ĐẠI HỌC THỦ DẦU MỘT

KHOA KỸ THUẬT CÔNG NGHỆ



**BÁO CÁO ĐỀ TÀI MÔN HỌC**

**CÔNG NGHỆ XML VÀ ỨNG DỤNG**

Đề tài

**XÂY DỰNG ỨNG DỤNG ANDROID ĐỌC DỮ LIỆU LƯU TRỮ BẰNG XML**

**Giảng viên:** ThS. Nguyễn Đình Thọ

**Nhóm sinh viên:** 9

Trần Hồ Bảo Hoàng – 152480103037

Nguyễn Thị Mộng Thường – 1524801030015

Phạm Minh Đức – 1524801030004

Lê Văn Chức –1424801030223

**Chuyên ngành:** Kỹ thuật phần mềm

**Lớp:** D15PM01

Bình Dương, Tháng 12/2018

ĐẠI HỌC THỦ DẦU MỘT

KHOA KỸ THUẬT CÔNG NGHỆ



**BÁO CÁO ĐỀ TÀI MÔN HỌC**

**CÔNG NGHỆ XML VÀ ỨNG DỤNG**

Đề tài

**XÂY DỰNG ỨNG DỤNG ANDROID ĐỌC DỮ LIỆU LƯU TRỮ BẰNG XML**

**Giảng viên:** ThS. Nguyễn Đình Thọ

**Nhóm sinh viên:** 9

Trần Hồ Bảo Hoàng – 152480103037

Nguyễn Thị Mộng Thường – 1524801030015

Phạm Minh Đức – 1524801030004

Lê Văn Chức –1424801030223

**Chuyên ngành:** Kỹ thuật phần mềm

**Lớp:** D15PM01

Bình Dương, Tháng 12/2018

**LỜI CẢM ƠN**

Tuy đã cố gắng tham khảo các tài liệu liên quan, cùng bài giảng của thầy cô và sự nỗ lực của cả nhóm nhưng em vẫn không thể tránh khỏi những sai sót. Vậy kính mong quý thầy cô giúp đỡ để chúng em có thể nắm vững hơn kiến thức mà mình học được.

Chân thành gửi lời cảm ơn đến Ban giám hiệu trường Đại học Thủ Dầu Một và quý thầy cô trong khoa kỹ thuật công nghệ đã đem tâm huyết truyền đạt kiến thức cho chúng em. Cảm ơn thầy **Nguyễn Đình Thọ** đã hướng dẫn chúng em hoàn thành bài đồ án này. Và cảm ơn tập thể lớp D15PM01 đã luôn ủng hộ và giúp đỡ em.

Một lần nữa chúng em xin chân thành cảm ơn!

Bình Dương, ngày 22 tháng 12 năm 2018

***Nhóm sinh viên thực hiện***

TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT

KHOA CÔNG NGHỆ THÔNG TIN

**NHẬN XÉT VÀ CHẤM ĐIỂM CỦA GIẢNG VIÊN**

Họ và tên giảng viên: **Nguyễn Đình Thọ**

Tên đề tài: Xây dựng ứng dụng android đọc dữ liệu lưu trữ bằng xml

Nội dung nhận xét:

**Điểm:**

Bằng số:

Bằng chữ:

|  |  |
| --- | --- |
|  | **GIẢNG VIÊN CHẤM**  *(Ký, ghi rõ họ tên)* |

**ThS. Nguyễn Đình Thọ**

**MỤC LỤC**

**DANH MỤC HÌNH**

**DANH SÁCH CÁC KÝ TỰ, CÁC CHỮ VIẾT TẮT**

|  |  |
| --- | --- |
| Từ viết tắt | Giải thích |
| XML | eXtensible Markup Language |
| DOM | Document Object Model |
|  |  |
|  |  |
|  |  |

# LỜI MỞ ĐẦU

Ngày nay, việc ứng dụng công nghệ thông tin vào đời sống được phát triển vô cùng mạnh mẽ. Ngành công nghệ thông tin nói chung và lĩnh vực công nghệ mạng nói riêng đã cho ra đời nhiều ứng dụng, dịch vụ đặc biệt kể đến là dịch vụ “aplication as service”. Các giao dịch, tương tác đều thông qua môi trường internet. Cùng với sự phát triển đó là sự ra đời của các ngôn ngữ và các giao thức giao tiếp để phục vụ những mục đích và dịch vụ nhất định. Có thể kể đến một số ngôn ngữ HTML, Java, PHP, .Net

Nền tảng Android là một nền tảng phát triển di động mã nguồn mở, hiện đại và là SDK cho các thiết bị di động. Với hệ điều hành này, chúng ta có thể tạo ra các ứng dụng di động rất mạnh. Nó giúp truy cập vào tất cả các khía cạnh của thiết bị di động mà nó chạy trên đó, từ các đồ họa cấp thấp, đến phần cứng như là thiết bị camera trên điện thoại. Với rất nhiều thứ có thể sử dụng Android và đặc biệt hơn các ứng dụng có thể truy cập các dịch vụ Web. Và XML thường được dùng như là một định dạng dữ liệu trên Internet. Nếu muốn truy cập dữ liệu từ Internet, các khả năng có thể là dữ liệu sẽ ở dạng XML. Nếu chúng ta muốn gửi dữ liệu đến một dịch vụ Web, có thể chúng ta cũng cần gửi cả dữ liệu XML. Nếu ứng dụng Android muốn mở rộng trên Internet, thì chúng ta cần phải làm việc với XML.

Chính vì vậy nhóm 9 đã làm bài báo cáo với đề tài “Xây Dựng Ứng Dụng Android Hiển Thị Dữ Liệu Từ XML”.

# TỔNG QUAN VỀ XML

## Lịch sử XML

Vào giữa những năm 1990, các chuyên gia SGML đã có kinh nghiệm với World Wide Web (vẫn còn khá mới vào thời đó). Họ tin tưởng rằng SGML có thể cung cấp giải pháp cho các vấn đề mà Web đang gặp phải. Jon Bosak đưa ra ý kiến W3C nên tài trợ một chương trình mang tên "SGML trên Web". Sau thời gian này một tập con của SGML ra đời mang tên XML.

## Định nghĩa

XML (viết tắt từ tiếng Anh: *eXtensible Markup Language*, tức "Ngôn ngữ đánh dấu mở rộng") là ngôn ngữ đánh dấu với mục đích chung do W3C đề nghị, để tạo ra các ngôn ngữ đánh dấu khác. Đây là một tập con đơn giản của SGML, có khả năng mô tả nhiều loại dữ liệu khác nhau. Mục đích chính của XML là đơn giản hóa việc chia sẻ dữ liệu giữa các hệ thống khác nhau, đặc biệt là các hệ thống được kết nối với internet. Các ngôn ngữ dựa trên XML (Ví dụ: RDF, RSS, MathML, XHTML, SVG, GML và cXML) được định nghĩa theo cách thông thường, cho phép các chương trình sửa đổi và kiểm tra hợp lệ bằng các ngôn ngữ này mà không cần có hiểu biết trước về hình thức của chúng.

## Các thành phần

Gồm có 02 phần:

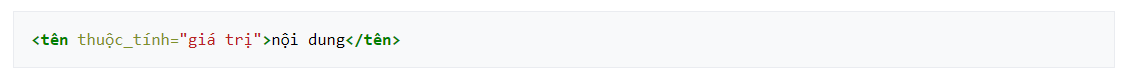
**Document Prolog:** lưu trữ metadata của XML gồm 2 phần: khai báo XML và khai báo kiểu dữ liệu trong XML.

* Phần khai báo XML (XML declararion) bao gồm các thông tin về version của XML, charset, encoding….
* Phần khai báo kiểu dữ liệu trong XML (DTD) dùng để khai báo cấu trúc của các thẻ dùng trong XML

**Root element:** hay còn gọi là Document Element: chứa tất cả các phần tử và nội dung của nó, một phần tử của XML phải có thẻ mở và thẻ đóng.

## Cú pháp

Cú pháp XML cơ bản cho một phần tử là:

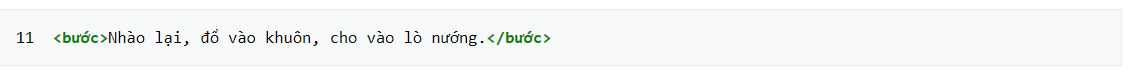




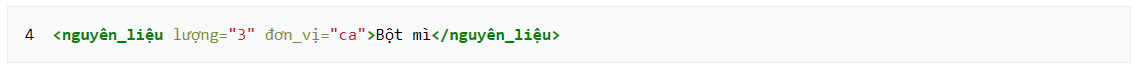
Dòng đầu tiên là **Khai báo XML** (*XML declaration*): đó là một dòng không bắt buộc, với nhiệm vụ thông báo phiên bản XML đang được sử dụng (thường là phiên bản 1.0), và còn có thể chứa thông tin về mã hóa ký tự và các phụ thuộc bên ngoài.

Phần còn lại của tài liệu này chứa các *phần tử* lồng nhau, một số phần tử trong đó có các *thuộc tính* và *nội dung*. Một **phần tử** thường bao gồm hai thẻ (*tag*), một *thẻ bắt đầu* và một *thẻ kết thúc*, có thể bao quanh văn bản và các phần tử khác. **Thẻ bắt đầu** bao gồm một cái tên đặt trong một cặp ngoặc nhọn, như "<bước>"; **thẻ kết thúc** bao gồm chính cái tên đó đặt trong một cặp ngoặc nhọn, với một dấu gạch chéo đứng trước, như "</bước>".

**Nội dung** của phần tử là tất cả những gì nằm giữa thẻ bắt đầu và thẻ kết thúc, bao gồm văn bản và các phần tử (con) khác. Dưới đây là một phần tử XML hoàn chỉnh, với thẻ bắt đầu, nội dung văn bản, và thẻ kết thúc:

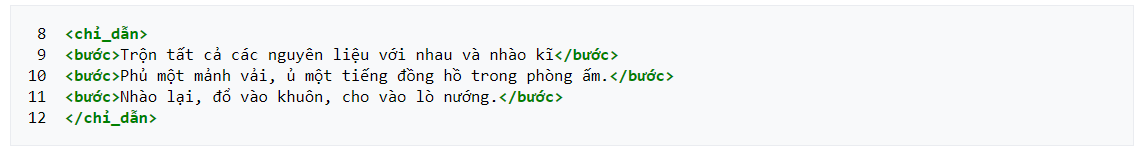


Bên cạnh nội dung, một phần tử có thể chứa các **thuộc tính** — các cặp tên - giá trị được đặt trong thẻ bắt đầu, ngay sau tên phần tử. Giá trị của thuộc tính phải được đặt trong cặp nháy đơn hoặc nháy kép, mỗi tên thuộc tính chỉ được xuất hiện một lần trong mỗi phần tử.

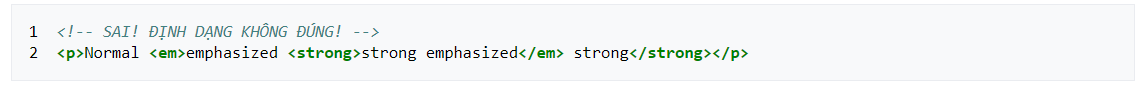


Trong ví dụ này, phần tử *nguyên\_liệu* có hai thuộc tính: *lượng* với giá trị "3", và *đơn vị* với giá trị "ca". Trong cả hai trường hợp, cũng như tên và nội dung của các phần tử, tại cấp độ đánh dấu, tên và giá trị của các thuộc tính cũng chỉ là dữ liệu text — các giá trị "3" và "ca" không phải một số lượng và một đơn vị đo lường mà chỉ là các chuỗi ký tự mà tác giả tài liệu có thể dùng để *biểu diễn* những thứ đó.

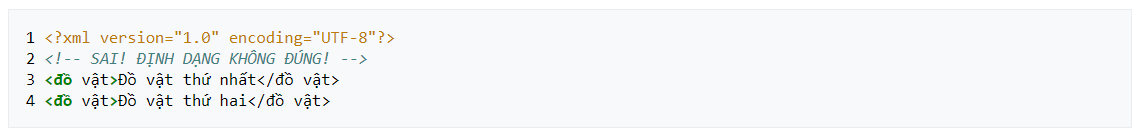
Ngoài văn bản, các phần tử còn có thể chứa các phần tử khác:



Trong đó, phần tử *chỉ\_dẫn* chứa ba phần tử *bước*. XML đòi hỏi rằng các phần tử phải được lồng nhau một cách đúng đắn — các phần tử không được có phần xen vào nhau. Ví dụ, đoạn dưới đây không phải XML định dạng đúng (*well-formed XML*) vì các phần từ *em* và *strong* xen vào nhau:



Mỗi tài liệu XML phải có đúng một **phần tử gốc** tại bậc trên cùng (còn gọi là *phần tử văn bản*), do đó đoạn sau cũng sẽ là một tài liệu XML định dạng sai:



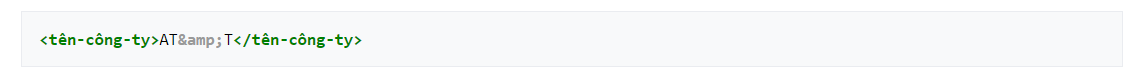
XML cung cấp cú pháp đặc biệt để biểu diễn một phần tử với nội dung rỗng. Thay vì viết một thẻ bắt đầu và một thẻ kết thúc ngay sau đó, tài liệu có thể chứa **thẻ phần tử rỗng** mà trong đó dấu gạch chéo đứng ngay sau tên phần tử. Hai ví dụ sau là tương đương về chức năng:



XML cung cấp cú pháp đặc biệt để biểu diễn một phần tử với nội dung rỗng. Thay vì viết một thẻ bắt đầu và một thẻ kết thúc ngay sau đó, tài liệu có thể chứa **thẻ phần tử rỗng** mà trong đó dấu gạch chéo đứng ngay sau tên phần tử. Hai ví dụ sau là tương đương về chức năng:

* &amp; (&)
* &lt; (<)
* &gt; (>)
* &apos; (')
* &quot; (")

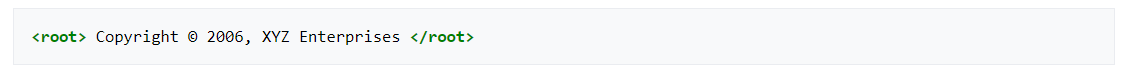
Dưới đây là một ví dụ sử dụng một thực thể XML khai báo trước để biểu diễn dấu & trong tên "AT&T":



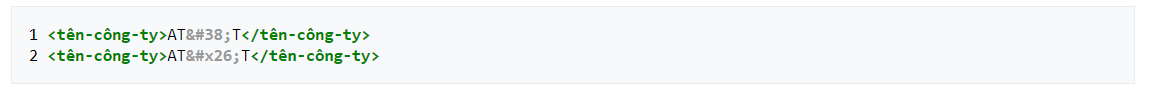
Nếu cần khai báo thêm các thực thể khác, việc đó được thực hiện tại DTD của tài liệu. Sau đây là một ví dụ cơ bản về khai báo thực thể tại một DTD nhỏ nội bộ. Các thực thể được khai báo có thể mô tả các ký tự đơn hay các đoạn văn bản, và có thể tham chiếu lẫn nhau.



Khi xem tại một trình duyệt thích hợp, tài liệu XML trên sẽ hiện ra như sau:



Các **tham chiếu ký tự số** trông giống như các thực thể. Nhưng thay cho một cái tên, chúng gồm một ký tự "#" và theo sau là một con số. Con số (theo [hệ thập phân](https://vi.wikipedia.org/wiki/H%E1%BB%87_th%E1%BA%ADp_ph%C3%A2n) hoặc hệ cơ số 16 với tiền tố "x") đại diện cho một mã hiệu Unicode (*Unicode code point*), và thường được dùng để đại diện cho các ký tự không dễ gõ trên máy tính, chẳng hạn một chữ cái Ả-rập trong một tài liệu được soạn trên một máy tính châu Âu. Dấu & trong ví dụ "AT&T" có thể được biểu diễn như sau (số 38 thập phân và 26 trong hệ cơ số 16 đều đại diện cho giá trị Unicode của dấu &):



## Ứng dụng XML

### Ứng dụng XML trong thương mại điện tử

*Trình bày tài liệu*

* Khi các công ty thực hiện các giao dịch, sẽ có nhiều tài liệu được trao đổi, đó là văn bản, hóa đơn, hợp đồng, các giấy tờ pháp lý ,..
* Cần đồng bộ các tài liệu này, đó là một lĩnh vực then chốt đối với thương mại điện tử và XML cung cấp giải pháp này.

*Quản lý dữ liệu*

* Một khối lượng lớn dữ liệu trong cơ sở dữ liệu hay nơi nào đó có thể được sử dụng cho nhiều giao dịch khác nhau.
* Nhiều tổ chức khác nhau sử dụng tất cả các loại dữ liệu để kiểm soát giao dịch thương mại điện tử, XML cung cấp một khả năng trình bày dữ liệu mà các tổ chức khác nhau có thể hiểu được.
* Một lĩnh vực khác của quản lý dữ liệu là khai phá dữ liệu.
* Thông tin trong các tài liệu XML phải được khai phá để có thể hiểu dữ liệu trong nhiều cơ sở dữ liệu khác nhau.
* Khía cạnh khác là khai phá các mẫu thông dụng cũng tập hợp trong các tài liệu XML cho các giao dịch thông minh.

*Tương tác*

* Các tổ chức khác nhau có thể cộng tác làm việc với nhau. Ở đây được gọi là cộng tác thương mại.

*Ví dụ, tổ chức A đặt một đơn đặt hàng với tổ chức B, tổ chức B lại đặt mua từ tổ chức C và D. Điều này có nghĩa là A, B, C và D cộng tác với nhau*

* XML là yếu tố cần thiết cho sự giao tiếp giữa hai tổ chức khác nhau.

Nhiều tài liệu, hợp đồng, và các hợp đồng phụ sẽ được trao đổi giữa hai tổ chức khác nhau. Trừ khi chúng được chỉ định trong một vài ngôn ngữ thông dụng nào đó, nếu không nó sẽ rất khó hiểu

*Một số kết quả liên quan*

Đã có nhiều sự nỗ lực nghiên cứu để phát triển các chuẩn XML cho thương mại điện tử. Ba trong số nhiều kết quả đạt được đó là:

- ebXML;

- RosettaNet;

- Commerce One.Net.

### Ứng dụng XML trong quản lý dữ liệu

*XML và xử lý truy vấn*

W3C đã yêu cầu đề xuất một ngôn ngữ vấn tin cho tài liệu XML và đã có nhiều kết quả đạt được:

* XMLQL (XML Query Language) là một ngôn ngữ vấn tin cho XML.
* XMLQL có nhiều cấu trúc có thể truy vấn các tài liệu dựa trên những DTD của chúng
* Tiện ích Oracle XMLSQL phát triển nhiều kỹ thuật cho phép biến đổi cấu trúc XML sang cấu trúc SQL và ngược lại
* IBM phát triển DB2 CXML mở rộng để lưu trữ và thao tác các tài liệu XML.
* Microsoft mở rộng SQL92 bằng cách phát triển OPENXML,…

Ngôn ngữ truy vấn và tối ưu hóa truy vấn là hai vấn đề lớn của các cơ sở dữ liệu XML.

*Dữ liệu phân tán*

* Các tài liệu XML cũng có thể là phân tán.
* Tiếp cận dữ liệu phân tán hoàn toàn tương tự với cơ sở dữ liệu quan hệ phân tán.
* Cần nghiên cứu nhiều trước khi quyết định cách tiếp cận nào phù hợp với tài liệu XML phân tán.

*Di chuyển và tương tác*

* XML rất thuận lợi trong việc di chuyển cơ sở dữ liệu; tuy nhiên không phải là lời giải đáp cho mọi vấn đề.

*Ví dụ, XML vẫn còn thiếu chức năng trình bày ngữ nghĩa.*

* Không thể giải quyết các xung đột ngữ nghĩa giữa các cơ sở dữ liệu khác nhau. Hy vọng rằng, RDF và ngữ nghĩa web phát triển sẽ hỗ trợ vấn đề này.

*Kho dữ liệu và khai phá dữ liệu*

Với kho dữ liệu có hai phương diện liên quan đến XML:

* Các tài liệu XML, cơ sở dữ liệu bán cấu trúc, cơ sở dữ liệu quan hệ và nhiều nguồn dữ liệu khác được tích hợp vào trong một kho dữ liệu Cho đến nay nhiều công việc trên CSDL quan hệ được tích hợp vào trong một kho dữ liệu dựa trên mô hình quan hệ. Khi CSDL là các tài liệu XML cũng như CSDL bán cấu trúc, làm thế nào để tích hợp chúng vào trong một kho dữ liệu. Mô hình dữ liệu nào là thích hợp cho một kho dữ liệu?
* Trình bày kho dữ liệu như một tập hợp các tài liệu XML, cần ánh xạ giữa các nguồn dữ liệu cơ sở. Ví dụ, ánh xạ các mô hình quan hệ và mô hình đối tượng đến các mô hình dữ liệu XML. Cần phát triển các kỹ thuật để truy cập, truy vấn và tạo chỉ mục trong kho dữ liệu.

Với khai phá tài liệu XML, có 2 phương diện cần quan tâm:

* Khai phá các tài liệu để trích rút thông tin hữu ích như mẫu và phương hướng. Ví dụ, các tài liệu XML có thể được khai phá cho hệ thống thương mại thông minh
* Khai phá các tài liệu cho tuỳ chọn quan hệ quản lý; cũng có thể khai phá các mẫu thông dụng để báo cho người dùng biết về các liên kết trong một tài liệu XML.

*Công nghệ đối tượng*

* Gồm quản lý cơ sở dữ liệu đối tượng, quản lý đối tượng phân tán, ngôn ngữ lập trình đối tượng, thành phần và khung đối tượng, phân tích và thiết kế hướng đối tượng. Các cấu trúc trong mô hình đối tượng bao gồm lớp (class), thuộc tính (attribute) và các thể hiện (instance) của đối tượng
* Coi DTD trong XML như các cấu trúc tài liệu tương tự như các lớp trong khi các tài liệu là các thể hiện; tuy nhiên, XML không phải là loại định kiểu mạnh. XML chỉ là tượng trưng và không có một mô hình thực thi. Vì vậy, nhiều phương thức không áp dụng được trong trường hợp này.
* Một phần của UML có thể được dùng cho mô hình dữ liệu XML, mặc dù nhiều phương diện UML trình bày trạng thái động và một số không áp dụng được.
* Các hệ thống quản lý đối tượng phân tán như OMG (Object Management Group), ORB (Object Request Broker) làm việc theo nhóm nghiên cứu để làm thế nào XML có thể sử dụng trong sự kết hợp với IDL (Interface Description Language).
* Các lược đồ của cơ sở dữ liệu thừa kế được định nghĩa trong XML, nhưng gọi các phương thức được tạo trong IDL.

### Ứng dụng XML trong quản lý thông tin

*Đa phương tiện và XML*

*Quản lý tri thức và XML*

*Hỗ trợ quyết định và XML*

* XML, RDF và ngữ nghĩa web sẽ trở thành những công nghệ then chốt cho hệ thống hỗ trợ quyết định.
* Nhiều kết quả của các tổ chức như ebXML có thể áp dụng được cho hỗ trợ quyết định.

*Tính toán trong môi trường vô tuyến*

* Các tập đoàn liên quan trong lĩnh vực tính toán vô tuyến đã phát triển ngôn ngữ đánh dấu vô tuyến WML (Wireless Markup Language).

XML và WAP là hai công nghệ cơ bản cho ngôn ngữ WML

# CÔNG NGHỆ

## Web Service

### Giới thiệu

Dịch vụ web là tập hợp các giao thức và tiêu chuẩn mở được sử dụng để trao đổi dữ liệu giữa các ứng dụng hoặc giữa các hệ thống. Các ứng dụng phần mềm được viết bằng các ngôn ngữ lập trình khác nhau và chạy trên các nền tảng khác nhau, có thể sử dụng các dịch vụ web để trao đổi dữ liệu qua mạng máy tính (như Internet) theo cách tương tự như liên lạc giữa các quá trình trên một máy tính. Khả năng tương tác này (ví dụ: giữa các ứng dụng Java và Python hoặc Windows và linux) là do việc sử dụng các tiêu chuẩn mở.

Tóm lại, một dịch vụ web hoàn chỉnh:

* Có sẵn trên mạng Internet hoặc mạng riêng (mạng nội bộ).
* Sử dụng hệ thống nhắn tin XML tiêu chuẩn hóa.
* Không được gắn với bất kỳ hệ điều hành hoặc ngôn ngữ lập trình nào.
* Tự mô tả thông qua một ngữ pháp XML phổ biến.
* Có thể khám phá thông qua cơ chế tìm kiếm đơn giản.

### Các thành phần của web service

Nền tảng web service cơ bản là XML HTTP. Tất cả các web service chuẩn đều hoạt động bằng các thành phần sau:

* SOAP (Giao thức truy cập đối tượng đơn giản): SOAP là một giao thức dựa trên XML đơn giản cho phép các ứng dụng trao đổi thông tin qua HTTP.
* UDDI (Mô tả, Khám phá và Tích hợp Toàn cầu): UDDI là một tiêu chuẩn dựa trên XML để mô tả, xuất bản và tìm kiếm các dịch vụ web.
* WSDL (Ngôn ngữ mô tả web service): WSDL là một ngôn ngữ dựa trên XML để mô tả các dịch vụ web và cách truy cập chúng.
* WSDL mô tả một dịch vụ web, cùng với định dạng thông báo và các chi tiết giao thức cho dịch vụ web.

### Cách thức hoạt động của Web Service

Một dịch vụ web cho phép giao tiếp giữa các ứng dụng khác nhau bằng cách sử dụng các tiêu chuẩn mở như HTML, XML, WSDL và SOAP.

Một dịch vụ web có sự giúp đỡ của:

* XML để gắn thẻ dữ liệu.
* SOAP để chuyển một tin nhắn.
* WSDL để mô tả tính khả dụng của dịch vụ.

Bạn có thể xây dựng một dịch vụ web dựa trên Java trên Solaris có thể truy cập từ chương trình Visual Basic của bạn chạy trên Windows.

Bạn cũng có thể sử dụng C# để xây dựng các dịch vụ web mới trên Windows có thể được gọi từ ứng dụng web của bạn dựa trên các trang Java Server (JSP) và chạy trên Linux.

Ví dụ:

Hãy xem xét một hệ thống xử lý đơn đặt hàng và quản lý tài khoản đơn giản. Nhân viên kế toán sử dụng ứng dụng khách được xây dựng với Visual Basic hoặc JSP để tạo tài khoản mới và nhập các đơn đặt hàng của khách hàng mới.

Logic xử lý cho hệ thống này được viết bằng Java và nằm trên máy Solaris, cũng tương tác với cơ sở dữ liệu để lưu trữ thông tin.

Các bước để thực hiện thao tác này như sau:

* Chương trình khách hàng kết hợp thông tin đăng ký tài khoản vào một thông điệp SOAP.
* Thông báo SOAP này được gửi đến dịch vụ web làm phần thân của yêu cầu HTTP POST.
* Dịch vụ web giải nén yêu cầu SOAP và chuyển đổi nó thành lệnh mà ứng dụng có thể hiểu được.
* Ứng dụng xử lý thông tin theo yêu cầu và phản hồi với số tài khoản duy nhất mới cho khách hàng đó.
* Tiếp theo, dịch vụ web gói trả lời vào một thông báo SOAP khác, nó gửi trở lại chương trình máy khách để đáp ứng yêu cầu HTTP của nó.
* Chương trình khách hàng mở gói thông điệp SOAP để có được kết quả của quá trình đăng ký tài khoản.

### Lợi ích của Web Service

* Hiển thị chức năng có trên mạng: Một dịch vụ web là một đơn vị mã được quản lý có thể được gọi từ xa bằng cách sử dụng HTTP, có nghĩa là, nó có thể được kích hoạt bằng cách sử dụng các yêu cầu HTTP. Các dịch vụ web cho phép bạn hiển thị chức năng của mã hiện có qua mạng. Khi nó được hiển thị trên mạng, ứng dụng khác có thể sử dụng chức năng của chương trình.
* Khả năng cộng tác: Các dịch vụ web cho phép các ứng dụng khác nhau nói chuyện với nhau và chia sẻ dữ liệu và dịch vụ với nhau. Các ứng dụng khác cũng có thể sử dụng các dịch vụ web. Ví dụ, một ứng dụng VB hoặc .NET có thể nói chuyện với các dịch vụ web Java và ngược lại. Các dịch vụ Web được sử dụng để làm nền tảng ứng dụng và công nghệ độc lập.
* Giao thức chuẩn hóa: Các dịch vụ Web sử dụng giao thức được chuẩn hóa cho giao tiếp. Tất cả bốn lớp (các dịch vụ Transport, XML Messaging, Service Description và Service Discovery) đều sử dụng các giao thức được xác định rõ trong stack giao thức dịch vụ web. Tiêu chuẩn này của protocol stack mang lại cho doanh nghiệp nhiều lợi thế, có thể kể đến như một loạt các lựa chọn, giảm chi phí do cạnh tranh, và tăng chất lượng.
* Giao tiếp chi phí thấp: Các dịch vụ web sử dụng SOAP qua giao thức HTTP, vì vậy bạn có thể sử dụng mạng internet hiện có của mình để triển khai các dịch vụ web. Giải pháp này ít tốn kém hơn nhiều so với các giải pháp độc quyền như EDI/B2B. Bên cạnh SOAP trên HTTP, các dịch vụ web cũng có thể được triển khai trên các cơ chế truyền tải đáng tin cậy khác như FTP.

## Microsoft Azure

### Giới thiệu về Microsoft Azure

#### Giới thiệu

Microsoft Azure (trước đây là Windows Azure / ˈæʒər /) là dịch vụ điện toán đám mây do Microsoft tạo ra để xây dựng, thử nghiệm, triển khai và quản lý các ứng dụng và dịch vụ thông qua mạng lưới trung tâm dữ liệu do Microsoft quản lý. Nó cung cấp phần mềm dưới dạng dịch vụ (SaaS), nền tảng là dịch vụ (PaaS) và cơ sở hạ tầng dưới dạng dịch vụ (IaaS) và hỗ trợ nhiều ngôn ngữ lập trình, công cụ và frameworks khác nhau, bao gồm cả phần mềm và hệ thống của bên thứ ba cụ thể của Microsoft.

Azure được công bố vào tháng 10 năm 2008, bắt đầu với tên mã là "Project Red Dog" và được phát hành vào ngày 1 tháng 2 năm 2010, dưới tên "Windows Azure" trước khi được đổi tên thành "Microsoft Azure" vào ngày 25 tháng 3 năm 2014.

#### Các dịch vụ

* Tính toán:
  + Máy ảo, cơ sở hạ tầng như một dịch vụ (IaaS) cho phép người dùng khởi chạy các máy ảo Microsoft Windows và Linux đa năng, cũng như hình ảnh máy được cấu hình sẵn cho các gói phần mềm phổ biến.
  + Các dịch vụ ứng dụng, nền tảng như một môi trường dịch vụ (PaaS) cho phép các nhà phát triển dễ dàng xuất bản và quản lý trang web.
  + Trang web, lưu trữ mật độ cao [không phải trình tự] của trang web cho phép nhà phát triển xây dựng trang web bằng ASP.NET, PHP, Node.js hoặc Python và có thể được triển khai bằng FTP, Git, Mercurial, Team Foundation Server hoặc tải lên thông qua cổng thông tin người dùng. Tính năng này đã được công bố dưới dạng xem trước vào tháng 6 năm 2012 tại sự kiện Gặp gỡ Microsoft Azure. Khách hàng có thể tạo các trang web bằng PHP, ASP.NET, Node.js hoặc Python hoặc chọn từ một số ứng dụng nguồn mở từ một thư viện để triển khai. Điều này bao gồm một khía cạnh của nền tảng dưới dạng dịch vụ (PaaS) cho Nền tảng Microsoft Azure. Nó đã được đổi tên thành Ứng dụng web vào tháng 4 năm 2015.
  + WebJobs, các ứng dụng có thể được triển khai vào môi trường Dịch vụ Ứng dụng để triển khai xử lý nền có thể được gọi theo lịch, theo yêu cầu hoặc chạy liên tục. Các dịch vụ Blob, Table và Queue có thể được sử dụng để liên lạc giữa WebApps và WebJobs và để cung cấp trạng thái.
* Dịch vụ di động
  + Mobile Engagement thu thập các phân tích thời gian thực làm nổi bật hành vi của người dùng. Nó cũng cung cấp thông báo đẩy cho các thiết bị di động.
  + HockeyApp có thể được sử dụng để phát triển, phân phối và thử nghiệm ứng dụng di động beta.
* Dịch vụ lưu trữ
  + Storage Services cung cấp API REST và SDK để lưu trữ và truy cập dữ liệu trên đám mây.
  + Table Service cho phép các chương trình lưu trữ văn bản có cấu trúc trong các bộ sưu tập các thực thể được phân vùng được truy cập bởi khóa phân vùng và khóa chính. Đó là một cơ sở dữ liệu phi quan hệ NoQuery.
  + Blob Service cho phép các chương trình lưu trữ dữ liệu văn bản và dữ liệu nhị phân không có cấu trúc dưới dạng các đốm màu có thể được truy cập bằng đường dẫn HTTP (S). Dịch vụ Blob cũng cung cấp các cơ chế bảo mật để kiểm soát truy cập dữ liệu.
  + Queue Service cho phép các chương trình giao tiếp không đồng bộ bằng tin nhắn bằng cách sử dụng hàng đợi.
  + File Service cho phép lưu trữ và truy cập dữ liệu trên đám mây bằng API REST hoặc giao thức SMB.
* Quản lý dữ liệu
  + Azure Search cung cấp tìm kiếm văn bản và một tập hợp con các bộ lọc có cấu trúc của OData bằng API REST hoặc SDK.
  + Cosmos DB là một dịch vụ cơ sở dữ liệu NoQuery, thực hiện một tập hợp con của câu lệnh SQL SELECT trên các tài liệu JSON.
  + Redis Cache là một triển khai được quản lý của Redis.
  + StorSimple quản lý các tác vụ lưu trữ giữa các thiết bị tại chỗ và lưu trữ đám mây.
  + Cơ sở dữ liệu SQL, trước đây gọi là Cơ sở dữ liệu SQL Azure, hoạt động để tạo, mở rộng và mở rộng các ứng dụng vào đám mây bằng công nghệ Microsoft SQL Server. Nó cũng tích hợp với Active Directory và Microsoft System Center và Hadoop.
  + Kho dữ liệu SQL là một dịch vụ lưu trữ dữ liệu được thiết kế để xử lý các truy vấn chuyên sâu về tính toán và dữ liệu trên các bộ dữ liệu vượt quá 1TB.
  + Azure Data Factory, là một dịch vụ tích hợp dữ liệu cho phép tạo ra các luồng công việc dựa trên dữ liệu trong đám mây để điều phối và tự động hóa chuyển động dữ liệu và chuyển đổi dữ liệu.
  + Azure Data Lake là dịch vụ lưu trữ và phân tích dữ liệu có thể mở rộng cho khối lượng công việc phân tích dữ liệu lớn, yêu cầu các nhà phát triển chạy các truy vấn song song liên tục.
  + Azure HDInsight là một dịch vụ liên quan đến dữ liệu lớn, triển khai Hortonworks Hadoop trên Microsoft Azure và hỗ trợ tạo các cụm Hadoop bằng Linux với Ubuntu.
  + Azure Stream Analytics là một công cụ xử lý sự kiện không có khả năng mở rộng cho phép người dùng phát triển và chạy các phân tích thời gian thực trên nhiều luồng dữ liệu từ các nguồn như thiết bị, cảm biến, trang web, phương tiện truyền thông xã hội và các ứng dụng khác.
* Học máy (Machine learning)
  + Dịch vụ Microsoft Azure Machine Learning (Azure ML) là một phần của Cortana Intelligence Suite cho phép phân tích dự đoán và tương tác với dữ liệu bằng ngôn ngữ tự nhiên và lời nói thông qua Cortana.
  + Dịch vụ nhận thức (trước đây là Project Oxford) là một bộ API, SDK và dịch vụ có sẵn cho các nhà phát triển để làm cho ứng dụng của họ thông minh hơn, hấp dẫn và có thể khám phá hơn.

#### Thiết kế

Microsoft Azure sử dụng một hệ điều hành chuyên dụng, được gọi là Microsoft Azure, để chạy "lớp cấu trúc" của nó: một cụm được lưu trữ tại các trung tâm dữ liệu của Microsoft quản lý tài nguyên tính toán và lưu trữ của máy tính và cung cấp tài nguyên (hoặc một tập hợp con của chúng) cho các ứng dụng đang chạy trên Microsoft Azure. Microsoft Azure đã được mô tả là "lớp đám mây" trên một số hệ thống Windows Server sử dụng Windows Server 2008 và phiên bản tùy chỉnh của Hyper-V, được gọi là Microsoft Azure Hypervisor để cung cấp dịch vụ ảo hóa.

Tỷ lệ và độ tin cậy được điều khiển bởi Bộ điều khiển vải Microsoft Azure, đảm bảo các dịch vụ và môi trường không bị lỗi nếu một hoặc nhiều máy chủ bị lỗi trong trung tâm dữ liệu của Microsoft và cũng cung cấp việc quản lý ứng dụng Web của người dùng như cấp phát bộ nhớ và cân bằng tải.

Azure cung cấp API được xây dựng trên REST, HTTP và XML cho phép nhà phát triển tương tác với các dịch vụ do Microsoft Azure cung cấp. Microsoft cũng cung cấp một thư viện lớp được quản lý phía máy khách, đóng gói các chức năng tương tác với các dịch vụ. Nó cũng tích hợp với Microsoft Visual Studio, Git và Eclipse.

Ngoài việc tương tác với các dịch vụ qua API, người dùng có thể quản lý các dịch vụ Azure bằng Cổng thông tin Azure dựa trên Web, đạt đến Tính khả dụng chung vào tháng 12 năm 2015. Cổng thông tin cho phép người dùng duyệt tài nguyên hoạt động, sửa đổi cài đặt, khởi chạy tài nguyên mới và xem giám sát cơ bản dữ liệu từ các máy và dịch vụ ảo đang hoạt động. Dịch vụ quản lý Azure tiên tiến hơn có sẵn.

#### Những mô hình triển khai

Microsoft Azure cung cấp hai mô hình triển khai cho tài nguyên đám mây: mô hình triển khai "cổ điển" và Trình quản lý tài nguyên Azure. Trong mô hình cổ điển, mỗi tài nguyên Azure (máy ảo, cơ sở dữ liệu SQL, v.v.) được quản lý riêng lẻ. Trình quản lý tài nguyên Azure, được giới thiệu vào năm 2014, cho phép người dùng tạo các nhóm dịch vụ liên quan để các tài nguyên được kết hợp chặt chẽ có thể được triển khai, quản lý và giám sát cùng nhau.

### Azure Web App Service

#### Giới thiệu

Azure App Service là dịch vụ lưu trữ các ứng dụng web, API REST và kết thúc di động. Bạn có thể phát triển bằng ngôn ngữ yêu thích của mình, có thể là .NET, .NET Core, Java, Ruby, Node.js, PHP hoặc Python. Các ứng dụng chạy và mở rộng quy mô dễ dàng trên các môi trường dựa trên Windows.

Được cung cấp bởi Azure App Service, Web service là một nền tảng được quản lý hoàn toàn cho phép bạn xây dựng, triển khai và mở rộng các ứng dụng web cấp doanh nghiệp trong vài giây.

#### Tính năng

Những tính năng chính:

* .NET, Java, PHP, Node.js, Python
* Tích hợp tự động và cân bằng tải
* Tính sẵn sàng cao với tính năng tự động vá
* Triển khai liên tục với Git, TFS, GitHub
* Các vị trí triển khai, thử nghiệm trong sản xuất, WebJobs và tích hợp liên tục
* Thư viện ứng dụng web: WordPress, Umbraco, Joomla, Drupal

Chúng không có gì khác hơn các dịch vụ Azure Websites trước đây, đó là lý do tại sao Azure Websites được triển khai cho Azure hiện được gọi là "web apps" và có thể chạy các ứng dụng này như trước đây. Nó cũng có thể lưu trữ WebJob như trước đây, với tần suất sao lưu tự động phụ thuộc vào tầng được khách hàng chọn.

### Azure SQL Database

Cơ sở dữ liệu Microsoft Azure SQL (trước đây là SQL Azure, SQL Server Data Services, SQL Services và Windows Azure SQL Database) là cơ sở dữ liệu đám mây được quản lý (SaaS) được cung cấp như một phần của Microsoft Azure.

Cơ sở dữ liệu đám mây là cơ sở dữ liệu chạy trên nền tảng điện toán đám mây và quyền truy cập vào cơ sở dữ liệu được cung cấp dưới dạng dịch vụ. Các dịch vụ cơ sở dữ liệu được quản lý đảm nhận khả năng mở rộng, sao lưu và tính sẵn sàng cao của cơ sở dữ liệu. Cơ sở dữ liệu Azure SQL là một dịch vụ cơ sở dữ liệu được quản lý khác với AWS RDS là dịch vụ chứa.

Cơ sở dữ liệu Microsoft Azure SQL bao gồm trí thông minh tích hợp tìm hiểu các mẫu ứng dụng và điều chỉnh để tối đa hóa hiệu suất, độ tin cậy và bảo vệ dữ liệu.

Nó ban đầu được công bố vào năm 2009 và phát hành vào năm 2010

* Các khả năng chính bao gồm:
  + Học hỏi liên tục các mẫu ứng dụng độc đáo của bạn, điều chỉnh hiệu suất thích ứng và tự động cải thiện độ tin cậy và bảo vệ dữ liệu
  + Thu nhỏ khi cần, hầu như không có thời gian chết ứng dụng
  + Quản lý và giám sát các ứng dụng đa năng với các lợi ích riêng biệt của một khách hàng trên mỗi cơ sở dữ liệu
  + Tận dụng các công cụ nguồn mở như cheetah, sql-cli, VS Code và các công cụ Microsoft như Visual Studio và SQL Server Management Studio, Azure Management Portal, PowerShell và REST API
  + Bảo vệ dữ liệu bằng mã hóa, xác thực, hạn chế quyền truy cập của người dùng vào tập hợp con dữ liệu phù hợp, theo dõi và kiểm tra liên tục để giúp phát hiện các mối đe dọa tiềm ẩn và cung cấp bản ghi các sự kiện quan trọng trong trường hợp vi phạm

## DOM

### Tổng quan về DOM

* DOM là một tiêu chuẩn được định nghĩa bởi tổ chức W3C, cũng giống như XML
* DOM không được thiết kế một cách cụ thể cho công nghệ java (không giống với SAX)
* DOM dùng cho mọi nền tảng(cross-platform) và mọi ngôn ngữ (crosslanguage)
  + Uses OMG's IDL to define interfaces
  + IDL to language binding

DOM là một tiêu chuẩn do một tổ chức W3C định nghĩa và mang tính quốc tế tất cả các ngôn ngữ tôn trọng các định nghĩa này và xây dựng các API dựa theo các tiêu chuẩn đó.Thực tế thì có rất nhiều ngôn ngữ như C++,C#,...đều hỗ trợ mô hình DOM .Do đó DOM không phải thiết kế dành riêng cho Java ,tuy nhiên nếu nói về các API cho phép phân tích các tài liệu XML thì chúng ta còn có mô hình SAX(Simple API for XML) một mô hình phân tích XML đơn giản hơn DOM rất nhiều ban đầu thiết kế cho Java nhưng hiện nay các ngôn ngữ khác cũng đã bắt đầu hỗ trợ mô hình này.

### Các đặc điểm và tính năng của mô hình DOM

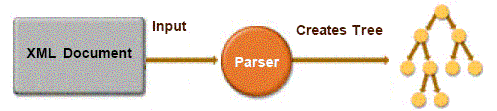
#### Tính năng của mô hình DOM

* Truy cập tài liệu XML như là một cấu trúc cây
* Được bao gồm hầu hết các nút element và các nút text
* Có thể "rà xoát" (Traversing) cây từ sau ra trước
* Đòi hỏi bộ nhớ lớn hơn
  + Khá nặng nề trong việc tải và lưu trữ
* Sử dụng nó khi rà xoát và hiệu chỉnh cây

Với một tài liệu XML mô hình DOM sẽ duyệt và chuyển nó thành một mô hình cây của các Object. Ví dụ một element tương ứng với một Object element, một thuộc tính tương ứng với một Object thuộc tính các Object này được tạo ra trong bộ nhớ (memory) và có cấu trúc cây. Chính vì vậy mà có thể rà xoát tài liệu XML này bằng cách rà xoát trên các Object trong bộ nhớ. Và có thể thêm bớt các thông tin, object để tạo ra một tài liệu XML mới hoặc một sản phẩm mới.

#### Hành động của DOM

Hình minh họa dưới cho thấy đầu vào là một tài liệu XML được bộ phân tích bởi mô hình DOM và một cây sẽ cây được tạo ra trong bộ nhớ mang thông tin của tài liệu đó. Việc phân tích tài liệu XML bây giờ đưa về phân tích, sử lý các nút của cây.



Hình 2‑1: Hành động của DOM

### Nút cây DOM và các kiểu nút (Node)

* Tài liệu XML được hình dung như là một cây
* Một cây được làm từ các nút cây(nodes)
* Có 12 kiểu nút cây khác nhau
* Các nút cây có thể chứa các nút cây khác (Phụ thuộc vào kiểu nút cây là gì)
  + Nút cha gồm có các nút con

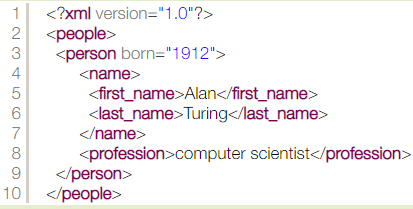
Có 12 kiểu Node

1. Document (Mô tả một nút lớn nhất đó là toàn bộ tài liệu XML)
2. DocumentFragment (Một đoạn tài liệu XML)
3. Element
4. Attr (Nút thuộc tính)
5. Text (Nút chứa text)
6. Comment (Ghi chú trong tài liệu XML)
7. ProcessingInstruction (Tương ứng với chỉ lệnh trong XML)
8. DocumentType (Định nghĩa XML)
9. Entity (Tương ứng với thực thể trong XML)
10. EntityReference (Tương ứng với các thực thể tham chiếu trong XML)
11. CDATASection (Tương ứng với các phân đoạn trong XML)
12. Notation (Tương ứng với các chú thích NOTATION trong XML)

#### Phân cấp cây DOM

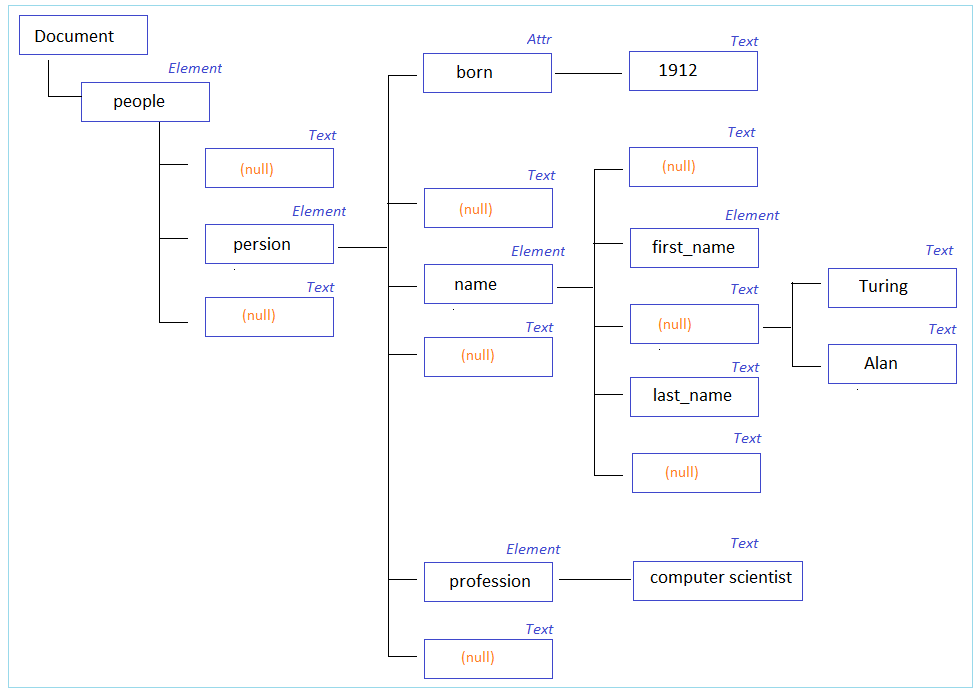
* Một nút Document bao gồm
  + Một nút Element (Đây chính là Element gốc của tài liệu)
  + Một hoặc nhiều các nút chỉ lệnh (ProcessingInstruction)
  + Một hoặc nhiều nút Comment
* Một nút Element có thể chứa
  + Các nút Element khác
  + Một hoặc nhiều nút Text
  + Một hoặc nhiều nút thuộc tính (Attr)
  + Một nút thuộc tính (Attr) chứa
* Một nút Text

Ví dụ:



Hình 2‑2: Ví dụ cây phân cấp DOM

Mô hình cây DOM:



### Mô hình DOM trong Java

Thực tế là W3C đã đưa ra danh sách 12 interface mô tả các nút của cây DOM. Mỗi interface có ra danh sách các hàm, các thuộc tính để can thiệp, sử dụng.

(Khi bạn làm việc với DOM trên một ngôn ngữ nào đó khác Java thì cũng phải làm việc với các interface có các thuộc tính, method tương tự đó chính là cái hay của DOM - một mô hình có tính thống nhất hành động giữa các ngôn ngữ).

Các interface này có một phân cấp rõ ràng.

Ví dụ:

* interface org.w3c.dom.Element mô tả một nút Element
* interface org.w3c.dom.Attr mô tả một nút thuộc tính

#### Phân cấp các interface

* Node
* Document
* DocumentFragment
* DocumentType
* ProcessingInstruction
* CharacterData
  + Comment
  + Text
    - CDATASection
* Element
* Attr
* EntityReference
* Entity
* Notation

#### Các interface khác cho DOM

* NodeList
* NamedNodeMap
* DOMImplementation

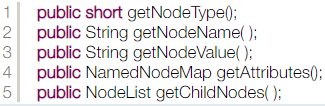
Ba interface quan trọng nằm trong org.w3c.dom:

* NodeList là một interface mô tả một tập hợp một số nút con của một nút nào đó. Method getChildNodes() của interface Node sẽ trả về một đối tượng NodeList chính là danh sách các nút con của nút gọi method này.
* Với một nút Element gọi method này sẽ trả về NodeList - một tập hợp trùu tượng chứa các nút con của Element trên. Trong NodeList này lại không có các thuộc tính của Element. Nút Element gọi method getAttributes() sẽ trả về đối tượng NamedNodeMap một danh sách trùu tượng chứa các thuộc tính của Element.
* NamedNodeMap còn có chứa các Entity mà ta sẽ thấy khi gọi method getEntities() của đối tượng.

#### Node interface

* Là kiểu dữ liệu chính trong DOM
* Mô tả một nút trong cây DOM
* Tất cả các nút đều là kiểu interface Node
  + Từ việc tất cả các kiểu đều là kiểu interface Node, tất cả các nút có thể xử lý một cách tương tự nhau.

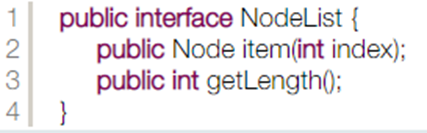
Một số method của Node trong Java:



Hình 2‑3: Một số phương thức của Node trong Java

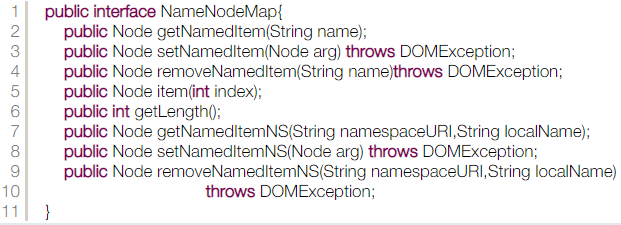
#### NodeList, NamedNodeMap interface

* NodeList Interface



Hình 2‑4: Ví dụ NodeList Interface

* NamedNodeMap Interface
  + Mô tả tập hợp các nút mỗi nút có thể phân biệt bởi tên
  + Có thể gọi bằng :
    - getAttributes() với Element
    - getEntities(), getNotations() với DocumentType
* NamedNodeMap Interface



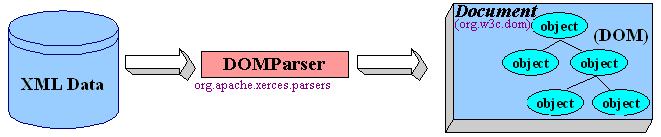
Hình 2‑5: Ví dụ NamedNodeMap Interface

### Lập trình với DOM

#### Các bước xử lý DOM

* Tạo ra một đối tượng bộ phân tích (parser object)
* Đặt các tính năng(Features) và đọc các thuộc tính (Properties)
* Phân tích tài liệu XML và nhận lấy đối tượng Document
* Thức hiện các toán tử
  + Rà soát DOM (Traversing)
  + Vận dụng DOM(Manipulating)
  + Tạo một DOM mới
  + Viết ra đầu ra các DOM
* Có một thực tế là trong Java có nhiều API để sử lý tài liệu XML theo mô hình DOM.
* Có thể sử dụng javax.xml.parsers, đã tích hợp sẵn vào Jdk1.5 trở lên.
* Có thể sử dụng org.apache.xerces.parsers của Apache.
* Tạo một đối tượng bộ phận không phải là một tiêu chuẩn vì:
  + DOM không định rõ làm thế nào để tạo một đối tượng bộ phân tích.
  + Có nhiều API khác nhau ngay trong nội bộ Java, để sử lý DOM, phải tuân thủ theo tiêu chuẩn DOM, implements các interface trong org.w3c.xml.
  + Các dòng code có thể khó khăn trong việc chuyển mang (Portability)
  + Nên sử dụng JAXP (javax.xml.parsers) làm API phân tích XML theo mô hình DOM vì thư viện tích hợp sẵn trong Jdk.

Dùng org.apache.xerces.parsers của Apache là thư viện phân tích DOM:



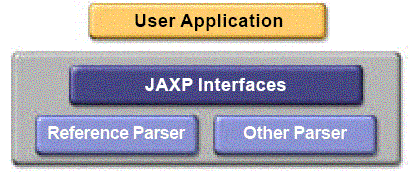
Hình 2‑6: Thư viện phân tích DOM org.apache.xerces.parsers

#### Các toán tử thao tác trên DOM

* Rà xoát DOM (Traversing DOM)
* Vận dụng DOM(Manipulating DOM)
  + Nối thêm các nút (Appending nodes)
  + Xóa bỏ các nút (Removing nodes)
  + Hiệu chỉnh các nút (Modifying nodes)
* Tạo ra một DOM mới (Generating a new DOM)
* Serializing DOMb
* Q: subclassing DOM

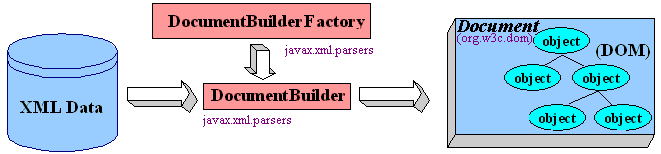
### 2.6. JAXP (Java API for XML Parsing )

JAXP một API cho phân tích XML nó bao gồm các API cho cả 2 mô hình SAX và DOM cung cấp bởi Sun.



Hình 2‑7: JAXP Interfaces

Ví dụ với JAXP/DOM



Hình 2‑8: Ví dụ với JAXP/DOM

## ****XML SCHEMA****

### ****Schema là gì?****

Sử dụng DTDs, chúng ta có thể kiểm tra cấu trúc file XML, tính đúng đắn của dữ liệu trong XML bằng DTDs. Tuy nhiên phương pháp này có nhiều hạn chế như:

  – Không có nhiều dạng kiểu dữ liệu

  – Không qui định được khoảng giá trị …

Chính vì thế **Schema** được tạo ra để khắc phục tất cả các nhược điểm của DTDs.

Như vậy về mặc cơ bản mục tiêu của Schema hoàn toàn giống như DTDs:

  – Định nghĩa cấu trúc các thành phần có trong XML

  – Định nghĩa các thuộc tính có trong Schema

  – Định nghĩa các thành phần con và thứ tự xuất hiện của chúng trong thành phần cha

  – Định nghĩa một thành phần là rỗng hay có chứa text

  – Định nghĩa kiểu dữ liệu cho các thành phần và thuộc tính

  – Định nghĩa giá trị mặc định cho thuộc tính

Tóm lại Schema được phát tirển trên nền tảng DTDs, tuy nhiên Schema có nhiều ưu điểm vượt trội hơn như:

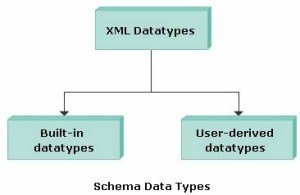
  –**Schema** viết theo định dạng như **XML** nên người dùng quen thuộc hơn so với **DTDs**, người dùng không cần phải làm quen với cấu trúc mới

  – **Schema** có thể mở rộng còn **DTDs** thì không: có thể sử dụng cho shema khác, người dùng có thể tự định nghĩa ra kiểu dữ liệu riêng từ các kiểu dữ liệu chuẩn, có thể dùng nhiều Schema cho 1 file XML cùng lúc

  – **Schema** hỗ trợ **Namespace**

  – **Schema** hỗ trợ nhiều dạng kiểu dữ liệu: dễ dàng kiểm tra tính đúng đắn của dữ liệu, dễ dàng khai báo định dạng, phạm vi của dữ liệu.

### ****Các kiểu dữ liệu trong Schema****

[](http://code5s.com/wp-content/uploads/2013/08/image002_2.jpg)

**Các kiểu dữ liệu có sẵn (Built-in datatypes) bao gồm các kiểu dữ liệu sau:**

  – string: dạng chuỗi

  – boolean: dạng luận lý

  – numeric: dạng số

  – dateTime: dạng ngày tháng, thời gian

  – binary: dạng nhị phân

  – anyURI: các chuỗi URI

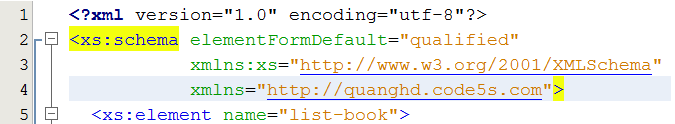
  – integer: số nguyên

  – decimal: số thập phân

  – time: thời gian

**Thẻ <schema>:**

Thẻ <schema> là thẻ gốc cho tất cả các file schema, thẻ <schema> có một số thuộc tính như sau:



**xmlns:xs=”http://www.w3.org/2001/XMLSchema”:** chỉ ra rằng các kiểu element, kiểu dữ liệu dùng trong schema được khai báo trong namespace <http://www.w3.org/2001/XMLSchema> với tiền tố xs

  – **xmlns=”http://code5s.com”:** namespace mặc định là [http://code5s.com](http://www.w3school.com/)

–**elementFormDefault=”qualified”:** chỉ ra rằng tên các thành phần được sử dụng trong file XML có khai báo schema phải thỏa mãn các khai báo namespace

### ****Khai báo Element trong Schema:****

Có 2 kiểu khai báo Element trong schema là: simple element và complex element

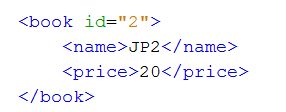
  – **Simple Element**: được dùng để khai báo cho các element chỉ chứa text không có khai báo các thẻ con hay thuộc tính

  – **Complex Element**: được dùng khai báo các khai báo Element có chứa element con, thuộc tính… có 4 dạng complex element:

+ Element rỗng:



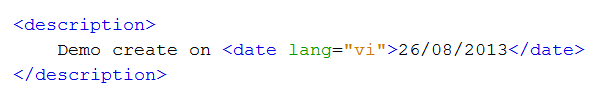
+ Element chỉ chứa các thẻ con:



 Element chỉ chứa text:

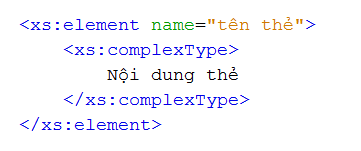


 Element chứa thẻ con và text:



### ****Complex Element:****

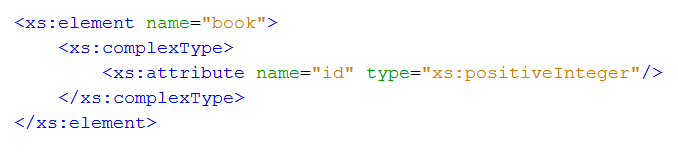
Cú pháp khai báo một complex element như sau:



#### Element rỗng:

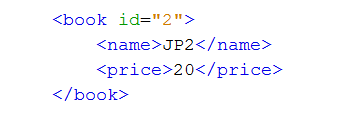
– Ví dụ: <book id=”1″/>

– Chúng ta thấy book  chỉ có 1 thuộc tính là id kiểu số nguyên, chúng ta khai báo như sau:



#### Element chỉ chứa các thẻ con:

 Ví dụ:



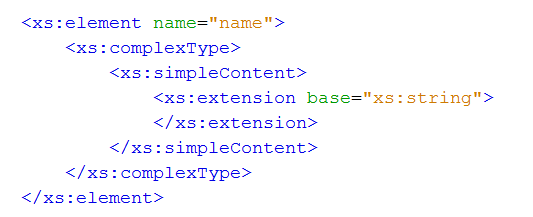
– Như khai báo XML, chúng ta thấy thẻ <book> có 2 thẻ con là <name> kiểu chuỗi và <price> kiểu số nguyên, có thuộc tính là id kiểu số nguyên



#### Element chỉ chứa text:

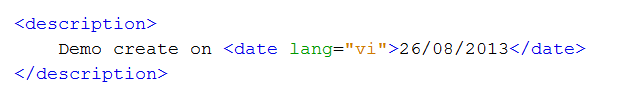
– Ví dụ: <name>Book 1</name>

– Trong thẻ <name> trên, chúng ta nội dung của thẻ có dạng chuỗi, schema đuuợc viết như sau:



#### Thẻ vừa có text vừa có element:

– Ví dụ:



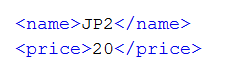
 Trong thẻ <description>, chúng ta thấy vừa có text vừa có thẻ con, schema được khai báo như sau:



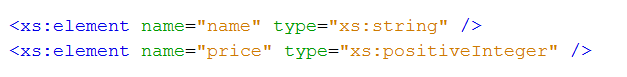
### ****Simple Element:****

– Cú pháp: <xs:element name=”aaa” type=”bbb”/>

– Ví dụ:



– Schema:



 Khai báo Attribute: Simple Element không thể có Attribute, khi 1 thẻ có Attribute, thẻ đó phải được khai báo ở dạng Complex Element, tuy nhiên 1 Attribute chúng ta có thể khai báo dưới dạng Simple Element theo cú pháp sau:

<xs:attribute name=”aaa” type=”bbb”/>

– Ví dụ: <book id=”1″/>

– Khai báo Schema:

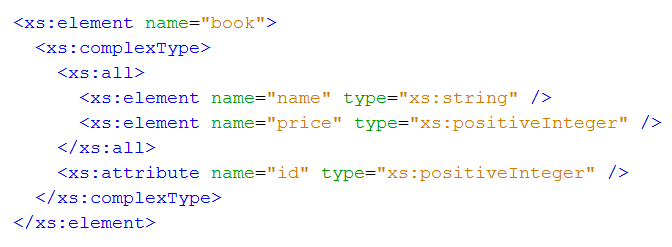


### ****. Schema Indicator:****

Chúng ta có thể kiểm soát các thẻ được sử dụng như thế nào trong XML bằng các Indicator. Có  5 dạng Indicator:

#### <xs:all>:

– Các thẻ con có thể xuất hiện với thứ tự bất kỳ và các thẻ con phải xuất hiện 1 lần duy nhất.



#### <xs:choice>

– Chỉ một trong số các thẻ con được xuất hiện



#### <xs:sequence>

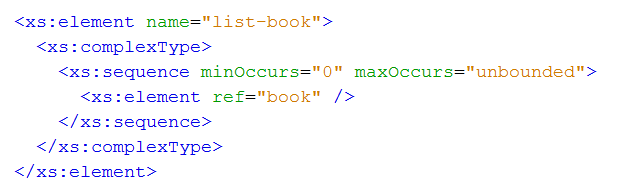
– Các thẻ con xuất hiện theo thứ tự



#### Số lần xuất hiện tối đa và tối thiểu:

– <xs:maxOccurs>: số lần xuất hiện tối đa

– <xs:minOccurs>: số lần xuất hiện tối thiểu



## Android

### Giới thiệu

Android là một gói phần mềm và hệ điều hành dựa trên linux cho các thiết bị di động như máy tính bảng và điện thoại thông minh.

Android được phát triển bởi Google và sau đó là OHA (Open Handset Alliance). Ngôn ngữ chủ yếu được sử dụng để viết mã Android là Java mặc dù các ngôn ngữ khác có thể được sử dụng để viết trên Android.

Mục tiêu của android là tạo ra một sản phẩm trong thế giới thực để cải thiện trải nghiệm di động cho người dùng.

Có rất nhiều phiên bản android như Lollipop, Kitkat, Jelly Bean, Ice Cream Sandwich, Froyo, Ecliar, Donut.

Open Handset Alliance (OHA) là một tập đoàn gồm 84 công ty như Google, Samsung, AKM, Synaptics, KDDI, Garmin, Teleca, Ebay, Intel etc.

Được thành lập vào ngày 5 tháng 11 năm 2007, do Google dẫn đầu. Cam kết nâng cao các tiêu chuẩn mở, cung cấp dịch vụ và triển khai các thiết bị cầm tay sử dụng Android Platform.

### Chức năng và danh mục ứng dụng

Android có một số chức năng quan trọng:

1. Android là mã nguồn mở
2. Bất kỳ ai cũng tùy biến lại Android Platform
3. Người sử dụng có thể lựa chọn nhiều ứng dụng trên Android.
4. Android cung cấp nhiều tính năng thú vị như: thời tiết, mở màn hình, RSS (Really Simple Syndication), …
5. Android cũng hỗ trợ cho các dịch vụ nhắn tin (SMS và MMS), trình duyệt web, lưu trữ dữ liệu (SQLite), kết nối (GSM, CDMA, Blue Tooth, Wi-Fi vv), media....

Hiện nay trên thị trường có nhiều danh mục ứng dụng trong Android:

* Entertainment
* Tools
* Communication
* Productivity
* Personalization
* Music and Audio
* Social
* Media and Video
* Travel and Local etc.

### Lịch sử phát triển và phiên bản

Tháng 10, năm 2003, Công ty Android được thành lập bởi Andy Rubin, Rich Miner, Nick Sears và Chris White. Sứ mệnh của công ty là tạo ra một hệ điều hành tiên tiến cho máy ảnh kỹ thuật số, biến chúng thành các thiết bị thông minh hơn và có khả năng nhận thức tốt hơn về vị trí cũng như sở thích của người sở hữu nó.

Công ty nhanh chóng nhận ra rằng, đây thực sự là một thị trường cằn cỗi, đầy sỏi đá. Và vì thế, Android Inc đã chú ý tới thị trường điện thoại thông minh với nỗ lực cạnh tranh với các nền tảng lớn lúc đó là Symbian và Windows Mobile.

Tính tới thời điểm hiện tại, Android đã xuất hiện trên thị trường di động được hơn 6 năm và trải qua 13 bản cập nhật lớn. Mỗi bản cập nhật này đều mang lại những tính năng mới góp phần quan trọng trong sự phát triển và thống trị của nền tảng này.

#### Android 1.5 Cupcake

Điểm nhấn đáng chú ý nhất ra đời cùng Android 1.5 Cupcake được cho là bàn phím ảo. Tuy nhiên trở lại thời điểm những năm 2008/2009, smartphone chủ yếu được ra đời với bàn phím QWERTY vật lý. Sự dịch chuyển từ yêu thích bàn phím cứng sang bàn phím ảo chỉ bắt đầu nhen nhóm khi iPhone đời đầu ra mắt vào năm 2007.

#### Android 1.6 Donut

Android 1.6 Donut vào thời điểm ra mắt không được đánh giá cao bởi người dùng vì những thay đổi nền tảng này mang lại không thực sự rõ rệt. Tuy nhiên, Android 1.6 đánh dấu việc hệ điều hành này đã hỗ trợ nhiều độ phân giải màn hình khác nhau. Đặc điểm này thực sự quan trọng trong quá trình phát triển của Android.

#### Android 2.0 Eclair

Trái ngược với Android 1.6 Donut, Android 2.0 có nhiều tính năng mới đến mức chỉ được chọn ra một tính năng ấn tượng nhất là rất khó khăn, dẫu vậy chuyên trang Phonearena đánh giá tính năng nổi bật nhất thuộc về phần mềm camera. Theo đó, trước khi Android 2.0 xuất hiện, Android không hỗ trợ đèn flash LED. Các tính năng như cân bằng trắng, chế độ lấy nét, chế độ chụp cảnh, hiệu ứng màu cũng hoàn toàn không khả dụng.

#### Android 2.2 Froyo

Hai điểm nhấn xuất hiện trên Android 2.2 Froyo là hiệu năng xử lý tuyệt vời và hỗ trợ kết nối Wi-Fi. Ở thời điểm nền tảng này ra mắt, Google khẳng định hiệu năng thiết bị vận hành trên nền tảng Android 2.2 có thể tăng lên gấp 2,5 lần.

Tính tới thời điểm hiện tại, Android đã xuất hiện trên thị trường di động được hơn 6 năm và trải qua 10 bản cập nhật lớn. Mỗi bản cập nhật này đều mang lại những tính năng mới góp phần quan trọng trong sự phát triển và thống trị của nền tảng này.

#### Android 2.3 Gingerbread

Android 2.3 Gingerbread được cho là một trong những phiên bản Android phổ biến nhất trong lịch sử hệ điều hành này. Thậm chí cho đến tận hôm nay, vẫn có tới 11% các thiết bị Android chạy phiên bản 2.3 Gingerbread.Android 2.3 Gingerbread mang đến cho thiết bị khả năng hỗ trợ nhiều loại cảm biến, bao gồm những cảm biến như cảm biến đa áp suất khí quyển (barometer) hay con quay hồi chuyển (gyroscope). Những cảm biến này ảnh hưởng rất nhiều đến tính năng và khả năng tương tác của người dùng với thiết bị Android về sau

#### Android 3.0 Honeycomb

Những đặc điểm đáng chú ý nhất trên từng phiên bản Android 6 Honeycomb là một trong những bản cập nhật Android ít được biết đến nhất, tuy nhiên không phải vì thế mà nền tảng này không có những đóng góp chung cho sự phát triển của Android. Nền tảng này đánh dấu sự quan tâm của Google đến trải nghiệm của người dùng các thiết bị máy tính bảng.

#### Android 4.0 Ice Cream Sandwich

Lần đầu tiên xuất hiện trong nền tảng Android 3.0 Honeycomb, giao diện "Holo" được Google cải thiện và trau chuốt thêm rất nhiều trong Android 4.0. Hiểu một cách đơn giản, giao diện người dùng Holo được Google đưa ra để giảm thiểu những khác biệt về thiết kế giữa phần mềm của các nhà sản xuất, nhà phát triển ứng dụng và phần mềm của Google.

Android 4.0 Ice Cream Sandwich để lại rất nhiều ảnh hưởng đến giao diện Android về sau.

#### Android 4.1 Jelly Bean

Android 4.1 Jelly Bean đánh dấu sự xuất hiện lần đầu tiên của Google Now, một thành phần quan trọng trong hệ sinh thái của Google. Tận dụng sức mạnh từ cỗ máy tìm kiếm của mình, Google Now mang đến cho người dùng những thông tin hữu ích ở giao diện dạng thẻ tùy thuộc vào ngữ cảnh. "Người trợ lý ảo" này còn có khả năng xử lý ngôn ngữ tự nhiên được đánh giá cao.

#### Android 4.4 KitKat

Với KitKat, Google không chỉ mang tới cho người dùng một giao diện tương tác hiện đại hơn mà còn tập trung phát triển một hệ điều hành có thể vận hành trơn tru trên cả các thiết bị giá rẻ, cấu hình khiêm tốn. Google khẳng định các thiết bị với chỉ 512MB cũng có thể cài đặt và chạy được hệ điều hành này. Đây là một bước tiến quan trọng trong việc giảm sự phân mảnh nền tảng trong thế giới các thiết bị Android.

#### Android 5.0 Lollipop

Lollipop rõ ràng là hệ điều hành Android có phần nhìn bắt mắt nhất từ trước đến nay với ngôn ngữ thiết kế mới được Google đặt tên là Material Design. Trong hệ điều hành này, giao diện Android được làm mới theo triết lý thiết kế phẳng cùng với hệ màu tươi sáng, khác hẳn với các hệ điều hành tiền nhiệm.

#### Android 6.0 Marshmallow

Android Marshmallow đã được Google công bố vào tháng 9 năm 2015, cải thiện tuổi thọ pin và thêm các tính năng mới như Hỗ trợ tính năng Hiện hành trên Tap và hỗ trợ cảm biến vân tay.

#### Android 7.0 Nougat

Các tính năng của Android Nougat được đưa ra trước tiên và cho cộng đồng chọn tên, được chính thức công bố vào cuối tháng 6 năm 2016. Cùng với việc cải thiện hiệu suất và quản lý pin nhờ một tính năng gọi là Doze on-the-go, Nougat cũng mang lại nhiều tiện ích như màn hình đa nhiệm.

Phiên bản 7,1 đã sớm được tung ra trên các điện thoại thông minh Pixel và Pixel XL của Google. Nó hỗ trợ thêm cho trình tạo Pixel mới, Google Assistant, Night Light (chủ yếu là bộ lọc ánh sáng màu xanh), chế độ Daydream VR, các phím tắt được gọi là 'Moves' và nhiều tính năng khác.

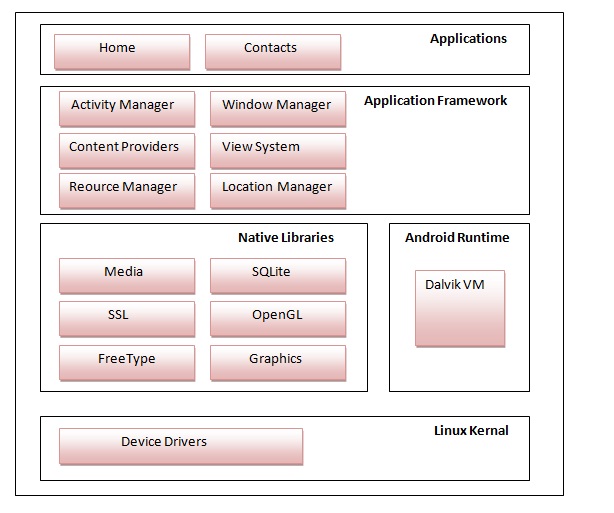
#### Android 8.0 Oreo

Vào năm 2017, Google chính thức ra phiên bản mới là Android Oreo. Phiên bản cập nhật mới nhất này cung cấp nhiều tính năng mới tuyệt vời. Trong phiên bản này Google đã có nhiều cải tiến bổ sung cho chất lượng âm thanh và nhập văn bản, cũng như quản lý tài nguyên.

### Kiến trúc Android

Kiến trúc Android (android architecture) hay còn gọi là Ngăn xếp phần mềm Android (Android software stack) được chia thành 5 phần:

1. linux kernel
2. native libraries (middleware),
3. Android Runtime
4. Application Framework
5. Applications



#### Linux kernel

Là trung tâm của kiến trúc Android. Linux kernel chịu trách nhiệm điều khiển thiết bị, quản lý điện năng, quản lý bộ nhớ, quản lý thiết bị và truy cập tài nguyên.

#### Native Libraries

Ở phía trên cùng của Linux kernel là các thư viện Native: WebKit, OpenGL, FreeType, SQLite, Media, C runtime library (libc) ...

Thư viện Webkit hỗ trợ duyệt web, thư viện SQLite hỗ trợ cơ sở dữ liệu, FreeType hỗ trợ font chữ...

#### Android Runtime

Android có các thư viện lõi và DVM (Dalvik Virtual Machine) có trách nhiệm chạy ứng dụng Android. DVM giống như JVM nhưng nó được tối ưu hóa cho các thiết bị di động. Nó tiêu thụ ít bộ nhớ hơn và mang lại hiệu suất nhanh.

#### Android Framework

Ở phía trên thư viện Native và Android Runtime là Android Framework gồm có các API Android như:UI (User Interface), telephony, resources, locations, Content Providers (data) and package managers. Android Framework cung cấp nhiều Class và Interface cho việc phát triển ứng dụng.

#### Applications

Trên Android Framework có nhiều ứng dụng: home, contact, settings, games, browsers được sử dụng Android Framework, Android Framework sử dụng Android runtime and các thư viện, Android runtime and các thư viện native đang sử dụng linux kernal.

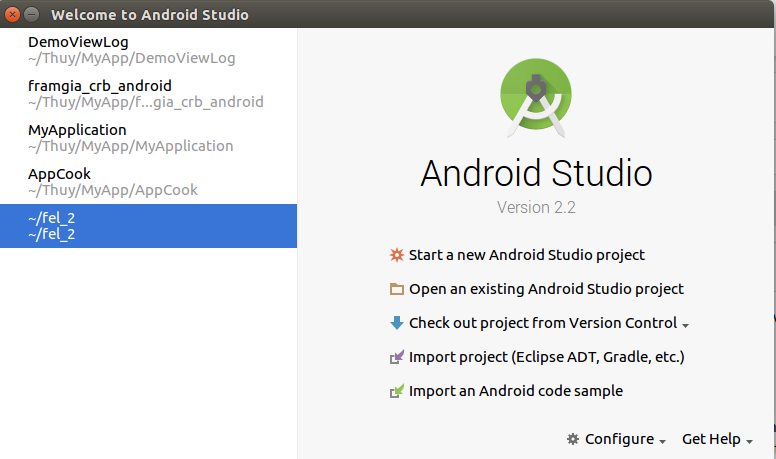
### Giới thiệu về Android Studio và Android SDK

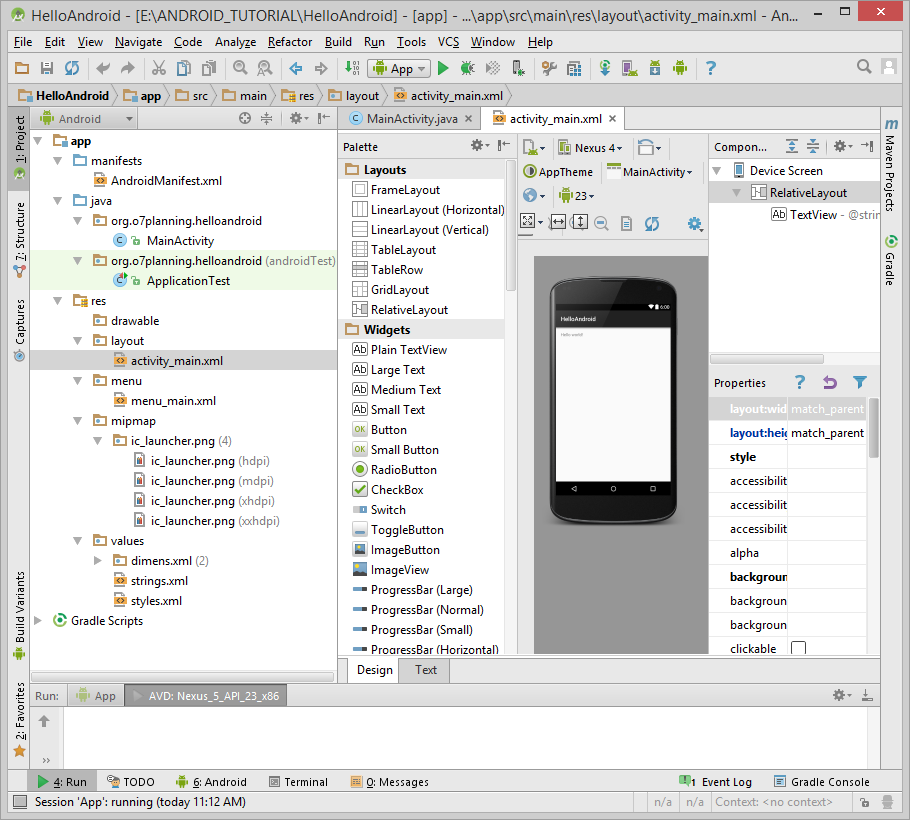
Tháng 5 năm 2013, Google công bố Android Studio, một môi trường phát triển ứng dụng tích hợp (IDE) dành riêng cho Android, mã nguồn mở, dựa trên IDE Java IntelliJ của hãng JetBrains (đối thủ với Eclipse và Netbeans, vốn khá quen thuộc với dân lập trình Java).

Android Studio chạy trên Windows, Mac và Linux, nhằm thay thế cho Eclipse Android Development Tool (ADT) vốn được sử dụng làm IDE chính trong các năm trước đó.

Một số tính năng nổi bật:

* Bộ công cụ build ứng dụng dựa trên Gradle (thay vì Maven).
* Chức năng dò và sửa lỗi nhanh, hướng Android.
* Công cụ chỉnh sửa màn hình dạng kéo thả tiện lợi.
* Các wizard tích hợp nhằm giúp lập trình viên tạo ứng dụng từ mẫu có sẵn.
* Tích hợp Google Cloud Platform, dễ dàng tích hợp với Google Cloud Messaging và App Engine của Google.





Và đi kèm với Android Studio là Android SDK. Nếu Android Studio là trình soạn thảo code (IDE) thì Android SDK là bộ tổng hợp các công cụ để build app, các bản mẫu máy ảo Android (sử dụng để test app) cần thiết để làm ra một ứng dụng Android hoàn chỉnh.

# XÂY DỰNG ỨNG DỤNG

## Mô tả ứng dụng

* Xây dựng Application lấy dữ liệu danh sách các sản phẩm và chi tiết sản phẩm theo mã sản phẩm. Kết quả trả về theo dạng XML.
* Xây dựng ứng dụng Android lấy dữ liệu trên Application và dùng Java DOM Parser để đọc dữ liệu XML và hiển thị dữ liệu đó lên màn hình.

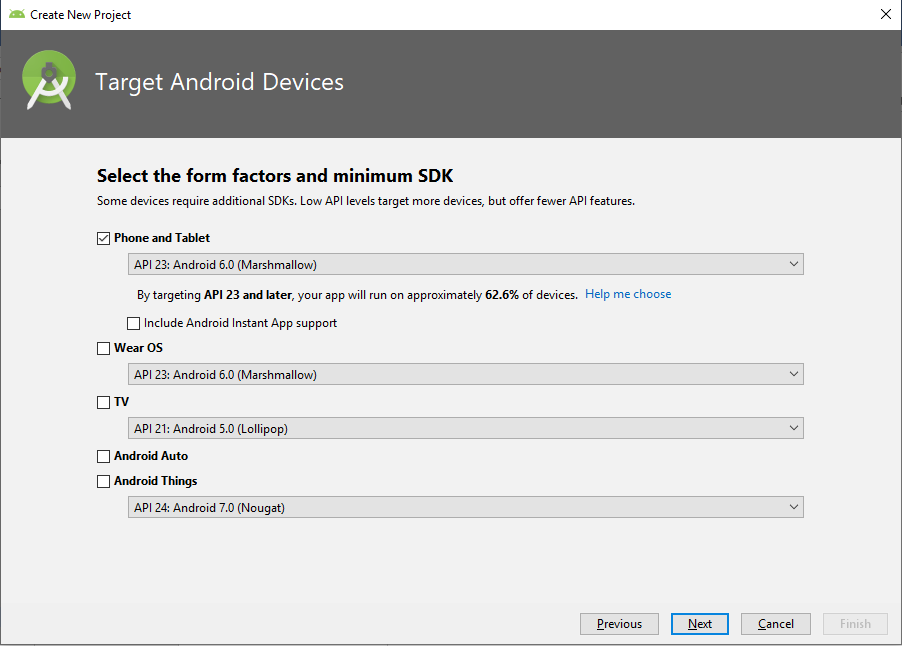
## Các thao tác triển khai ứng dụng

* Xây dụng Application bằng Android Studio
* Xây dựng các chức năng: Lấy danh sách sản phẩm, Lấy chi tiết sản phẩm theo mã sản phẩm

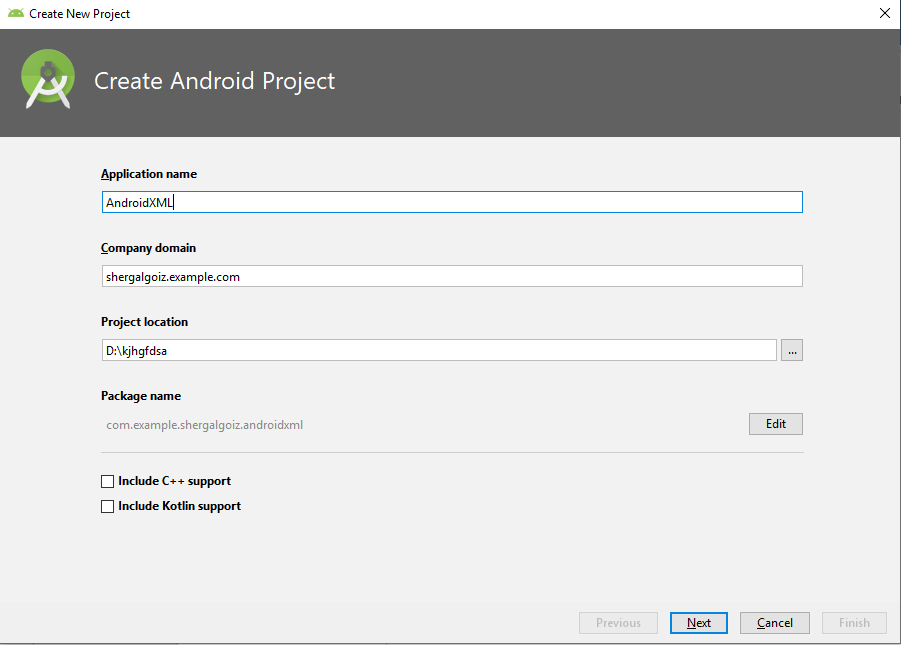
Chạy Android Studio trên thiết bị, chọn new project

Chọn version Android . Lưu ý bản Android 6.0 trở lên chúng ta phải import thêm dòng lệnh cho phép thiết bị sử dụng cuối cùng, để cho phép thiết bị đó kết nối mạng

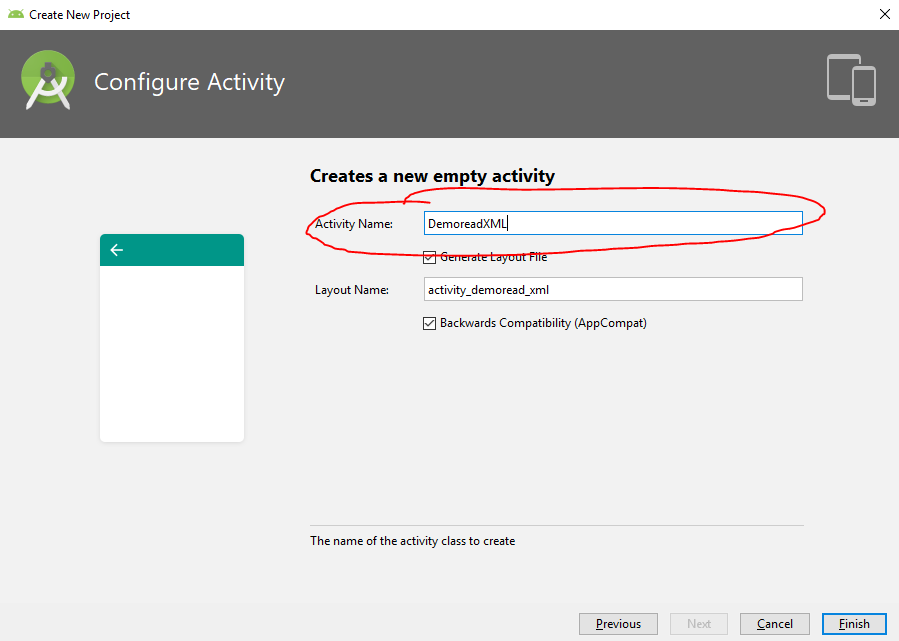
<**manifest xmlns:android="http://schemas.android.com/apk/res/android"**



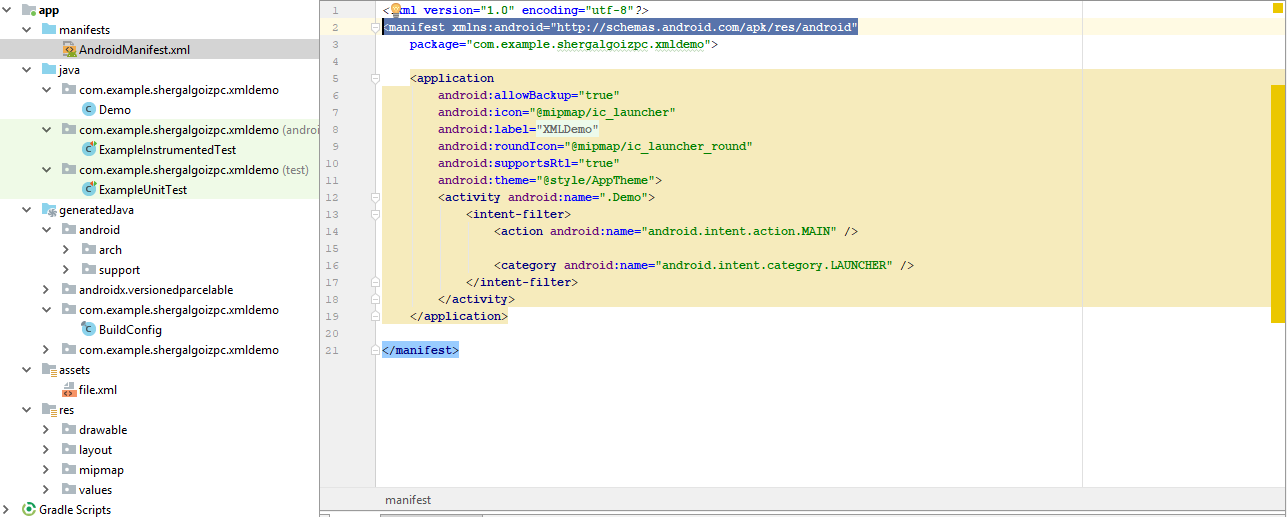
Chọn đúng version để khởi tạo



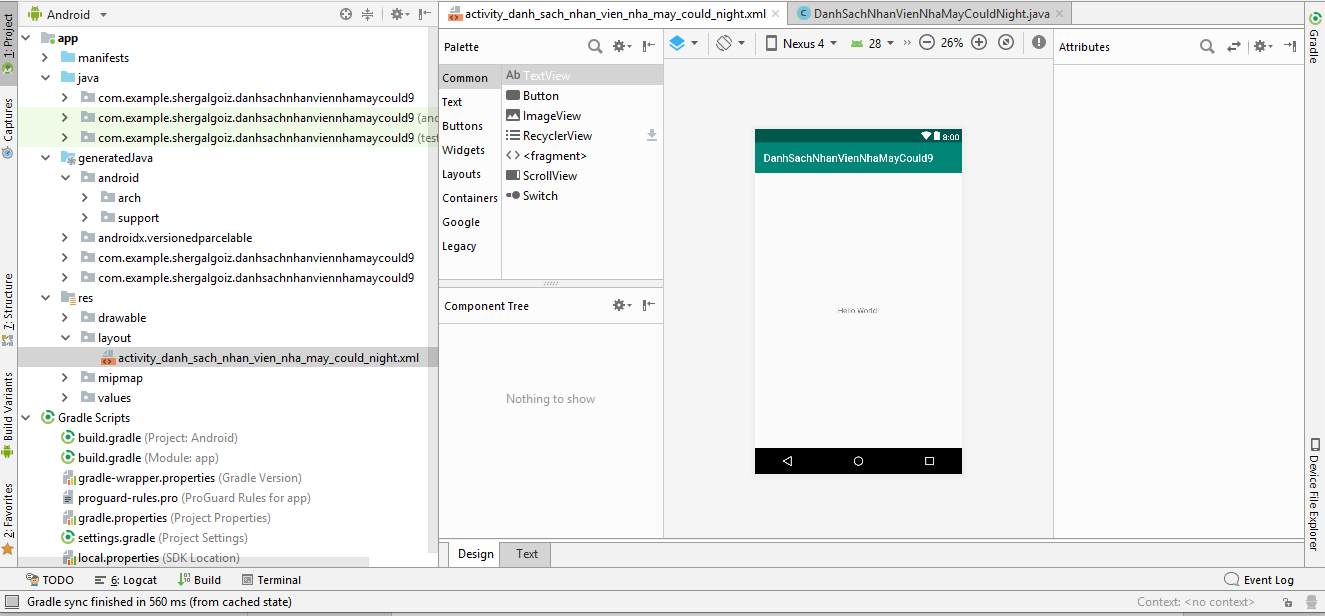
Đặt tên công ty ở phần Company Domain



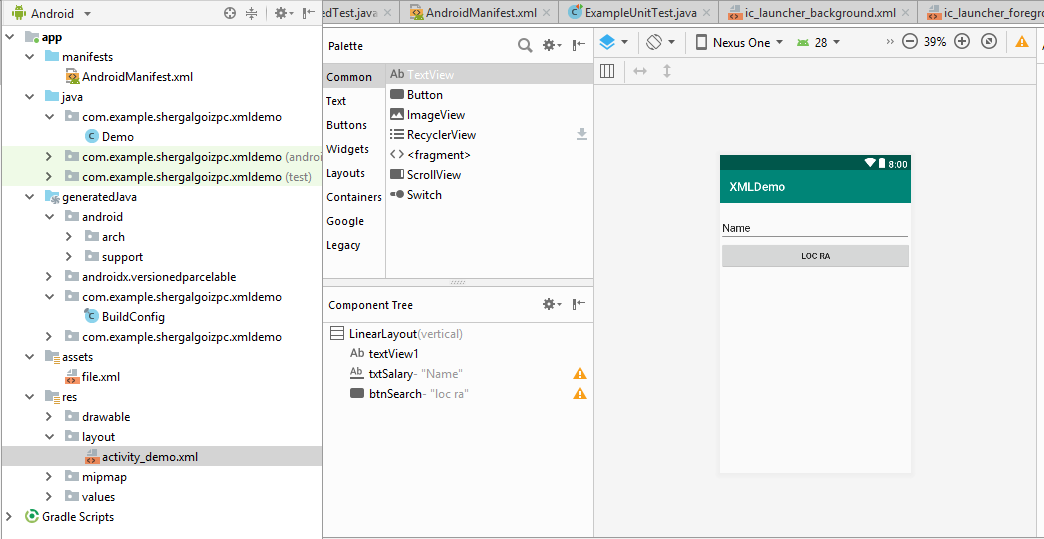
Bước đặt tên Activity Name là bước quan trọng, tên này thể hiện tiêu đề khi app thực thi



Thêm dòng lệnh này để android 6.0 trở lên đi build đc phép giao tiếp với mạng internet. (có paste trên đầu phần giới thiệu)



Xây dựng giao diện



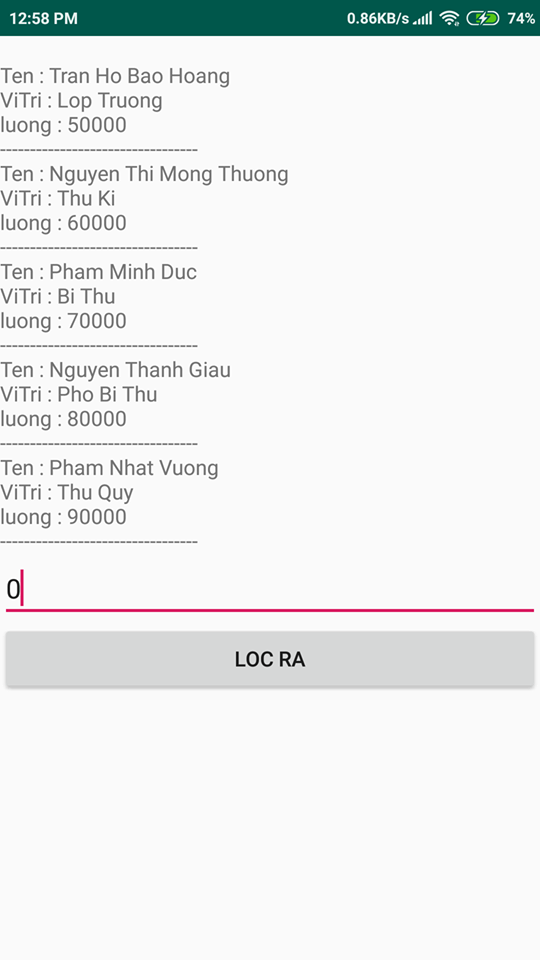
Dùng LinearLayout để xây dựng các items từ trên xuống. Hệ thổng phản hồi nhanh và đơn giản.

Gồm có 3 items:

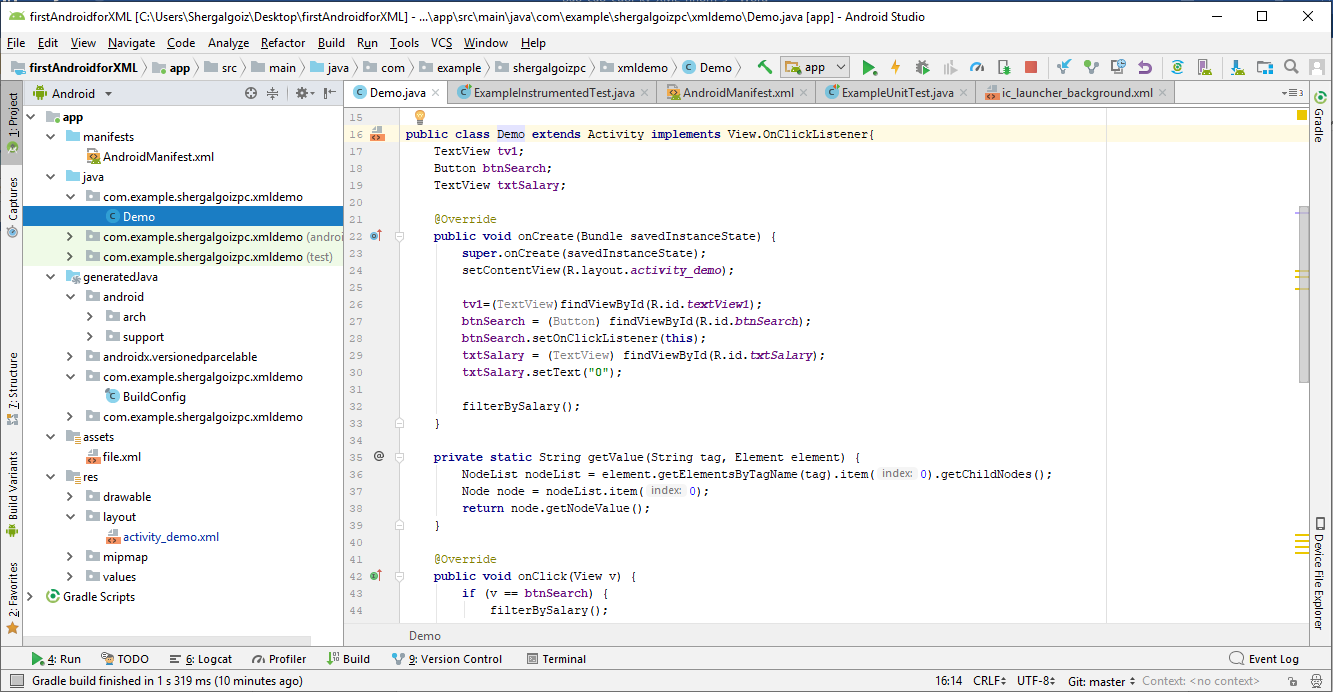
Một TextView

Một textBox  
Một Button

Giao diện thể hiện trên Android



Giao diện trên client

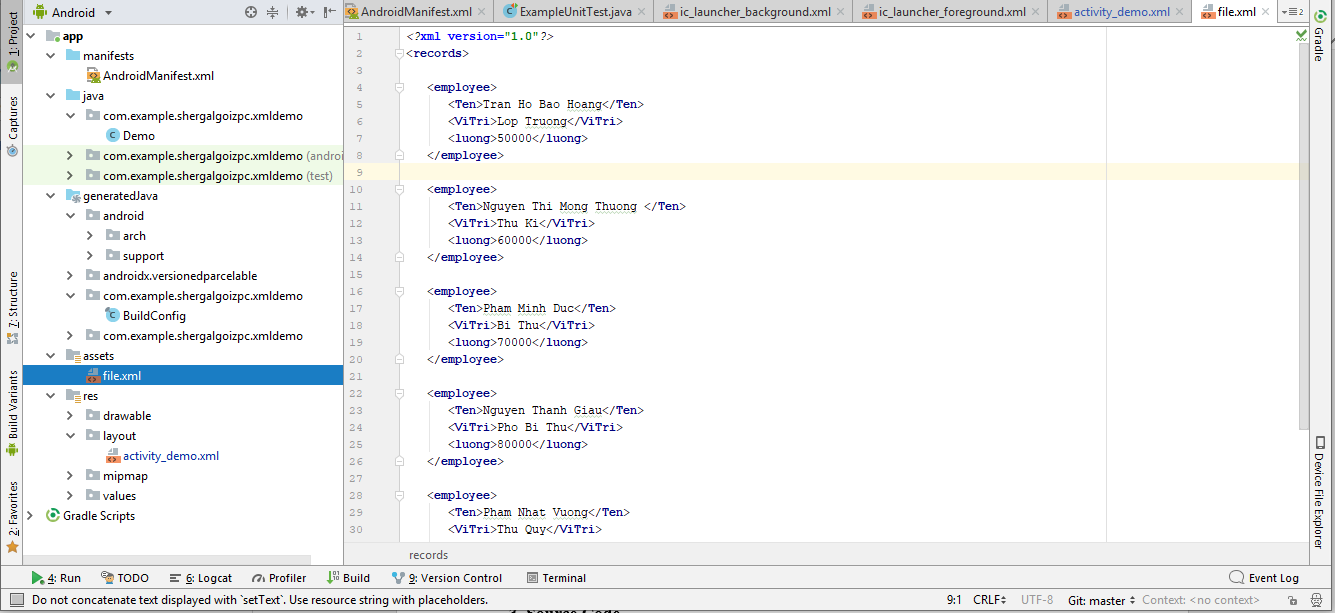
Viết code để xứ lí trên đây 

**package** com.example.shergalgoizpc.xmldemo;  
  
**import** java.io.InputStream;  
**import** javax.xml.parsers.DocumentBuilder;  
**import** javax.xml.parsers.DocumentBuilderFactory;  
**import** org.w3c.dom.Document;  
**import** org.w3c.dom.Element;  
**import** org.w3c.dom.Node;  
**import** org.w3c.dom.NodeList;  
**import** android.app.Activity;  
**import** android.os.Bundle;  
**import** android.view.View;  
**import** android.widget.Button;  
**import** android.widget.TextView;  
  
**public class** Demo **extends** Activity **implements** View.OnClickListener{  
 TextView **tv1**;  
 Button **btnSearch**;  
 TextView **txtSalary**;  
  
 @Override  
 **public void** onCreate(Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 setContentView(R.layout.***activity\_demo***);  
  
 **tv1**=(TextView)findViewById(R.id.***textView1***);  
 **btnSearch** = (Button) findViewById(R.id.***btnSearch***);  
 **btnSearch**.setOnClickListener(**this**);  
 **txtSalary** = (TextView) findViewById(R.id.***txtSalary***);  
 **txtSalary**.setText(**"0"**);  
  
 filterBySalary();  
 }  
  
 **private static** String getValue(String tag, Element element) {  
 NodeList nodeList = element.getElementsByTagName(tag).item(0).getChildNodes();  
 Node node = nodeList.item(0);  
 **return** node.getNodeValue();  
 }  
  
 @Override  
 **public void** onClick(View v) {  
 **if** (v == **btnSearch**) {  
 filterBySalary();  
 }  
 }  
  
 **private void** filterBySalary() {  
 **tv1**.setText(**""**);  
 Long inputSalary = Long.*parseLong*(**txtSalary**.getText().toString());  
 **try** {  
 InputStream is = getAssets().open(**"file.xml"**);  
  
 DocumentBuilderFactory dbFactory = DocumentBuilderFactory.*newInstance*();  
 DocumentBuilder dBuilder = dbFactory.newDocumentBuilder();  
 Document doc = dBuilder.parse(is);  
  
 Element element=doc.getDocumentElement();  
 element.normalize();  
  
 NodeList nList = doc.getElementsByTagName(**"employee"**);  
  
 **for** (**int** i=0; i<nList.getLength(); i++) {  
 Node node = nList.item(i);  
 **if** (node.getNodeType() == Node.***ELEMENT\_NODE***) {  
 Element element2 = (Element) node;  
  
 **long** curSalary = Long.*parseLong*(*getValue*(**"luong"**, element2));  
 **if** (inputSalary > curSalary) **continue**;  
  
 **tv1**.setText(**tv1**.getText()+**"\nTen : "** + *getValue*(**"Ten"**, element2)+**"\n"**);  
 **tv1**.setText(**tv1**.getText()+**"ViTri : "** + *getValue*(**"ViTri"**, element2)+**"\n"**);  
 **tv1**.setText(**tv1**.getText()+**"luong : "** + *getValue*(**"luong"**, element2)+**"\n"**);  
 **tv1**.setText(**tv1**.getText()+**"---------------------------------"**);  
 }  
 }  
  
 } **catch** (Exception e) {e.printStackTrace();}  
 }  
}

Bởi vì có thêm 1 tính năng nho nhỏ, là lọc những người có số lương cao hơn, bỏ đi những người có số lương đã thấp hơn. Chúng e đã thêm code vào duyệt và lọc ra.  
So sánh vs 1 giá trị. Nếu như ko có giá trị thì chương trình sẽ out ra khỏi try đi vào catch .

**txtSalary**.setText(**"0"**);

Bắt buộc phải có giá trị. Thấp nhất là số “ 0 ”:



Đọc từ file XML này được để trong thư mục của App

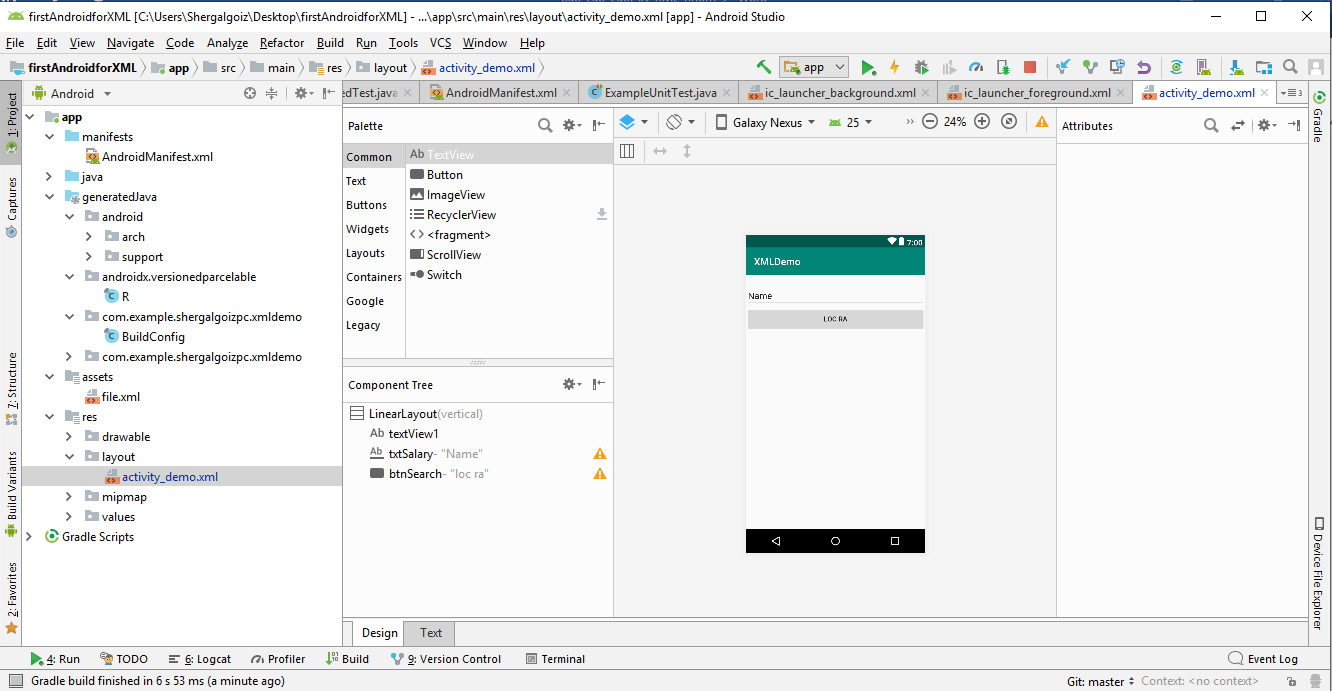
C:\Users\Shergalgoiz\Desktop\ {firstAndroidforXML\app\src\main\assets}  
đây là đường dẫn tuyệt đối ở các client khác nhau sẽ khác nhau  
{..\..\..\app\src\main\assets}

Đường dẫn tương đối.

Nội dung file XML thử

*<?***xml version="1.0"***?>*<**records**>  
  
 <**employee**>  
 <**Ten**>Tran Ho Bao Hoang</**Ten**>  
 <**ViTri**>Lop Truong</**ViTri**>  
 <**luong**>50000</**luong**>  
 </**employee**>  
  
 <**employee**>  
 <**Ten**>Nguyen Thi Mong Thuong </**Ten**>  
 <**ViTri**>Thu Ki</**ViTri**>  
 <**luong**>60000</**luong**>  
 </**employee**>  
  
 <**employee**>  
 <**Ten**>Pham Minh Duc</**Ten**>  
 <**ViTri**>Bi Thu</**ViTri**>  
 <**luong**>70000</**luong**>  
 </**employee**>  
  
 <**employee**>  
 <**Ten**>Nguyen Thanh Giau</**Ten**>  
 <**ViTri**>Pho Bi Thu</**ViTri**>  
 <**luong**>80000</**luong**>  
 </**employee**>  
  
 <**employee**>  
 <**Ten**>Pham Nhat Vuong</**Ten**>  
 <**ViTri**>Thu Quy</**ViTri**>  
 <**luong**>90000</**luong**>  
 </**employee**>  
  
</**records**>

## Giao diện chụp kết quả triển khai

**

# TÀI LIỆU KHAM KHẢO

1. <https://www.tutorialspoint.com/java_xml/java_dom_parser.htm>
2. <https://www.howkteam.vn/Course/Khoa-hoc-lap-trinh-Android-co-ban/Gioi-thieu-lap-trinh-Android-11>
3. <https://androidcoban.com/gioi-thieu-ve-dieu-hanh-android.html>
4. <http://hiepsiit.com/khoa-hoc/android/androidlagi/53/1>
5. <https://o7planning.org/en/11007/android>
6. <https://stackoverflow.com/questions/9037605/how-i-can-clear-the-value-of-textview?fbclid=IwAR1B7HbQjeZkePwipfCbb3NHVDAk2sitSKHk39ijvafrTVjf7b63l2c3FQE>
7. <https://openweathermap.org/forecast5?fbclid=IwAR1X1c4QIOJXd-VvsIuV6SVp6LaCNWrI07B8ZVkAeAD39tYaMlYvKUCM-Ko>
8. <https://samples.openweathermap.org/data/2.5/forecast?q=London,us&mode=xml&appid=b6907d289e10d714a6e88b30761fae22>
9. <https://www.howkteam.vn/course/khoa-hoc-lap-trinh-android-co-ban-3>