Some programming languages like javascript have a feature called callbacks. Using callbacks, you can pass a function as variable into another function. For example :

```
var success = function() {
  console.log("Wooo, it worked!");
}

var task = function(onSuccess) {
  /// Do some task
  if(task.complete) {
    if(typeof onSuccess == "function") {
      onSuccess();
    }
  }
}

task(success)
```

What the above code does is it performs the task, and it calls the onSuccess method if one is passed in.

Java 1.7 on the other hand doesn't have callbacks.

So lets say we have a class Task and 2 classes A and B which both call Task. Both A and B have a method that needs to be called right when the Task is finished and the Task is run asynchronously. Implement a way to do this "callback" in Java without using inheritance

```
class A{
  public void taskFinished() {
    System.out.print('hello');
  }

  public void runTask() {
    Task task = new Task();
    task.run()
  }
```

```
}

class B{
  public void taskFinished() {
    System.out.print('hi');
  }

  public void runTask() {
    Task task = new Task();
    task.run()
  }
}

class Task{
  public void run() {
    //Do some task
  }
}
```