***Giáo trình***

**Lập trình Ứng dụng Di động**

**GS. Phan Viết Hoàng**

**Bài học 2 (Buổi 2)**

**Lập trình Java**

### References

<http://iviettech.vn/blog/4247-hoc-lap-trinh-huong-doi-tuong-voi-java.html>

<http://iviettech.vn/blog/4632-cai-dat-mo-hinh-mvcmodel-view-controller-trong-java.html>

Hầu hết sinh viên chuyên ngành CNTT đã được học về lập trình hướng đối tượng và cả ngôn ngữ lập trình Java. Tuy nhiên, nhiều bạn vẫn thấy khó hiểu về lập trình hướng đối tượng cũng như cách áp dụng nó vào lập trình thực tế.

Để giúp các bạn dễ dàng tiếp cận hơn với chủ đề quan trọng này, trong seri bài viết chúng ta sẽ bàn về việc sử dụng lập trình hướng đối tượng trong Java từ cơ bản như cách sử dụng lập trình hướng đối tượng đến cài đặt mô hình MVC cũng như chuyển đổi giữa đối tượng và CSDL (ORM: Object Relation Mapping). Tôi sẽ cố gắng trình bày theo hướng thực hành với ví dụ cụ thể để các bạn dễ hiểu và dễ làm theo.

### 1. Lập trình hướng đối tượng là gì?

Trong lập trình thông thường chúng ta có 02 hướng tiếp cận để phân tích bài toán trước khi tiến hành lập trình là hướng cấu trúc(Structured Programming) và hướng đối tượng (Object Oriented Programming).

Lập trình có cấu trúc đã phát triển từ đầu những năm 1970 và đã khẳng định được ưu điểm của nó thông qua hàng loạt ứng dụng đã được phát triển và các ngôn ngữ lập trình hỗ trợ như Pascal, C… Với ưu điểm là cấu trúc rõ ràng, đơn giản, lập trình có cấu trúc đã phát triển mạnh trong giai đoạn đầu của ngành phần mềm với các ứng dụng có độ phức tạp thấp nhưng khi phải giải quyết các bài toán phức tạp hơn thì Lập trình có cấu trúc trở nên khó điều kiển và dần mất ưu thế về Lập trình hướng đối tượng.

Lập trình hướng đối tượng là phương thức tiếp cận bài toán theo hướng xem hệ thống là hoạt động của các đối tượng như trong thực tế, sau đó mô tả chúng và cho chúng tương tác với nhau để tạo nên hệ thống. Với sự tiếp cận như vậy giúp lập trình hướng đối tượng tư duy gần với thực tế hơn (lập trình có cấu trúc tư duy thiên về toán hơn) và độ phức tạp cũng chia ra cho việc mô tả các đối tượng và hành vi của nó nên độ phức tạp của bài toán cũng được chia nhỏ ra.

Chúng ta xem xét ví dụ sau để thấy được ưu điểm của Lập trình hướng đối tượng:

Ví dụ chúng ta cần viết một ứng dụng để mô tả một trận đấu bóng đá. Chúng ta biết trận đấu gồm 22 cầu thủ, 04 trọng tài, khán giả, sân bóng và quả bóng. Trong đó mỗi hoạt động của quá bóng đều chi phối hoạt động của các thành phần còn lại. Ví dụ khi quả bóng được phát đi và chuẩn bị rơi xuống 1 điểm trên sân mỗi cầu thủ phải có phản ứng tương ứng hợp lý và trọng tài cũng phải di chuyển hợp lý, khán giả cũng phải có hành động phù hợp …

Nếu tiếp cận theo lập trình có cấu trúc chúng ta phải xét từng trường hợp cụ thể (22 cầu thử + 04 trọng tài + sân bong = 27 đối tượng phải xem xét) để điều khiển cho hợp lý làm cho bài toán vô cùng phức tạp. Còn lập trình hướng đối tượng không làm vậy, nó chia hệ thống ra thành các nhóm đối tượng như sau: Cầu thủ (có thể phân biệt thủ môn riêng), Trọng tài, Khán giả, Sân bóng và Quả bóng. Sau đó định nghĩa tính chất và hành vi của từng loại đối tượng. Ví dụ: cầu thủ có các chỉ số như Vị trí đảm nhận, tốc độ chạy, khả năng sút, mức độ tranh chấp, đánh đầu … và định nghĩa ngữ cảnh và hành vi của các cầu thủ theo ngữ cản đó. Ví dụ một cầu thủ thấy bóng cách xa 30 mét thì không cần di chuyển theo bóng để giữ vị trí, nếu cách xa 20 mét thì di chuyển theo hướng bóng để đón bóng, tốc độ di chuyển càng gần bóng càng nhanh chẳng hạn …. Sau đó mô tả cho 22 cầu thủ. Sau đó tạo ngữ cảnh cho trận đấu và các đối tượng sẽ hành xử theo hành vi đã được mô tả để tạo nên ứng dụng.

Tất nhiên, đây chỉ là ví dụ để bạn có ý niệm còn cài đặt nó không hề đơn giản cho người mới học lập trình hướng đối tượng, chúng ta sẽ thực hiện với bài toán dễ hơn để bạn dễ hình dung và thực hiện.

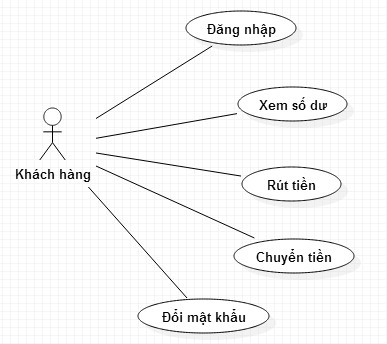
Lập trình hướng đối tượng đã thể hiện tính ưu việt của nó mà bằng chứng là hầu hết các ngôn ngữ lập trình hiện đại đều hỗ trợ mô hình này như C++, Java, C#, PHP… Ở thời điểm này nếu không rành về Lập trình hướng đối tượng thì quả là thiệt thòi cho một Lập trình viên.

Các lý thuyết chi tiết về Lập trình hướng đối tượng bạn có thể tìm đọc thêm ở nhiều tài liệu khác, trong bài này Tôi sẽ hướng dẫn bạn cài đặt một ứng dụng, hy vọng qua đó bạn sẽ hiểu rõ hơn về lập trình hướng đối tượng và cách cài đặt nó trên ngôn ngữ lập trình cụ thể là Java.

### 2. Bài toán và phân tích

Trong bài viết này, chúng ta sẽ cài đặt giả lập hoạt động của một máy ATM. Máy ATM cho phép người dùng đăng nhập, rút tiền, chuyển tiền, xem số dư, đổi mật khẩu.. Trong ví dụ này chúng ta giới hạn các chức năng của máy ATM gồm: đăng nhập, rút tiền, chuyển tiền, xem số dư, đổi mật khẩu và sẽ giảm độ phức tạp đến mức tối đa.

Biểu đồ chức năng của hệ thống ATM như sau:

[](http://iviettech.vn/wp-content/uploads/2015/03/ATM-Use-Case-Diagram.png)

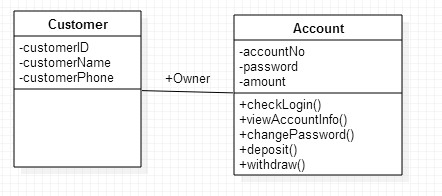
Bạn có thể xem thêm về cách xây dựng bản vẽ này tại đây: [**Use Case Diagram**](http://iviettech.vn/blog/543-ban-ve-use-case-use-case-diagram.html).

Bây giờ chúng ta xem trong hệ thống này có những đối tượng nào nhé. Quan sát các đối tượng trong hệ thống chúng ta thấy có những đối tượng sau:

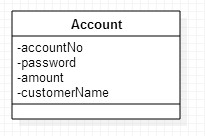
(Làm thế nào để thấy được các bạn vui lòng đọc bài này: [**Class Diagram**](http://iviettech.vn/blog/813-classs-diagarm-ban-ve-ve-classlop.html))

* Khách hàng (Customer)
* Tài khoản (Account)

Thiết lập thêm các thuộc tính của các lớp này chúng ta có sơ đồ lớp như sau:

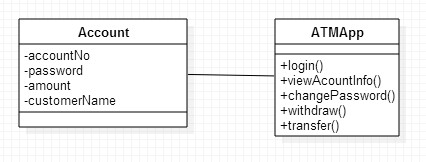
[](http://iviettech.vn/wp-content/uploads/2015/03/ATM-Class-Diagram.png)

Để đơn giản cho quá trình cài đặt, tôi sẽ đơn giản quan hệ này bằng cách cho tên khách hàng vào lớp tài khoản luôn là biết khách hàng nào sở hữu tài khoản này và bỏ qua đối tượng khách hàng. Lúc đó hệ thống chỉ cần mô tả 01 lớp là Account như sau:

[](http://iviettech.vn/wp-content/uploads/2015/03/ATM-Class-Diagram-2.png)

Mọi chuyện có vẻ đơn giản hơn rồi.

Để cài đặt ứng dụng trong Java chúng ta phải cài đặt một lớp ứng dụng có tên là ATMApp(ứng dụng ATM) để sử dụng lớp Account này nhằm tạo ra chức năng cho khách hàng sử dụng. Lúc đó sơ đồ lớp như sau:

[](http://iviettech.vn/wp-content/uploads/2015/03/ATM-Class-Diagram-3.png)

Các phương thức login(đăng nhập), viewAccountInfo(xem thông tin khách hàng), changePassword(thay đổi mật khẩu), withdraw(rút tiền), transfer(chuyển tiền) của lớp ATM sẽ tương tác với các đối tượng tài khoản được tạo ra từ lớp Account để tạo nên hệ thống.

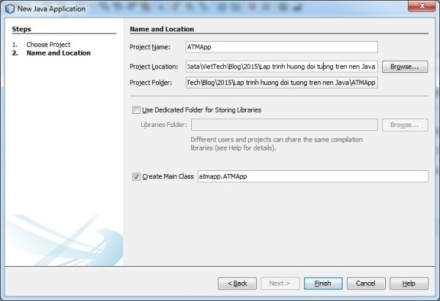
Bây giờ chúng ta sẽ cài đặt ứng dụng này trên Java.

### 3. Cài đặt ứng dụng ATM trên Java

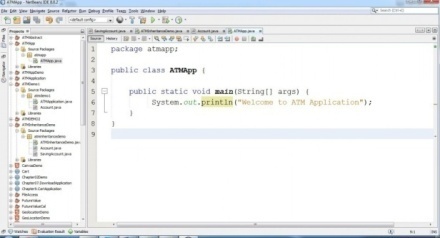
Để thực hiện các bước theo bài này bạn phải cài đặt Java và NetBean trên máy tính của mình. [Xem thêm bài hướng dẫn cài đặt Java và NetBean để cài đặt môi trường.](http://www.khamphacntt.com.vn/java-co-ban-bai-2-cai-dat-jdk-va-netbean-ide.html%20)

#### Bước 1. Tạo dự án ATMApp

Khởi động NetBean và tạo một dự án ATMApp như hình bên dưới.

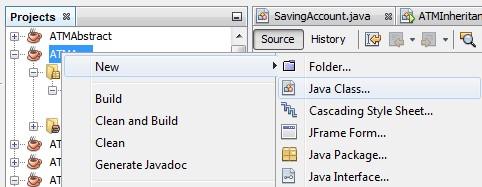
[](http://iviettech.vn/wp-content/uploads/2015/03/Tao-du-an-1.jpg)

Bỏ qua các Comment và thêm vào dòng lệnh System.out.print như bên dưới để bắt đầu ứng dụng. Nhấn Play để chạy và kiểm tra ứng dụng.

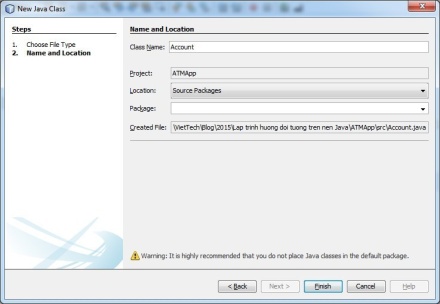
[](http://iviettech.vn/wp-content/uploads/2015/03/Tao-du-an-2.jpg)

#### Bước 2. Xây dựng lớp Account

Kích phải chuột lên gói ATMApp bên trái – > New -> Java Class

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-1.jpg)

Nhập tên lớp là Account, chọn Package là ATMApp và kích Finish để tạo lớp.

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-2.jpg)

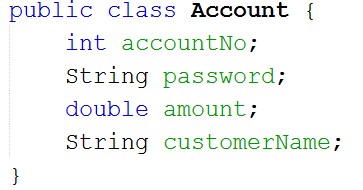
Lớp ban đầu chỉ có:

public class Account {

}

Chúng ta sẽ xem hình mô tả lớp Account ở trên để tiến hành cài đặt.

* Tiếp theo chúng ta sẽ code các thuộc tính cho lớp. Mặc định cách sử dụng kiểu dữ liệu và khai báo bạn đã biết. Nếu chưa biết bạn nên đọc thêm sách hướng dẫn lập trình cơ bản trên Java trước.

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-3.jpg)

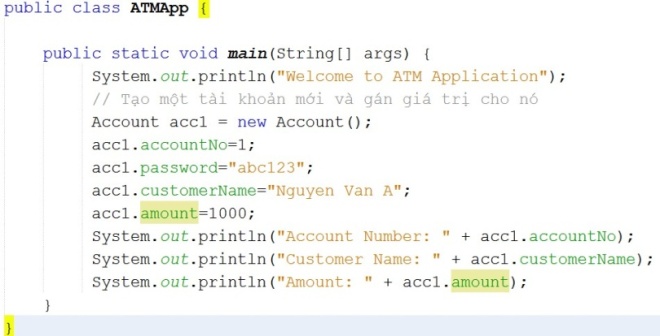
* Tiếp tục chúng ta sẽ code Contructor để khởi tạo giá trị cho lớp:

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-4.jpg)

Constructor là một phương thức đặc biệt được sử dụng để khởi tạo giá trị ban đầu cho đối tượng, phương thức này sẽ tự chạy khi tạo một đối tượng. Contructor có tên trùng với tên lớp và không có giá trị trả về.

* Thử sử dụng lớp này:

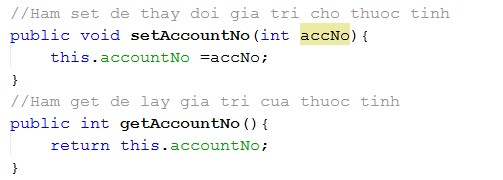
Quay trở lại lớp ATMApp và sử dụng lớp vừa tạo theo đoạn mã dưới đây:

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-5.jpg)

Đoạn mã trên tạo ra đối tượng acc1 của lớp Account và sử dụng chúng để in thông tin ra màn hình. Cho chạy ứng dụng nó vẫn hoạt động tốt. Tuy nhiên, theo tinh thần lập trình hướng đối tượng thì không khuyến khích cách truy cập trực tiếp vào các thuộc tính của lớp như thế này.

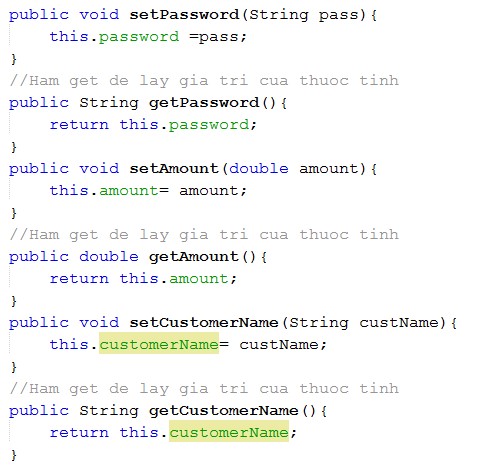
Điều này cũng giống như chúng ta đeo bảng tên mà tất cả các thông tin cá nhân của mình hiên ra trước mắt mọi người và nguy hiểm hơn là người khác có thể thay đổi thông tin này mà không cần hỏi ý kiên của chúng ta. Ví dụ ai đó tự nhiên lại thay đổi tên của bạn mà không hỏi ý kiến bạn có đồng ý hay không thì quá nguy hiểm.

Do vậy, lập trình hướng đối tượng khuyến khích bạn viết private cho các thuộc tính của đối tượng và viết các hàm get() và set() để lấy hoặc thay đổi giá trị của thuộc tính.  Ví dụ đối với thuộc tính accountNo cần 02 phương thức như sau:

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-6.jpg)

Như vậy muốn lấy giá trị thuộc tính của đối tượng phải hỏi qua chủ thể và nếu không thích chủ thể có thể dùng hàm if để từ chối trả về thông tin.

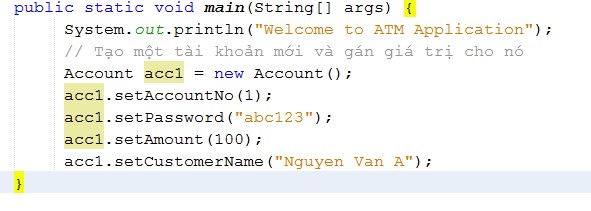
Tiếp tục bổ sung các hàm Get và Set cho các thuộc tính còn lại chúng ta sẽ hoàn thiện lớp Account:

[](http://iviettech.vn/wp-content/uploads/2015/03/Create-Account-Class-7.jpg)

#### Bước 3. Sử dụng lớp Account để xây dựng ứng dụng ATM

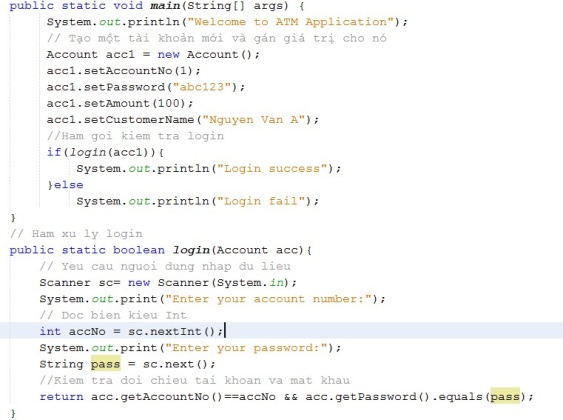
##### **– Tạo ra một tài khoản (đối tượng của lớp Account)**

Trước tiên chúng ta phải tạo ra một tài khoản của khách hàng cần giao tiếp. Trong thực tế chúng ta phải lấy từ CSDL lên(phần này sẽ bàn ở các bài sau) nhưng ở đây để đơn giản chúng ta gán trực tiếp.

[](http://iviettech.vn/wp-content/uploads/2015/03/ATMApp-Class-1.jpg)

Như vậy chúng ta đã có 01 tài khoản là acc1, chúng ta sẽ xây dựng chức năng login để khách hàng sử dụng đăng nhập.

##### **Thủ tục đăng nhập (login)**

[](http://iviettech.vn/wp-content/uploads/2015/03/ATMApp-Class-2.jpg)

Để đoạn mã trên chạy được bạn phải sử dụng dòng lệnh dưới trước lệnh khai báo lớp để ATMApp để sử dụng lớp Scanner.

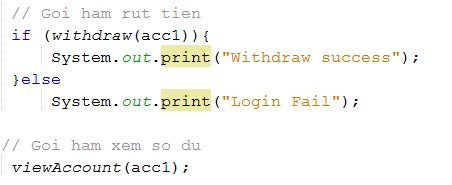
import java.util.Scanner;

Thực hiện đoạn lệnh trên để kiểm tra chức năng Login.

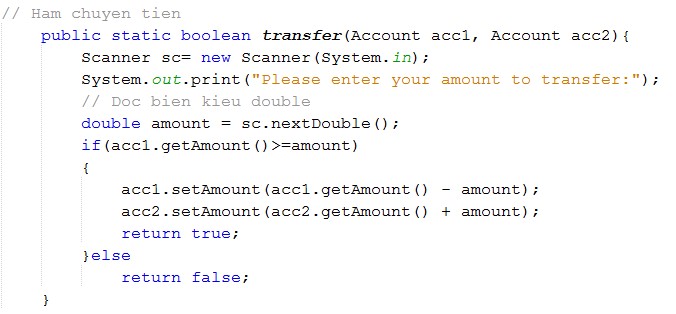
##### -Tiếp theo chúng ta sẽ viết các thủ tục rút tiền, xem số dư tài khoản như sau:

##### **[ATMApp - Class-3](http://iviettech.vn/wp-content/uploads/2015/03/ATMApp-Class-3.jpg)**

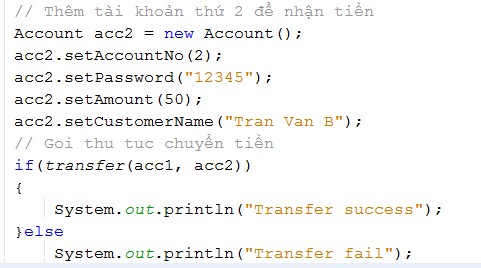
Để gọi 2 hàm trên chúng ta vào hàm Main và thêm đoạn lệnh sau:

[](http://iviettech.vn/wp-content/uploads/2015/03/ATM-Class-Diagram-41.jpg)

Tiếp theo chúng ta sẽ viết hàm chuyển tiền. Hàm chuyển tiền là hàm chuyển từ tài khoản này qua tài khoản khác nên chúng ta phải tạo thêm tài khoản nhận tiền để thực hiện thao tác này và viết hàm chuyển tiền để thực hiện.

[](http://iviettech.vn/wp-content/uploads/2015/03/ATMApp-Class-5.jpg)

Chúng ta sẽ quay trở lại hàm main() và code để thêm tài khoản và sử dụng hàm chuyển tiền ở trên:

[](http://iviettech.vn/wp-content/uploads/2015/03/ATMApp-Class-7.jpg)

Chúng ta thấy việc tạo ra đối tượng thứ 2 của tài khoản rất dễ dàng và chúng ta dễ dàng giao tiếp với nó đã cho thấy sự tiện lợi của Lập trình hướng đối tượng.

Qua ứng dụng trên các bạn có thể thấy việc lập trình theo mô hình hướng đối tượng giúp chúng ta mô tả bài toán sát với cách thực hiện trong mô hình thực tế từ đó giúp bạn dễ hình dung hơn.

Thêm các câu lệnh điều khiển vào hàm main() để hoạt động của ứng dụng trên gần với với hoạt động  thực tế của ATM hơn. Bạn có thể download đoạn mã của cả ứng dụng trên tại đây: [**Download**](http://iviettech.vn/wp-content/uploads/2015/03/atmapp.rar).

Trong phần trên để bạn dễ hiểu tôi đã đưa tất cả các phương thức về lớp ATMApp. Tuy nhiên, điều này làm hạn chế khả năng của các đối tượng sinh ra từ lớp Account cũng như giảm ưu thế về sử dụng lại trong lập trình hướng đối tượng. Để cải tiến cho ứng dụng trên tốt hơn chúng ta cần đưa những phương thức như withdraw, deposit, checkLogin, changePassword vào thành phương thức của lớp Account. Sau đó các phương thức của lớp ATMApp sử dụng lại các phương thức này. Bạn có thể download code sau khi đã cải tiến tại đây: [**Download.**](http://iviettech.vn/wp-content/uploads/2015/03/atmimprove.rar)

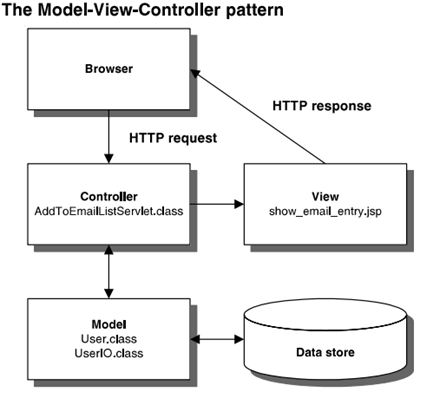
Bài viết này mới là phần ứng dụng cơ bản về lập trình hướng đối tượng trên Java, trong loạt bài tiếp theo Tôi sẽ giới thiệu các bạn cách cài đặt theo MVC và chuyển đổi dữ liệu từ Database lên đối tượng qua ORM(Object Relation Mapping). Mời các bạn đón đọc.

## [CÀI ĐẶT MÔ HÌNH MVC(MODEL – VIEW – CONTROLLER) TRONG JAVA](http://iviettech.vn/blog/4632-cai-dat-mo-hinh-mvcmodel-view-controller-trong-java.html)

**MVC là Design Pattern(mẫu thiết kế) được áp dụng rộng rãi trên hầu hết các ngôn ngữ lập trình hướng đối tượng hiện nay từ Java, C#, PHP, Visual C++ …. Trong bài viết này chúng ta sẽ làm rõ về MVC cũng như cách áp dụng nó từ thiết kế đến cài đặt trên ngôn ngữ lập trình Java. Hy vọng sẽ giúp các bạn rõ hơn về mô hình này.**

## 1. ****MVC là gì?****

Model – View – Controller (MVC) là một mẫu thiết kế nhằm mục tiêu chia tách phần Giao diện và Code để dễ quản lý, phát triển và bảo trì. MVC chia ứng dụng phần mềm ra làm 3 phần có tương tác với nhau là Model(Dữ liệu), View(Giao diện), Controller( Code điều khiển tương tác giữa Model và View cũng như nghiệp vụ (Business)).

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh-1.png)

Hình 1 – Mô hình MVC

**Trong đó:**

* **Model** là lớp chứa thông tin đối tượng (Dữ liệu), tương tác với Database. Chịu trách nhiệm chính trong mô hình hóa đối tượng.
* **View** là giao diện của hệ thống tương tác trực tiếp với người dùng.
* **Controller**nhận yêu cầu từ người dùng và sử dụng Model và View để xử lý và trả kết quả cho người dùng.

Vì sao cần có MVC?

Sự nhập nhằng giữa giao diện và code trong các ngôn ngữ lập trình Scripting như JSP, PHP, ASP làm cho code chương trình khó hiểu và khó bảo trì và khó hình dung trong quá trình cài đặt hệ thống từ góc nhìn thiết kế. Ví dụ: trong thiết kế chúng ta mô hình hóa lớp Account nhưng khi cài đặt lại phải có giao diện theo chức năng và một số qui định về Bussiness nên rất khó hình dung. MVC ra đời với mục tiêu khắc phục những vấn đề đó. Bạn sẽ thấy rõ hơn trong các phần tiếp theo khi chúng ta tiến hành áp dụng MVC để xây dựng ứng dụng.

MVC có thể dùng trên Desktop, tuy nhiên trên mô hình ứng dụng Web thì ưu điểm của nó mới được thể hiện rõ ràng hơn.

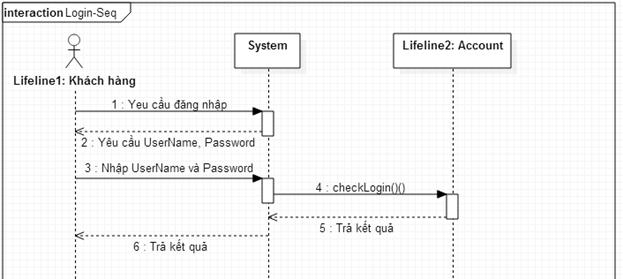
## ****2. Bài toán và phân tích****

Để bạn dễ hiểu MVC và cách sử dụng nó, chúng ta sẽ sử dụng MVC để cài đặt cho chức năng Login của ứng dụng ATM mà chúng ta đã nói ở bài [**Học lập trình hướng đối tượng với Java**](http://iviettech.vn/blog/4247-hoc-lap-trinh-huong-doi-tuong-voi-java.html).

Mô tả chức năng Login như sau:

* Màn hình Login của ATM sẽ hiển thị thông tin yêu cầu số PIN(AccountNo) và Password.
* Hệ thống sẽ nhận thông tin từ người dùng và chuyển cho lớp Account để kiểm tra.
* Nếu thông tin số PIN và Password đúng thì trả về Login thành công, Nếu sai thì trả về Login thất bại.

**Bản vẽ thiết kế Sequence Diagram cho chức năng này như sau:**

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh2.png)

Hình 2- Bản vẽ Sequence Diagram của chức năng Login

Trong đó:

* Lớp **System** đại diện cho các giao diện màn hình.
* Lớp **Account** mô tả tài khoản trong hệ thống.

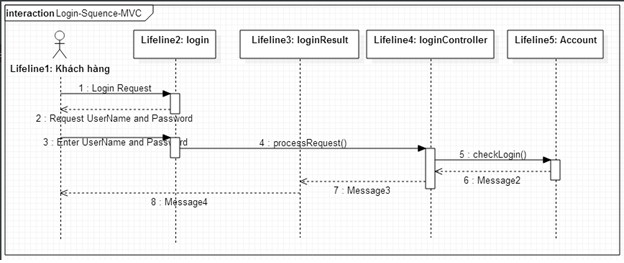
Để hiểu rõ hơn về bản vẽ này bạn đọc thêm bài **Bản vẽ Class Diagram**

Theo mô hình trên thì phần Code business và tương tác giữa 2 lớp bạn sẽ viết trong lớp giao diện hoặc lớp Account nên hầu hết các ngôn ngữ cho bạn viết code xen lẫn với HTML. Chính điều này gây khó hiểu cho người mới học.

Bây giờ chúng ta sẽ áp dụng thiết kế MVC để thiết kế cho chức năng này. Bằng cách tách lớp Giao diện ra thành View và Controller ta sẽ có bản vẽ Sequence Diagram như sau:

Bản vẽ thiết kế Sequence Diagram sau khi áp dụng MVC:

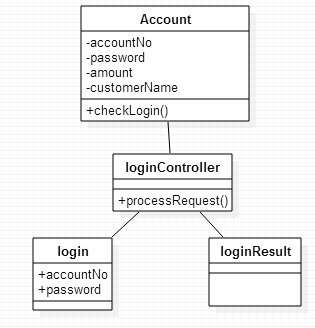
* Lớp login: thể hiện giao diện có thể cài đặt bằng file login.html hoặc login.jsp
* loginResult: cũng là file HTML hoặc JSP
* loginController: là lớp điều khiển cho ứng dụng này. Sử dụng Sevlet.
* Lớp Account: sử dụng Java bình thường.

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh3.png)

Hình 3- Chức năng Login sau khi thiết kế theo MVC

Sơ đồ này cho chúng ta thấy gần với quá trình cài đặt hơn.

**Cập nhật các phương thức và lớp mới vào sơ đồ lớp chúng ta sẽ có:**

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh4.png)

Hình 4- Class Diagram của chức năng MVC

Trong đó:

* Login và loginResult: là các file HTML thuần túy đóng vai trò là View.
* loginController đóng vai trò là Controller viết bằng Servlet
* Account: đóng vai trò là Model viết bằng Java.

Thoạt nhìn bạn thấy khá phức tạp vì nó phát sinh thêm lớp Control và loginResult nhưng thực ra nó sẽ tách phần code Java ra riêng với code HTML sẽ làm cho ứng dụng của chúng ta sáng sủa hơn và gần với quá trình cài đặt ứng dụng hơn.

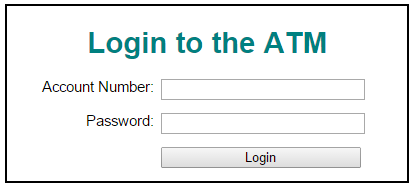
## ****Cài đặt ứng dụng ATM trên Java sử dụng MVC****

Phần trên chúng ta đã phân tích về mặt thiết kế, bây giờ chúng ta sẽ đi cài đặt ứng dụng trên với Java. Hy vọng sẽ giúp các bạn hiểu sau hơn về mô hình này. Để thực hiện được bài này bạn phải biết cơ bản về lập trình Java và Sevlet.

Chúng ta sử dụng sơ đồ lớp ở trên và cài đặt nó trên ngôn ngữ lập trình Java. Trước tiên để cài đặt ứng dụng này máy tính của bạn phải cài đặt Tomcat và NetBean và cấu hình chúng chạy thành công. [Xem cách cài đặt Tomcat và NetBean ở đây](http://congdongjava.com/forum/threads/c%E1%BA%A5u-h%C3%ACnh-tomcat-trong-netbeans.680/).

### ****a. Cài đặt phần View****

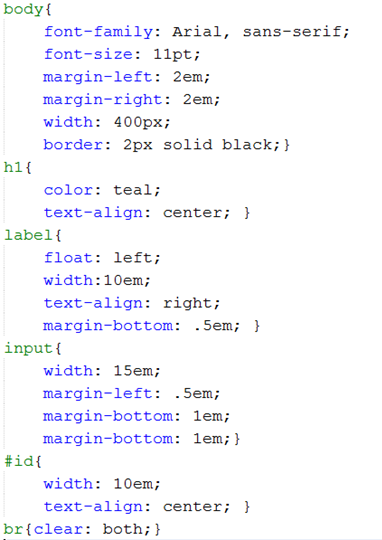
Trước tiên chúng ta sẽ cài đặt giao diện cho ứng dụng. Phần View là trang login.html như sau:

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh5.png)

Hình 5- Trang Login

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh6.png)

Hình 6- Đoạn code HTML của trang Login

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh7.png)

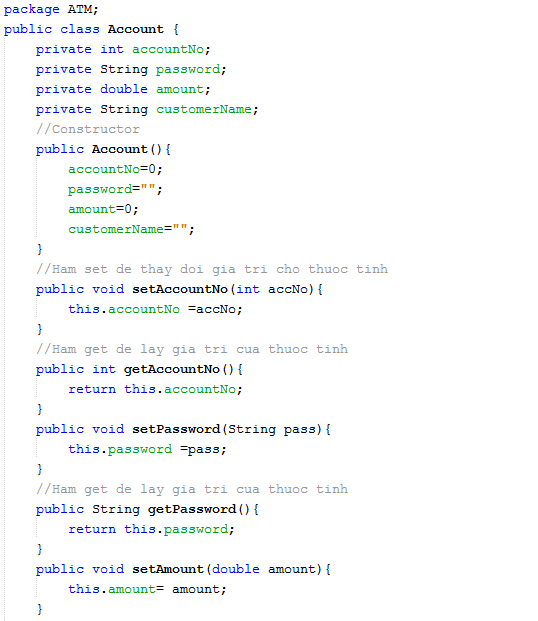
Hình 7- Đoạn code CSS của trang Login

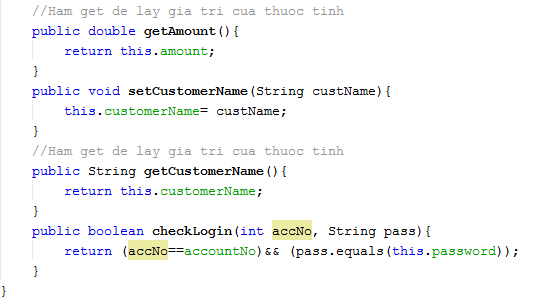
Tất nhiên bạn cần có kiến thức về HTML, CSS để cài đặt đoạn này. Nếu chưa biết về HTML và CSS bạn có thể xem thêm bài [**Giới thiệu về HTML**](http://iviettech.vn/blog/3809-lam-quen-voi-html.html).

### ****b. Cài đặt phần Model****

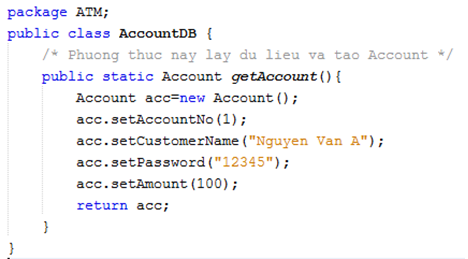
Phần model bao gồm 2 lớp như sau:

* **Account**: là lớp mô tả về tài khoản. Chúng ta đã mô tả kỹ ở bài “Học lập trình hướng đối tượng với Java”.
* **AccountDB**: là lớp bổ sung để kết nối với CSDL. Chúng ta sẽ nhận AccountID do người dùng nhập vào, sau đó vào CSDL để lấy thông tin của ID tương ứng và tạo đối tượng của lớp Account. Phần này gọi là Object Relational Mapping(ORM) sẽ được mô tả kỹ ở bài tiếp theo. Bài này chúng ta tạm thời thiết lập cứng giá trị.

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh8-1.png)

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh8-2.png)

Hình 8- Đoạn code lớp Account

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh9.png)

Hình 9- Đoạn code lớp AccountDB

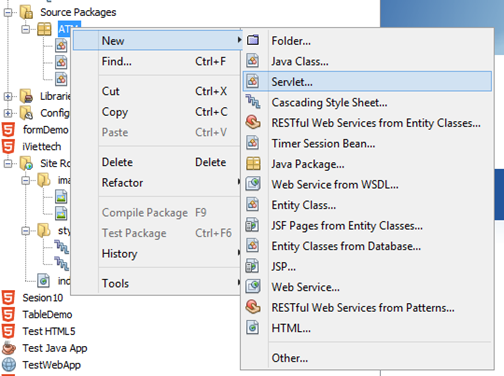
### 

### c. ****Cài đặt phần Controller****

Như đã giới thiệu ở trên Controller có trách nhiệm kết nối phần View và Model cũng như xử lý các vấn đề nghiệp vụ liên quan. Phần này viết bằng Java. Có nhiều cách cài đặt phần này như sử dụng Sevlet hoặc dùng Spring Framework… ở đây chúng ta sử dụng Servlet để cài đặt. Để tạo ra Controller chúng ta thực hiện các bước sau:

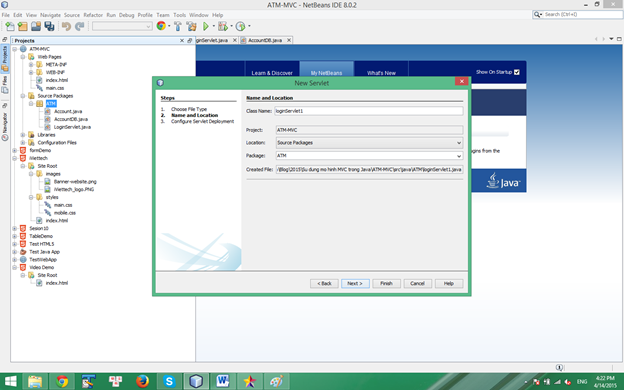
**Bước 1: Tạo Servlet**

Từ cửa sổ Project của NetBean, kích chuột phải -> New -> Sevlet

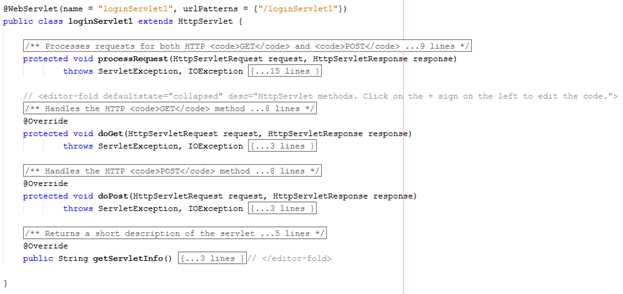
[](http://iviettech.vn/wp-content/uploads/2015/05/hinh10.png)

Hình 10- Tạo Sevlet

Đặt tên Sevlet và kích Finish để tạo một Servlet

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh11.png)

Hình 11- Tạo Sevlet -2

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh12.png)

Hình 12- Sevlet khi mới tạo

Trong đó:

@WebServlet(name = “loginServlet1”, urlPatterns = {“/loginServlet1”})

**WebServlet annotation**: có mục đích cấu hình Sevlet. Khi có annotation này đặt trước class thì trình biên dịch nhận biết đó là một Sevlet class. Bạn có thể dùng cách này hoặc dùng file web.xml để thay thế.

**processRequest method:**đây là phương thức được sử dụng để viết Code cho Sevlet thực hiện chức năng Controller.

**doGet, doPost methods**: là 2 phương thức đáp ứng request của người dùng khi người dùng đặt phương thức get hay post trong file login.html. Hai phương thức này đều gọi lại **processRequest()**nên bạn chỉ cần viết lệnh trong **processRequest()**là đủ. Chi tiết vì sao sử dụng override doGet(), doPost bạn nên tìm đọc một bài về Sevlet để hiểu rõ hơn.

**getServletInfo() method**: nhằm mục đích mô tả thông tin về Sevlet.

**Bước 2: Viết mã lệnh cho Sevlet**

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh13.png)

Hình 13- Code cua Sevlet

Trong đó:

* Lấy Account Number và Password từ trang Login gửi sang thông qua HTTP Request. Mỗi khi người dùng kích nào nút Submit trên form HTML thì các thông tin từ Form sẽ được gói gửi kèm đến trang xử lý được chỉ định trong phần action của form.
* Gọi phương thức getAccount() từ lớp AccountDB để tạo ra đối tượng tương ứng với AccountID đề nghị đăng nhập.
* Gọi kiểm tra lấy kết quả và xây dựng trang kết quả trả về. Nếu bạn muốn trả về một trang có sẵn thì sử dụng phương thức .forward() của getSevletContext() để trả về kết quả cho trang đó.

**Bước 3: Cấu hình ứng dụng để chạy**

Để ứng dụng chạy được, bạn phải khai báo đúng urlPatterns vào thuộc tính action của form login. Lúc đó hệ thống mới chạy được.

[hinh13-2](http://iviettech.vn/wp-content/uploads/2015/05/hinh13-2.png)

@WebServlet(name = “loginServlet1”, urlPatterns = {“/**LoginServlet**“})

Phần bôi đậm và bôi đen này phải trùng nhau.

Trong trường hợp bạn đã có sẵn trang kết quả như LoginResult.jsp hoặc Thanks.jsp bạn có thể sửa lại đoạn code trên processRequest như sau:

[](http://iviettech.vn/wp-content/uploads/2015/05/hinh14.png)

Hình 14- Đoạn Code của Sevlet khi trang kết quả đã có sẵn tên là thanks.jsp

Sau đó bạn dùng jsp để truy cập dữ liệu từ lớp account chuyển sang để hiển thị kết quả.

Như vậy chúng ta đã xây dựng được một ứng dụng nhỏ về MVC trên Java. Bạn có thể lấy đoạn mã lệnh của ứng dụng ở đây để tham khảo thêm. [**Download**](http://iviettech.vn/wp-content/uploads/2015/05/ATM-MVC.rar).