**TRƯỜNG ĐẠI HỌC KIẾN TRÚC HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

------- \*\*\* -------

**ĐỒ ÁN MÔN HỌC CÔNG NGHỆ JAVA**

**ĐỀ TÀI: NGHIÊN CỨU VÀ XÂY DỰNG TRÒ CHƠI CARO**

NHÓM SINH VIÊN THỰC HIỆN:

Lê Việt Anh MSV:2155010007

Vũ Việt Dũng

**Giảng viên hướng dẫn:**Th.S Nguyễn Hồng Thanh

Hà Nội, tháng 1 năm 2024

MỤC LỤC

LỜI MỞ ĐẦU 3

[**CHƯƠNG 1:CƠ SỞ LÍ THUYẾT** 4](#_Toc155212404)

[1.1 Tổng quan về Java 4](#_Toc155212405)

[1.1.1 Khái niệm 4](#_Toc155212406)

[1.1.2 Đặc điểm ngôn ngữ lập trình Java 5](#_Toc155212407)

[1.1.3 Các phiên bản của Java 6](#_Toc155212408)

[1.1.4 Các thành phần của Java 6](#_Toc155212409)

[1.1.5 Lợi thế và hạn chế 8](#_Toc155212411)

[1.2 Tổng quan về đề tài 9](#_Toc155212413)

[1.2.1 Tổng quan về cờ caro 9](#_Toc155212406)

[1.2.2 Giao diện 9](#_Toc155212407)

[1.2.3 Cách chơi 10](#_Toc155212408)

[**CHƯƠNG 2:XÂY DỰNG VÀ TRIỂN KHAI ỨNG DỤNG** 12](#_Toc155212414)

[2.1 Xây dựng ứng dụng 13](#_Toc155212415)

[2.1.1 Giao diện tương tác với người dùng 13](#_Toc155212415)

[2.1.1.1 Gói Java AWT 14](#_Toc155212416)

[2.1.1.2 Gói Java Swing 15](#_Toc155212416)

[2.2 Thực thi và cài đặt 16](#_Toc155212417)

[2.2.1 Sơ đồ lớp 16](#_Toc155212418)

[2.2.2 Công nghệ được áp dụng 18](#_Toc155212419)

[**Kết luận** 21](#_Toc155212423)

LỜI MỞ ĐẦU

Java là một công nghệ xây dựng các ứng dụng phần mềm có vị trí rất lớn trong những năm cuối thế kỉ 20, đầu thế kỉ 21. Nó được coi là công nghệ mang tính cách mạng và khả thi nhất trong việc tạo ra các ứng dụng có khả năng chạy thống nhất trên nhiều nền tảng mà chỉ cần biên dịch một lần.

Lần đầu tiên xuất hiện vào năm 1992 như là một ngôn ngữ dùng trong nội bộ tập đoàn Sun Microsystems để xây dựng ứng dụng điều khiển các bộ xử lý bên trong máy điện thoại cầm tay, lò vi sóng, các thiết bị điện tử dân dụng khác. Không chỉ là một ngôn ngữ, Java còn là một nền tảng phát triển và triển khai ứng dụng trong đó máy ảo Java, bộ thông dịch có vai trò trung tâm.

Sun, công ty đã phát minh ra ngôn ngữ Java, chính thức ban hành bản Java Development Kit 1.0 vào năm 1996 hoàn toàn miễn phí để các nhà phát triển có thể tải về, học Java, xây dựng các ứng dụng Java và triển khai chúng trên các hệ điều hành có hỗ trợ Java. Khi nhắc đến Java người ta không còn chỉ nhắc đến Java như là một ngôn ngữ mà nhắc đến Java như là một công nghệ hay một nền tảng phát triển.

Ứng dụng Game cờ Caro được xây dựng nhằm mục đích làm quen với ngôn ngữ, các câu lệnh cơ bản và một số công nghệ hướng đối tượng tiên tiến.

Ứng dụng còn có mục đích chính nhằm giải trí nhẹ nhàng cho những ai đang căng thẳng.

Chúng em hy vọng rằng báo cáo này sẽ mang lại cho quý thầy/cô và các bạn thích thú và đem lại một cái nhìn sâu rộng về sức mạnh của lập trình Java trong việc phát triển trò chơi và ứng dụng đa năng.

Cuối cùng. chúng em xin chân thành cảm ơn thầy Nguyễn Hồng Thanh - giảng viên bộ môn Công nghệ Java đã dành thời gian và sự hướng dẫn để chúng em có thể hoàn thành đồ án này. Chúng em hy vọng rằng báo cáo này sẽ đáp ứng được sự mong đợi của thầy và sẽ góp phần nhỏ vào sự nghiên cứu và phát triển trong lĩnh vực nền tảng lập trình Java.

Trân trọng cảm ơn!

**CHƯƠNG 1: CƠ SỞ LÝ THUYẾT**

* 1. **Tổng quan về Java**
     1. **Khái niệm về Java**

Java là một trong những ngôn ngữ lập trình hướng đối tượng. Nó được sử dụng trong phát triển phần mềm, trang web, game hay ứng dụng trên các thiết bị di động. Java được khởi đầu bởi James Gosling và bạn đồng nghiệp ở Sun MicroSystem năm 1991. Ban đầu Java được tạo ra nhằm mục đích viết phần mềm cho các sản phẩm gia dụng, và có tên là Oak. Java được phát hành năm 1994, đến năm 2010 được Oracle mua lại từ Sun MicroSystem.

* + 1. **Đặc điểm ngôn ngữ lập trình Java**

**Tương tự C++, hướng đối tượng hoàn toàn**

Trong quá trình tạo ra một ngôn ngữ mới phục vụ cho mục đích chạy được trên nhiều nền tảng, các kỹ sư của Sun MicroSystem muốn tạo ra một ngôn ngữ dễ học và quen thuộc với đa số người lập trình. Vì vậy họ đã sử dụng lại các cú pháp của C và C++.

Tuy nhiên, trong Java thao tác với con trỏ bị lược bỏ nhằm đảo bảo tính an toàn và dễ sử dụng hơn. Các thao tác overload, goto hay các cấu trúc như struct và union cũng được loại bỏ khỏi Java.

**Độc lập phần cứng và hệ điều hành**

Một chương trình viết bằng ngôn ngữ Java có thể chạy tốt ở nhiều môi trường khác nhau. Gọi là khả năng “cross-platform”. Khả năng độc lập phần cứng và hệ điều hành được thể hiện ở 2 cấp độ là cấp độ mã nguồn và cấp độ nhị phân.

Ở cấp độ mã nguồn: Kiểu dữ liệu trong Java nhất quán cho tất cả các hệ điều hành và phần cứng khác nhau. Java có riêng một bộ thư viện để hỗ trợ vấn đề này. Chương trình viết bằng ngôn ngữ Java có thể biên dịch trên nhiều loại máy khác nhau mà không gặp lỗi.

Ở cấp độ nhị phân: Một mã biên dịch có thể chạy trên nhiều nền tảng khác nhau mà không cần dịch lại mã nguồn. Tuy nhiên cần có Java Virtual Machine để thông dịch đoạn mã này.

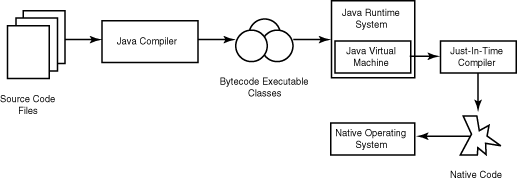
**Ngôn ngữ thông dịch**

Ngôn ngữ lập trình thường được chia ra làm 2 loại (tùy theo các hiện thực hóa ngôn ngữ đó) là ngôn ngữ thông dịch và ngôn ngữ biên dịch.

* Thông dịch (Interpreter) : Nó dịch từng lệnh rồi chạy từng lệnh, lần sau muốn chạy lại thì phải dịch lại.
* Biên dịch (Compiler): Code sau khi được biên dịch sẽ tạo ra 1 file thường là .exe, và file .exe này có thể đem sử dụng lại không cần biên dịch nữa.

Ngôn ngữ lập trình Java thuộc loại ngôn ngữ thông dịch. Chính xác hơn, Java là loại ngôn ngữ vừa biên dịch vừa thông dịch. Cụ thể như sau

Khi viết mã, hệ thống tạo ra một tệp .java. Khi biên dịch mã nguồn của chương trình sẽ được biên dịch ra mã byte code. Máy ảo Java (Java Virtual Machine) sẽ thông dịch mã byte code này thành machine code  (hay native code) khi nhận được yêu cầu chạy chương trình.



Ưu điểm : Phương pháp này giúp các đoạn mã viết bằng Java có thể chạy được trên nhiều nền tảng khác nhau. Với điều kiện là JVM có hỗ trợ chạy trên nền tảng này.

Nhược điểm : Cũng như các ngôn ngữ thông dịch khác, quá trình chạy các đoạn mã Java là chậm hơn các ngôn ngữ biên dịch khác (tuy nhiên vẫn ở trong một mức chấp nhận được).

**Cơ chế thu gom rác tự động**

Khi tạo ra các đối tượng trong Java, JRE sẽ tự động cấp phát không gian bộ nhớ cho các đối tượng ở trên heap.

Với ngôn ngữ như C C++, bạn sẽ phải yêu cầu hủy vùng nhớ mà bạn đã  cấp phát, để tránh việc thất thoát vùng nhớ. Tuy nhiên vì một lý do nào đó, bạn không hủy một vài vùng nhớ, dẫn đến việc thất thoát và làm giảm hiệu năng chương trình.

Ngôn ngữ lập trình Java hỗ trợ cho bạn điều đó, nghĩa là bạn không phải  tự gọi hủy các vùng nhớ. Bộ thu dọn rác của Java sẽ theo vết các tài nguyên đã được cấp. Khi không có tham chiếu nào đến vùng nhớ, bộ thu dọn rác sẽ tiến hành thu hồi vùng nhớ đã được cấp phát.

**Đa luồng**

Java hỗ trợ lập trình đa tiến trình (multithread) để thực thi các công việc đồng thời. Đồng thời cũng cung cấp giải pháp đồng bộ giữa các tiến trình (giải pháp sử dụng priority…).

**Tính an toàn và bảo mật**

* Tính an toàn

Ngôn ngữ lập trình Java yêu cầu chặt chẽ về kiểu dữ liệu.

Dữ liệu phải được khai báo tường minh.

Không sử dụng con trỏ và các phép toán với con trỏ.

Java kiểm soát chặt chẽ việc truy nhập đến mảng, chuỗi. Không cho phép sử dụng các kỹ thuật tràn. Do đó các truy nhập sẽ không vượt quá kích thước của mảng hoặc chuỗi.

Quá trình cấp phát và giải phóng bộ nhớ được thực hiện tự động.

Cơ chế xử lý lỗi giúp việc xử lý và phục hồi lỗi dễ dàng hơn.

* Tính bảo mật

Java cung cấp một môi trường quản lý chương trình với nhiều mức khác nhau.

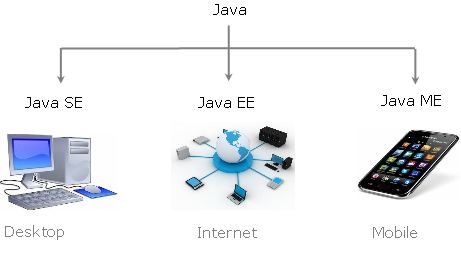
Mức 1 : Chỉ có thể truy xuất dữ liệu cũng như phương phức thông qua giao diện mà lớp cung cấp.

Mức 2 : Trình biên dịch kiểm soát các đoạn mã sao cho tuân thủ các quy tắc của ngôn ngữ lập trình Java trước khi thông dịch.

Mức 3 : Trình thông dịch sẽ kiểm tra mã byte code xem các đoạn mã này có đảm bảo được các quy định, quy tắc trước khi thực thi.

Mức 4: Java kiểm soát việc nạp các lớp vào bộ nhớ để giám sát việc vi phạm giới hạn truy xuất trước khi nạp vào hệ thống.

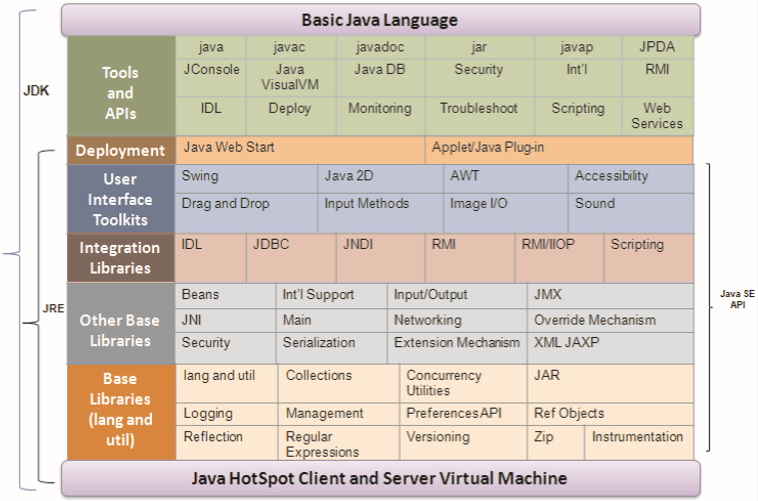
* + 1. **Các phiên bản của Java**

1. 
2. **Java Standard Edition (Java SE)** – Là một nền tảng cơ bản cho phép phát triển giao diện điều khiển, các ứng dụng mạng và các ứng dụng dạng Win Form.
3. **Java Enterprise Edition (Java EE)** – Được xây dựng trên nền tảng Java SE, giúp phát triển các ứng dụng web, các ứng dụng ở cấp doanh nghiệp, …
4. **Java Mobile Edition (Java ME)** – Là một nền tảng cho phép phát triển các ứng dụng nhúng vào các thiết bị điện tử như mobile,…
   * 1. **Các thành phần của Java**

**Các thành phần của Java SE Platform**

Gồm 2 thành phần:

* **JRE (Java Runtime Environment)**: cung cấp *JVM  (Java Virtual Machine)*và thư viện được sử dụng để chạy chương trình Java.
* **JDK (Java Development Kit)***:*được biết đến như bộ cung cụ phát triển Java, bao gồm: trình biên dịch và trình gỡ rối được sử dụng để phát triển các ứng dụng Java.



* + 1. **Lợi thế và hạn chế của Java**

**Lợi thế**

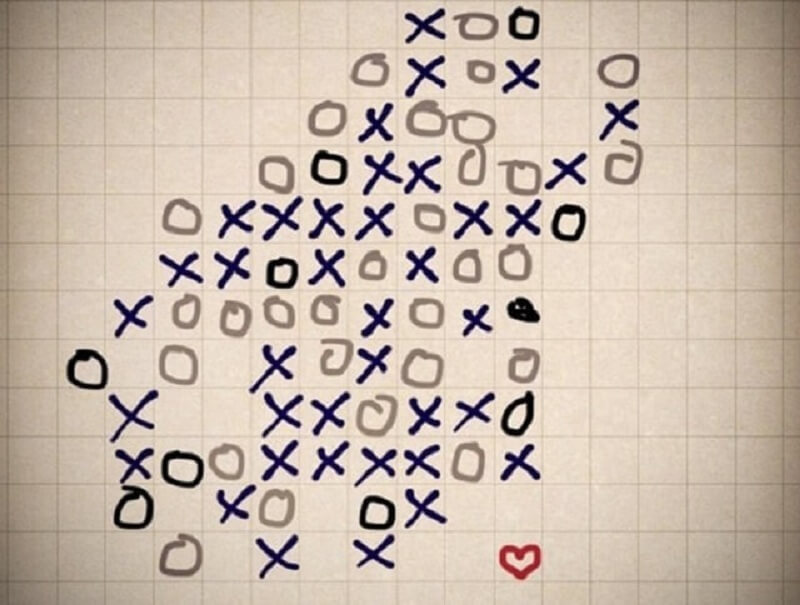
* Cho phép xử lí nhiều thành phần dữ liệu trong cùng 1 thời điểm.
* Bộ nhớ chỉ được cấp cho mảng khi mảng thực sự được sử dụng.
* **Truy cập ngẫu nhiên**: chúng ta có thể lấy bất cứ dữ liệu nào ở tại bất cứ vị trí chỉ mục nào.

**Hạn chế**

* **Giới hạn kích cỡ**: Kích thước mảng cố định không thay đổi. Nó không tăng kích thước của nó tại runtime. Để xử lý vấn đề này, **Collection Framework** được sử dụng trong Java.
  + Các phần tử của một mảng là được đặt và tham chiếu liên tiếp nhau trong bộ nhớ, điều đó là khó khăn khi ta cố tình bỏ đi một phần tử nào đó trong mảng, nó mất tính liên tiếp. Thông thường một kỹ thuật mà thường sử dụng là tạo một mảng mới lưu trữ các đối tượng của mảng ban đầu và bỏ đi các phần tử không cần thiết, tuy nhiên điều này làm giảm hiệu năng của chương trình.
  + Với trường hợp mở rộng mảng cũng với kỹ thuật tương tự là khởi tạo một mảng mới với kích cỡ lớn hơn sau đó thì copy các phần tử mảng cũ cho mảng mới .
* Chỉ lưu trữ được duy nhất cùng một kiểu dữ liệu trong mảng
  1. **Tổng quan về đề tài**

**1.2.1 Tổng quan về game cờ Caro**

Game cờ caro hay còn được biết đến với tên gọi là chơi XO. Đây được xem là một trò chơi dân gian và xuất hiện từ lâu đời tại Việt Nam. Biết bao nhiêu thế hệ người dân Việt Nam đều đã chơi loại cờ này. Đặc biệt là ở độ tuổi học sinh.



Game cờ caro

Hiện tại, về cơ bản thì với game cờ caro, người chơi sẽ chỉ cần làm sao ghi được 4 nước liên tiếp không bị chặn ở đầu nào theo hàng ngang, hàng dọc hoặc hàng chéo là có thể giành chiến thắng. Nghe có vẻ đơn giản thế nhưng, game cờ caro lại là những màn đấu trí vô cùng căng thẳng. Ngày nay, game cờ caro cũng có khá nhiều biến thể khác nhau. Điều này nhằm tạo sự phù hợp với người chơi ở từng quốc gia, khu vực hay vùng miền khác nhau. Và ở kỷ nguyên 4.0 thì sẽ không thể thiếu được game cờ caro online tiện lợi và hấp dẫn.

**1.2.2 Giao diện**

1. Thuận tiện: các nút chức năng phải dễ nhìn ,dễ sử dụng.
2. Hình ảnh: ảnh nền,ảnh bàn cờ ,ảnh quân cờ,ảnh nút,….đơn giản, dễ nhìn.
3. Sinh động:có âm thanh nước đi trong khi chơi.
4. Giao diện:
   1. Giao diện chính:

* Hiện bàn cờ :bàn cờ gồm 225 ô chia làm 15 hàng dọc và 15 hàng ngang( mỗi hàng gồm 15 ô vuông liên tiếp nhau, thẳng hàng 1 ô vuông = 30 px. )
* Bên cạnh giao diện chính là bàn cờ, cần hiển thị các chức năng để xử lí chương trình (ví dụ: 1 thanh Menu công cụ bao gồm tất cả các chức năng xử lí chương trình), hiển thị về thời gian chơi, hiển thị tỷ số ván thắng của cả 2 bên, hiển thị tọa độ các nước đã đi của hai bên.
  1. Giao diện về bảng “Chức năng”: hiển thị các chữ viết và hình ảnh: hỏi xem người chơi có muốn chọn âm thanh, người chơi muốn đi trước hay để máy đi trước, người chơi muốn chọn loại quân cờ nào và chọn kinh nghiệm máy khi chơi.
  2. Giao diện bảng “Hướng dẫn chơi”: đưa ra luật chơi bẳng cả chữ viết và hình ảnh hướng dẫn người chơi.
  3. Giao diện bảng “Ghi danh”: đưa ra thông tin vê người chơi chiến thắng máy: tên người chơi, số nước đi người chơi và thời gian chơi, kinh nghiệm chơi của máy.

**1.2.3 Cách chơi**

**Luật cơ bản của game cờ caro**

Bởi vì cờ caro có sự phổ biến ở toàn châu Á nên tại mỗi quốc gia sẽ có những luật cụ thể riêng tùy theo quan niệm của họ. Những thông tin được chia sẻ sau đây sẽ là những điều mà áp dụng với game cờ caro tại Việt Nam.

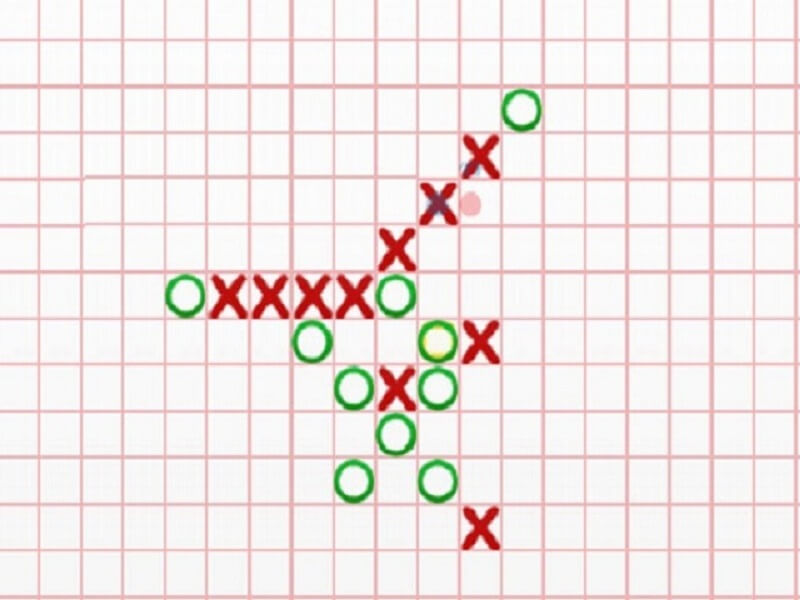
Ở nước ta, game cờ caro sẽ được thể hiện dưới 2 ký tự tiêu biểu là X và O. Chính vì thế mà game này còn được gọi là chơi XO. Sẽ có 2 người chơi với nhau, một người đại diện là quân X và người còn lại sẽ là quân O. Thông thường, quân X sẽ được đi trước.

Cách chơi game cờ caro

Với tựa game này, bạn có thể đánh ở bất kỳ đâu với việc vẽ bàn cờ là những ô vuông. Có thể là trên giấy, trên bảng hay bất kỳ mặt phẳng nào cũng được. Hiện nay thì chúng ta có thể hoàn toàn chơi game cờ caro online trên máy tính bảng hay điện thoại di động, đều rất tiện lợi.

Luật chơi và cách chơi cơ bản của game cờ caro khá đơn giản. Bạn chỉ cần đánh bất kỳ vào ô nào trên bàn cờ đều được. Lần lượt từng người đưa ra nước cờ của mình. Tuy nhiên, bạn cần tính toán làm sao để có thể tạo được 4 nước trắng liền kề, không bị chặn ở bất kỳ đầu nào theo hàng ngang, dọc hay chéo đều được là có thể giành chiến thắng. Nếu bị chặn thì bạn sẽ phải tiếp tục ván cờ của mình.

Trường hợp bị chặn 1 đầu thì bạn sẽ cần có 5 quân cờ liên tiếp để chiến thắng, còn nếu chặn 2 đầu thì bạn cần tìm hướng đi mới cho mình. Với game cờ caro offline thì ta sẽ có luật phân thắng bại là 5 đen hơn 4 trắng. Tức là nếu bạn có 5 nước bị chặn 1 dầu thì bạn sẽ là người giành chiến thắng chứ không phải là người có chiến thắng là 4 nước trắng không bị chặn và đánh thắng trước.

Luật chơi

Đối với cờ caro online thì khi chơi trên điện thoại hay máy tính bảng thì mỗi người chơi sẽ có 20 giây để suy nghĩ và đưa ra nước cờ cho mình. Nếu hết thời gian suy nghĩ mà bạn vẫn chưa quyết định thì hệ thống sẽ mặc định bạn thua trong bàn đó.

**CHƯƠNG 2:XÂY DỰNG VÀ TRIỂN KHAI ỨNG DỤNG**

* 1. **Xây dựng ứng dụng**
     1. **Giao diện tương tác với người dùng**

Các ứng dụng phần mềm hiện nay rất thân thiện vì được trình bày nhiều màn hình giao diện đồ họa đẹp mắt. Các ngôn ngữ lập trình hiện nay cung cấp các đối tượng đồ họa, chúng có thể được điều khiển bởi người lập trình, hay bởi người sử dụng. Một trong số những kết quả quan trọng nhất chính là các ngôn ngữ hiện nay được dựa trên Giao diện người dùng đồ họa (Graphical User Interface - GUI).

  GUI cung cấp chức năng nhập liệu theo cách thân thiện với người dùng. GUI đa dạng từ ứng dụng đến ứng dụng và có thể chứa nhiều điều khiển như hộp văn bản, nhãn, hộp danh sách hay các điều khiển khác. Các ngôn ngữ lập trình khác nhau cung cấp nhiều cách khác nhau để tạo GUI. Các ngôn ngữ như VB hay VC++ có thể cung cấp chức năng kéo và thả trong khi đó phần mềm giống như C++ yêu cầu người lập trình phải viết toàn bộ mã để xây dựng GUI.

Một phần tử (element) GUI được thiết lập bằng cách sử dụng thủ tục sau:

* Tạo đối tượng
* Xác định sự xuất hiện ban đầu của đối tượng
* Chỉ ra nó nằm ở đâu
* Thêm phần tử vào giao diện trên màn hình

Một thành phần (component) GUI là một đối tượng trực quan. Người dùng tương tác với đối tượng này thông qua con trỏ chuột hay bàn phím. Các thành phần như là button, label v.v… có thể được nhìn thấy trên màn hình. Bất kỳ cái gì chung cho tất cả các thành phần GUI đều được tìm thấy trong lớp Component. Để tạo các đối tượng GUI chúng ta cần nhập gói **java.awt** hoặc gói **javax.swing**.

Giao diện tương tác với người dùng của ứng dụng game cờ Caro được xây dựng dựa trên các gói **java.awt** và **javax.swing**. Sau đây là các lớp được sử dụng trong ứng dụng từ 2 gói trên.

* + - 1. **Gói Java AWT**

**Lớp Color**

* Màu được tạo từ 3 màu: đỏ , xanh lá cây hoặc xanh da trời. Mỗi màu được biểu diễn bởi một giá trị kiểu byte, từ 0 (tối nhất) đến 255 (sáng nhất) -> *RGB model*.

Color c = new Color(r, g, b);

* r, g, and b chỉ tới 1 màu đỏ, xanh lá cây, xanh da trời bởi các components của nó.

Ví dụ:

Color c = new Color(228, 100, 255);

* 13 màu chuẩn **(black, blue, cyan, darkGray, gray, green, lightGray, magenta, orange, pink, red, white, yellow)** được định nghĩa như hằng số trong java.awt.Color.
* Từ JDK 1.4: BLACK, BLUE, CYAN, DARK\_GRAY, GRAY, GREEN, LIGHT\_GRAY, MAGENTA, ORANGE, PINK, RED, WHITE, and YELLOW
* Thiết lập màu nền (background) và nổi (foreground) :

**setBackground(Color c)**

**setForeground(Color c)**

Ví dụ:

**jbt.setBackground(Color.yellow);**

**jbt.setForeground(Color.red);**

**Font:**

Ví dụ:

**batdau.setFont(new java.awt.Font("Times New Roman", 0, 24));**

**Lớp Graphics:**

Lớp trừu tượng Graphics cung cấp giao tiếp đồ hoạ độc lập với thiết bị để hiện thị hình minh hoạ và ảnh trên màn hình của các nền tảng khác nhau. Bất kể thành phần nào (button, label ... ) được hiển thị, một đối tượng Graphic sẽ được tạo trên nền tảng gốc (native). Có thể lấy về đối tượng này thông qua phương thức getGraphics(). Ví dụ, đề lấy về graphic của một đối tượng nhãn jlblBanner

Graphics graphics = jlblBanner.getGraphics();

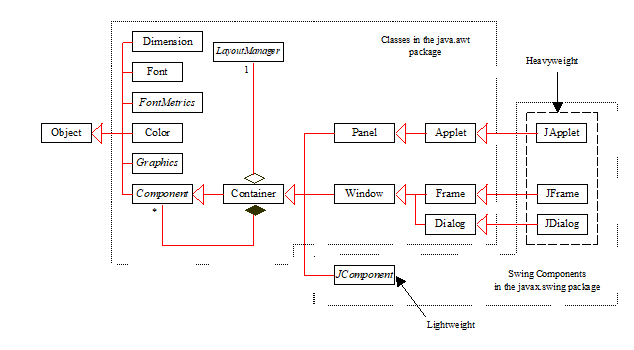
Sau đó, có thể sử dụng các phương thức của lớp Graphics để vẽ lên nhãn này.

Phương thức paintComponent thuộc lớp JComponent :

**protected void paintComponent(Graphics g)**

Phương thức này được gọi mỗi khi hiển thị hoặc hiện thị lại.

* + - 1. **Gói Java Swing**

****

Cấu trúc cây của gói Swing

**Frame:**

* cửa sổ không nằm bên trong cửa sổ khác; thành phần cơ bản để chứa các GUI khác.
* Lớp JFrame được dùng đề tạo cửa sổ.
* Với các chương trình GUI Swing -> sử dụng JFrame -> tạo cửa sổ.

Cách tạo một Frame:

Ví dụ:

import javax.swing.\*;

public class MyFrame {

public static void main(String[] args) {

JFrame frame = new JFrame("Test Frame");

frame.setSize(400, 300);

frame.setVisible(true);

frame.setDefaultCloseOperation(JFrame.

EXIT\_ON\_CLOSE);

}

}

Thêm một button vào frame

**frame.getContentPane().add(new JButton("OK"));**

**Layout managers**

* Trình quản lý layout (sự bố trí) của Java cung cấp một mức độ trừu tượng hoá cho phép định hình giao diện lên tất cả các hệ thống cửa sổ.
* Các thành phần UI được đặt trong containe. Mỗi container có một trình quản lý layout để sắp xếp các thành phần này.
* Sử dụng phương thức setLayout(LayoutManager) của container.
* Grouplayout làm việc với các layout theo chiều ngang và chiều dọc riêng rẽ. Layout được xác định cho mỗi chiều một cách độc lập. Khi tập trung vào chỉ là một chiều, bạn chỉ có thể giải quyết một nửa vấn đề cùng một lúc. Điều này dễ dàng hơn so với việc xử lý cả hai chiều cùng một lúc. Điều này có nghĩa là, tất nhiên, rằng mỗi thành phần cần phải được định nghĩa hai lần trong layout. Nếu bạn quên làm điều này, GroupLayout sẽ tạo ra một ngoại lệ.

**Panels:**

* Là Container con để nhóm các thành phần UI.
* Recomend: đặt component, panel -> panel -> frameIl.
* Để thêm thành phần -> JFrame: thêm vào thành nội dung của JFrame. Để thêm thành phần vào panel -> thêm trực tiếp vào panel bằng phương thức add.

Tạo một Jpanel:

* Sử dụng new JPanel() : layout mặc định là FlowLayout
* Sử dụng new JPanel(LayoutManager).
* Sử dụng phương thức add(Component)

**JPanel p = new JPanel();**

**p.add(new JButton("OK"));**

**Image Icon**

* Java sử dụng các lớp javax.swing.ImageIcon để biểu diễn cho một biểu tượng. Một biểu tượng là một hình ảnh kích thước cố định.
* Hình ảnh được thường được lưu trong tập tin hình ảnh. Bạn có thể sử dụng ImageIcon mới (tên tập tin) để xây dựng một biểu tượng hình ảnh.
* Ví dụ, câu lệnh sau đây tạo ra một Icon từ một file ảnh us.gif trong thư mục hình ảnh theo đường dẫn lớp hiện tại:

**Icon ImageIcon = new ImageIcon ("image / us.gif");**

**2.1.2. Khai báo và định nghĩa các hằng và biến cho trò chơi Caro**

1. **Container cn;**: Đối tượng chứa các thành phần giao diện.
2. **JPanel pn, pn2;**: Đối tượng panel để chứa các thành phần giao diện.
3. **JLabel lb;**: Đối tượng label để hiển thị thông báo về lượt chơi và kết quả.
4. **JButton newGame\_bt, undo\_bt, exit\_bt;**: Các đối tượng nút để thực hiện các chức năng.
5. **JCheckBox switchMode\_cb;**: Đối tượng checkbox để chọn chế độ chơi với máy hoặc người chơi.
6. **private JButton b[][] = new JButton[column + 2][row + 2];**: Mảng 2 chiều chứa các ô cờ là các đối tượng JButton.

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

1. **Color background\_cl = Color.white;**: Màu nền của bảng cờ.
2. **Color x\_cl = Color.red;**: Màu của người chơi X.
3. **Color o\_cl = Color.blue;**: Màu của người chơi O.
4. **int column = 20, row = 30, count = 0;**: Số cột, số dòng của bảng cờ, và biến đếm lượt chơi.
5. **int xUndo[] = new int[column \* row];**: Mảng lưu trữ tọa độ x của các nước đi đã thực hiện.
6. **int yUndo[] = new int[column \* row];**: Mảng lưu trữ tọa độ y của các nước đi đã thực hiện.
7. **boolean tick[][] = new boolean[column + 2][row + 2];**: Mảng kiểm tra ô cờ đã được đánh hay chưa.
8. **int Size = 0;**: Biến lưu trữ kích thước của mảng tọa độ nước đi đã thực hiện.
9. **boolean isVsComputer = false;**: Biến kiểm tra chế độ chơi với máy.

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

**2.1.3.Hàm kiểm tra xem người chơi nào đã thắng**

1. **Kiểm tra ô trống:** Nếu ô cờ tại **(i, j)** là trống (**" "**), thì không cần kiểm tra và trả về **false** vì không thể thắng khi ô cờ trống.
2. **Kiểm tra hàng ngang và dọc:** Sử dụng vòng lặp để kiểm tra số lượng ô liên tiếp có giá trị giống nhau (người chơi hiện tại) theo chiều hàng ngang và dọc. Nếu số lượng ô liên tiếp là 5 hoặc hơn, trả về **true** vì người chơi đã thắng.
3. **Kiểm tra đường chéo xuôi và ngược:** Sử dụng hai vòng lặp while để kiểm tra số lượng ô liên tiếp theo đường chéo xuôi và ngược. Nếu số lượng ô liên tiếp là 5 hoặc hơn, trả về **true** vì người chơi đã thắng.
4. **Trả về false nếu không có chiến thắng:** Nếu qua tất cả các kiểm tra mà không có người chơi nào thắng, hàm trả về **false**.

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

**2.1.3.Hàm kiểm tra số lượng ô liên tiếp có giá trị giống nhau**

1. **Khởi tạo biến player:** Lấy giá trị của người chơi hiện tại tại ô cờ **(x, y)**.
2. **Khởi tạo biến count:** Biến này sẽ đếm số lượng ô liên tiếp có giá trị giống nhau theo hướng đã xác định.
3. **Khởi tạo biến i và j:** Đặt giá trị ban đầu cho các biến **i** và **j** theo hướng **dx** và **dy**.
4. **Vòng lặp while:** Thực hiện vòng lặp while để đi theo hướng đã xác định và đếm số lượng ô liên tiếp có giá trị giống nhau. Dừng lại khi gặp ô trống hoặc vượt ra khỏi biên của bảng cờ.
5. **Trả về giá trị của count:** Số lượng ô liên tiếp có giá trị giống nhau theo hướng đã xác định.

**Ảnh có chứa văn bản, ảnh chụp màn hình, phần mềm, Phần mềm đa phương tiện

Mô tả được tạo tự động**

**2.1.4. Hàm hoàn tác nước đi**

1. **Kiểm tra điều kiện:** Hàm kiểm tra xem có nước đi nào để undo không (biến **Size** phải lớn hơn 0).
2. **Giảm kích thước mảng Size:** Giảm giá trị của biến **Size** xuống một để trỏ đến phần tử cuối cùng của mảng **xUndo** và **yUndo** chứa tọa độ của nước đi cuối cùng.
3. **Lấy tọa độ i và j:** Lấy tọa độ **i** và **j** từ phần tử cuối cùng của mảng **xUndo** và **yUndo**.
4. **Đặt lại giá trị của ô cờ:** Đặt lại giá trị của ô cờ tại **(i, j)** thành khoảng trắng (**" "**) và đặt lại màu nền của ô cờ.
5. **Cập nhật trạng thái của ô cờ:** Đặt lại trạng thái của ô cờ trong mảng **tick** về **true** để cho phép việc đánh nước đi mới tại ô cờ này.
6. **Cập nhật thông báo lượt chơi:** Cập nhật lại label **lb** để hiển thị lượt chơi của người chơi tiếp theo.
7. **Tắt chức năng undo nếu không còn nước đi để undo:** Nếu không còn nước đi nào để undo (kích thước của mảng **Size** bằng 0), thì tắt chức năng undo bằng cách vô hiệu hóa nút undo (**undo\_bt.setEnabled(false)**).

**Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ

Mô tả được tạo tự động**

**2.1.5.Hàm thực hiện nước đi của máy**

1. **Lấy tọa độ nước đi cuối cùng của người chơi (lastX, lastY):** Lấy tọa độ của nước đi cuối cùng của người chơi từ mảng **xUndo** và **yUndo**.
2. **Xác định phạm vi để máy thực hiện nước đi (range):** Đặt giá trị mặc định cho phạm vi là 2 ô xung quanh nước đi cuối cùng của người chơi.
3. **Khởi tạo biến attempts:** Biến này sẽ đếm số lần thử chọn ô trong phạm vi xung quanh.
4. **Vòng lặp while:** Thực hiện vòng lặp để chọn một ô trống trong phạm vi xung quanh nước đi cuối cùng của người chơi.
5. **Kiểm tra nếu vị trí nằm trong bảng và ô đó trống:** Kiểm tra xem vị trí mới được chọn có nằm trong bảng và có phải là ô trống không.
6. **Mở rộng phạm vi nếu cần:** Nếu sau một số lần thử mà không tìm thấy ô trống, mở rộng phạm vi và tiếp tục thử lại.
7. **Logic dự phòng nếu không tìm thấy ô trong phạm vi gần:** Nếu vẫn không tìm thấy ô trống sau một số lần mở rộng phạm vi, thì chạy hàm **randomMove** để chọn một ô ngẫu nhiên.



**2.1.6. Giao diện**

1. **Gọi constructor của lớp cha (super(s)):** Gọi constructor của lớp cha (**JFrame**) để thiết lập các thuộc tính của cửa sổ như tiêu đề.
2. **Khởi tạo và cấu hình giao diện (cn, pn):** Khởi tạo container (**cn**) và panel (**pn**) của giao diện.
3. **Tạo và cấu hình ô cờ (b):** Sử dụng hai vòng lặp để tạo ra các ô cờ (JButton) và cấu hình chúng, bao gồm xác định ActionCommand, màu nền, và thiết lập ActionListener để xử lý sự kiện khi người chơi nhấp vào một ô cờ.
4. **Thêm ô cờ vào panel (pn):** Dùng vòng lặp để thêm các ô cờ vào panel.
5. **Tạo và cấu hình các thành phần khác (lb, newGame\_bt, undo\_bt, exit\_bt, switchMode\_cb):** Tạo và cấu hình các thành phần khác như label, nút "New Game", nút "Undo", nút "Exit", và checkbox "Play vs Computer". Đặt ActionListener để xử lý sự kiện khi người chơi tương tác với chúng.
6. **Thêm các thành phần vào container (cn):** Thêm panel (**pn**) và panel chứa các thành phần khác (**pn2**) vào container.
7. **Cấu hình các thuộc tính của cửa sổ (this):** Thiết lập kích thước, hiển thị cửa sổ, vị trí cửa sổ, và xử lý sự kiện khi người chơi đóng cửa sổ.
8. **Vô hiệu hóa nút Undo (undo\_bt):** Đặt trạng thái ban đầu của nút "Undo" là không hoạt động.

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình, phần mềm

Mô tả được tạo tự động Ảnh có chứa văn bản, ảnh chụp màn hình

Mô tả được tạo tự động

**2.1.7. Xử lý giao diện**

1. **Xử lý sự kiện từ checkbox chế độ chơi với máy (switchMode\_cb):** Kiểm tra xem sự kiện được kích hoạt bởi checkbox chế độ chơi với máy hay không. Nếu có, cập nhật biến **isVsComputer** và gọi hàm **updateModeButton** để cập nhật nút chế độ chơi.
2. **Xử lý sự kiện từ nút "New Game" (newGame\_bt):** Kiểm tra xem sự kiện được kích hoạt bởi nút "New Game" hay không. Nếu có, tạo một trò chơi mới bằng cách gọi constructor **CaRo** và đóng cửa sổ trò chơi hiện tại.
3. **Xử lý sự kiện từ nút "Undo" (undo\_bt):** Kiểm tra xem sự kiện được kích hoạt bởi nút "Undo" hay không. Nếu có, gọi hàm **undo** để hoàn tác nước đi cuối cùng.
4. **Xử lý sự kiện từ nút "Exit" (exit\_bt):** Kiểm tra xem sự kiện được kích hoạt bởi nút "Exit" hay không. Nếu có, thoát ứng dụng bằng cách gọi **System.exit(0)**.
5. **Xử lý sự kiện từ ô cờ được nhấp (b[i][j]):** Kiểm tra xem sự kiện có phải từ một ô cờ hay không. Nếu có, lấy tọa độ **(i, j)** của ô cờ và gọi hàm **addPoint(i, j)** để thực hiện nước đi.
6. **Xử lý khi có người chơi thắng:** Nếu sau nước đi, có người chơi thắng, thực hiện các hành động liên quan như hiển thị thông báo chiến thắng, vô hiệu hóa ô cờ và nút undo, và thay đổi màu nền của nút "New Game".

Ảnh có chứa văn bản, ảnh chụp màn hình, màn hình

Mô tả được tạo tự động

**2.1.8. Hàm làm mới trò chơi**

1. **Kiểm tra trạng thái của biến isVsComputer:** Sử dụng câu lệnh **if** để kiểm tra giá trị của biến **isVsComputer**.
2. **Cập nhật văn bản của checkbox (switchMode\_cb):** Nếu **isVsComputer** là **true**, đặt văn bản của checkbox thành "Play vs Human". Ngược lại, nếu **isVsComputer** là **false**, đặt văn bản của checkbox thành "Play vs Computer". Điều này giúp hiển thị thông tin chính xác về chế độ chơi hiện tại cho người chơi khi họ xem giao diện người dùng.

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, phần mềm

Mô tả được tạo tự động

* 1. **Thực thi và cài đặt**

Ta sẽ chạy ứng dụng trên Visual Studio Code. Để chạy được chương trình thì trước tiên phải cài đặt JDK

Bước 1: Cài đặt JDK



Hình 2. File JDK khi tải về

Bước 2: Mở ứng dụng Visual Studio Code :

Ảnh có chứa Phông chữ, biểu tượng, Đồ họa, Xanh điện

Mô tả được tạo tự động

Hình 3. Phần mềm Visual Studio Code

Bước 2: Cài đặt môi trường Java

Ảnh có chứa văn bản, phần mềm, Biểu tượng máy tính, Trang web

Mô tả được tạo tự động

Hình 4. Giao diện cài đặt môi trường Java

Bước 3: Nhấn vào File -> Open File

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, số

Mô tả được tạo tự động

Hình 5. Mở File

Bước 4: Tìm đến file chứa chương trình và nhấn mở

Ảnh có chứa văn bản, ảnh chụp màn hình, số, thực đơn

Mô tả được tạo tự động

Hình 6. Chọn file cần mở

Bước 5: Sau khi mở file, chúng ta sẽ thấy file có tên là Gamecaro.java



Hình 7. Tên file

Bước 6: Nhấn Ctrl + Alt + N hoặc biểu tượng mũi tên góc trên bên trái màn hình để chạy chương trình

Ảnh có chứa văn bản, ảnh chụp màn hình, Phông chữ, hàng

Mô tả được tạo tự động

Hình 8. Thực thi file code được mở

Bước 7: Kết quả hiện ra:

Ảnh có chứa văn bản, hàng, ảnh chụp màn hình, Sơ đồ

Mô tả được tạo tự động

Hình 9. Giao diện Game caro

**Kết luận**

Hiện nay, lập trình game đang là một lĩnh vực mới thu hút nhiều lập trình viên. Việc xây dựng các ứng dụng vui chơi giải trí là rất cần thiết cho sự phát triển công nghệ. Trong phạm vi đề tài, chúng em chỉ trình bày những phần công nghệ cơ bản có liên quan đến ứng dụng như lập trình giao diện người dùng với Swing và AWT và bắt sự kiện với AWT.

Nhìn chung game tạo ra đáp ứng nhu cầu giải trí của người chơi. Giao diện thân thiện, đơn giản, dễ sử dụng và thao tác. Tuy nhiên vẫn còn một số lỗi như lỗi font chữ chưa được khắc phục, bàn cờ còn hạn chế, chưa đẹp mắt.

Tuy vậy, do những hạn chế về trình độ, thời gian cũng như thiết bị nên chúng em không khỏi những vướng mắc, sai sót trong quá trình tìm hiểu và nghiên cứu. Chúng em rất mong nhận được những đánh giá và chỉnh sửa từ các thầy cô hướng dẫn.

Chúng em xin chân thành cảm ơn!