**Giới thiệu**

**tight-coupling** hay "liên kết ràng buộc" là một khái niệm trong Java ám chỉ việc mối quan hệ giữa các Class quá chặt chẽ. Khi yêu cầu thay đổi logic hay một class bị lỗi sẽ dẫn tới ảnh hưởng tới toàn bộ các Class khác.

**loosely-coupled** là cách ám chỉ việc làm giảm bớt sự phụ thuộc giữa các Class với nhau.

**Ví dụ dễ hiểu**

Lấy một ví dụ:

Bạn có một Class thực thi một nhiệm vụ cực kỳ phức tạp, và một trong số đó là việc sắp xếp dữ liệu trước khi xử lý.

**Cách code level 1:**

public class BubbleSortAlgorithm{

public void sort(int[] array) {

// TODO: Add your logic here

System.out.println("Đã sắp xếp bằng thuật toán sx nổi bọt");

}

}

public class VeryComplexService {

private BubbleSortAlgorithm bubbleSortAlgorithm = new BubbleSortAlgorithm();

public VeryComplexService(){

}

public void complexBusiness(int array[]){

bubbleSortAlgorithm.sort(array);

// TODO: more logic here

}

}

Với cách làm ở trên, VeryComplexService đã hoàn thiện được nhiệm vụ, tuy nhiên, khi có yêu cầu **thay đổi** thuật toán sắp xếp sang QuickSort thì nghe vẻ chúng ta sẽ phải sửa lại hoàn toàn cả 2 Class trên.

Ngoài ra BubbleSortAlgorithm sẽ chỉ tồn tại nếu VeryComplexService tồn tại, vì VeryComplexService tạo đối tượng BubbleSortAlgorithm bên trong nó (hay nói cách khác là sự sống chết của BubbleSortAlgorithm sẽ do VeryComplexService quyết định), theo như cách implement này, nó là liên kết rất chặt với nhau.

**Cách làm level 2:**

public interface SortAlgorithm {

/\*\*

\* Sắp xếp mảng đầu vào

\* @param array

\*/

public void sort(int array[]);

}

public class BubbleSortAlgorithm implements SortAlgorithm{

@Override

public void sort(int[] array) {

// TODO: Add your logic here

System.out.println("Đã sắp xếp bằng thuật toán sx nổi bọt");

}

}

public class VeryComplexService {

private SortAlgorithm sortAlgorithm;

public VeryComplexService(){

sortAlgorithm = new BubbleSortAlgorithm();

}

public void complexBusiness(int array[]){

sortAlgorithm.sort(array);

// TODO: more logic here

}

}

Với cách làm này, VeryComplexService sẽ chỉ quan hệ với một interface SortAlgorithm. Với cách này thì mỗi quan hệ giảm bớt sự liên kết, nhưng nó không thay đổi được việc thuật toán vẫn đang là BubbleSortAlgorithm.

**Cách làm level 3:**

public interface SortAlgorithm {

/\*\*

\* Sắp xếp mảng đầu vào

\* @param array

\*/

public void sort(int array[]);

}

public class BubbleSortAlgorithm implements SortAlgorithm{

@Override

public void sort(int[] array) {

// TODO: Add your logic here

System.out.println("Đã sắp xếp bằng thuật toán sx nổi bọt");

}

}

public class QuicksortAlgorithm implements SortAlgorithm {

@Override

public void sort(int[] array) {

// TODO: Add your logic here

System.out.println("Đã sắp xếp bằng thuật sx nhanh");

}

}

public class VeryComplexService {

private SortAlgorithm sortAlgorithm;

public VeryComplexService(SortAlgorithm sortAlgorithm){

this.sortAlgorithm = sortAlgorithm;

}

public void complexBusiness(int array[]){

sortAlgorithm.sort(array);

// TODO: more logic here

}

}

public static void main(String[] args) {

SortAlgorithm bubbleSortAlgorithm = new BubbleSortAlgorithm();

SortAlgorithm quickSortAlgorithm = new QuicksortAlgorithm();

VeryComplexService business1 = new VeryComplexService(bubbleSortAlgorithm);

VeryComplexService business2 = new VeryComplexService(quickSortAlgorithm);

}

Cách thứ ba này cũng là cách làm phổ biển nhất. Mối liên hệ giữa 2 Class đã "lỏng lẻo" hơn trước rất nhiều. VeryComplexService sẽ không quan tâm tới việc thuật toán sắp xép là gì nữa, mà chỉ cần tập trung vào nghiệp vụ. Còn SortAlgorithm sẽ được đưa vào từ bên ngoài tùy theo nhu cầu sử dụng.

**Dependency Injection**

Sau khi bạn đã nắm được 2 khái niệm **tight-coupling** và **loosely-coupled** thì sẽ có thể hiểu dễ dàng khái niệm **Dependency Injection**. Một trong những nhân tố chính giúp cuộc đời lập trình Java của bạn trở nên tươi sáng hơn.

Xem thêm tại:

[🌛](https://loda.me/articles/core-gii-thch-dependency-injection-di-v-ioc-bng-ngc-trinh)

[[core] Giải thích Dependency Injection (DI) và IoC bằng Ngọc Trinh](https://loda.me/articles/core-gii-thch-dependency-injection-di-v-ioc-bng-ngc-trinh)