**BỘ CÔNG THƯƠNG**

**TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP TP.HCM**



**Tìm Hiểu Công Nghệ WPF và viết ứng dụng học tiếng anh cho trẻ em**

***Nhóm 39 - Sinh viên thực hiện:***

1. Phan Hữu Quý – 16073931
2. Hoàng Hữu Cương – 16048731

# MỤC LỤC

[MỤC LỤC 2](#_Toc40971771)

[DANH MỤC CÁC HÌNH VẼ 2](#_Toc40971772)

[DANH MỤC CÁC BẢNG BIỂU 2](#_Toc40971773)

[LỜI MỞ ĐẦU 2](#_Toc40971774)

[CHƯƠNG 1 : GIỚI THIỆU 2](#_Toc40971775)

[1.1 Tổng quan 2](#_Toc40971776)

[1.1.1 Một số ứng dụng học tiếng anh hiện nay: 2](#_Toc40971777)

[1.1.2 Giới thiệu về Ứng dụng học tiếng anh cho trẻ em Big Fun: 2](#_Toc40971778)

[1.2 Mục tiêu đề tài 2](#_Toc40971779)

[1.3 Phạm vi đề tài 2](#_Toc40971780)

[1.4 Mô tả yêu cầu chức năng 2](#_Toc40971781)

[CHƯƠNG 2 : CƠ SỞ LÝ THUYẾT 2](#_Toc40971782)

[2.1 Công nghệ WPF 2](#_Toc40971783)

[2.1.1 Giới thiệu về WPF 2](#_Toc40971784)

[2.1.2 Mục tiêu của WPF 2](#_Toc40971785)

[2.1.3 Các tính năng nổi bật của WPF 2](#_Toc40971786)

[2.1.4 Ngôn ngữ XAML 2](#_Toc40971787)

[2.1.5 WPF cho Window và trình duyệt Web 2](#_Toc40971788)

[2.1.6 Các thành phần của WPF 2](#_Toc40971789)

[2.2 Material Design 2](#_Toc40971790)

[2.2.1 Material Design là gì? 2](#_Toc40971791)

[2.2.2 Đặc điểm của Material Design 2](#_Toc40971792)

[2.2.3 Ba mảnh ghép của Material Design 2](#_Toc40971793)

[CHƯƠNG 3 : PHÂN TÍCH 2](#_Toc40971794)

[3.1 Phân tích yêu cầu bằng UML 2](#_Toc40971795)

[3.1.1 Use case Tổng Quát 2](#_Toc40971796)

[3.1.2 Danh sách các actor 2](#_Toc40971797)

[3.1.3 Danh sách các Use case 2](#_Toc40971798)

[3.2 Đặc tả các yêu cầu chức năng 2](#_Toc40971799)

[3.2.1 Use case UC1: Concentration 2](#_Toc40971800)

[3.2.2 Use case UC2: Matching 2](#_Toc40971801)

[3.2.3 Use case UC3: Sorting 2](#_Toc40971802)

[3.2.4 Use case UC4: Multiple Choice 2](#_Toc40971803)

[3.2.5 Use case UC5: Painting 2](#_Toc40971804)

[3.2.6 Use case UC6: Story time 2](#_Toc40971805)

[3.2.7 Use case UC7: Looking and find 2](#_Toc40971806)

[CHƯƠNG 4 : THIẾT KẾ VÀ HIỆN THỰC 2](#_Toc40971807)

[4.1 Tiêu đề mục 2](#_Toc40971808)

[4.1.1 Tiêu đề tiểu mục 2](#_Toc40971809)

[4.1.2 Tiêu đề tiểu mục 2](#_Toc40971810)

[4.2 Tiêu đề mục 2](#_Toc40971811)

[4.2.1 Tiêu đề tiểu mục 2](#_Toc40971812)

[4.2.2 Tiêu đề tiểu mục 2](#_Toc40971813)

[CHƯƠNG 5 : KẾT LUẬN 2](#_Toc40971814)

[5.1 Kết quả đạt được 2](#_Toc40971815)

[5.2 Hạn chế của đồ án 2](#_Toc40971816)

[5.3 Hướng phát triển 2](#_Toc40971817)

[TÀI LIỆU THAM KHẢO 2](#_Toc40971818)

[PHỤ LỤC 2](#_Toc40971819)

# DANH MỤC CÁC HÌNH VẼ

[Hình 2‑1 Tương tác giữa nhà thiết kế và lập trình viên thông qua XAML 2](#_Toc36835874)

[Hình 2‑2 Các thành phần cơ bản của WPF. 2](#_Toc36835875)

[Hình 2‑3 Quan hệ giữa đối tượng dữ liệu và đối tượng phụ thuộc. 2](#_Toc36835876)

[Hình 3‑1 Use case tổng quát 2](#_Toc36835877)

# DANH MỤC CÁC BẢNG BIỂU

[Bảng 3‑1 Danh sách các Actor. 2](#_Toc36835866)

[Bảng 3‑2 Danh Sách Các Use case chức năng. 2](#_Toc36835867)

[Bảng 3‑3 Đặc tả Use case Concentration 2](#_Toc36835868)

[Bảng 3‑4 Đặc tả Use case Matching 2](#_Toc36835869)

[Bảng 3‑5 Đặc tả Use case Sorting. 2](#_Toc36835870)

[Bảng 3‑6 Đặc tả Use case Multiple Choice 2](#_Toc36835871)

[Bảng 3‑7 Đặc tả Use case Story time. 2](#_Toc36835872)

[Bảng 3‑8 Đặc tả Use case Looking and find. 2](#_Toc36835873)

# LỜI MỞ ĐẦU

1. Tổng quan tình hình nghiên cứu thuộc lĩnh vực của đề tài

Trong những năm gần đây với sự phát triển vượt trội của công nghệ thông tin, những ứng dụng của Công Nghệ Thông Tin vào khắp các lĩnh vực đời sống đã đóng góp to lớn cho sự phát triển của nhân loại về nhiều mặt. Đi kèm với nó là yêu cầu của người dùng cũng ngày càng khắt khe về mọi mặt, họ không chỉ muốn một ứng dụng hỗ trợ đầy đủ chức năng nghiệp vụ đơn giản mà còn yêu cầu giao diện ứng dụng trực quan, sinh động…

**Ví dụ như trong trường hợp một ứng dụng học tiếng anh cho trẻ em, một giao diện trực quan, sinh động với nhiều hiệu ứng bắt mắt đi kèm những âm thanh, hình ảnh, video thú vị sẽ góp phần rất lớn vào việc thu hút trẻ em sử dụng ứng dụng để tiếp thu kiến thức mà không bị nhàm chán.**

Với công nghệ trước đây, một ứng dụng với các yêu cầu như vậy thật sự là thách thức đòi hỏi rất nhiều tài nguyên và công sức, và WPF ra đời chính là để xây dựng một nền tảng chung giúp cho việc xây dựng các ứng dụng trở nên trực quan và thuận tiện hơn bao giờ hết.

Cũng ví lí do đó mà chúng em đã chọn đề tài “Tìm hiểu về công nghệ WPF và xây dựng ứng dụng học tiếng anh cho trẻ em”.

2. Ý nghĩa khoa học và thực tiễn của đề tài

Có cơ hội tiếp cận, tìm hiểu một công nghệ mới để xây dựng một ứng dụng, hiểu rõ những điểm mạnh, điểm yếu của WPF và so sánh với Winform.

Củng cố kiến thức của bản thân về lập trình, về thiết kế ứng dụng và kiểm thử.

Thiêt kế và xây dựng một ứng dụng học tiếng anh cho trẻ em với tiêu chuẩn công nghiệp và có khả năng đưa vào áp dụng trong thực tế.

# : GIỚI THIỆU

## Tổng quan

Hiện nay tiếng anh được coi là ngôn ngữ quốc tế và ở Việt Nam, tiếng anh đã chiếm được vị trí quan trọng từ khi đất nước thực hiện chính sách mở cửa hội nhập; Nói riêng ở Việt Nam thì có tới hàng triệu người đều đang nỗ lực học tiếng anh mỗi ngày, trong số đó, ngoài lứa tuổi học sinh, sinh viên, hay những người đã trưởng thành, đã đi làm thì vẫn có một bộ phận là lứa tuổi trẻ em được bố mẹ định hướng việc học tiếng anh từ rất sớm và đặc biệt là việc chọn một phương pháp phù hợp cho trẻ học tiếng anh là một vấn đề rất được các bậc phụ huynh quan tâm, ngoài việc tự dạy cho bé tại nhà, đưa bé đến những trung tâm tiếng anh uy tín… thì ngày nay bố mẹ còn có thể cho con học thông qua một ứng dụng dạy tiếng anh trên máy tình hoặc điện thoại thông minh.

### Một số ứng dụng học tiếng anh hiện nay:

1. **MonkeyJunior:**

Monkey Junior có nội dung chương trình phân chia thành 3 cấp độ từ dễ đến khó nên thích hợp với trẻ em ở nhiều lứa tuổi ( từ 0-10 tuổi). Trải dài với hàng trăm chủ đề khác nhau từ những đồ vật ở nhà, hình dạng, bộ phận cơ thể, đồ chơi, các hành động, trái cây....

Mỗi bài học gồm 5 phần và chỉ kéo dài 5-7 phút phù hợp với mức độ tiếp thu và tập trung của trẻ nhỏ. Kết thúc mỗi phần bé được tặng sao, hết bài học được nhận sticker giúp trẻ luôn hào hứng, học mà chơi, chơi mà học.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Phần 1** | **Phần 2** | **Phần 3** | **Phần 4** | **Phần 5** |
| Học từ và câu theo chủ đề 1 | Học từ phổ biến (Sight words) | Học từ và câu theo chủ đề 2 | Phát âm (phonics) | Trò chơi ôn tập |

1. **English for kid:**

English for kids là ứng dụng hỗ trợ bé học tiếng Anh thông qua các trò chơi: Flashcard, Sentences Games, Word games,…Với Flashcard, các bé được học thêm từ vựng của rất nhiều chủ đề khác nhau: động vật (animal), quần áo (clothes), số đếm ( numbers),các loài hoa (flower),…Còn với Sentences games, các bé sẽ tập sắp xếp các từ thành câu hoàn chỉnh, sửa lỗi sai của câu để dần hình thành trong hình dung của trẻ thế nào là một câu hoàn chỉnh. Trong khi đó Word games là trò chơi đoán từ vựng ngẫu nhiên. Mỗi từ vựng đưa ra bất kỳ thì nhiệm vụ của bé là đưa ra đáp án đúng. Đây cũng là một cách vừa kiểm tra vốn từ vựng vừa gia tăng vốn từ vựng cho bé.

### Giới thiệu về Ứng dụng học tiếng anh cho trẻ em Big Fun:

Ứng dụng học tiếng anh cho trẻ em Big Fun được xây dựng trên nền tảng ngôn ngữ C#, áp dụng công nghệ WPF. Các bài học được đưa ra dựa trên chương trình dạy tiếng anh cho trẻ em Big fun, một chương trình giảng dạy tiếng anh cho trẻ em từ 3-6 tuổi với những chủ đề gần gũi như Lớp học của em ( My class), Cơ thể của em ( My body),… Ở mỗi bài học sẽ được lồng ghép một số câu giao tiếp, mẫu câu thông dụng; bên cạnh đó, giao diện của ứng dụng được thiết kế kết hợp với những hình ảnh hoạt hình thú vị, những âm thanh sống động cùng với các hiệu đặc sắc giúp tạo sự hứng thú, lôi cuốn, giúp cho trẻ em tiếp thu nhanh và hiệu quả các bài học mà không cảm thấy nhàm chán.

## Mục tiêu đề tài

Trọng tâm tìm hiểu công nghệ lập trình WPF để thiết kế giao diện, kết hợp với các thư viện hỗ trợ đồ họa, âm thanh, hình ảnh, các giáo trình giảng dạy tiếng anh để áp dụng vào đề tài.

Tìm hiểu những thói quen, sở thích của người dùng để tạo ra ứng dụng thân thiện, dễ sử dụng, truyền tải được nội dung học tập.

## Phạm vi đề tài

Để hoàn thành ứng dụng học tiếng anh cần nghiên cứu và tìm hiểu rõ công nghệ WPF, tham khảo một số ứng dụng, chương trình giảng dạy tiếng anh cho trẻ em như: ứng dụng MonkeyJunior, chương trình dạy tiếng anh Big Fun, bộ sách Family and friend…

## Mô tả yêu cầu chức năng

Ứng dụng học tiếng anh cho trẻ em từ 3-6 tuổi cung cấp phương pháp học tiếng anh theo từng chủ đề gần gũi với những điều bé quan tâm ở lứa tuổi của mình. Mỗi bài học sẽ là một chủ đề gần gũi với bé, được lồng ghép một số mẫu câu cơ bản và được chia thành rất nhiều hoạt động nhỏ, trong đó sẽ bao gồm những hình ảnh, âm thanh sôi động, những trò chơi đơn giản yêu cầu bé phải lắng nghe, tưởng tượng và tương tác với ứng dụng, từ đó tiếng anh sẽ tự nhiên đi vào trí não của bé:

* Hoạt động 1- Concentration: Bé sẽ tham gia trò chơi Lật hình để tìm các cặp hình giống nhau. Ở mỗi lần lật hình, phát âm từ vựng của hình sẽ được phát để bé có thể lắng nghe và ghi nhớ.
* Hoạt động 2- Matching: trò chơi nối ô âm thanh với ô hình ảnh tương ứng. Bé sẽ chọn một ô âm thanh, âm thanh phát âm của từ vựng sẽ được phát ngay sau đó và bé phải lắng nghe và chọn một ô hình ảnh tương ứng để nối 2 ô đó lại với nhau.
* Hoạt động 3- Sorting: Nghe và xếp hình ảnh vào ô đúng. Ứng dụng sẽ phát âm thanh yêu cầu đặt một hình ảnh vào ô có số thứ tự nào đó, bé phải chọn hình ảnh đó và kéo nó vào ô có số thứ tự đúng theo yêu cầu.
* Hoạt động 4- Multiple Choice: Chọn hình ảnh của từ vựng nghe được. Ứng dụng sẽ phát âm thanh phát âm của từ vựng và bé phải chọn hình ảnh đúng với từ vựng đó.
* Hoạt động 6- Story Time: Ứng dụng sẽ hiển thị một số câu giao tiếp, mẫu câu thông dụng kèm hình ảnh và âm thanh để người dùng lắng nghe và ghi nhớ.
* Hoạt động 5- Look and Find: Nghe và tìm kiếm những gì nghe được từ trong hình.

1. Yêu cầu chức năng:

* Concentration: Ứng dụng sẽ hiển thị sáu hình ảnh bị lật úp, trong đó bao gồm ba cặp hình ảnh, khi người dùng chọn hai hình bị úp để mở hình ảnh, âm thanh phát âm từ vựng của hình ảnh đó sẽ được phát. Lật lần lượt hai hình để tìm các hình giống nhau, nếu lật hai hình khác nhau thì sẽ phát âm thanh báo sai và hai hình đó sẽ bị úp trở lại. Dùng phải lật đúng cả ba cặp hình để hoàn thành trò chơi
* Matching: Ứng dụng sẽ hiển thị ba ô âm thanh và ba ô hình ảnh tương ứng, khi người dụng chọn ô âm thanh, phát âm của một từ vựng sẽ được phát, dùng phải lắng nghe để chọn ô hình ảnh đúng với từ vựng đó để nối hai ô lại với nhau. Ghép thành công ba cặp để hoàn thành trò chơi.
* Sorting: Ứng dụng sẽ hiển thị ba hình ảnh và ba ô trống được đánh số theo thứ tự, lần lượt phát âm thanh yêu cầu đặt hình một hình vào một ô có số thứ tự nhất định, dùng phải đặt đúng theo yêu cầu cả ba hình ảnh vào ba ô để hoàn thành trò chơi.
* Multiple Choice: Ứng dụng sẽ hiển thị ba hình ảnh và lần lượt phát âm từ vựng của một trong ba hình ảnh đó, người dùng phải chọn hình ảnh theo từ vựng nghe được. Thực hiện ba lượt chọn để hoàn thành trò chơi.
* Story Time: Ứng dụng sẽ hiển thị một số câu giao tiếp, mẫu câu thông dụng kèm hình ảnh và âm thanh minh họa, người dùng sẽ lắng nghe và ghi nhớ.
* Look and find: Ứng dụng sẽ hiển thị một hình ảnh nền và yêu cầu người dùng tìm và chọn hình ảnh của một từ vựng nào đó trong hình nền.

1. Yêu cầu phi chức năng:
   * Âm thanh, hình ảnh phù hợp cho lứa tuổi từ 3-6.
   * Giao diện đẹp, nhiều hiệu ứng thú vị để thu hút trẻ em.
   * Dễ sử dụng: các thao tác học tập, chọn bài học… Phải đơn giản để bé có thể dễ dàng tự sử dụng ngay lần đầu tiên sử dụng sau khi được hướng dẫn.
   * Ứng dụng có thể cài đặt và chạy ổn định trên hệ điều hành win 8,10

# : CƠ SỞ LÝ THUYẾT

## Công nghệ WPF

### Giới thiệu về WPF

WPF, viết tắt của Windows Presentation Foundation, là hệ thống API mới hỗ trợ việc xây dựng giao diện đồ họa trên nên Windows. Được xem như thế hệ kế tiếp của WinForms, WPF tăng cường khả năng lập trình giao diện của lập trình viên bằng cách cung cấp các API cho phép tận dụng những lợi thế về đa phương tiện hiện đại. Là một bộ phận của .NET Framework 3.5, WPF sẵn có trong Windows Vista và Windows Server 2008. Đồng thời, WPF cũng có thể hoạt động trên nền Windows XP Pack 2 hoặc mới hơn, và cả trên Windows Server 2003.

WPF là nền tảng lập trình mới, hiện đại của Microsoft, phần lớn các ứng dụng Desktop của Microsoft đều viết trên nền WPF.

### Mục tiêu của WPF

WPF được xây dựng nhằm vào 3 mục tiêu cơ bản:

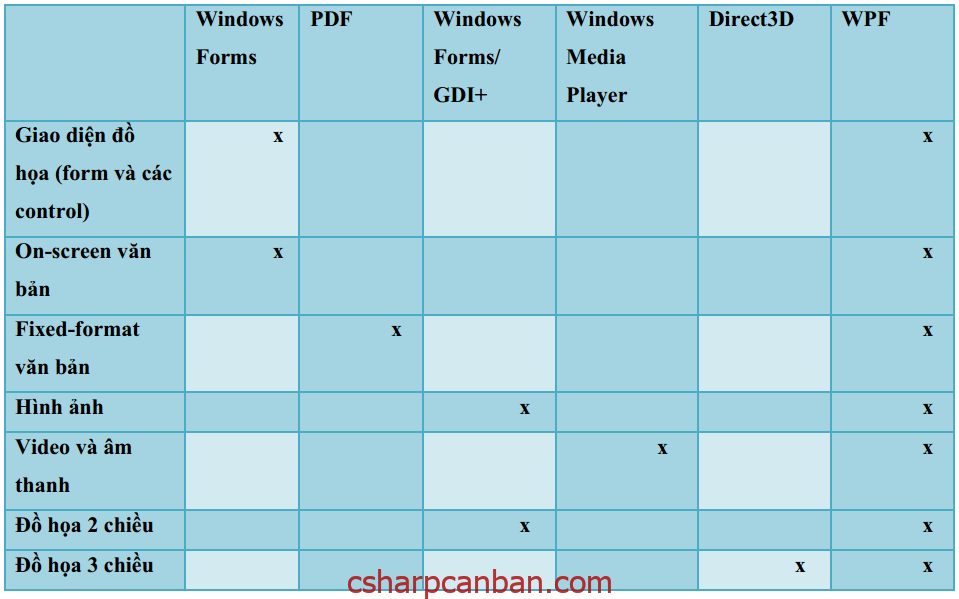
* Cung cấp một nền tảng thống nhất để xây dựng giao diện người dùng.
* Cho phép người lập trình và người thiết kế giao diện làm việc cùng nhau một cách dễ dàng.
* Cung cấp một công nghệ chung để xây dựng giao diện người dùng trên cả Windows và trình duyệt Web.

### Các tính năng nổi bật của WPF

WPF mang đến cho việc lập trình trở nên đơn giản hơn từ thiết kế giao diện, xử lý dữ liệu (sử dụng Linq kết hợp với cơ chế Binding) giúp việc lập trình dễ dàng và mạnh mẽ hơn hẳn WinForm.

Cùng với nền tảng .NET 4.5 hỗ trợ Web API (trước đó, .NET 3.5 dùng WCF, hay Web Service cho .NET trước đó nữa), giúp chúng ta dễ dàng xây dựng các ứng dụng điện toán đám mây.

Bảng dưới đây liệt kê ra những đặc điểm ưu việt của công nghệ lập trình WPF so với một số ngôn ngữ lập trình khác:



Hình ‑ So sánh công nghệ wpf và một số công nghệ khác

Tuy nhiên, WPF ra đời không có nghĩa là tất cả những công nghệ nêu trên bị thay thế. Windows Forms vẫn có giá trị, thậm chí trong WPF, một số ứng dụng mới vẫn sẽ sử dụng Windows Forms. Windows Media Player vẫn đóng một vai trò công cụ độc lập để chơi nhạc và trình chiếu video. PDF cho văn bản vẫn tiếp tục được sử dụng. Direct3D vẫn là công nghệ quan trọng trong games và các dạng ứng dụng khác (Trong thực tế, bản thân WPF dựa trên Direct3D để thực hiện mọi biểu diễn đồ họa).

Việc tạo ra một giao diện người dùng hiện đại không chỉ là việc hợp nhất các công nghệ sẵn có khác nhau. Nó còn thể hiện ở việc tận dụng lợi điểm của card đồ họa hiện đại. Để giải phóng những hạn chế của đồ họa bitmap, WPF dựa hoàn toàn trên đồ họa vector, cho phép hình ảnh tự động thay đổi kích thước để phù hợp với kích thước và độ phân giải của màn hình mà nó được hiển thị.

Bằng việc hợp nhất tất cả các công nghệ cần thiết để tạo ra một giao diện người dùng vào một nền tảng đơn nhất, WPF đơn giản hóa đáng kể công việc của lập trình viên giao diện. Với việc yêu cầu lập trình viên học một môi trường phát triển duy nhất, WPF góp phần làm giảm chi phí cho việc xây dựng và bảo trì ứng dụng. Và bằng việc cho phép tích hợp đa dạng nhiều cách biểu diễn thông tin trên giao diện người dùng, WPF góp phần nâng cao chất lượng, và theo đó là giá trị công việc, của cách thức người dùng tương tác với ứng dụng trên Windows.

### Ngôn ngữ XAML

XAML, viết tắt của Extensible Application Markup Language, là ngôn ngữ đặc tả dựa trên XML được dùng để định nghĩa các đối tượng và thuộc tính của chúng, mối quan hệ cũng như sự tương tác. XAML đặc biệt được dùng trong các công nghệ của .NET Framework 3.0 (trong đó có WPF) như ngôn ngữ đặc tả giao diện người dùng (user interface - UI) nhằm mô tả cấu trúc và đặc tính của các phần tử UI, sự liên kết dữ liệu, các sự kiện và các đặc tính khác. XAML là mô hình có tính đột phá trong lĩnh vực tính toán trên Internet được chấp nhận rộng rãi trong nhiều hệ thống và bởi nhiều nhà cung cấp phần mềm.

XAML định ra một tập các phần tử XML như Button, TextBox, Label…, nhằm định nghĩa các đối tượng đồ họa tương ứng như nút bấm, hộp thoại, nhãn…, và nhờ đó cho phép mô tả chính xác diện mạo của giao diện người dùng. Các phần tử XAML cũng chứa các thuộc tính, cho phép thiết lập nhiều tính chất khác nhau của đối tượng đồ họa tương ứng. Ví dụ, đoạn mã sau sẽ tạo ra một nút bấm màu đỏ có nhan đề “No”.

*<Button Background="Red">No</Button>*

Mỗi phần tử XAML lại tương ứng với một lớp WPF, và mỗi thuộc tính của phần tử đó lại tương ứng với thuộc tính hay sự kiện của lớp này. Chẳng hạn, nút bấm màu đỏ trong ví dụ trên có thể tạo bằng C# code như sau:

*Button btn = new Button();*

*btn.Background = Brushes.Red;*

*btn.Content = "No";*

Nếu như mọi thứ có thể biểu diễn bằng XAML thì cũng có thể biểu diễn bằng đoạn mã, thì câu hỏi đặt ra là XAML có ý nghĩa gì? Câu trả lời là việc xây dựng các công cụ sinh và sử dụng các đặc tả bằng XML dễ dàng hơn nhiều so với xây dựng một công cụ tương tự làm việc với đoạn mã. Bởi vậy, XAML mở ra một cách thức tốt hơn để lập trình viên và người thiết kế làm việc với nhau.



Hình ‑ Tương tác giữa nhà thiết kế và lập trình viên thông qua XAML

**XAML** đóng vai trò một ngôn ngữ chung giữa môi trường thiết kế giao diện và môi trường lập trình. Đối với người thiết kế, XAML không những cho phép người thiết kế dễ dàng mô tả, chỉnh sửa các đối tượng UI và các đặc tính của chúng, mà còn cho phép họ tương tác với các đối tượng này ở mức độ nhất định. Do vậy, tăng khả năng cảm quan của người thiết kế đối với giao diện. Ngoài ra, nó cũng giúp người thiết kế hạn chế những ý tưởng đồ họa không khả thi khi lập trình. Đối với người lập trình, nhờ một môi trường lập trình có khả năng tự động tái tạo giao diện đã thiết kế dựa trên file đặc tả bằng XAML do người thiết kế chuyển sang, người lập trình không cần tự mình tái tạo lại giao diện. Điều này giảm đi nhiều công sức và thời gian để phát triển giao diện, cũng như tránh những sai lệch giữa giao diện do người thiết kế và giao diện do người lập trình tái tạo. Vì cả môi trường thiết kế và lập trình đều có khả năng hiểu và sử dụng XAML, ứng dụng WPF có thể chuyển qua lại giữa hai môi trường phát triển để sửa đổi hay bổ sung giao diện một cách dễ dàng. Với tất cả những lợi điểm này, vị thế của người thiết kế trong việc xây dựng giao diện được nâng cao.

### WPF cho Window và trình duyệt Web

Trong thời đại bùng nổ của Internet, các ứng dụng Web ngày một phát triển. Việc trang bị giao diện người dùng với đầy đủ tính năng như một ứng dụng desktop sẽ thu hút nhiều người sử dụng;, và do đó góp phần làm tăng giá trị doanh nghiệp. Tuy nhiên, như đã nêu trong phần đầu, với những công nghệ truyền thống, để phát triển một giao diện đồ họa vừa hoạt động trên desktop vừa trên trình duyệt Web, đòi hỏi phải sử dụng những công nghệ hoàn toàn khác nhau, giống như việc xây dựng hai giao diện hoàn toàn độc lập. Điều này tạo ra chi phí không cần thiết để phát triển giao diện.

WPF là một giải pháp cho vấn đề này. Lập trình viên có thể tạo ra một ứng dụng trình duyệt *XAML* (XBAP) sử dụng WPF chạy trên Internet Explore. Trên thực tế, cùng đoạn code này có thể được dùng để sinh ứng dụng WPF chạy độc lập trên Windows.

Phần giao diện của ứng dụng dạng XBAP được trình duyệt chia thành các frame thay vì chạy trên các cửa sổ riêng, ngoài ra, các chức năng đều được bảo toàn. Cùng một đoạn mã được sử dụng chung cho cả hai trường hợp sẽ làm giảm khối lượng công việc cần thiết để phát triển hai dạng giao diện. Thêm vào đó, sử dụng cùng một đoạn mã cũng có nghĩa là sử dụng cùng kỹ năng của lập trình viên. Do đó, lập trình viên chỉ cần có học một kiến thức chung là có thể sử dụng trong cả hai trường hợp. Một lợi điểm nữa của việc dùng chung công nghệ cho cả giao diện Windows và giao diện Web là người xây dựng ứng dụng không nhất thiết phải quyết định trước loại giao diện nào được sử dụng. Miễn là máy client đáp ứng được những yêu cầu hệ thống để chạy XBAP, một ứng dụng có thể cung cấp cả giao diện Windows và giao diện Web, mà chỉ sử dụng phần lớn những đoạn mã giống nhau.

Mỗi ứng dụng XBAP được download khi cần từ một Web server, nên nó phải tuân theo những yêu cầu về an ninh khắt khe hơn đối với một ứng dụng Windows độc lập. Theo đó, XBAP chạy trong phạm vi sandbox an ninh do hệ thống an ninh truy nhập mã của. NET Framework cung cấp. XBAP chỉ chạy với các hệ thống Windows có cài đặt WPF và chỉ với Internet Explore phiên bản 6 và 7 trở lên.

### Các thành phần của WPF

Giống như các thành phần khác của .NET Framework, WPF tổ chức các chức năng theo một nhóm namespace cùng trực thuộc namespace System.Windows. Bất kể chức năng nào được sử dụng, cấu trúc cơ bản của mọi ứng dụng WPF đều gần như nhau. Là ứng dụng Windows độc lập hay là một XBAP, một ứng dụng WPF điển hình bao giờ cũng gồm một tập các trang XAML và phần code tương ứng được viết bằng C# hoặc Visual Basic, còn gọi là các file code-behind. Tất cả các ứng dụng đều kế thừa từ lớp chuẩn Application của WPF. Lớp này cung cấp những dịch vụ chung cho mọi ứng dụng, chẳng hạn như các biến lưu trữ trạng thái của ứng dụng, các phương thức chuẩn để kích hoạt hay kết thúc ứng dụng.

Mặc dù WPF cung cấp một nền tảng thống nhất để tạo giao diện người dùng, những công nghệ mà WPF chứa đựng có thể phân chia thành những thành phần độc lập. Nhân của WPF là cơ chế tạo sinh đồ họa dựa trên vector và độc lập với độ phân giải nhằm tận dụng những lợi thế của phần cứng đồ họa hiện đại. WPF được mở rộng với các tập tính năng phát triển ứng dụng bao gồm XAML, các control, cơ chế móc nối dữ liệu, layout, đồ họa 2 chiều, ba chiều, hoạt họa, style, khuôn dạng mẫu, văn bản, media, text và in ấn. WPF nằm trong .NET Framework, nên ngoài ra, ứng dụng WPF có thể kết hợp các thành phần khác có trong thư viện lớp của .NET Framework.



Hình ‑ Các thành phần cơ bản của WPF.

#### Layout và Control

##### Layout

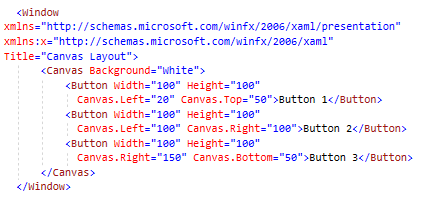
Để sắp đặt các thành phần khác nhau trên giao diện, ứng dụng WPF sử dụng panel. Mỗi panel có thể chứa các thành phần con, bao gồm các control như nút bấm hay hộp thoại, hay bản thân những panel khác. Những loại panel khác nhau cho phép sắp xếp thành phần con theo những cách khác nhau. Sau đây là một số layout panel thông dụng trong wpf:

###### Canvas:

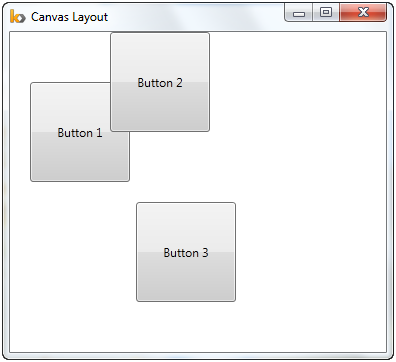
Canvas cho phép bố trí các control bằng cách xác định vị trí cố định của chúng. Sử dụng cách này khiến cho giao diện trở nên thiếu linh hoạt và gây ra nhiều hạn chế, như khi độ phân giải hoặc kích thước cửa sổ thay đổi. Vì thế bạn nên hạn chế dùng control này nếu có thể.Trong Canvas, bạn gán tọa độ cho các control con thông qua các thuộc tính (attached property) Canvas.Left, Canvas.Right, Canvas.Top, Canvas.Bottom.

Cần lưu ý là Left và Top có độ ưu tiên cao hơn Right và Bottom. Nghĩa là nếu bạn gán giá trị cho cả Canvas.Left và Canvas.Right, khi đó tọa độ theo chiều ngang của control sẽ lấy từ Canvas.Left, tương tự với Canvas.Top và  Canvas.Bottom.

Ví dụ:



Kết quả:

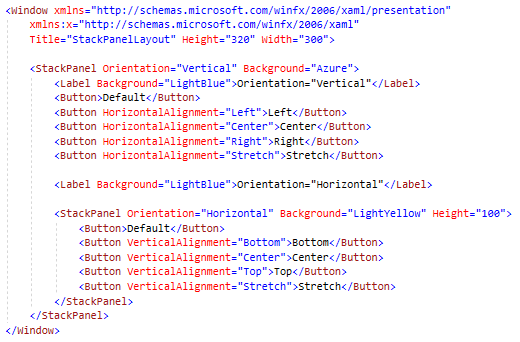


###### StackPanel:

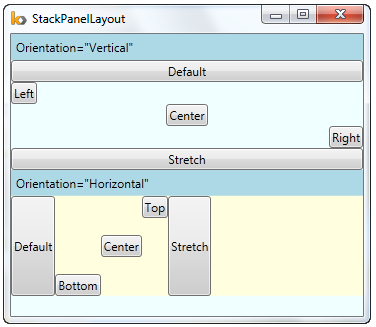
Control này sẽ sắp xếp các control con của nó theo dòng hoặc cột tùy theo giá trị của thuộc tính **Orientation** là Vertical hay Horizontal. Các control con sẽ được sắp xếp với vị trí liên tiếp và không chồng lên nhau.

Giá trị mặc định của thuộc tính Orientation là Vertical, các control sẽ được sắp xếp theo chiều dọc từ trên xuống dưới. Ngược lại là Horizontal, chúng sẽ được sắp xếp từ trái sang phải.

Ví dụ dưới đây cho thấy hai StackPanel lồng nhau. StackPanel bên ngoài sẽ co giãn khít với kích thước của Window và xếp các control theo chiều dọc và StackPanel bên trong xếp theo chiều ngang:



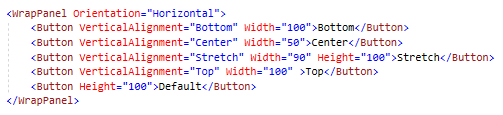
Kết Quả:



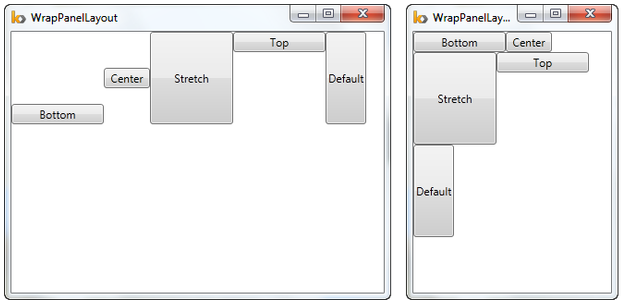
###### WrapPanel:

Cũng sắp xếp các control lần lượt theo hàng hoặc cột, nhưng đặc điểm chính của WrapPanel là sẽ tự động cho các control sang hàng/cột mới nếu như kích thước của hàng/cột còn lại không đủ chứa control.

Để thay đổi chiều sắp xếp các control con, bạn sử dụng thuộc tính Orientation với hai giá trị là Horizontal và Vertical; giá trị mặc định là Horizontal.



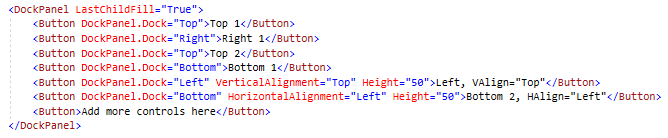
Kết quả:



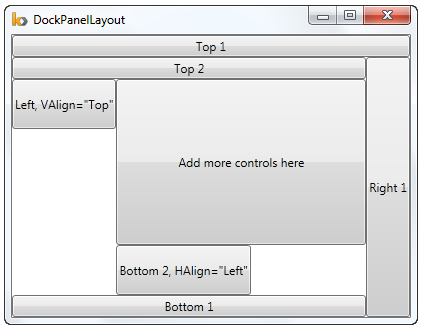
###### DockPanel:

Khái niệm Dock không lạ lẫm gì nếu bạn đã lập trình GUI trong .NET hoặc Java. Khi sử dụng container này, các control con sẽ được gắn thêm một attached property là DockPanel.Dock cho phép bạn gán các giá trị: Left, Right, Top, Bottom. Các giá trị tương ứng với mỗi phần không gian trong DockPanel, phần còn lại ở giữa sẽ được dùng để chứa các control khác.

Thuộc tính LastChilFill có giá trị mặc định true nhằm xác định các control thêm vào sau sẽ lấp đầy khoảng trống còn lại. Nếu bạn muốn giữ lại khoảng trống này, hãy gán giá trị của nó trở thành false.



Kết quả:



###### Grid:

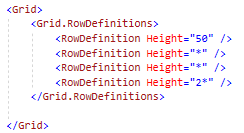
Control này sắp xếp các control con trong một lưới giống như table trong html. Grid sẽ chia không gian thành các phần với kích thước tùy ý được ngăn cách bởi cách đường lưới. Mặc định các đường lưới này sẽ không hiển thị, để hiển thị chúng bạn chỉ cần thay đổi giá trị của thuộc tính Grid.ShowGridLines thành true.

Để sử dụng control này, trước tiên bạn cần định nghĩa cấu trúc của Grid thông qua hai collection là Grid.RowDefinitions và Grid.ColumnDefinitions. Với mỗi dòng, cột bạn cần thêm một đối tượng RowDefinition/ColumnDefinition vào collection tương ứng.

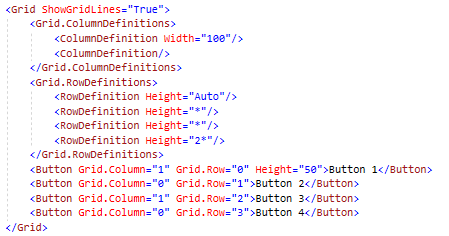
Với mỗi dòng/cột, bạn có thể thiết lập kích thước cho chúng bằng cách dùng giá trị cố định, tự động (Auto) hoặc tỷ lệ. Với kiểu Auto, kích thước của ô lưới chỉ dùng vừa đủ để chứa các control bên trong nó.

Mặc dịnh các ô lưới sẽ được chia theo tỷ lệ bằng nhau dựa theo số lượng dòng/cột. Cách chia này này sử dụng dấu sao “\*” làm đơn vị tính.

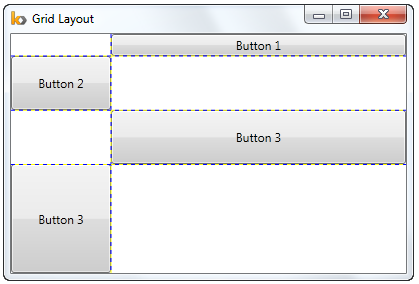
Ví dụ sau chia Grid làm 4 dòng, trong đó dòng 1 và dòng 2 bằng nhau, dòng 3 có kích thước gấp đôi dòng 1. Vì dòng đầu tiên có chiều cao là 100, nên chiều cao của dòng 1 sẽ bằng chiều cao của Grid chia cho 4.



Sau khi đã định nghĩa cấu trúc của Grid, bạn thêm các control vào một ô xác đinh trong lưới thông qua hai attached property là Grid.Row và Grid.Colum, như ví dụ sau:



Kết quả:



##### Control

Giống như bất kỳ một công nghệ giao diện người dùng nào, WPF cung cấp một số lượng lớn các control. Ngoài ra, người dùng có thể tùy ý định nghĩa các control theo ý mình. Các control chuẩn gồm Button, Label, TextBox, ListBox, Menu, Slider, hay phức tạp hơn có SpellCheck, PasswordBox… Các sự kiện do người dùng tạo ra, như di chuyển chuột hay ấn phím, có thể được các control nắm bắt và xử lý. Trong khi các control và các thành phần giao diện khác có thể được đặc tả đầy đủ bằng XAML, các sự kiện bắt buộc phải được xử lý bằng mã trình.

###### Label

Nhãn (Label) là các điều kiển để hiển thị các văn bãn tĩnh, thường được sử dụng để làm nhãn cho các control khác như Textbox, ListBox, ComboBox,…

|  |  |
| --- | --- |
| <Grid>  <Label Height=”30″ HorizontalAlignment=”Left” Margin=”10,15,0,0″ Name=”label1″  VerticalAlignment=”Top” Width=”60″>Họ đệm:</Label>  </Grid> | untitled |

Nhãn đuợc bắt đầu <Label> và kết thúc là </Label>, nội dung cũa nhãn là đoạn văn bản đặt giữa cặp thẻ này. Trong ví dụ này “Họ đệm:” là nội dung của nhãn.

Bên trong thẻ <Label> có rất nhiều đặc tính để mô tả về thẻ, trong đó:

* Height=”30” : Độ cao của khung nhãn là 30px
* HorizontalAlignment=”Left” : Nhãn được căn trái trong cửa sổ
* Margin=”10,15,0,0″ : có 4 giá trị là Left,Top,Right,Bottom
* Name=”label1″ : Tên của nhãn là lablel1
* VerticalAlignment=”Top” :Nhãn được căn theo đỉnh của cửa sổ.
* Width=”60″: Chiều rộng của nhãn là 60px

###### TextBox

Hộp soạn thảo (TextBox) là control cho phép người dùng nhập dữ liệu dạng văn bản:

|  |  |
| --- | --- |
| <Grid>  <TextBox Height=”30″ Margin=”80,17,30,0″ Name=”textBox1″ VerticalAlignment=”Top” />Hộp soạn thảo</TextBox>  <TextBox Height=”30″ Margin=”80,52,30,0″ Name=”textBox2″ VerticalAlignment=”Top” />  </Grid> | untitled |

Hộp soạn thảo được tạo nên bởi thẻ <TextBox/>. Nếu muốn thiết lập sẵn nội dung mặc định cho hộp soạn thảo, ta đặt nội dung này vào giữa cặp thẻ <TextBox/> Nội dung </TextBox>.  Nếu không muốn đặt giá trị mặc định thì không cần thẻ đóng </TextBox>. Thẻ <TextBox/>  cũng có nhiều đặc tính, trong đó:

* + Margin=”80,17,30,0“: Cách lề trái 80, đỉnh cửa sổ 17, cạnh phải 30
  + Name=”textBox1″: Tên của hộp soạn thảo là textBox1
  + VerticalAlignment=”Top”: Căn theo đỉnh cửa sổ

Đặc điểm của hộp soạn thảo với các đặc tính trên là khi người dùng co dãn, thay đổi kích thước cửa sổ, chiều rộng của hộp soạn thảo tự động co dãn theo.

###### Button

Nút bấm (Button) là loại điều khiển cho phép người dùng nhấn chuột để chọn lệnh, khi nhấn vào nút bấm, nó sẽ sinh ra sự kiện Click  và sẽ chạy các lệnh gắn với sự kiện này.

|  |  |
| --- | --- |
| <Grid>          <Button Height=”35″ HorizontalAlignment=”Left” Margin=”16,0,0,27″ Name=”button1″ VerticalAlignment=”Bottom” Width=”110″ Click=”button1\_Click”>Xem thông tin</Button>          <Button Height=”35″ HorizontalAlignment=”Right” Margin=”0,0,24,27″ Name=”button2″ VerticalAlignment=”Bottom” Width=”110″>Nhập lại</Button>  </Grid> | untitled |

Nút bấm được bắt đầu bằng thẻ <Button> và kết thúc bằng thẻ </Button>. Nhãn của nút bấm được đặt trong cặp thẻ <Button> Nhãn nút bấm </Button>.

Nút bấm có nhiều đặc tính, trong đó:

* + Height=”35″: Chiều cao nút bấm là 35
  + Width=”110″: Chiều rộng là 110
  + HorizontalAlignment=”Left”: Căn theo lề trái
  + VerticalAlignment=”Bottom”: Căn theo đáy cửa sổ
  + Margin=”16,0,0,27″: Cách lề trái 16, cách đáy 27
  + Name=”button1″: Tên nút bấm là button1
  + Click=”button1\_Click”: Khi nhấn chuột vào nút sẽ kích hoạt phương thức button1\_Click()

###### RadioButton và CheckBox

Radio Button và CheckBox đều là điều khiển dạng hộp chọn.  Tuy nhiên, điểm khác biệt cơ bản giữa hai loại điều khiển này là:

* **Radio Button**: là hộp chọn theo nhóm, nghĩa là các hộp trong cùng một nhóm sẽ loại trừ nhau, tại một thời điểm người dùng chỉ được chọn một trong các mục. Vi dụ như hộp chọn giới tính, ta phải sử dụng radio vì tại một thời điểm chỉ cho phép chọn Nam hoặc Nữ
* **CheckBox**:  là hộp chọn mà người dùng có thể chọn một hoặc nhiều mục cùng một lúc. Vi dụ như mục chọn Ngoại ngữ, cho phép người dùng chọn đồng thời nhiều mục.

<Grid>

        <Label Height=”30″ HorizontalAlignment=”Left” Margin=”10,94,0,0″ Name=”label3″ VerticalAlignment=”Top” Width=”60″>Giới tính:</Label>

        <RadioButton Height=”22″ Margin=”80,99,0,0″ Name=”radioButton1″ VerticalAlignment=”Top” HorizontalAlignment=”Left” Width=”79″ GroupName=”GioiTinh” IsChecked=”True”>Nam</RadioButton>

        <RadioButton Height=”22″ HorizontalAlignment=”Right” Margin=”0,99,30,0″ Name=”radioButton2″ VerticalAlignment=”Top” Width=”79″ GroupName=”GioiTinh”>Nữ</RadioButton>

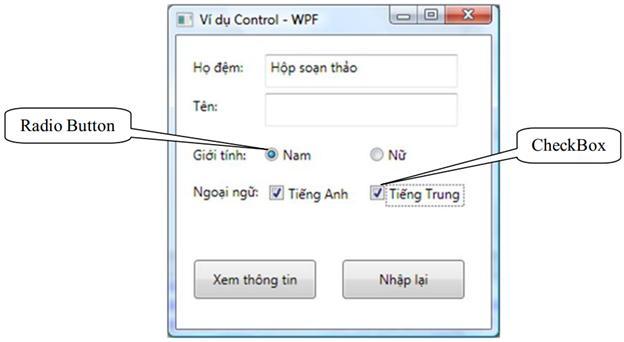
        <Label HorizontalAlignment=”Left” Margin=”10,127,0,105″ Name=”label4″

Width=”69″>Ngoại ngữ:</Label>

        <CheckBox Margin=”84,0,119,110″ Name=”checkBox1″ Height=”20″ VerticalAlignment=”Bottom” IsChecked=”True”>Tiếng Anh</CheckBox>

        <CheckBox HorizontalAlignment=”Right” Margin=”0,0,24,110″ Name=”checkBox2″ Width=”85″ Height=”20″ VerticalAlignment=”Bottom”>Tiếng Trung</CheckBox>

    </Grid>



**Radio Button** được tạo bởi thẻ <RadioButton> và kết thúc bởi</RadioButton>, giữa cặp thẻ này là nhãn của Radio Button <RadioButton> Nhãn </RadioButton>.

CheckBox được tạo bởi thẻ <CheckBox > và kết thúc bởi</CheckBox>, giữa cặp thẻ này là nhãn của CheckBox <CheckBox> Nhãn </CheckBox>.

Cả hai thẻ này đều có đặc tính IsChecked=”True” hoặc IsChecked=”False”. Mục nào có thuộc tính này sẽ được tự động chọn khi cửa sổ bắt đầu hiển thị.

Đối với Radio Button, vì là hộp chọn loại trừ, nếu trong một cửa sổ có nhiều nhóm Radio Button khác nhau thì các Radio Button của mỗi nhóm đuợc phân biệt bởi đặc tính GroupName=”TenNhom”. Ví dụ, trên cùng một cửa sổ có hai Radio Button chọn Giới tính (Nam; Nữ) và ba Radio Button khác chọn nghề nghiệp (Kinh doanh; Kỹ Thuật; Marketing) thì các Radio  Button Nam,  Nu phải có cùng GroupName với nhau, ba Radio  Button Kinh doanh, Ky Thuat, Marketting phải có cùng GroupName và khác với GroupName của nhóm giới tính.

###### ListBox

Hộp danh sách (ListBox) và là điều khiển hiển thị một danh sách các mục theo từng dòng và cho phép người dùng chọn một hay nhiều phẩn tử của danh sách.

|  |  |
| --- | --- |
| <Grid>          <Label Height=”30″ HorizontalAlignment=”Left” Margin=”10,0,0,126″ Name=”label5″ VerticalAlignment=”Bottom” Width=”69″>Quê quán:</Label>          <ListBox Height=”68″ Margin=”80,0,33,77″ Name=”listBox1″ VerticalAlignment=”Bottom” SelectedIndex=”0″>          <ListBoxItem>Hà nội</ListBoxItem>          <ListBoxItem>TP. Hồ Chí Minh</ListBoxItem>          <ListBoxItem>Hải Phòng</ListBoxItem>          <ListBoxItem>Đà Nẵng</ListBoxItem>          </ListBox>  </Grid> | untitled |

ListBox được tạo bởi thẻ <ListBox>  và kết thúc bằng thẻ đóng  </ListBox> .

Mỗi phần tử của danh sách nằm trong cặp thẻ <ListBoxItem> Nhãn </ListBoxItem> lồng bên trong cặp thẻ trên.

Đặc tính SelectedIndex=”k” để yêu cầu tự động chọn phần thử thứ n trong danh sách khi mở cửa sổ. Phần tử đầu tiên của danh sách có giá trị là 0, phần tử cuối cùng là n-1. Nếu muốn khi mở cửa sổ không chọn phần tử nào thì đặt giá trị k bằng -1.

###### Combobox

Hộp danh sách thả xuống (ComboBox) là các điều khiển hiển thị một danh sách theo từng dòng cho người dùng chọn. Tuy nhiên, khác với ListBox, ComboBox gọn gàng hơn bởi vì nó chỉ hiển thị 1 dòng và khi nhấn vào biểu tượng tam giác bên cạnh thì danh sách mới được mở ra. Combox chỉ cho phép chọn 1 dòng tại 1 thời điểm.

|  |  |
| --- | --- |
| <Grid>          <Label Height=”30″ HorizontalAlignment=”Left” Margin=”10,0,0,126″ Name=”label5″ VerticalAlignment=”Bottom” Width=”69″>Quê quán:</Label>          <ComboBox Height=”26″ Margin=”84,0,27,126″ Name=”comboBox1″ VerticalAlignment=”Bottom” SelectedIndex=”0″>          <ComboBoxItem>Hà nội</ComboBoxItem>          <ComboBoxItem>TP. Hồ Chí Minh</ComboBoxItem>          <ComboBoxItem>Hải Phòng</ComboBoxItem>          <ComboBoxItem>Đà Nẵng</ComboBoxItem>         </ComboBox>  </Grid> | untitled |

ComboBox được tạo bởi thẻ <ComboBox>  và kết thúc bằng thẻ đóng  </ComboBox> .

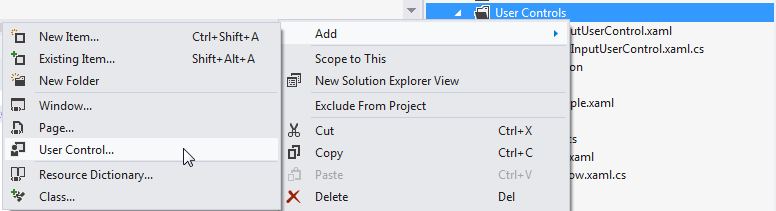
Mỗi phần tử của danh sách nằm trong cặp thẻ <ComboBoxItem> Nhãn </ComboBoxItem> Lồng bên trong cặp thẻ trên.

Đặc tính SelectedIndex=”k” để yêu cầu tự động chọn phần thử thứ n trong danh sách khi mở cửa sổ. Phần tử đầu tiên của danh sách có giá trị là 0, phần tử cuối cùng là n-1. Nếu muốn khi mở cửa sổ không chọn phần tử nào thì đặt giá trị k bằng -1.

##### UserControl

UserControl là một loại control do người dùng định nghĩa, thiết kế giao diện, chức năng cho nó. Cách sử dụng của UserControl cũng tương tự như các loại control thông thường khác. Sau khi xây dựng hoàn chỉnh, lập trình viên có thể sử dụng lại UserControl đó cho nhiều ứng dụng, nhiều cửa sổ khác nhau rất đơn giản. Các loại UserControl đều kế thừa từ class System.Windows.Controls.UserControl.

Cách tạo một usercontrol:



Trong ví dụ này, chúng ta sẽ tạo một User control có thể giới hạn số ký tự văn bản trong text box, đồng thời hiện thị bao nhiêu ký tự đã nhập và tổng cộng là bao nhiêu. Thấy đơn giản, nhưng nó được sử dụng trong rất nhiều ứng dụng web như Tweeter. Chức năng này cũng có thể thêm vào từng window riêng lẻ, nhưng sẽ hợp lý hơn nếu bạn muốn sử dụng chức năng cho nhiều nơi khác nhau trong ứng dụng, thế nên chúng sẽ wrap lại thành user control.

Code usercontrol:

<Grid>

<Grid.RowDefinitions>

<RowDefinition Height="Auto" />

<RowDefinition Height="\*" />

</Grid.RowDefinitions>

<Grid.ColumnDefinitions>

<ColumnDefinition Width="\*" />

<ColumnDefinition Width="Auto" />

</Grid.ColumnDefinitions>

<Label Content="{Binding Title}" />

<Label Grid.Column="1">

<StackPanel Orientation="Horizontal">

<TextBlock Text="{Binding ElementName=txtLimitedInput, Path=Text.Length}" />

<TextBlock Text="/" />

<TextBlock Text="{Binding MaxLength}" />

</StackPanel>

</Label>

<TextBox MaxLength="{Binding MaxLength}" Grid.Row="1" Grid.ColumnSpan="2" Name="txtLimitedInput" ScrollViewer.VerticalScrollBarVisibility="Auto" TextWrapping="Wrap" />

</Grid>

Và:

using System;

using System.Windows.Controls;

namespace WpfTutorialSamples.User\_Controls

{

public partial class LimitedInputUserControl : UserControl

{

public LimitedInputUserControl()

{

InitializeComponent();

this.DataContext = this;

}

public string Title { get; set; }

public int MaxLength { get; set; }

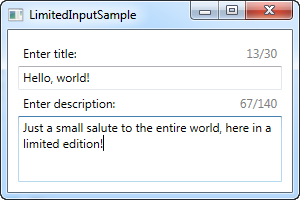
}

}

Từ trên xuống dưới vài từ ngoài vào trong: một Grid với 2 hàng và 2 cột. Hàng bên trên của Grid chứa 2 labels, một hiện thị tiêu đề và cái còn lại là thống kê. Chúng đều sủ dụng datta binding để liên kết những thông tin cần sử dụng - **Title** và **MaxLength** là những thuộc tính của một class, bên trong code.

Số ký tự hiện tại có thể lấy được bằng cách liên kết trực tiếp với thuộc tính Text.Length của TextBox sẽ được đặt ở hàng thứ 2, Kết quả giống như ảnh chụp màn hình bên trên. Lưu ý, do databinding được sử dụng, không cần update các label bên trong code C# nữa, và cả MaxLength của textBox, thay vào đó, chúng sẽ được liên kết trực tiếp với các thuộc tính.

Với đoạn code bên trên, chúng ta cần sử dụng User Control bên trong một window. Chúng ta thêm tham chiếu tới namespace chứa UserControl, ở phần trên cùng của file XAML trong Window:



#### Style và Template

Giống như sử dụng Cascading Style Sheets (CSS) đối với HTML, việc định ra thuộc tính đồ họa cho các đối tượng giao diện một lần, rồi sau đó áp dụng lại cho các đối tượng khác cùng loại thường rất tiện lợi. WPF cũng cung cấp tính năng tương tự bằng việc sử dụng thành phần Style của XAML. Ví dụ, kiểu ButtonStyle có thể được định nghĩa như sau:

<Style x:Key="ButtonStyle">

<Setter Property="Control.Background" Value="Red"/>

<Setter Property="Control.FontSize" Value="16"/>

</Style>

Bất kỳ nút bấm nào sử dụng kiểu này sẽ có nền màu đỏ và sử dụng font chữ kích thước 16.

Ví dụ:

<Button Style="{StaticResource ButtonStyle}"> Click Here </Button>

Một Style có thể được dẫn xuất từ một Style khác, thừa kế hoặc chồng lên những thuộc tính đã thiết lập. Mỗi style có thể định nghĩa các trigger cho phép tạo ra những hiệu ứng tương tác đặc biệt, chẳng hạn như khi lướt chuột qua nút bấm, nút bấm chuyển thành màu vàng. WPF cũng hỗ trợ sử dụng template. Mỗi template tương tự như một style, và ở hai dạng:

- Template cho dữ liệu: sử dụng thành phần DataTemplate của XAML để thiết lập một nhóm thuộc tính hiển thị của dữ liệu như màu sắc, phương thức căn lề...

- Template cho control: sử dụng thành phần ControlTemplate của XAML để định ra diện mạo của một control.

#### Text

Giao diện người dùng ít nhiều đều hiển thị chữ hay text. Đối với phần lớn mọi người, đọc text trên màn hình thường khó hơn đọc trên giấy in. Đó là do chất lượng hiển thị text trên màn hình kém hơn so với khi in ra giấy. WPF tập trung giải quyết vấn đề này, làm chất lượng text hiển thị trên màn hình tương đương trên giấy in. Cụ thể, WPF hỗ trợ các font chữ OpenType chuẩn, cho phép sử dụng các thư viện font đã có. WPF cũng hỗ trợ công nghệ font chữ mới ClearType, cho phép hiển thị các ký tự mịn hơn đối với mắt người, đặc biệt là trên màn hình tinh thể lỏng (LCD).

Để nâng cao hơn nữa chất lượng hiển thị text, WPF cho phép một số công nghệ khác như chữ ghép, theo đó một nhóm ký tự được thay thế bằng một ảnh đơn nhất, tạo tâm lý thoải mái hơn khi đọc đối với người dùng.

#### Văn bản

WPF hỗ trợ ba dạng văn bản: văn bản cố định (fixed), văn bản thích nghi (flow/adaptive) và văn bản XPS (XML Paper Specification). Kèm theo đó, WPF cũng cung cấp các dịch vụ để tạo, xem, quản lý, ghi chú, đóng gói và in ấn văn bản.

Văn bản cố định trông không đổi bất kể chúng được hiển thị trên màn hình hay in ra máy in. Trong WPF, những văn bản dạng này được định nghĩa bằng phần tử FixedDocument trong XAML và được hiển thị bằng control DocumentViewer.

Trong khi đó, văn bản thích nghi thường chỉ dùng để đọc trên màn hình, và có khả năng tự động thay đổi các thuộc tính hiển thị ảnh và text cho phù hợp với kích thước cửa số hay các yếu tố môi trường khác nhằm nâng cao chất lượng đọc cho người dùng. Văn bản thích nghi được định nghĩa bằng phần tử FlowDocument. Để hiển thị văn bản thích nghi, WPF sử dụng một số control khác nhau, chẳng hạn như FlowDocumentPageViewer, FlowDocumentScrollViewer, FlowDocumentReader…

Văn bản XPS xây dựng trên cơ sở văn bản bất động của WPF. XPS là một định dạng mở theo đặc tả XML, có khả năng sử dụng trên nhiều nền tảng khác nhau, được thiết kế nhằm tạo thuận lợi cho việc xây dựng, chia sẻ, in ấn và lưu trữ văn bản. Cũng như văn bản cố định, văn bản XPS được hiển thị bằng DocumentViewer.

#### Hình ảnh

Trong WPF, hình ảnh được hiển thị nhờ control **Image**, ví dụ:

<ImageWidth="200 Source="C:\DocumentsandSettings\AllUsers\Documents\MyPictures\Ava.jpg" />

Control Image có thể hiển thị hình ảnh lưu trữ dưới nhiều khuôn dạng khác nhau, bao gồm JPEG, BMP, TIFF, GIF và PNG. Nó cũng có thể hiển thị hình ảnh dạng Windows Media Photo mới được sử dụng trong Windows Vista. Bất kể ở khuôn dạng nào, WPF sử dụng Windows Imaging Component (WIC) để tạo ra hình ảnh. Cùng với các codec dùng cho các khuôn dạng ảnh kể trên, WIC cũng cung cấp một nền tảng chung để bổ sung codec khác.

#### Video và âm thanh

Khi tốc độ của các bộ xử lý và truyền thông mạng ngày một nâng cao, video trở thành một phần tương tác lớn của người dùng với phần mềm. Người dùng cũng sử dụng nhiều thời gian để nghe nhạc và các dạng âm thanh khác trên máy tính. Do đó, WPF cung cấp tính năng hỗ trợ cả hai dạng media này thông qua phần tử MediaElement. Control này có thể chơi các định dạng video WMV, MPEG và AVI, và nhiều định dạng âm thanh khác nhau. Việc lập trình để chạy một đoạn video trở nên khá đơn giản, như trong ví dụ sau:

<MediaElement Source= "..\My Videos\Ruby.wmv" />

#### Đồ họa hai chiều

Trong 20 năm gần đây, việc tạo ra đồ họa hai chiều trên Windows dựa trên Graphics Device Interface (GDI) và phiên bản sau của nó GDI+. Các ứng dụng Windows Forms phải sử dụng chức năng này thông qua một namespace khác hoàn toàn, bởi bản thân Windows Forms không tích hợp đồ họa 2 chiều. Đối với đồ họa 3 chiều thì càng tồi hơn, Windows Forms phải dựa trên công nghệ hoàn toàn biệt lập là Direct3D. Với WPF, vấn đề trở nên đơn giản hơn nhiều. Cả đồ họa 2 chiều và 3 chiều đều có thể được tạo ra trực tiếp trong XAML hoặc trong code sử dụng thư viện WPF tương ứng.

Đối với đồ họa 2 chiều, WPF định ra nhóm control của các khuôn hình (shapes) mà ứng dụng có thể sử dụng để tạo nên hình ảnh, gồm:

* + **Line**: vẽ đường thẳng qua 2 điểm.
  + **Elllipse**: vẽ ellipse.
  + **Rectangle**: vẽ chữ nhật.
  + **Polygon**: vẽ đa giác.
  + **Polyline**: vẽ đa giác mở.
  + **Path**: vẽ hình theo một đường bất kỳ.

Mỗi khuôn hình đều có các thuộc tính phong phú cho phép hiển thị với nhiều tính chất khác nhau: màu nền, màu biên… Một đặc điểm quan trọng trong WPF là: vì mọi thứ đều được xây dựng trên một nền chung, việc kết hợp các đặc tính và đối tượng khác nhau, chẳng hạn, lồng một ảnh vào một hình chữ nhật, trở nên đơn giản. Điểm thú vị nữa là các đối tượng hình học này còn có thể thu nhận các sự kiện từ phía người dùng như một control, chẳng hạn sự kiện nhắp chuột.

Ngoài ra, WPF cũng cung cấp một nhóm chức năng hình học khác, gọi là geometries, để làm việc với đồ họa hai chiều, như LineGeometry, RectangleGeometry, EllipseGeometry, và PathGeometry. Dạng hình học này có nhiều thuộc tính và chức năng tương tự như các khuôn hình đã nêu trên. Điểm khác biệt quan trọng nhất là các geometries không được dùng để hiển thị, chúng được dùng chủ yếu để tính toán hình học, ví dụ như để định ra các vùng miền, theo dõi vị trí bấm chuột...

Thêm vào đó, WPF cung cấp lớp Transform cho phép thực hiện các biến đổi hình học như xoay, dịch chuyển, co giãn đối tượng đồ họa; hoặc cho phép thực hiện các hiệu ứng hoạt họa theo thời gian thông qua các lớp Animation và Timing.

#### Đồ họa ba chiều

WPF hỗ trợ đồ họa 3 chiều bằng việc gói các lời gọi API của Direct3D, và do vậy, việc sử dụng chúng trở nên thống nhất và đơn giản hơn đáng kể. Để hiển thị đồ họa ba chiều, ứng dụng WPF sử dụng control Viewport3D. Để tạo ra các cảnh ba chiều, lập trình viên mô tả một hay nhiều mô hình, sau đó, phân định cách thức các mô hình này được chiếu sáng hay hiển thị. Như thường lệ, điều này được thực hiện bằng XAML, bằng code hay trộn cả hai. Để mô tả mô hình, WPF cung cấp lớp GeometryModel3D để tạo ra hình dạng của mô hình. Khi mô hình đã được định hình, diện mạo bên ngoài của nó có thể được điều khiển bằng việc phủ lên các vật liệu (material). Chẳng hạn, lớp SpecularMaterial cho phép tạo bóng trên bề mặt mô hình.

Bất kể được làm từ vật liệu gì, một mô hình có thể được chiếu sáng theo nhiều cách. Lớp DirectionalLight cho phép ánh sáng tới từ một hướng xác định, trong khi lớp AmbientLight tạo ra ánh sáng đồng đều trên mọi vật trong cảnh. Cuối cùng, để định ra cách nhìn cảnh, lập trình viên phải định ra một *camera*. Ví dụ, PerspectiveCamera cho phép phân định khoảng cách từ vị trí nhìn tới vật thể và kiểu nhìn phối cảnh (tuân theo luật gần xa).

Xây dựng cảnh ba chiều trực tiếp bằng XAML hay mã trình đều không đơn giản. Do đó, chỉ nên dùng ứng dụng WPF để hiển thị cảnh ba chiều, việc xây dựng cảnh nên được thực hiện bằng những công cụ đồ họa chuyên biệt.

#### Móc nối dữ liệu

Phần lớn các ứng dụng được tạo ra đều cung cấp cho người dùng phương tiện để xem và sửa đổi dữ liệu. Trong các ứng dụng WPF, việc lưu trữ và truy xuất dữ liệu đã được thực hiện bởi các công nghệ như Microsoft SQL Server và ADO.NET. Sau khi dữ liệu được truy xuất và tải vào các đối tượng quản lý dữ liệu trên ứng dụng, phần việc khó khăn của ứng dụng WPF mới bắt đầu. Về cơ bản, có hai công việc phải thực hiện:

1) Sao chép dữ liệu từ các đối tượng quản lý dữ liệu vào các control trên giao diện, qua đó, dữ liệu có thể được hiển thị hay sửa đổi.

2) Đảm bảo rằng những thay đổi trên dữ liệu từ các control được cập nhật trở lại các đối tượng quản lý dữ liệu.

Để đơn giản hóa quá trình phát triển ứng dụng, WPF cung cấp một cơ chế móc nối dữ liệu để thực hiện tự động những bước này. Phần nhân của cơ chế móc nối dữ liệu là lớp Binding mà nhiệm vụ của nó là liên kết control trên giao diện (đích) với đối tượng quản lý dữ liệu (nguồn). Mối quan hệ này được minh họa trong hình 4 dưới đây:



Hình ‑ Quan hệ giữa đối tượng dữ liệu và đối tượng phụ thuộc.

Việc hỗ trợ móc nối dữ liệu được xây dựng ngay từ nhân của WPF. Tất cả các đối tượng đồ họa trong WPF đều kế thừa từ DependencyObject, chúng là các đối tượng phụ thuộc. Chức năng mà lớp cơ sở này hỗ trợ cho phép thực hiện hiệu ứng hoạt họa, tạo kiểu mẫu (styling) và móc nối dữ liệu. Các đối tượng này đều mang một thuộc tính đặc biệt gọi là DependencyProperty, thuộc tính phụ thuộc. Phần lớn các thuộc tính hay dùng như Text, Content, Width, Height, vân vân đều là các thuộc tính phụ thuộc. Tất cả các thuộc tính phụ thuộc đều có thể tạo hiệu ứng hoạt họa, tạo kiểu và kết nối dữ liệu.

Cơ chế móc nối dữ liệu trong WPF còn cung cấp thêm những tính năng như xác thực tính hợp lệ, sắp xếp, lọc và phân nhóm dữ liệu. Thêm vào đó, tính năng móc nối dữ liệu cũng hỗ trợ sử dụng khuôn mẫu dữ liệu (data template) để tạo ra các đối tượng giao diện tùy biến có kết nối dữ liệu, khi các control chuẩn không phù hợp. Móc nối dữ liệu và khuôn dạng dữ liệu có thể được coi là tính năng mạnh nhất của WPF.

## Material Design

### Material Design là gì?

**Material Design**  là một ngôn ngữ thiết kế được phát triển vào năm 2014 bởi Google, được mở rộng dựa trên mô típ "Flat" . Thay vì mang đến cảm giác ‘phẳng lì’ trên toàn bộ giao diện, Material Design là những lớp Flat xếp chồng lên nhau, tạo chiều sâu và điểm nhấn hơn thiết kế phẳng thông thường.

Material Design chủ yếu tập trung vào những đường nét đơn giản, sử dụng những gam màu đậm, nổi bật. Đồng thời, chúng thường sử dụng những yếu tố đồ họa có cảm giác 3D, có hiệu ứng ‘nổi lên’ trên giao diện. Ngoài ra nó còn bao gồm những hiệu ứng, chuyển động như khi các button, các menu hiện lên màn hình…

### Đặc điểm của Material Design

Một số đặc điểm có thể dùng để ‘nhận dạng’ Material Design là:

1. Thường sử dụng những gam màu nổi bật, tuy nhiên vẫn cần thống nhất một mảng màu chủ đạo
2. Các biểu tượng (icon) phẳng, theo phong cách tối giản và dễ hiểu
3. Giao diện phẳng, ít hoặc không có hiệu ứng chuyển màu hay đổ bóng
4. Tận dụng nhiều khoảng trắng, giúp không gian giao diện khá thoáng và dễ chịu.
5. Hiệu ứng chuyển động tự nhiên, mượt mà, dễ để hình dung
6. Tránh đi sự nhàm chán: Nhờ có các màu sắc vui vẻ và sinh động hơn, hiệu ứng chuyển động lôi cuốn hơn

### Ba mảnh ghép của Material Design

1. Không gian: Để tạo chiều sâu cho một thiết kế, designer cần điều chỉnh ánh sáng một cách phù hợp, sao cho thành phẩm có chiều sâu như ‘chứa đựng’ một không gian bên trong vậy.
2. Ánh sáng: Có 2 loại nguồn sáng được kết hợp:
3. Nguồn sáng chiếu trực tiếp: nguồn sáng này rất quan trọng. Giống như nguồn sáng từ đèn pin, nó sẽ mang hiệu ứng đổ bóng mạnh và sắc nét.
4. Ánh sáng môi trường: để phân biệt, ánh sáng này thường nhẹ nhàng và không rõ nguồn, tạo viền bóng nhẹ xung quanh.

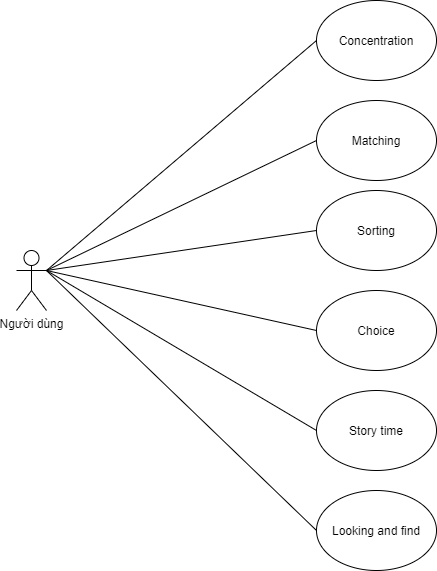
Thông thường, Material Design thường kết hợp cả 2 nguồn sáng, mang đến hiệu ứng bóng tổng hợp, mô phỏng không gian thực tế.

1. Material: Là những mặt phẳng có độ dày tương đồng, xếp chồng lên nhau. Trên mỗi Material là một nội dung được thiết kế theo nguyên tắc Flat Design. (ví dụ như button hay Animation chẳng hạn.

# : PHÂN TÍCH

## Phân tích yêu cầu bằng UML

### Use case Tổng Quát



Hình ‑ Use case tổng quát

### Danh sách các actor

|  |  |  |
| --- | --- | --- |
| **Tác nhân** | **Mô tả tác nhân** | **Ghi chú** |
| Người dùng | Người sử dụng ứng dụng để học tiếng anh |  |

Bảng ‑ Danh sách các Actor.

### Danh sách các Use case

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **Tên Use case** | **Mô tả ngắn gọn Use case** | **Chức năng** |
| UC1 | Concentration | Ứng dụng sẽ hiển thị sáu hình ảnh bị lật úp, trong đó bao gồm ba cặp hình ảnh, khi người dùng chọn hai hình bị úp để mở hình ảnh, âm thanh phát âm từ vựng của hình ảnh đó sẽ được phát. Lật lần lượt hai hình để tìm các hình giống nhau, nếu lật hai hình khác nhau thì sẽ phát âm thanh báo sai và hai hình đó sẽ bị úp trở lại. Dùng phải lật đúng cả ba cặp hình để hoàn thành trò chơi |  |
| UC2 | Matching | Ứng dụng sẽ hiển thị ba ô âm thanh và ba ô hình ảnh tương ứng, khi người dụng chọn ô âm thanh, phát âm của một từ vựng sẽ được phát, dùng phải lắng nghe để chọn ô hình ảnh đúng với từ vựng đó để nối hai ô lại với nhau. Ghép thành công ba cặp để hoàn thành trò chơi. |  |
| UC3 | Sorting | Ứng dụng sẽ hiển thị ba hình ảnh và ba ô trống được đánh số theo thứ tự, lần lượt phát âm thanh yêu cầu đặt hình một hình vào một ô có số thứ tự nhất định, người dùng phải đặt đúng theo yêu cầu cả ba hình ảnh vào ba ô để hoàn thành trò chơi. |  |
| UC4 | Multiple Choice | Ứng dụng sẽ hiển thị ba hình ảnh và lần lượt phát âm từ vựng của một trong ba hình ảnh đó, người dùng phải chọn hình ảnh theo từ vựng nghe được. Thực hiện ba lượt chọn để hoàn thành trò chơi. |  |
| UC5 | Painting | Ứng dụng sẽ hiển thị 3 hình ảnh trắng cần tô màu và danh sách các màu sắc cho người dùng chọn, hệ thống sẽ yêu cầu người dùng tô màu cho các hình ảnh đó, người dùng phải lắng nghe và chọn màu cho để tô hình ảnh theo yêu cầu. |  |
| UC6 | Story time | Ứng dụng sẽ hiển thị một số câu giao tiếp, mẫu câu thông dụng kèm hình ảnh và âm thanh minh họa, người dùng sẽ lắng nghe và ghi nhớ. |  |
| UC7 | Looking and find | Ứng dụng sẽ hiển thị một hình ảnh nền và yêu cầu người dùng tìm và chọn hình ảnh của một từ vựng nào đó trong hình nền. |  |

Bảng ‑ Danh Sách Các Use case chức năng.

## Đặc tả các yêu cầu chức năng

### Use case UC1: Concentration

#### Mô tả:

|  |  |  |
| --- | --- | --- |
| Use case: Concentration | | |
| Mục đích | Use case cho phép người dùng học tiếng anh qua trò chơi gép các cặp hình giống nhau. | |
| Mô tả | Use case sẽ hiển thị sáu hình ảnh bị lật úp, trong đó bao gồm ba cặp hình ảnh, khi người dùng chọn hai hình bị úp để mở hình ảnh, âm thanh phát âm từ vựng của hình ảnh đó sẽ được phát. Lật lần lượt hai hình để tìm các hình giống nhau, nếu lật hai hình khác nhau thì sẽ phát âm thanh báo sai và hai hình đó sẽ bị úp trở lại; nếu 2 hình giống nhau thì phát âm thanh thông báo chính xác và ẩn 2 hình đó đi. Người dùng phải lật đúng cả ba cặp hình để hoàn thành trò chơi | |
| Tác Nhân | Người dùng | |
| Điều kiện trước | Đã chọn bài học | |
| Điều kiện sau | Tất cả các hình đã đoán đúng bị ẩn đi.  Phát thông báo chúc mừng | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 1. Chọn hình thứ nhất.  3. chọn hình thứ hai. | 2. Phát âm thanh phát âm từ vựng của hình ảnh đó và lật hình lên.  4. Phát âm thanh phát âm từ vựng của hình ảnh đó và lật hình lên.  5. Kiểm tra hai hình ảnh đã chọn.  5.1 Nếu hai hình ảnh giống nhau thì phát âm thanh thông báo chính xác và ẩn hai hình đó đi.  5.2 Nếu hai hình ảnh khác nhau thì phát âm báo sai, lật úp hai hình đó lại và chuyển sang b7.  6. Nếu đã ghép đúng ba cặp hình thì thông báo chúc mừng và kết thúc.  7. Quay lại b1. |
| Luồng sự kiện phụ |  | |

Bảng ‑ Đặc tả Use case Concentration.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC2: Matching

#### Mô tả:

|  |  |  |
| --- | --- | --- |
| Use case: Matching | | |
| Mục đích | Use case cho phép người dùng học tiếng anh qua trò chơi gép nối các ô âm thanh với hình ảnh tương ứng. | |
| Mô tả | Use case này sẽ hiển thị ba ô âm thanh và ba ô hình ảnh tương ứng, khi người dụng chọn ô âm thanh, phát âm của một từ vựng sẽ được phát, dùng phải lắng nghe để chọn ô hình ảnh đúng với từ vựng đó để nối hai ô lại với nhau. Ghép thành công ba cặp để hoàn thành trò chơi. | |
| Tác Nhân | Người dùng. | |
| Điều kiện trước |  | |
| Điều kiện sau |  | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 1. Người dùng chọn một ô âm thanh.  3. Người dùng chọn một hình ảnh. | 2. Hệ thông phát âm thanh của ô được chọn.  4. Hệ thống kiểm tra ô âm thanh và hình ảnh đã chọn.  4.1 Nếu ô âm thanh chứa phát âm tương ứng của hình ảnh đã chọn thì phát âm thanh chính xác và nối ô âm thanh với hình ảnh đã chọn lại với nhau.  4.2 Nếu ô âm thanh không chứa phát âm tương ứng của hình ảnh đã chọn thì phát âm thanh báo sai và chuyển sang b6.  5. Nếu đã nối được ba cặp thì thông báo chúc mừng và kết thúc use case.  6. Quay lại b1. |
| Luồng sự kiện phụ |  | |
| Ghi chú |  | |

Bảng ‑ Đặc tả Use case Matching.

#### 3.2.2.2 Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC3: Sorting

#### Mô tả:

|  |  |  |
| --- | --- | --- |
| Use case: Sorting | | |
| Mục đích | Use case cho phép người dùng học tiếng anh qua trò chơi sắp xếp hình ảnh vào các ô được đánh số theo thứ tự được yêu cầu. | |
| Mô tả | Use case này sẽ hiển thị ba hình ảnh và ba ô trống được đánh số theo thứ tự, lần lượt phát âm thanh yêu cầu đặt hình một hình vào một ô có số thứ tự nhất định, người dùng phải đặt đúng theo yêu cầu cả ba hình ảnh vào ba ô để hoàn thành trò chơi. | |
| Tác Nhân | Người dùng | |
| Điều kiện trước |  | |
| Điều kiện sau |  | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 2. Người dùng kéo hình ảnh vào một ô trống. | 1. Hệ thống phát âm thanh yêu cầu đặt một hình ảnh vào ô có số thứ tự nào đó.  3. Hệ thống kiểm tra  3.1 Nếu người dùng xếp đúng hình ảnh vào ô đúng theo yêu cầu thì phát thông báo chính xác.  3.2 Nếu người dùng xếp không đúng theo yêu cầu thì phát thông báo sai, đặt hình ảnh về vị trí ban đầu.  4. Nếu cả 3 hình ảnh đều được đặt đúng vị trí thì hiển thị thông báo chúc mừng và kết thúc use case.  5. Quay lại b1. |
| Luồng sự kiện phụ |  | |

Bảng ‑ Đặc tả Use case Sorting.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC4: Multiple Choice

#### Mô tả

|  |  |  |
| --- | --- | --- |
| Use case: Multiple Choice | | |
| Mục đích | Use case cho phép người dùng học tiếng anh qua trò chơi lựa chọn hình ảnh theo từ vựng mà ứng dụng đưa ra. | |
| Mô tả | Use case này sẽ hiển thị ba hình ảnh và lần lượt phát âm từ vựng của một trong ba hình ảnh đó, người dùng phải chọn hình ảnh theo từ vựng nghe được. Thực hiện ba lượt chọn đúng để hoàn thành trò chơi. | |
| Tác Nhân | Người dùng | |
| Điều kiện trước |  | |
| Điều kiện sau |  | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 2. Chọn hình ảnh | 1. Hiển thị ba hình ảnh và phát âm thanh phát âm của một từ vựng.  3. Kiểm tra hình được chọn  3.1 Nếu hình ảnh đúng với từ vựng ứng dụng vừa phát thì phát thông báo chính xác.  3.2 Nếu hình ảnh không đúng với từ vựng được phát thì phát thông báo không chính xác và chuyển sang b5.  4. Nếu đã chọn đúng ba lần thì phát thông báo chúc mừng hoàn thành và kết thúc.  5. Quay lại B1. |
| Luồng sự kiện phụ |  | |

Bảng ‑ Đặc tả Use case Multiple Choice.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC5: Painting

#### Mô tả

|  |  |  |
| --- | --- | --- |
| Use case: Painting | | |
| Mục đích | Use case cho phép người dùng học từ vựng thông qua trò chơi tô màu những hình ảnh theo yêu cầu. | |
| Mô tả | Use case này sẽ hiển thị những hình ảnh chưa được tô màu và một danh sách các màu sắc để người dùng sử dụng, hệ thống sẽ phát âm thanh yêu cầu người dùng chọn màu để tô cho các hình ảnh theo yêu cầu được đưa ra, người dùng chọn đúng màu để tô cho hình ảnh theo yêu cầu đó. | |
| Tác Nhân | Người dùng | |
| Điều kiện trước | Chọn Unit và chọn mục Painting | |
| Điều kiện sau |  | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 2. Người dùng lắng nghe và kéo màu vào hình ảnh để tô màu cho hình ảnh đó. | 1. Phát âm thanh yêu cầu tô một hình ảnh bằng một màu trong danh sách các màu sắc hiển thị trên giao diện.  3. Kiểm tra màu và hình ảnh người dùng vừa chọn.  3.1 nếu đúng theo yêu cầu vừa phát thì phát âm thanh báo chính xác, chuyển màu cho hình ảnh.  3.2 Nếu người dùng chọn sai màu hoặc tô sai hình ảnh thì phát âm thanh báo không chính xác và chuyển sang b5.  4. Nếu đã tô đúng màu cho cả 3 hình ảnh thì hiển thị thông báo chúc mừng hoàn thành và kết thúc usecase.  5. Quay lại b1. |
| Luồng sự kiện phụ |  | |
| Ghi chú | Có 3 hình ảnh cần được tô màu trong use case này | |

Bảng ‑ Đặc tả Use case Painting.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC6: Story time

#### Mô tả

|  |  |  |
| --- | --- | --- |
| Use case: Story time | | |
| Mục đích | Use case cho phép người dùng học những câu giao tiếp, những mẫu câu thông dụng cơ bản bằng các hình ảnh và âm thanh minh họa. | |
| Mô tả | Use case này sẽ hiển thị những câu giao tiếp, những mẫu câu cơ bản kèm với hình ảnh và âm thanh đọc minh họa | |
| Tác Nhân | Người dùng | |
| Điều kiện trước | Chọn bài học và chọn mục story time | |
| Điều kiện sau |  | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 3. Người dùng chọn từ. | 1. Hiển thị các câu giao tiếp, mẫu câu của bài học và danh sách hình ảnh minh họa.  2. Phát file âm thanh tổng của bài học.  4. Phát file âm thanh phát âm của từ đó. |
| Luồng sự kiện phụ |  | |

Bảng ‑ Đặc tả Use case Story time.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

### Use case UC7: Looking and find

#### Mô tả

|  |  |  |
| --- | --- | --- |
| Use case: Looking and find | | |
| Mục đích | Use case cho phép người dùng học từ vựng qua trò chơi tìm vật trong hình. | |
| Mô tả | Use case này sẽ hiển thị một hình nền lớn và yêu cầu người dùng tìm một vật trong hình đó. Người dùng phải lắng nghe và tìm vật được yêu cầu từ trong hình. | |
| Tác Nhân | Người dùng | |
| Điều kiện trước | Chọn bài học và chọn mục Looking and find | |
| Điều kiện sau | Thông báo chúc mừng hoàn thành | |
| Luồng sự kiện chính | Người dùng | Hệ thống |
| 3. Chọn vật từ trong hình. | 1. Hiển thị hình ảnh nền.  2. Phát âm thanh phát âm của vật cần tìm trong hình.  4. Kiểm tra vật được chọn.  4.1 Nếu đó là vật cần tìm thì phát thông báo chính xác.  4.2 Nếu đó không phải vật cần tìm thì sang bước 6.  5. Nếu đã tìm được ba vật được yêu cầu thì thông báo chúc mừng hoàn thành và kết thúc use case.  6. Quay lại B3. |
| Luồng sự kiện phụ |  | |

Bảng ‑ Đặc tả Use case Looking and find.

#### Biểu đồ

##### Activity Diagram



##### Sequence Diagram

# : THIẾT KẾ VÀ HIỆN THỰC

Chương này sinh viên trình bày quá trình thiết kế, thử nghiệm và hiện thực hệ thống hoặc bài toán.

## Tiêu đề mục

### Tiêu đề tiểu mục

Nội dung tiểu mục

### Tiêu đề tiểu mục

Nội dung tiểu mục

## Tiêu đề mục

### Tiêu đề tiểu mục

Nội dung tiểu mục

### Tiêu đề tiểu mục

Nội dung tiểu mục

# : KẾT LUẬN

## Kết quả đạt được

Trình bày các chức năng đã hoàn thành trong đồ án hoặc kết quả nghiên cứu.

## Hạn chế của đồ án

Trình bày những hạn chế, công việc chưa hoàn tất trong đồ án

## Hướng phát triển

Trình bày các định hướng phát triển cho hệ thống hoặc hướng nghiên cứu trong tương lai.

# TÀI LIỆU THAM KHẢO

Các tài liệu Tiếng Việt

1. Giáo trình học tiếng anh Big Fun theo hệ thống trường học [TOMATO Children’s Home](https://tomato.edu.vn/)

Các tài liệu Tiếng Anh

1. Amy Apon. Lecture for Cluster and Grid Computing. University of Arkansas, 2004.

Các tài liệu từ Internet

1. Website: <https://csharpcanban.com/>
2. ...

# PHỤ LỤC