[*Mô hình MVC?*](https://pikachu06.wordpress.com/2015/06/15/mo-hinh-mvc/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/mo-hinh-mvc/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/mo-hinh-mvc/#respond)

1.1. Model (Mô hình).  
– Mô hình gồm các lớp java có nhiệm vụ:  
· Biểu diễn data và cho phép truy cập tới để get và set data trong (JAVABEAN), Thường thì phần layer này mô phỏng 1 cách đầy đủ đối tượng từ thế giới thực.  
· Nhận các yêu cầu từ khung nhìn  
· Thi hành các yêu cầu đó (tính toán, kết nối CSDL …)  
· Trả về các giá trị tính toán theo yêu cầu của Controller

1.2. View (khung nhìn) :

· Bao gồm các mã tương tự như JSP, HTML, CSS, XML, Javascript, JSON… để hiển thị giao diện người dùng, các dữ liệu trả về từ Model thông qua Controller…

1.3. Controller (Bộ điều khiển) :

· Đồng bộ hoá giữa Khung nhìn và Mô hình. Tức là với một trang JSP này thì sẽ tương ứng với lớp java nào để xử lý nó và ngược lại, kết quả sẽ trả về trang jsp nào.Nó đóng vai trò điều tiết giữa View và Model.

· Như vậy, chúng ta có thể tách biệt được các mã java ra khỏi mã html. Do vậy, nó đã giải quyết được các khó khăn đã nêu ra trong Mô hình 1. Người thiết kế giao diện và người lập trình java có thể mang tính chất độc lập tương đối. Việc debug hay bảo trì sẽ dễ dàng hơn, việc thay đổi các theme của trang web cũng dễ dàng hơn …

**MVC là gì ?**

Mô hình MVC (Model – View – Controller) là một kiến trúc phần mềm hay mô hình thiết kế được sử dụng trong kỹ thuật phần mềm. Nó giúp cho các developer tách ứng dụng của họ ra 3 thành phần khác nhau Model, View và Controller. Mỗi thành phần có một nhiệm vụ riêng biệt và độc lập với các thành phần khác

**Model**

Đây là thành phần chứa tất cả các nghiệp vụ logic, phương thức xử lý, truy xuất database, đối tượng mô tả dữ liệu như các Class, hàm xử lý…

**View**

Đảm nhận việc hiển thị thông tin, tương tác với người dùng, nơi chứa tất cả các đối tượng GUI như textbox, images…Hiểu một cách đơn giản, nó là tập hợp các form hoặc các file HTML.

**Controller**

Giữ nhiệm vụ nhận điều hướng các yêu cầu từ người dùng và gọi đúng những phương thức xử lý chúng… Chẳng hạn thành phần này sẽ nhận request từ url và form để thao tác trực tiếp với Model.

Advertisements

REPORT THIS AD

REPORT THIS AD

[*Câu hỏi phỏng vấn Java Core (Phần 5)*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-java-core-phan-5/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-java-core-phan-5/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-java-core-phan-5/#respond)

**1. Kể các collection biết. Phân biệt list và linklist. Add dãy 10 số từ 0 -10 thì nên dùng cái nào?**  
– Collection, Set, List, Map, ArrayList, Vector, Hashtable, Hashset, HashMap  
\*Phân biệt list và linkedlist, array, arrraylist:  
  
*Array*  
– Ưu điểm:  
+ Dễ cài đặt & truy cập đến các phần tử dữ liêu.  
+ Tốc độ truy cập đến 1 vị trí bất kỳ trên mảng nhanh & hiệu quả.

– Nhược điểm:  
+ Cần phải xác định số phần tử mảng trước khi sử dụng(kể cả mạng động)-> ko phù hợp với bài toán chưa biết số lượng các phần tử.  
+ Khó khăn trong khi chèn or xoá một phần tử bất kỳ. Nếu việc chèn & xoá diễn ra nhiều -> xử lý chậm.

*Linkedlist*  
– Ưu điểm:  
+ Do DSLK dùng tới đâu cấp phát tới đó -> phù hợp với bài toán chưa xđ số phần tử cho trước.  
+ Dễ dàng trong việc xoá, chèn các phần tử trong DS(thay đổi lại địa chỉ, quan hệ của phần tử).

– Nhược điểm:  
+ Khó cài đặt và truy cập đến phần tử trong danh sách.  
+ Tốc độ truy cập đến phần tử trong DS chậm( người ta thường dùng cây quyết định để phục vụ cho việc tìm kiếm)

*ArrayList:*  
– truy cập trực tiếp các phần tử qua set và get methods  
– có thể thêm, tìm kiếm đối tượng dể dàng hơn array

**2. Phân biệt Vector và ArrayList?**  
Vector có cơ chế đồng bộ hóa còn ArrayList thì khôngà ArrayList nhanh hơn vector

**3. Arraylist hơn array như thế nào?**  
Năng động, cung cấp cơ chế chèn và tìm kiếm mạnh mẽ hơn array

**4. an Iterator là gì?**  
Về cơ bản,bộ duyệt Iterator giúp ta duyệt các phần tử trong 1 cái danh sách, ví dụ mình có cái danh sách các tên:  
– khi khởi tạo Iterator, sẽ có 1 con trỏ được khởi tạo  
– hàm hasNext để kiểm tra xem mình còn duyệt được tiếp nữa không  
– hàm next trả ra phần tử kế tiếp của bộ duyệt

Mình hoàn toàn có thể làm một class duyệt theo ý mình, ví dụ như thêm vào đó 2 biến là vị trí bắt đầu và chiều duyệt thì hoàn toàn có thể làm được duyệt nhiều kiểu

**5. Các câu hỏi khác**

+ Subclass: class kế thừa tất cá các thuộc tính public và phương thức protected  
+ Break, continute:  
Break: cham dứt hẳn  
Continute: chấm dứt câu lệnh và quay trở lại vòng lặp

Q: Can you call one constructor from another if a class has multiple constructors?  
A: Yes, use this() syntax.

Q: What is the default value of an object reference declared as an instance variable?  
A: Null, unless it is defined explicitly.

Q: Is it necessary that each try block must be followed by a catch block?  
A: It is not necessary that each try block must be followed by a catch block. It should be followed by either a catch block or a finally block.

Q: a private constructor?  
A: ngăn chặn subclass và ko muốn lớp khác khởi tạo đối tượng

Q: What is difference between Path and Classpath?  
A: Path and Classpath are operating system level environment variales. Path is defines where the system can find the executables(.exe) files and classpath is used to specify the location of .class files.

Q: Can an Interface extend another Interface?  
A: Yes an Interface can inherit another Interface, for that matter an Interface can extend more than one Interface

Q: Em hay sử dụng những loại Collections nào và một chút đặc điểm:  
A: Có 4 loại là Lists (lưu một danh sách các đối tượng), Sets (một danh sách các đối tượng không trùng nhau), Maps (một danh sách các đối tượng có ID), Queues (một danh sách các đối tượng được xắp sếp theo thứ tự). Mỗi loại này có nhiều lớp con.  
Dùng gì để truy cập CSDL và thiết kế như thế nào: JDBC, Hibernate… sau đó sử dụng DAO pattern (CRUD: creat, read, insert, update, delete)  
– Mô hình n-tier: view, bussiness logic, data  
– Một vài loại pattern thông dụng: singleton, factory, proxy, adapter, observer …  
– Một vài loại frameworks: Struts: MVC+ Front Controller pattern. Hibernate: object relational mapping.

Q: các tính chất của OOP  
A: có 4 tính chất là: Tính trừu tượng, tính đa hình, tính thừa kế và tính đóng gói

Q: sự khác nhau giữa JSP và Servlet  
A: JSP khi biên dịch thì sẽ convert sang code java, Servlet là thuần code java nên servlet sẽ nhanh hơn. Hơn nữa trong mô hình MVC thì servlet sẽ đóng vai trò controller và trả về một view JSP. Trên serlvet mà trả về code HMTL thì sẽ rất là nhàm chán nhưng JSP thì sẽ dễ hơn nhiều

Q: Khác nhau giữa equals và == trong java  
A: equals là để so sánh giá trị giữa 2 object còn == là để so sánh reference của 2 object

Q: sư khác nhau khi anh lấy 1 phần từ từ 1 list ra và lấy phần tử tứ set ra như thế nào  
A: lấy phần tử từ một list ra sẽ trả về 1 reference của object trong list đó, còn set thì sẽ nhận được giá trị clone (copy) của nó

A: có 2 loại là Runtime Exception (unchecked) là những ngoại lệ xảy ra trong lúc thực thi và checked exception là những ngoại lệ xảy ra trong lúc biên dịch

**6. Kể các kinh nghiệm về Java**

**7. Jad là gì? build file ở đâu?**

**8. HashMap and Map?**

**9. Phân biệt HashMap và HashTable**

**10. Phân biệt a constructor và a method?**

[*Câu hỏi phỏng vấn Java Core (Phần 4)*](https://pikachu06.wordpress.com/2015/06/15/cau-7-try-catch-finally-java-gap-ngoai-le-trong-try-thi-tim-den-catch-ngoai-le-hop-voi-catch-nao-thi-lam-theo-khoi-lenh-trong-catch-do-va-bo-qua-cac-catch-con-lai-khac-khoi-lenh-trong-finally/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-7-try-catch-finally-java-gap-ngoai-le-trong-try-thi-tim-den-catch-ngoai-le-hop-voi-catch-nao-thi-lam-theo-khoi-lenh-trong-catch-do-va-bo-qua-cac-catch-con-lai-khac-khoi-lenh-trong-finally/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-7-try-catch-finally-java-gap-ngoai-le-trong-try-thi-tim-den-catch-ngoai-le-hop-voi-catch-nao-thi-lam-theo-khoi-lenh-trong-catch-do-va-bo-qua-cac-catch-con-lai-khac-khoi-lenh-trong-finally/#respond)

*Tags*

[*java*](https://pikachu06.wordpress.com/tag/java/)

**Câu 11: Kiểu nguyên thủy và lớp wrapper(loại nào tham biến, loại nào tham trị)**

với những kiêu nguyên thủy của Java (int, boolean..) chỉ có thể được truyền theo kiểu tham trị, tức là giá trị của x vẫn giữ nguyên sau khi gọi hàm.  
Các đối tượng như String thì có thể truyền tham biến  
  
**Câu 12. Kiểu dữ liệu tham chiếu:là các kiểu dữ liệu đối tượng**.

Trong java có 3 kiểu dữ liệu tham chiếu  
– Mảng(Array): Tập hợp các dữ liệu cùng loại  
– Lớp(Class): Tập hợp các biến và phương thức  
– Giao diện(Interface): Là 1 lớp trừu tượng được tạo ra để bổ sung cho các kế thừa đa lớp trong java

**Câu 13. Các khai báo quyền truy xuất và kế thừa của Lớp**

-public:được phép truy xuất từ bất cứ nơi nào và bất cứ lớp nào cũng được quyền kế thừa  
-protected:chỉ có phương thức cùng gói được phép truy xuất và kế thừa  
-private:chỉ có phương thức cùng gói được phép truy xuất nhưng không lớp nào được phép kế thừa  
-nếu không khai báo,mặc định là protected

**Câu 14. Ý nghĩa của từng chỉ định thuộc tính**

Có 6 chỉ định thuộc tính là static, abstract, final, native, synchronized (đồng bộ) và volatile (linh hoạt)  
Nếu là một phương thức không tĩnh, đầu tiên bạn phải khởi tạo một đối tượng,sau đó mới được phép gọi phương thức  
Nếu là một phương thức tĩnh,bạn được phép gọi trực tiếp từ lớp  
Một phương thức trừu tượng không có nội dung.Nội dung của nó sẽ được các lớp con tùy biến và phát triển theo hướng của riêng nó.  
– final: không thể được extends hay override (ghi đè)  
– native: thân phương thức viết bằng C hay C++  
– synchronized: chỉ cho phép 1 thread truy cập vào khối mã ở cùng một thời điểm  
– volatile: sử dụng với biến để thông báo rằng giá trị của biến có thể được thay đổi vài lần vì vậy không ghi vào thanh ghi  
.Từ thứ 3 là giá trị trả về.Nếu không có giá trị trả về thì là void

**Câu 15. Interface, abstract class**

What is an Abstract Class?  
Lớp trừu tượng đơn giản được xem như một class cha cho tất cả các Class có cùng bản chất. Do đó mỗi lớp dẫn xuất (lớp con) chỉ có thể kế thừa từ một lớp trừu tượng. Bên cạnh đó nó không cho phép tạo instance, nghĩa là sẽ không thể tạo được các đối tượng thuộc lớp đó.

What is an Interface?  
Lớp này được xem như một mặt nạ cho tất cả các Class cùng cách thức hoạt động nhưng có thể khác nhau về bản chất. Từ đó lớp dẫn xuất có thể kế thừa từ nhiều lớp Interface để bổ sung đầy đủ cách thức hoạt động của mình (đa kế thừa – Multiple inheritance).  
Ví dụ:  
– Abstract class ConVat có các lớp con Chim, Ca.  
– Abstract class MayMoc có các lớp con MayBay, Thuyen  
– Interface: iBay, iBoi, iChay.  
=> MayBay, Chim sẽ có cùng Interface là iBay. Rõ ràng mặc dù MayBay, Chim có cùng cách thức hoạt động là bay nhưng chúng khác nhau về bản chất.  
=> MayBay cũng có interface là iChay nhưng Chim không thể nào kế thừa thêm abstract classMayMoc

Giao diện được khai báo giống như 1 lớp, cũng có state và behavior. Nhưng state của giao diện là final còn behavior là abstract

\* Bạn chỉ có thể thừa kế (extend) từ một class và chỉ có thể hiện thực (implement) các chức năng (interface) cho class của mình.

**Câu 16. Phân biệt abstract class với class (như trên)**

**Câu 17. Biến của interface khai báo thế nào?**

static – because Interface cannot have any instance. and final – because we do not need to change it

**Câu 18. phân biệt final và static**

Từ khóa final:  
Nếu lớp A là final thì lớp A sẽ không được kế thừa bởi bất kỳ lớp nào.  
Thuộc tính final: có giá trị không đổi

Từ khóa static:  
Truy cập: Không cần tạo đối tƣợng  
Method: Không sử dụng từ khóa this, Gọi mà không cần khởi tạo đối tƣợng

**Câu 19. static và non-static**

Biến static chỉ là một biến không phụ thuộc vào một đối tượng nào cả, tức là mình có thể truy vấn trực tiếp mà không cần phải khời tạo đối tượng.

VD: Mình có một class tên là A trong class A có một biến static tên là abc=10;  
Khi mình gọi biến này mình bằng cách: A obj = new A(); và truy vấn đến biến đó bằng cách thông thường là obj.abc. Điều đó là dư thừa, mà thay vì vậy mình có thể truy vấn một cách trực tiếp mà không cần khởi tạo đối tượng như mình đã đề cập, bạn có thể truy vấn trực tiếp tới biến đó bằng cách A.abc.

Tương tự cho hàm static, mình cũng chỉ cần gọi <tên lớp>.<tên hàm>.  
Vậy cũng có thể dễ dàng đoán được rằng khi chạy ứng dụng thì thằng static đã được tạo ra mà không cần phải khởi tạo

**Câu 20: Phân biệt biến class và đối tượng**

[*Câu hỏi tuyển dụng Java Core (Phần 3)*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-3/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-3/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-3/#respond)

**Câu 7. Try-catch-finally (Java)**

– Gặp ngoại lệ trong try thì tìm đến catch, ngoại lệ hợp với catch nào thì làm theo khối lệnh trong catch đó và bỏ qua các catch còn lại khác.  
– Khối lệnh trong finally luôn được thực hiện. (Nếu không có catch mà gặp ngoại lệ thì chỉ lệnh đầu tiên trong khối finally được thực hiện, sau đó java sẽ báo lỗi ngoại lệ!)

**Câu 8. Kế thừa, đa hình, trừu tượng, đóng gói**,  
– Đóng gói: là tiến trình che giấu việc thực thi những chi tiết của một đối tượng đối với người sử dụng đối tượng ấy  
– Đa hình:  
+ Nạp chồng (overload): là hiện tượng các phương thức cùng tên, khác tham số  
+ Ghi đè(override): phương thức có mặt ở lớp cha, được xác định là phương thức chung cho các lớp con, rồi xuất hiện ở các lớp con

**Câu 9. Mục đích thu gom rác thải trong Java là gì, và khi nào nó được sử dụng**  
Làm sạch những đối tượng không còn tham chiếu bởi bất kỳ chương trình nào để giải phóng bộ nhớ

**Câu 10. phân biệt jdk, jre, file javax.exe là gì**  
Build = javac thì đó là biên dịch ra bytecode nên javac là trình biên dịch. Khi chạy java = java.exe thì thông dịch ra mã máy, đó là trình thông dịch

Java Development Kit là một bộ bao gồm các công cụ cần thiết trong quá trình phát triển các ứng dụng Java + JRE. Các công cụ đó là: javac.exe, java.exe, applet viewer,… vai trò của từng công cụ như sau:  
· javac.exe – giúp biên dịch các file nguồn viết bằng ngôn ngữ Java (\*.java) thành mã máy (byte code)  
· java.exe – gọi tới JRE, tải nạp các file mã máy đã được biên dịch và gọi tới method main để thực thi các mã đó.  
· Applet Viewer – để chạy các ứng dụng viết bằng applet.

JRE là một ứng dụng nền giúp thực thi các file mã máy đã được biên dịch từ file nguồn \*.java. Các thành phần của JRE chỉ bao gồm các gói Java và thư viện thực thi ứng dụng (runtime libraries) nên JRE không có khả năng biên dịch file Java thành mã máy chỉ có khả năng thực thi các file byte code sau khi đã được SDK biên dịch

[*Câu hỏi tuyển dụng Java Core (Phần 2)*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-2/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-2/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-2/#respond)

**Câu 4: Regular Expressions là gì? Pattern và Matcher ?**

**Regular Expression** hay còn gọi là biểu thức chính quy dùng để chỉ cách thức chúng ta nhận biết, kiểm tra một chuỗi (biểu thức) có đúng như định dạng chúng ta mong muốn hay không.

Trong java hiện nay có 2 lớp là **Pattern và Matcher** (java.util.regex) giúp ta xử lý các vấn đến liên quan đến regular expression.  
+ **Pattern** là lớp dùng để biên dịch biểu thức, kết quả biên dịch để tạo ra đối tượng thuộc lớp  **Matcher** để kiểm tra chuổi ký tự có đúng biểu thức đó  
+ **Matcher** là lớp được tạo ra từ Pattern và dùng để thực hiện những việc như match, tìm kiếm.

**Câu 5: Có bao nhiu loại exception. Exception kế thừa từ đối tượng nào?**  
Có 3 loại : Checked, Unchecked, Error. Lớp Throwable là lớp cha của tất cả các ngoại lệ và lỗi trong ngôn ngữ Java  
Có hai loại ngoại lệ chính trong ngôn ngữ lập trình Java:

**Ngoại lệ có kiểm soát (Checked Exception)**: java.lang.Exception  
Ngoại lệ có kiểm soát được tạo ra trong các tình huống thực hiện bình thường của một chương trình. Một số ví dụ về ngoại lệ có kiểm soát là – yêu cầu một tập tin không có, người dùng nhập liệu sai, và mạng hỏng. Những ngoại lệ này được xử lý để tránh các lỗi biên dịch. Nếu một ngoại lệ xảy ra trong quá trình thực hiện phương thức, phương thức có thể điều quản ngoại lệ hoặc chuyển ngoại lệ cho phương thức đang gọi để chỉ rõ vấn đề xảy ra. Phương thức đang gọi lại có thể điều quản ngoại lệ hoặc chuyển cho phương thức đang gọi nó. Quá trình này có thể tiếp tục đến khi ngoại lệ đạt đến đỉnh của luồng (thread) và luồng bị kết thúc thực hiện. Kỹ thuật này gọi là Call-stack. Ưu điểm chính của kỹ thuật là nhà phát triển có thể đặt các đoạn mã xử lý lỗi ở bất cứ vị trí nào họ muốn.

**Ngoại lệ không kiểm soát (Unchecked Exception)**: java.lang.RuntimeException  
Ngoại lệ không kiểm soát được tạo ra trong các tình huống được xem là không thể phục hồi đối với chương trình. Ví dụ thông thường về các tình huống là hành động truy nhập một phần tử nằm ngoài mảng. Một ứng dụng không được yêu cầu kiểm soát các loại ngoại lệ kiểu này. Ngoại lệ thực thi (runtime exception) cũng là ví dụ của ngoại lệ không kiểm soát. Thường thì chúng phát sinh do các lỗi logic (logical bugs). Ngoại lệ không kiểm soát phát sinh do các vấn đề môi trường hoặc các lỗi không thể được phục hồi và được gọi là Error (lỗi). Sử dụng hết bộ nhớ được cấp phát của chương trình là ví dụ về lỗi (error). Khi bạn gọi một phương thức mà phương thức đó quăng ra ngoại lệ thuộc loại **Checked Exception** thì bạn phải xử lý lỗi bằng cách sử dụng try{…}catch{} hoặc quăng ra ngoài phương thức chứa đoạn mã(throws Exception). Ngược lại nếu đó là Unchecked Exception (tương tự các ngôn ngữ khác như C#, PHP, JavaScript…), thì bạn có thể không cần phải xử lý lỗi này mà chương trình vẫn được dịch tốt (tất nhiên lỗi sẽ xảy ra lúc chạy nếu có).  
Xét về mặt phân cấp thừa kế thì Unchecked Exception được dẫn suất từ **RuntimeException**

**Câu 6: Throw và throws**

Throw: “quăng ra”, có thể “quăng ” ở bất kỳ dòng nào trong phương thức.  
Throws: “Ném đi”, chỉ có phương thức mới sử dụng throws, ném đi cho thằng khác xử lý  
void test(String txt) throws NumberFormatException, Exception{  
Integer.parseInt(txt);//NumberFormatException quăng ra ở đây nếu txt không phải là số  
}  
phương thức nào gọi phương thức test() phải try catch xử lý cả 2 NumberFormatException, Exception vì phương thức test() throws đi

[*Câu hỏi tuyển dụng Java Core (Phần 1)*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-1/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-1/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-tuyen-dung-java-core-phan-1/#respond)

**Câu 1: Các đặc điểm của java?**

*Thông dịch***:**  
Java là một ngôn ngữ lập trình vừa biên dịch vừa thông dịch. Chương trình nguồn viết bằng ngôn ngữ lập trình Java có đuôi \*.java đầu tiên được biên dịch thành tập tin có đuôi \*.class và sau  
đó sẽ được trình thông dịch thông dịch thành mã máy

*Độc lập nền***:**  
Một chương trình viết bằng ngôn ngữ Java có thể chạy trên nhiều máy tính có hệ điều hành khác nhau (Windows, Unix, Linux, …) với điều kiện ở đó có cài đặt máy ảo java (Java Virtual Machine).

*Hướng đối tượng***:**  
Hướng đối tượng trong Java tương tự như C++ nhưng Java là một ngôn ngữ lập trình hướng đối tượng hoàn toàn. Tất cả mọi thứ đề cập đến trong Java đều liên quan đến các đối tượng được định nghĩa trước, thậm chí hàm chính của một chương trình viết bằng Java (đó là hàm main) cũng phải đặt bên trong một lớp. Hướng đối tượng trong Java không có tính đa kế thừa (multi inheritance) như trong C++ mà thay vào đó Java đưa ra khái niệm interface để hỗ trợ tính đa kế thừa.

*Đa nhiệm – đa luồng (MultiTasking – Multithreading)***:**  
Java hỗ trợ lập trình đa nhiệm, đa luồng cho phép nhiều tiến trình, tiểu trình có thể chạy song song cùng một thời điểm và tương tác với nhau

*Khả chuyển (portable)***:**  
Chương trình ứng dụng viết bằng ngôn ngữ Java chỉ cần chạy được trên máy ảo Java là có thể chạy được trên bất kỳ máy tính, hệ điều hành nào có máy ảo Java. “Viết một lần, chạy mọi nơi” (Write Once, Run Anywhere).

**Câu 2: Các loại biến có thể có của class?**

*Biến local*: Biến sẽ được khai báo và khởi tạo bên trong hàm. Bị hủy khi hàm kết thúc

*Instance biến*:  biến bên trong class nhưng ngoài phương thức. Biến dduoc khởi tạo khi  class được load

*Class biến:*  khai báo bên trong class, bên ngoài phương thức với keyword static

**Câu 3: Phân biệt string với stringbuffer/stringbuilder**

Khác biệt đáng kể nhất giữa String và StringBuffer\StringBuilder trong Java đó là đối tượng String là không thể thay đổi (**immutable** – không thể thay đổi giá trị) trong khi đó các đối tượng StringBuffer\StringBuilder lại có thể thay đổi (**mutable –**có thể thay đổi giá trị).

Việc thay đổi giá trị của đối tượng String được thực hiện bằng cách tạo một đối tượng mới có giá trị mà bạn muốn đổi sang.Vì thế,  việc thay đổi giá trị đối với một đối tượng String sẽ dẫn đến phát sinh hàng loạt các đối tượng String mới trong bộ nhớ, càng tạo ra nhiều đối tượng tạm thì bộ thu dọn rác (Garbage Collection) càng phải làm việc nhiều hơn để xây dựng lại bộ nhớ

StringBuilder và StringBuffer về cơ bản là giống nhau, chúng đều có các phương thức giống nhau để làm việc với chuỗi. Sự khác biệt giữa StringBuilder và StringBuffer nằm ở chổ:  
+ StringBuffer : thuộc loại**synchronized** do đó các phương thức của nó đều là “thread safe” (thích hợp với xử lý đa luồng – multi thread)  
+ StringBuilder thì ngược lại, không **synchronized.**  
Với đặc tính “thread safe”, các phương thức của StringBuffer sẽ chạy chậm hơn so với StringBuilder

[*Câu hỏi phỏng vấn SQL*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-sql/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-sql/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-sql/#respond)

**1. Phân biệt inner join với left join**

*INNER JOIN*: Kết quả trả về chỉ gồm những bản ghi thỏa mãn điều kiện nối hai bảng  
*LEFT JOIN*: select tất cả bản ghi có trong bảng thứ nhất trong mệnh đề FROM, bất kể nó có match với bảng thứ hai hay không. Những bản ghi không match với bảng thứ hai nó sẽ để giá trị NULL  
  
*RIGHT JOIN*:select tất cả những bản ghi có trong bảng thứ hai trong mệnh đề FROM, bất kể những bản ghi đó có match với bảng thứ nhất hay không

*FULL OUTER JOIN*: là sự kết hợp của LEFT OUTER JOIN và RIGHT OUTER JOIN. Kết quả trả về tất cả những bản ghi match hoặc không match giữa hai bảng

*CROSS JOIN*  
Cái này chính là tích Đề các. Mỗi bản ghi trong bảng thứ nhất của mệnh đề FROM sẽ kết hợp (vô điều kiện) với các bản ghi trong bảng thứ ha

*EQUI JOIN*: chính là INNER JOIN cộng thêm điều kiện: nó select toàn bộ các trường có trong hai bảng tham gia nối.

*NATURAL JOIN*:giống EQUI JOIN cộng thêm điều kiện: nếu hai bảng có hai trường giống nhau thì nó chỉ lấy ra 1 trường.

**2. index trong sql**  
Index (chỉ mục) thực chất là một dạng tương tự như phần mục lục của 1 cuốn sách  
Trong SQL, nó đóng vai trò tăng tốc truy xuất  
index trong CSDL có hai loại: Clustered Index và Non-Clustered Index  
Clustered index: thường được tự tạo ra khi bảng có primary key, clustered index chính là unique index  
Non-Clustered Index: foreign key là non-clustered index

Tham khảo:  
<http://sinhvienit.net/forum/clustered-index-va-non-clustered-index.41251.html>  
<http://www.sqlviet.com/blog/index-giup-tang-hieu-nang-thuc-hien-nhu-the-nao>

**3. Phân biệt stored procedure ,function và trigger,view**

1. Khung nhìn (view) có thể được xem như là một bảng “ảo” trong cơ sở dữ liệu có nội dung được định nghĩa thông qua một truy vấn (câu lệnh SELECT). Như vậy, một khung nhìn trông giống như một bảng với dữ liệu quan sát được là một tập bao gồm các dòng và các cột. Điểm khác biệt giữa khung nhìn và bảng là: khung nhìn không được xem là một cấu trúc lưu trữ dữ liệu tồn tại trong cơ sở dữ liệu. Thực chất dữ liệu quan sát được trong khung nhìn được lấy từ các bảng thông qua câu lệnh truy vấn dữ liệu.

2. Một thủ tục là một đối tượng trong cơ sở dữ liệu bao gồm một tập nhiều câu lệnh SQL được nhóm lại với nhau thành một nhóm (như tập tin BAT trong DOS vậy). Khi một thủ tục lưu trữ đã được định nghĩa, nó có thể được gọi thông qua tên thủ tục, nhận các tham số truyền vào, thực thi các câu lệnh SQL bên trong thủ tục và có thể trả về các giá trị sau khi thực hiện xong.  
3. Một trigger là một đối tượng với nội dung là một tập các câu lệnh SQL (tương tự thủ tục). Mỗi một trigger gắn liền với một bảng nào đó trong CSDL và được tự động kích hoạt khi xảy ra những giao tác làm thay đổi dữ liệu trong bảng (INSERT, UPDATE, DELETE).

Điểm khác biệt giữa thủ tục lưu trữ và trigger là: các thủ tục lưu trữ được thực thi khi người sử dụng có lời gọi đến chúng còn các trigger lại được “gọi” tự động khi xảy ra những giao tác làm thay đổi dữ liệu trong các bảng

4. Function: giống như produre nhưng có kết quả trả về

**4. Phân biệt JOIN and UNION**

A Join: cho phép kết nối dữ liệu liên quan trong các bảng khác nhau  
A Union: hợp dữ liệu của 2 bảng mà có các cột loại data giống nhau

**5. Phân biệt UNION and UNION ALL**  
UNION và UNION ALL đều dùng để hợp hai tập bản ghi cùng cấu trúc, nhưng giữa hai mệnh đề có một khác biệt khá tinh tế: UNION loại bỏ các bản ghi trùng lặp trước khi trả lại kết quả, còn UNION ALL giữ lại tất cả các bản ghi từ hai tập ban đầu

**6. WHERE clause and HAVING clause**

where là câu lệnh dùng để đặt điều kiện lọc trên từng bộ.  
Having cũng là câu lệnh đặt điều kiện nhưng là ở trên 1 nhóm xác định. thường là đi kèm với câu lệnh group by

**7. How to select first 5 records from a table?**

SELECT TOP 5 \* FROM EMP

**8. What is difference between DELETE and TRUNCATE commands?**

DELETE luôn luôn chậm hơn TRUNCATE. Càng có nhiều bản ghi DELETE càng chậm, còn TRUNCATE thì không phụ thuộc vào lượng dữ liệu. DELETE có phạm vi ứng dụng rộng hơn; còn TRUNCATE chỉ dùng được mỗi một việc, nhưng nó lại làm rất nhanh

Tham khảo: <http://www.sqlviet.com/blog/delete-va-truncate>

\*Viết vài câu SQL từ đơn giản tới phức tạp

[*Câu hỏi phỏng vấn AJAX, Jquery*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-ajax-jquery/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-ajax-jquery/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-ajax-jquery/#respond)

**1. Ajax (lợi và hại)**  
– Ưu điểm:  
\* Nó giúp việc thiết kế web đa dạng hơn và tăng tính tương tác của website với người dùng  
\* Nó sử dụng các công nghệ đã có sẵn nên dễ học và sử dụng  
\* Nhờ tính phổ biến của nó, đã khuyến khích việc phát triển các khuôn mẫu mà sẽ giúp lập trình viên tránh khỏi các vết xe đổ trước.  
\* Được hỗ trợ trong các trình duyệt phổ biến hiện nay

– Hạn chế  
\* Bạn không thể bookmark nó vào favourite trên trinh duyệt hay gởi link đến cho bạn bè, vì tất cả quá trình nó thực hiện ngầm và không hiển thị trên address  
\* Khó debug  
\* không bảo mật  
\* Không thể hiện thị nội dung trên các trang tìm kiếm vì các trang tìm kiếm hiện nay vẫn chưa hỗ trợ tìm vì rất khó tìm và gần như không thể tìm đc.  
\* Không thể sử dụng nút back vì back cũng là chính nó  
\* Với một số trình duyệt, do nhu cầu bảo mật, sẽ tắt chức năng thực hiện javascript nên ajax không thể chạy, hay trong một vài host, không hỗ trợ vào sâu cấu hình server nên hay bị lỗi “Access denied”

**2. Cơ chế bảo mật trong ajax**

Vấn đề bảo mật có thể gặp phải:  
Khi ajax được gửi thông qua các text đơn giản, và nó có thể dẫn đến sẽ biết được các chi tiết của database  
Hacker có thể chèn các scripting độc hại và dễ dàng xập nhập vào hệ thống

**3. Side control nào ajax tương tác: server side hay client side**

**4. Công nghệ nào được sử dụng dể tạo ajax**  
+ JavaScript  
+ XMLHttpRequest  
+ Document Object Model (DOM)  
+ Extensible HTML (XHTML)  
+ Cascading Style Sheets (CSS)

**5. Một số freamwwork về ajax**

**6. Json trong ajax**

JSON là một kiểu mô tả dữ liệu dựa trên cơ sở là Javascript dùng để truyền tải dữ liệu . Các loại json hổ trợ: string, object..

**7. Post, get- cái nào tốt hơn (http,ajax,jquery)**

+ POST: Bảo mật hơn GET vì dữ liệu được gửi ngầm, không xuất hiện trên URL (sai)  
+ GET: Dữ liệu được gửi tường minh, chúng ta có thể nhìn thấy trên URL, đây là lý do khiến nó không bảo mật so với POST. Nó còn bị giới hạn số ký tự bởi URL của web browsers.  
+ GET thực thi nhanh hơn POST vì nhứng dữ liệu gủi đi luôn được Webbrowser cached lại  
+ Khi dùng phương thức POST thì server luôn thực thi và trả về kết quả cho client, còn phương thức GET ứng với cùng 1 yêu cầu đó webbrowser sẽ xem trong cached có kết quả tương ứng với yêu cầu đó ko và trả về ngay không cần phải thực thi các yêu cầu đó ở phía server  
+ Đối với những dữ liệu luôn được thay đổi thì chúng ta nên sử dụng phương thức POST, còn dữ liệu ít thay đổi chúng ta dùng phương thức GET để truy xuất và xử lý nhanh hơn.  
Dùng POST trong: insert, update,delete; Get : search..

Bất lợi của post:  
– Không copy link hay lưu lại link được.  
– Không SEO được (robot không thể post như người)  
– Người dùng không thể back/forward được

Vì vậy dùng GET khi:  
– Không sợ bị lưu vết hay bị nhìn thấy bởi người đứng sau lưng  
– Độ dài trong giới hạn cho phép  
– Ngoài ra thì dùng GET trong mọi trường hợp có thể được

Chỉ dùng POST khi:  
– Không thể dùng GET

\* XmlHttpRequest object is used for Ajax requests  
**8. Jquery : cơ chế**

Gần như là một mã nguồn mở sử dụng JS, Jquery tổng hợp các vấn đề cần phải làm trong JS thành một thư viện, và sau đó nó cho phép phát biểu theo cách gọi riêng của nó

[*Câu hỏi phỏng vấn JSP*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-jsp/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-jsp/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-jsp/#respond)

**1. Scriptlet trong jsp**  
Scriplet có thể chứa bất kỳ ngôn ngữ, khai báo biến, phương thức, biểu thức trong thẻ page scripting languega.  
Scriptlets dc thực hiện tại thời điểm dc yêu cầu, khi JSP xử lý yêu cầu từ client request. Nếu scriptlet thực hiện cho ra 1 output, output sẽ dc lưu trữ trong out objectbạn có thể hiển thị nó  
**2. Lợi và hại trong jsp**

Ưu điểm:  
– Dễ phát triển hơn Java servlet: Do trang JSP trước khi thực hiện nó được biên dịch thành servlet nên những gì servlet làm được chắc chắn JSP cũng có thể làm được. Viết một trang JSP đơn giản hơn dễ dàng hơn cho người mới bắt đầu với JSP  
– JSP có tất cả các lợi thế(các phần import) của java  
– JSP sử dụng ngôn ngữ kịch bản đơn giản dựa trên cú pháp cho HTML nhúng vào JSP  
– Dễ dàng truy cập đối tượng  
– Sử dụng chuẩn HTTP request/response àJSP lý tưởng cho công nghệ Web

Nhược điểm  
– Khó kiểm tra lỗi  
– Không áp dụng dc mô hình MVC-> chỉ thực hiện trong dự án nhỏ  
– Nếu tất cả xử lý mã java được tập trung ở JSP thì việc mở rộng hay nâng cấp dự án sẽ rất khó khăn, việc phân chia các modun cũng gặp nhiều khó khăn, phức tạp hơn so với servlet. Mã JSP ở dạng thuần văn bản nên việc che dấu mã nguồn logic thường rất kém

**3. Lưu ý khác trong jsp**  
*Q: What is a Expression?*  
A: <%= someexpression %>: a expression là string, ko có dấu ; phía sau

*Q: What is a Declaration?*  
A: <%! int i = 0; %>

*Q: Life-circye jsp:*  
A: – Compilation (biên dịch)  
– Initialization  
– Execution  
– Cleanup

Tham khảo: <http://ht117.wordpress.com/2014/02/12/jsp-life-cycle/>

*Q: Difference between SendRedirect() and Forward() in JSP Servlet*  
A:  
SendRedirect(url): chuyển tới 1 file view khác  
Forward(): chuyển tới 1 requestDispatcher khác

<http://javarevisited.blogspot.com/2011/09/sendredirect-forward-jsp-servlet.html>

*Q: Is it possible for one JSP to extend another java class if yes how?*  
A: <%@ include page extends=”classname” %>

*Q: What is < jsp:usebean >tag why it is used*

usebean là khai báo 1 javabean trong jsp. Mỗi lần khai báo, bean này có thể đc truy cập trong các thẻ scripting khác trong jsp.bạn có thể get phương thức,set thuộc tính của bean

A javabean:  
+ construtor không có tham số  
+ có set,get properties  
+ có 1 số thuộc tính có thể read hoặc written  
+ nên serializable và implement serializable interface

*Q: what are the implicit Object*  
A: request, response, pageContext, session, and application, out, config, page, and exception

*Q: How can you pass information form one jsp to included jsp*  
A:  
Sử dụng: <jsp:param>  
<jsp:include page=”newbid.jsp” flush=”true”>  
<jsp:param name=”price” value=”123.7″/>  
<jsp:param name=”quantity” value=”4″/>

*Q:  Is it possible for one JSP to extend another java class if yes how*  
A: yes. <%@ include page extends=”classname”%>

Tham khảo thêm: <http://www.tutorialspoint.com/jsp/jsp_interview_questions.htm>

[*Câu hỏi phỏng vấn Hibernate*](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-hibernate/)

*[15MondayJun 2015](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-hibernate/)*

Posted by [pikachu06](https://pikachu06.wordpress.com/author/pikachu06/) in [Câu hỏi phỏng vấn Java/J2ee](https://pikachu06.wordpress.com/category/cau-hoi-phong-van-javaj2ee/), [Hibernate framework](https://pikachu06.wordpress.com/category/hibernate-framework/)

**≈**[**Leave a comment**](https://pikachu06.wordpress.com/2015/06/15/cau-hoi-phong-van-hibernate/#respond)

**1. ORM là gì**  
ORM là một phương pháp lập trình để chuyển đổi từ mô hình database sang mô hình đối tượng

**2. Hibernate là gì**  
Hibernate framework là một framework cho persistence layer, chuyển đổi từ mô hình database sang mô hình đối tượng

**3. Ưu điểm và nhược điểm của Hibernate**  
*\*Lợi thế khi dùng Hibernate:*  
+ Nâng suất: không viết code sql, ít viết code java  
+ Hiệu suất: cahce  
+ Dễ bảo trì  
+ linh hoạt: do generate code sql, có thể tùy chỉnh sql  
*\*Nhược điểm của Hibernate*  
Chậm hơn jdbc do phải generate code sql

**6. HQL là gì**  
Ngôn ngữ để truy vấn dưới hình thức hướng đối tượng khi bạn đang phát triển ứng dụng hướng đối tượng tương tác với một cơ sở dữ liệu quan hệ thông qua Hibernate

**7. Hibernate có các interface core gì:**  
Session  
SessionFactory  
Configuration  
Transaction  
Query và criteria

**8. Sự khác nhau giữa load() và get()**  
(lấy 1 object )  
load(): chắc chắn object tồn tại  
ném exception nếu id ko tìm thấy trong database  
get(): ko chắc object có tồn tại  
nếu id ko tìm thấy sẽ return null