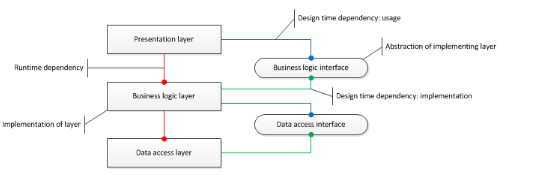
[Bài đọc] Dependency Injection

Dependency Injection

Có một quy tắc thiết kế quan trọng là các tầng khác nhau của hệ thống chỉ phụ thuộc tới interface.



Việc tuân thủ quy tắc này giúp cho phần mềm dễ thay đổi hay mở rộng hơn. Tuy vậy, nếu không có phương pháp phù hợp, việc đảm bảo nó là không dễ.

Để dễ hiểu, hãy soi chiếu trên một thiết kế quen thuộc, phân tầng presentation và tầng service. Giả sử trong tầng presentation có class ControllerA, và class này sử dụng đến - đồng nghĩa với phụ thuộc vào - class ServiceX nằm trong tầng service.

main() {  
 controller.doA();  
}  
  
class ControllerA {  
 public void doA() {  
 ServiceX service = new ServiceX();  
 service.doX();  
 }  
}  
  
class ServiceX {  
 public void doX() {  
  
 }  
}

Như vậy là ControllerA nằm trong presentation đang phụ thuộc tới một class triển khai nằm trong service, và rõ là vi phạm nguyên tắc thiết kế đã nói ở trên.

Kể cả khi tái cấu trúc để trừu tượng hóa api của service, việc phụ thuộc vào lớp triển khai vẫn diễn ra.

main() {  
 controller.doA();  
}

class ControllerA {  
 public void doA() {  
 ServiceX service = new ServiceXImlement();  
 service.doX();  
 }  
}  
  
interface ServiceX {  
 void doX();  
}  
  
class ServiceXImlement implements ServiceX {  
 public void doX() {  
 }  
}

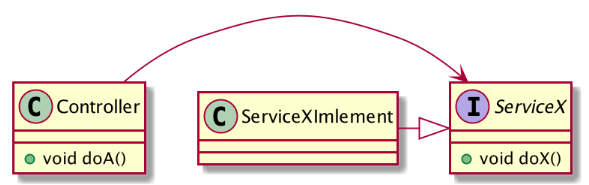
Nguyên tắc vẫn bị vi phạm:

Bây giờ, tại một vỏ bọc nằm bên ngoài cả presentation lẫn service, chẳng hạn như tại main(), mối phụ thuộc mà presenter cần đến được tạo ra, và sau đó được truyền vào cho presenter:

main() {  
 ServiceX service = new ServiceXImplement();  
 controller.setService(service);  
 controller.doA();  
}

class ControllerA {  
 private ServiceX service;  
  
 public void setService(ServiceX service) {  
 this.service = service;  
 }  
  
 public void doA() {  
 service.doX();  
 }  
}

Giờ đây presentation không còn phụ thuộc vào service nữa:



Đây chính là cơ chế tiêm phụ thuộc nổi tiếng. Và Spring ứng dụng triệt để cơ chế này để hỗ trợ nhà phát triển tạo ra các chương trình dễ mở rộng, dễ bảo trì, và dễ dàng kiểm thử.

Bạn hãy tìm hiểu cách để sử dụng cơ chế tiêm phụ thuộc này bằng vỏ bọc của Spring.