**GIải thích các Mẫu thiết kế - Bài tập tự làm**

Mục tiêu của tài liệu này là cung cấp một kinh nghiệm làm việc với mẫu thiết kế trong tầm tay, và cho thấy làm thế nào các mẫu có thể xuất hiện xuyên suốt vòng đời của một dự án phần mềm, cũng như sự thay đổi yêu cầu, phát triển, và trở nên rõ ràng hơn.

Như tất cả các bài tập có dạng này, điều quan trọng là cần cân bằng độ phức tạp, điều làm cho các bài tập trở nên thú vị và thực tế hơn, với việc tập chung vào vấn đề chúng ta đang cố gắng nhấn mạnh. Do đó, trong khi dự án liên quan có thể đơn giản, ý tưởng thiết kế được đưa ra hy vọng sẽ phong phú, thú vị và mạnh mẽ.

Cách thức hoạt động:

Đây là một tài liệu độc lập (DPE\_Self-Paced\_Exercise\_Solutions.pdf) chứa code mẫu dự án của chúng tôi. Không được tham khảo code này trong khoảng thời gian sắp tới, vì chúng sẽ ngăn bạn tiếp cận với việc trải nghiệm hiện thực hóa các mẫu thiết kế.

Bạn sẽ nhận được các yêu cầu trong tài liệu này. Hãy để bản thân bạn có thời gian suy nghĩ về vấn đề, nhưng đừng over-design – ghi nhớ, nếu bạn giữ code có phẩm chất, nguyên tắc và thực hành như đã học đươc trong khóa học này trong đầu của bạn một cách lâu dài, bạn không cần phải tiếp cận những thứ vụn vặt, và bạn ko cần phải làm quá thiết kế của bạn.

Khi bạn đã có giải pháp cho vấn đề, hãy nhìn vào tài liệu solution. Ở đó bạn sẽ tìm thấy giải pháp code được chúng tôi đề xuất, kèm theo các giải thích. Cái mà chúng tôi đã làm sẽ không cần thiết phải tốt hơn cái bạn đã làm (trong thực tế, chúng tôi không hi vọng vậy!), nhưng chúng có thể sẽ hữu ich cho bạn để cho dự án của bạn đồng bộ với các mà các ví dụ thể hiện, do đó bài tập nãy sẽ là cho chúng ta có cảm giác tiến bộ.

Một khi bạn đã xem xét quá tài liệu mẫu và làm cho code của bạn giống như của chúng tôi, lật úp trang tài liệu đó lại và tiếp tục với một tập các yêu cầu khác. Một lần nữa, thay đổi giải pháp của bạn để chứa các yêu cầu mới đó, nhưng đừng có tham khảo tài liệu.

Chúng ta bắt đầu từ đây, và chúng ta sẽ thấy các mẫu xuất hiện, miễn là chúng ta giữ cho code ở chất lượng tốt nhất.

Sang trang, và chúng ta sẽ bắt đầu.

**Bước 1: Khởi đầu đơn giản**

Bạn đang code một đối tượng doanh nghiệp cho hệ thống Package Handling. Một team khác đang tạo giao diện người dung (GUI), nhưng chúng chưa sẵn sang. Một team khác nữa đang làm việc với một lớp persistence, cùng với nhiều thúc khác, và sẽ tạo ra một Shipment ID duy nhất cho những lúc mà bạn cần, nhưng mà nó vẫn chưa xong.

Bạn cần, ít nhất, 2 đối tượng. Một đối tượng để thể hiện việc điều khiển, hay đối tượng Client đó sẽ tương tác với GUI. Một đối tượng khác thể hiện đối tượng đang được chuyển đi (shipped)

Trạng thái từ người dung thông quấ GUI:

* ShipmentID (một gia trị int thể hiện một ID đã tồn tại, hay 0, nghĩa là bạn cần phải tạo ra một ID mới, duy nhất tại thời điểm xây dựng.
* ToAddress (một string chứa tên đường, thành phố, bang) – có thể thay đổi được.
* FromAddress (một string chứa tên đường, thành phố, bang) – có thể thay đổi được.
* ToZipCode (một chuỗi chứa chính xác 5 ký tự) - có thể thay đổi được.
* FromZipCode (một chuỗi chứa chính xác 5 ký tự) - có thể thay đổi được.
* Weight (một giá trị double, thể hiện cân nặng của đôi tượng bằng đơn vị ounces).

Đối tượng client sẽ chứa một thể hiện của đối tượng thể hiện vật cần được chuyển, và sau đó yêu cầu nó “tự chuyển nó đi”. Yêu cầu này sẽ trả về một string duy nhất chỉ ra ID của lô hàng, nơi đối tượng được chuyển đến, nơi đối tượng sẽ đi đến, và tốn bao nhiêu tiền.

Chi phí sẽ được tính bằng trọng lượng. tỉ lệ là 39 cent cho một ounce sẽ được áp dụng.

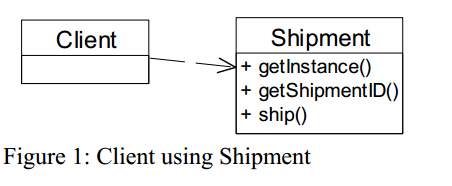
Một khi client đã nhận được các thông tin trả về, nó cần phải hiện thị chúng lên màn hình console.

Nếu bạn quen thuộc với unit test, hãy viết chúng. Ví dụ của chúng ta trong thực tế là theo hướng test-driven. Tuy nhiên unit test sẽ không được yêu cầu vì đây không phải là khóa học về phát triển hướng kiểm thử (TDD). Điều đó nói rằng, xem xét các khả năng kiểm thử cho thiết kế của bạn luôn luôn cần thiết.

Các thứ cần ghi chú:

* Chắc chắn rằng bạn đã theo đúng các practice tốt. luôn luôn lập trình theo mục đích, và luôn luôn đóng gói phần khởi tạo đối tượng,
* Vì bạn khôn có một GUI cụ thể, bạn cần phải dummy lên đâu đó. Chi cần chắc chắn ràng “đâu đó” sẽ dễ dàng thay đổi.
* Vì bạn không có một lớp persistence cụ thể nơi sẽ tạo ra các ID duy nhất cho bạn, bạn cần chỉ đơn giản là tự tạo một ID duy nhất cho thời điểm hiện tại. Một lần nữa, chắc chắn rang nó sẽ dễ dàng thay đổi khi code thực sự đã sẵn sàng. Gợi ý: tạo một method tên là getShipmentID, bây giờ sẽ tăng giá trị của một biến tĩnh lên 1 và trả nó về nhưng trong tương lai sẽ lấy một shipment ID thực sự.

Hình 1 thể hiện những ý tưởng trên trong UML.



DÙNG LẠI Ở ĐÂY VÀ BẮT ĐẦU CODE SOLUTION CỦA BẠN, KHI BẠN HOÀN THÀNH, MỞ TÀI LIỆU THAM KHẢO VÀ XEM XÉT BƯỚC 1.

**Bước 2: Nhiều Shipper.**

Bây giờ bạn được thông báo rằng bây giờ sẽ có nhiều công ty thành viên sẽ vận chuyển đối tượng trong câu hỏi. Điều nãy sẽ làm ảnh hưởng đến tỉ lệ cent/ounces khi giá được xác định.

Các công ty bao gồm:

* Air East: Trụ sở tại Atlanta
* Chicago Sprint: Trụ sở tại Chicago
* Pacific Parcel: Trụ sở tại San Diego

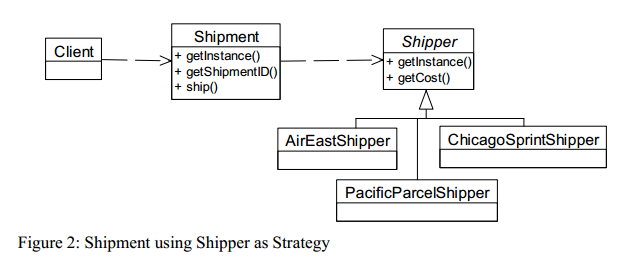
Người vận chuyển sẽ được xác định chính xác bằng zip code của người gởi (from zip code). Nếu zip code bắt đầu bằng 1, 2, 3 thì nguồn sẽ từ phía đông, do đó Air East sẽ được dùng. Nếu kí tự đầu tiên của zip code là 4, 5, 6 thì Chicago Sprint sẽ được lựa chọn. Còn 7, 8, 9 sẽ có nghĩa là phía tây nước Mỹ, và do đó sẽ là Pacific Parcel.

Air East sẽ có giá 39 cent cho một ounces, Chicago Sprint là 42 cent, Pacific Parcel là 51 cent

Ghi chú rằng “to” zip code không ảnh hưởng đến giá. Tât cả các nhà cung cấp của chúng ta đều có thỏa thuận cho việc vận chuyển xuyên quốc gia,

Nếu zip code không được xác định, mặc định sẽ là Air East.

Hình 2 thể hiện chúng trên UML



**Bước 3: Nhiều thể loại hàng hóa khác nhau**

Bây giờ có 3 loại hàng hóa khác nhau: thư, gói, và quá cỡ. Chúng ta xác định loại hàng hóa nào sẽ chuyển căn cứ vào trọng lượng của chúng.

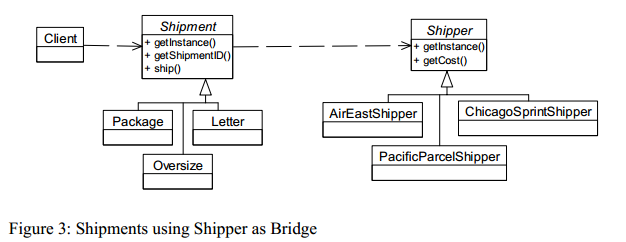
Thường thường chúng ta áp dụng các luật lệ sau:

* Một lá thư thường ko nặng quá 15 ounce (ít hơn 1 pound)
* Một gói thường ko nặng quá 160 ounce.
* Một hàng quá cỡ là bất cứ thứ gì nặng hơn 160 ounc.

Các dịch vụ vận chuyển của chúng ta sẽ có các cách tính toán khác nhau, dựa trên loại hàng hóa sẽ được chuyển.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Air West | Chicago Sprint | Pacific Parcel |
| Thư | 39 cent/ounce | 42 cent/ounce | 51 cent/ounce |
| Gói | 25 cent/ounce | 20 cent/ounce | 19 cent/ounce |
| Quá cỡ | Tính theo kiểu gói, cộng thêm 10$ | Không tính phí | Tính bình thường theo kiểu gói, cộng thêm chi phí 2 cent cho mỗi ounce. |

Hình 3 mô tả chúng trên UML



**Bước 4: Tạo lô hàng với mã đặc biệt**

Nếu được lựa chọn bằng GUI, hàng hóa có thể được đánh dấu là Dễ vỡ, Không được làm rớt, Yêu cầu trả về tiếp nhận, hoặc là bất cứ sự kết hợp nào của 3 loại này với nhau. Môt lần nữa, GUI chưa sẵn sàng, do đó bạn cần làm điều này ở đâu đó, đâu cũng được.

Hiệu ứng của vấn đề này sẽ là khi Khách hàng yêu cầu hàng hóa chuyển đi, các thông tin này sẽ hiện ra ở cuối dòng, và tất cả là chữ hoa được đánh dấu bởi các dấu sao. Ví dụ khi cả 3 đánh dấu được áp dụng:

Your Shipment with the ID 17263  
will be picked up from 12292 4th Ave SE, Bellevue, Wa 92021  
and shipped to 1313 Mockingbird Lane, Tulsa, OK 67721  
Cost = 5.1  
\*\*MARK FRAGILE\*\*  
\*\*MARK DO NOT LEAVE IF ADDRESS NOT AT HOME\*\*  
\*\*MARK RETURN RECEIPT REQUESTED\*\*

Hình 4 mô tả các điều trên

