, hoặc những tương tác thầm lặng trong Monument Valley. Hãy nghĩ về câu chuyện sẽ được người chơi cảm nhận trong trò chơi của bạn.

Mood – cảm xúc: Trò chơi của bạn tạo ấn tượng gì? Hình ảnh? Âm thanh? Ấn tượng đầu tiên rất là quan trọng. Ấn tượng đầu tiên sẽ cuốn người chơi vào trò chơi. Có lẽ, bạn sẽ cung cấp cho trò chơi của bạn một sự rung cảm retro với đồ họa pixel và âm nhạc chiptune.

### 1.2 Các chức năng chính của game

1.2.1 Chơi game

Người chơi điều khiển một chú chim, cố gắng vượt qua các hàng ống màu xanh lá cây mà không chạm vào chúng. Mỗi khi vượt qua một ô thử thách sẽ được cộng thêm một điểm vào tổng điểm của người chơi.

1.2.2 Khởi động lại game:

Game sẽ dừng lại khi người chơi va vào nền hoặc ống màu xanh lá, khi đó người chơi ấn nút R để restart lại game.

Ngoài ra người chơi cũng có thể restart lại game ngay trong quá trình chơi mà không cần thiết phải chết mới restart lại được.

## 2. Đặc tả

Trò chơi hay ở mức độ khó khi điều khiển con chim nhảy lên bằng dấu spacebar và đợi cho con chim bay xuống ở mức nhất định mới điều khiển nó nhảy lên tiếp, nếu nhảy lên quá cao sẽ bị va vào ống.

Game đơn giản cho cả trẻ con chơi, và dân văn phòng trong những giờ nghỉ trưa chơi được. Không yêu cầu cốt truyện khi người chơi tham gia vào game.

### 2.1 Các class chính dự định thiết kế

2.1.1 App

Dùng để khởi động làm main class, màn hình chính của game, ứng dụng sẽ được khởi động từ class này.

2.1.2 Bird

Là một model trong game, model này đóng vai trò, số hóa hình ảnh con chim, để dễ dàng tính tọa độ của nhân vật khi di chuyển trong game.

2.1.3 Game

Là class chịu trách nhiệm monitor trong game, class này dựa vào các hành động của player để phân tích xử lý các hành động tiếp theo xảy ra trong game.

Ngoài ra game class cũng chịu trách nhiệm điều hành game theo đúng flow đã được thiết kế ra từ đầu.

2.1.4 GamePanel

GamePanel chịu trách nhiệm vẽ nên khung game, khung game này sẽ định hình tọa độ cũng như những thứ sẽ hiển thị trong khung game đó.

2.1.5 Keyboard

Nhiệm vụ chính của Keyboard class là listen các key được player ấn xuống, handle các key ấy ném ra các event như thế nào. Vd: khi key up thì sao, khi key down thì sao, khi player released một key sẽ thế nào.

2.1.6 Pipe

Tạo ra model Pipe nhằm mô phỏng ống nước, mô phỏng độ cao, tốc độ tới của ống nước và điều kiện va chạm của chim.

2.1.7 Render

Lớp Render mục đích để vẽ các lớp khác.

2.1.8 Util

Lớp Util để thực hiện việc load các image có sẵn.

## 3. Sơ đồ hoạt động hệ thống

Khi khởi động Flappy Bird, sẽ gọi vào class App đầu tiên, tiếp theo gọi đến lớp Keyboard, rồi gọi đến GamePanel.

## 4. Sơ đồ phân luồn chức năng

4.1 App

Method: main

Dùng để khởi động làm main class, màn hình chính của game, ứng dụng sẽ được khởi động từ class này.

4.2 Bird

Method: Bird, update, getRender.

Là một model trong game, model này đóng vai trò, số hóa hình ảnh con chim, để dễ dàng tính tọa độ của nhân vật khi di chuyển trong game.

4.3 Game

Method: Game, restart, update, getRenders, watchForStart, watchForPause, watchForReset, movePipes, checkForCollisions.

Là class chịu trách nhiệm monitor trong game, class này dựa vào các hành động của player để phân tích xử lý các hành động tiếp theo xảy ra trong game.

Ngoài ra game class cũng chịu trách nhiệm điều hành game theo đúng flow đã được thiết kế ra từ đầu

4.4 GamePanel

Method: GamePanel, update, paintComponent, run.

GamePanel chịu trách nhiệm vẽ nên khung game, khung game này sẽ định hình tọa độ cũng như những thứ sẽ hiển thị trong khung game đó.

4.5 Keyboard

Method: Keyboard, getInstance, keyPressed, keyReleased, keyTyped, isDown.

Nhiệm vụ chính của Keyboard class là listen các key được player ấn xuống, handle các key ấy ném ra các event như thế nào. Vd: khi key up thì sao, khi key down thì sao, khi player released một key sẽ thế nào.

4.6 Pipe

Method: Pipe, reset, update, collides, getRender,

Tạo ra model Pipe nhằm mô phỏng ống nước, mô phỏng độ cao, tốc độ tới của ống nước và điều kiện va chạm của chim.

4.7 Render

Method: Render, Render (overload)

Lớp Render mục đích để vẽ các lớp khác.

4.8 Util

Method: loadImage.

Lớp Util để thực hiện việc load các image có sẵn.

## 5. Phân tích thao tác người dùng

* Khởi động game.
* Ấn Spacebar bắt đầu game
* Ấn spacebar khi chơi game
* Ấn R khi chết.
* Tắt game.

# Chương IV. CHI TIẾT HỆ THỐNG

## 1. Một số mã nguồn

### 1.1. App

package flappybird.app.code;  
  
import javax.swing.\*;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class App {  
  
 public static int *WIDTH* = 500;  
 public static int *HEIGHT* = 510;  
  
 public static void main(String[] args) {  
 JFrame frame = new JFrame();  
 frame.setVisible(true);  
 frame.setDefaultCloseOperation(JFrame.*EXIT\_ON\_CLOSE*);  
 frame.setSize(*WIDTH*, *HEIGHT*);  
 frame.setLocationRelativeTo(null);  
  
 Keyboard keyboard = Keyboard.*getInstance*();  
 frame.addKeyListener(keyboard);  
  
 GamePanel panel = new GamePanel();  
 frame.add(panel);  
 }  
}

### 1.2. Bird

package flappybird.app.code;  
  
import java.awt.Image;  
import java.awt.event.KeyEvent;  
import java.awt.geom.AffineTransform;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Bird {  
  
 public int x;  
 public int y;  
 public int width;  
 public int height;  
  
 public boolean dead;  
  
 public double yvel;  
 public double gravity;  
  
 private int jumpDelay;  
 private double rotation;  
  
 private Image image;  
 private Keyboard keyboard;  
  
 public Bird() {  
 x = 100;  
 y = 150;  
 yvel = 0;  
 width = 45;  
 height = 32;  
 gravity = 0.5;  
 jumpDelay = 0;  
 rotation = 0.0;  
 dead = false;  
  
 keyboard = Keyboard.*getInstance*();  
 }  
  
 public void update() {  
 yvel += gravity;  
  
 if (jumpDelay > 0)  
 jumpDelay--;  
  
 if (!dead && keyboard.isDown(KeyEvent.*VK\_SPACE*) && jumpDelay <= 0) {  
 yvel = -10;  
 jumpDelay = 10;  
 }  
  
 y += (int)yvel;  
 }  
  
 public Render getRender() {  
 Render r = new Render();  
 r.x = x;  
 r.y = y;  
  
 if (image == null) {  
 image = Util.*loadImage*("F:\\Project I\_Flappy\_Bird\\app\\src\\main\\resources\\lib\\bird.png");  
 }  
 r.image = image;  
  
 rotation = (90 \* (yvel + 20) / 20) - 90;  
 rotation = rotation \* Math.*PI* / 180;  
  
 if (rotation > Math.*PI* / 2)  
 rotation = Math.*PI* / 2;  
  
 r.transform = new AffineTransform();  
 r.transform.translate(x + width / 2, y + height / 2);  
 r.transform.rotate(rotation);  
 r.transform.translate(-width / 2, -height / 2);  
  
 return r;  
 }  
}

### 1.3. Game

package flappybird.app.code;  
  
import java.awt.event.KeyEvent;  
import java.util.ArrayList;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Game {  
  
 public static final int *PIPE\_DELAY* = 100;  
  
 private Boolean paused;  
  
 private int pauseDelay;  
 private int restartDelay;  
 private int pipeDelay;  
  
 private Bird bird;  
 private ArrayList<Pipe> pipes;  
 private Keyboard keyboard;  
  
  
 public int score;  
 public Boolean gameover;  
 public Boolean started;  
  
 public Game() {  
 keyboard = Keyboard.*getInstance*();  
 restart();  
 }  
  
 public void restart() {  
 paused = false;  
 started = false;  
 gameover = false;  
  
 score = 0;  
 pauseDelay = 0;  
 restartDelay = 0;  
 pipeDelay = 0;  
  
 bird = new Bird();  
 pipes = new ArrayList<>();  
 }  
  
 public void update() {  
 watchForStart();  
  
 if (!started)  
 return;  
  
 watchForPause();  
 watchForReset();  
  
 if (paused)  
 return;  
  
 bird.update();  
  
 if (gameover)  
 return;  
  
 movePipes();  
 checkForCollisions();  
 }  
  
 public ArrayList<Render> getRenders() {  
 ArrayList<Render> renders = new ArrayList<Render>();  
 renders.add(new Render(0, 0, "F:\\Project I\_Flappy\_Bird\\app\\src\\main\\resources\\lib\\background.png"));  
 for (Pipe pipe : pipes)  
 renders.add(pipe.getRender());  
 renders.add(new Render(0, 0, "F:\\Project I\_Flappy\_Bird\\app\\src\\main\\resources\\lib\\foreground.png"));  
 renders.add(bird.getRender());  
 return renders;  
 }  
  
 private void watchForStart() {  
 if (!started && keyboard.isDown(KeyEvent.*VK\_SPACE*)) {  
 started = true;  
 }  
 }  
  
 private void watchForPause() {  
 if (pauseDelay > 0)  
 pauseDelay--;  
  
 if (keyboard.isDown(KeyEvent.*VK\_P*) && pauseDelay <= 0) {  
 paused = !paused;  
 pauseDelay = 10;  
 }  
 }  
  
 private void watchForReset() {  
 if (restartDelay > 0)  
 restartDelay--;  
  
 if (keyboard.isDown(KeyEvent.*VK\_R*) && restartDelay <= 0) {  
 restart();  
 restartDelay = 10;  
 return;  
 }  
 }  
  
 private void movePipes() {  
 pipeDelay--;  
  
 if (pipeDelay < 0) {  
 pipeDelay = *PIPE\_DELAY*;  
 Pipe northPipe = null;  
 Pipe southPipe = null;  
  
 // Look for pipes off the screen  
 for (Pipe pipe : pipes) {  
 if (pipe.x - pipe.width < 0) {  
 if (northPipe == null) {  
 northPipe = pipe;  
 } else if (southPipe == null) {  
 southPipe = pipe;  
 break;  
 }  
 }  
 }  
  
 if (northPipe == null) {  
 Pipe pipe = new Pipe("north");  
 pipes.add(pipe);  
 northPipe = pipe;  
 } else {  
 northPipe.reset();  
 }  
  
 if (southPipe == null) {  
 Pipe pipe = new Pipe("south");  
 pipes.add(pipe);  
 southPipe = pipe;  
 } else {  
 southPipe.reset();  
 }  
  
 northPipe.y = southPipe.y + southPipe.height + 175;  
 }  
  
 for (Pipe pipe : pipes) {  
 pipe.update();  
 }  
 }  
  
 private void checkForCollisions() {  
  
 for (Pipe pipe : pipes) {  
 if (pipe.collides(bird.x, bird.y, bird.width, bird.height)) {  
 gameover = true;  
 bird.dead = true;  
 } else if (pipe.x == bird.x && pipe.orientation.equalsIgnoreCase("south")) {  
 score++;  
 }  
 }  
  
 // Ground + Bird collision  
 if (bird.y + bird.height > App.*HEIGHT* - 80) {  
 gameover = true;  
 bird.y = App.*HEIGHT* - 80 - bird.height;  
 }  
 }  
}

### 1.4. GamePanel

package flappybird.app.code;  
  
import javax.swing.\*;  
import java.awt.\*;  
import java.awt.event.\*;  
import java.awt.geom.\*;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class GamePanel extends JPanel implements Runnable {  
  
 private Game game;  
  
 public GamePanel() {  
 game = new Game();  
 new Thread(this).start();  
 }  
  
 public void update() {  
 game.update();  
 repaint();  
 }  
  
 protected void paintComponent(Graphics g) {  
 super.paintComponent(g);  
  
 Graphics2D g2D = (Graphics2D) g;  
 for (Render r : game.getRenders())  
 if (r.transform != null)  
 g2D.drawImage(r.image, r.transform, null);  
 else  
 g.drawImage(r.image, r.x, r.y, null);  
  
  
 g2D.setColor(Color.*BLACK*);  
  
 if (!game.started) {  
 g2D.setFont(new Font("TimesRoman", Font.*PLAIN*, 20));  
 g2D.drawString("Press SPACE to start", 150, 240);  
 } else {  
 g2D.setFont(new Font("TimesRoman", Font.*PLAIN*, 24));  
 g2D.drawString(Integer.*toString*(game.score), 10, 465);  
 }  
  
 if (game.gameover) {  
 g2D.setFont(new Font("TimesRoman", Font.*PLAIN*, 20));  
 g2D.drawString("Press R to restart", 150, 240);  
 }  
 }  
  
 public void run() {  
 try {  
 while (true) {  
 update();  
 Thread.*sleep*(25);  
 }  
 }  
 catch (Exception e) {  
 e.printStackTrace();  
 }  
 }  
}

### 1.5. Keyboard

package flappybird.app.code;  
  
import java.awt.event.KeyEvent;  
import java.awt.event.KeyListener;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Keyboard implements KeyListener {  
  
 private static Keyboard *instance*;  
  
 private boolean[] keys;  
   
 private Keyboard() {  
 keys = new boolean[256];  
 }  
  
 public static Keyboard getInstance() {  
  
 if (*instance* == null) {  
 *instance* = new Keyboard();  
 }  
   
 return *instance*;  
 }  
  
 public void keyPressed(KeyEvent e) {  
 if (e.getKeyCode() >= 0 && e.getKeyCode() < keys.length) {  
 keys[e.getKeyCode()] = true;  
 }  
 }  
   
 public void keyReleased(KeyEvent e) {  
 if (e.getKeyCode() >= 0 && e.getKeyCode() < keys.length) {  
 keys[e.getKeyCode()] = false;  
 }  
 }  
  
 public void keyTyped(KeyEvent e) {}  
  
 public boolean isDown(int key) {  
  
 if (key >= 0 && key < keys.length) {  
 return keys[key];  
 }  
   
 return false;  
 }  
}

### 1.6. Pipe

package flappybird.app.code;  
  
import java.awt.Image;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Pipe {  
  
 public int x;  
 public int y;  
 public int width;  
 public int height;  
 public int speed = 3;  
  
 public String orientation;  
  
 private Image image;  
  
 public Pipe(String orientation) {  
 this.orientation = orientation;  
 reset();  
 }  
  
 public void reset() {  
 width = 66;  
 height = 400;  
 x = App.*WIDTH* + 2;  
  
 if (orientation.equals("south")) {  
 y = -(int)(Math.*random*() \* 120) - height / 2;  
 }  
 }  
  
 public void update() {  
 x -= speed;  
 }  
  
 public boolean collides(int \_x, int \_y, int \_width, int \_height) {  
  
 int margin = 2;  
  
 if (\_x + \_width - margin > x && \_x + margin < x + width) {  
  
 if (orientation.equals("south") && \_y < y + height) {  
 return true;  
 } else if (orientation.equals("north") && \_y + \_height > y) {  
 return true;  
 }  
 }  
  
 return false;  
 }  
  
 public Render getRender() {  
 Render r = new Render();  
 r.x = x;  
 r.y = y;  
  
 if (image == null) {  
 image = Util.*loadImage*("F:\\Project I\_Flappy\_Bird\\app\\src\\main\\resources\\lib\\pipe-" + orientation + ".png");  
 }  
 r.image = image;  
  
 return r;  
 }  
}

### 1.7. Render

package flappybird.app.code;  
  
import java.awt.Image;  
import java.awt.geom.AffineTransform;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Render {  
 public int x;  
 public int y;  
 public Image image;  
 public AffineTransform transform;  
  
 public Render() {  
 }  
  
 public Render(int x, int y, String imagePath) {  
 this.x = x;  
 this.y = y;  
 this.image = Util.*loadImage*(imagePath);  
 }  
}

### 1.8. Util

package flappybird.app.code;  
  
import java.awt.Image;  
import javax.imageio.ImageIO;  
import java.io.File;  
import java.io.IOException;  
import java.util.HashMap;  
  
*/\*\*  
 \* Flappy Bird replicate is simple a under developing program.  
 \* Hope you will enjoy this game.  
 \*  
 \** ***@author*** *Nguyễn Tiến Tú - HUBT - 15150064  
 \** ***@version*** *1.0  
 \** ***@since*** *6/2019  
 \*/*public class Util {  
  
 private static HashMap<String, Image> *cache* = new HashMap<String, Image>();  
  
 public static Image loadImage(String path) {  
 Image image = null;  
  
 if (*cache*.containsKey(path)) {  
 return *cache*.get(path);  
 }  
  
 try {  
 image = ImageIO.*read*(new File(path));  
  
 if (!*cache*.containsKey(path)) {  
 *cache*.put(path, image);  
 }  
 }   
 catch (IOException e) {  
 e.printStackTrace();  
 }  
  
 return image;  
 }  
}

# CHƯƠNG V. TỔNG KẾT

## 1. Đánh giá kết quả thực hiện

- Sau khi hoàn thành Game Flappy Bird bản thân em đã học hỏi thêm rất nhiều kiến thức về lập trình(hướng đối tượng).

- Tuy nhiên Game Flappy Bird (ver 1.0) vẫn còn nhiều hạn chế về giao diện chơi và cách thức chơi.

- Trong quá trình thực hiện em đã tham khảo nhiều nhờ thầy cô và các website trên mạng, có một vài kiến thức em vẫn chưa thật sự hiểu nhưng cũng đã làm được nhờ sự hỗ trợ.

## 2. Hướng phát triển tiếp theo

- Đối với các phiên bản sau, em sẽ thêm các chức năng thú vị hơn ví dụ như: Điểm số cao nhất, bấm quay lại chơi tiếp, Play Game(khi người dùng bận), Pause Game (tiếp tục Game khi người dùng quay lại)….

# LỜI KẾT

Các thầy(cô) đã truyền đạt những kiến thức và những kinh nghiệm cùng với sự nỗ lực phấn đấu của em, em đã hoàn thành đồ án tốt nghiệp này. Từ việc quan sát phân tích hệ thống, phân tích chức năng, thiết kế và cài đặt giao diện cho đến công việc lập trình biên soạn các ngôn ngữ mã nguồn đều được sự hướng dẫn của các thầy cô trong khoa Công nghệ thông tin và các sự trợ giúp của các trang website học lập trình online.

Tuy nhiên, trong thời gian làm đồ án tốt nghiệp, do thời gian và trình độ kiến thức vẫn còn nhiều hạn chế nên đồ án tốt nghiệp này của Em không tránh khỏi sai sót. Em rất mong được sự hướng dẫn chỉ bảo thêm của toàn thể các thầy ( cô ) giáo trong khoa Công Nghệ Thông Tin để đồ án tốt nghiệp của Em ngày càng hoàn thiện hơn.

# TÀI LIỆU THAM KHẢO

Phân tích yêu cầu: <https://vi.wikipedia.org>

Video game: <https://vi.wikipedia.org/>

JDK: https://o7planning.org/

<https://tech.vccloud.vn/>

Flappy Bird: <https://vi.wikipedia.org/wiki/Flappy_Bird>

<https://www.w3schools.com/>

<http://hoclaptrinhweb.org/>

<http://hocwebgiare.com/>

<https://freetuts.net/hoc-php>

<https://www.draw.io/>