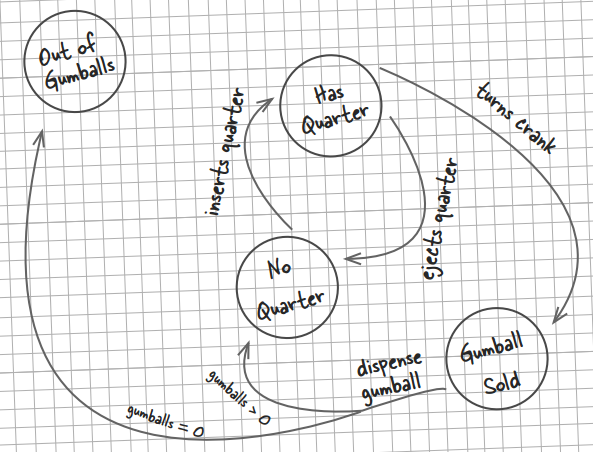
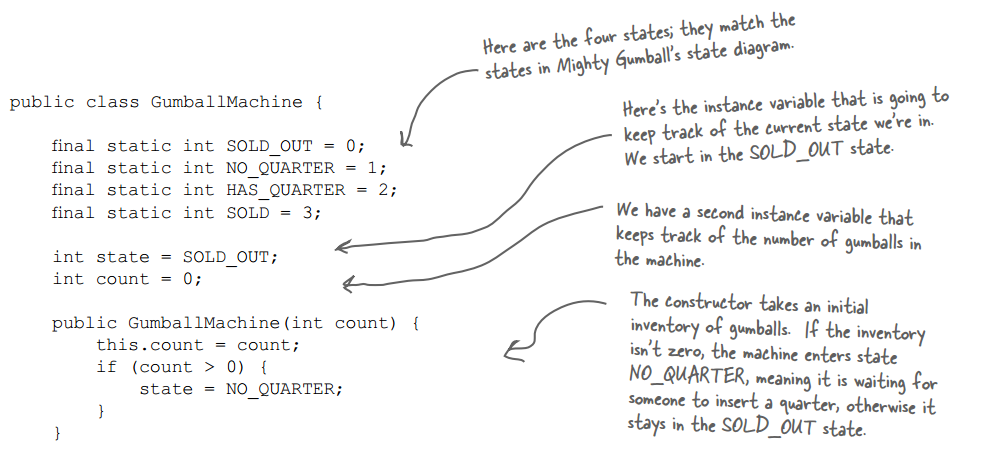
Gumball Machine

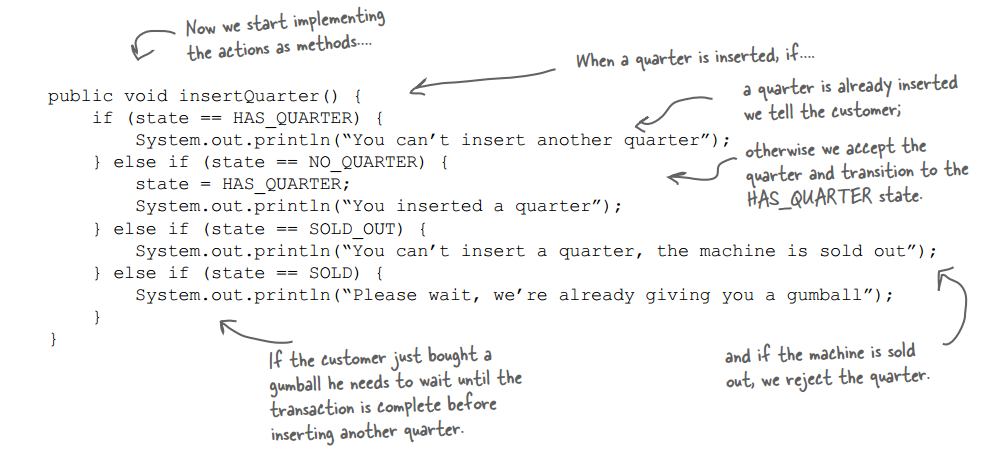
La empresa Mighty Gumball Inc. nos ha dado la responsabilidad de implementar el software para sus máquinas de goma de mascar.

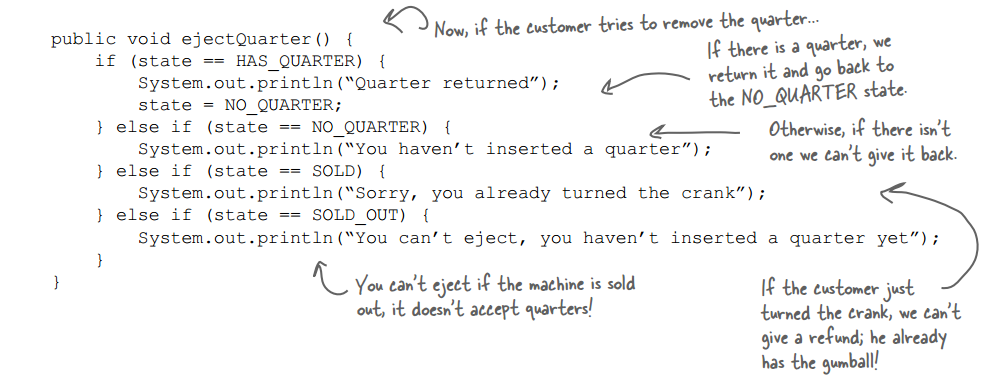
Los especialistas de Mighty Gumball Inc. esperan que el controlador de la máquina de goma maneje la siguiente lógica. Ellos esperan agregar más comportamiento en el futuro, entonces se necesita mantener el diseño lo más flexible y mantenible posible.

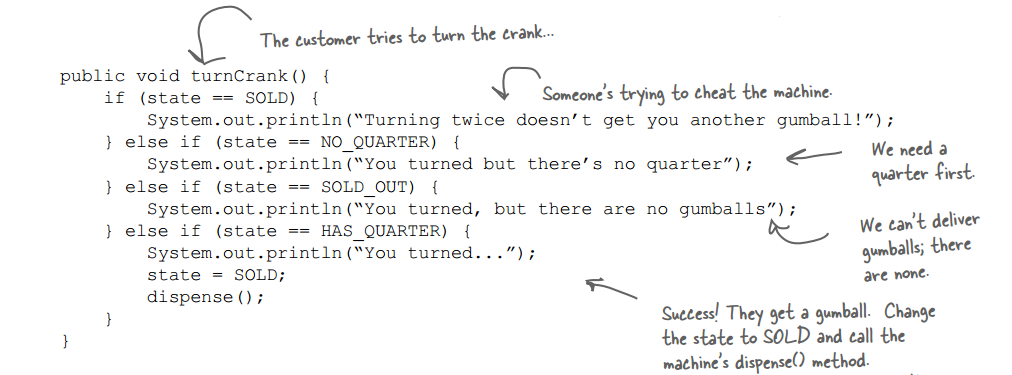


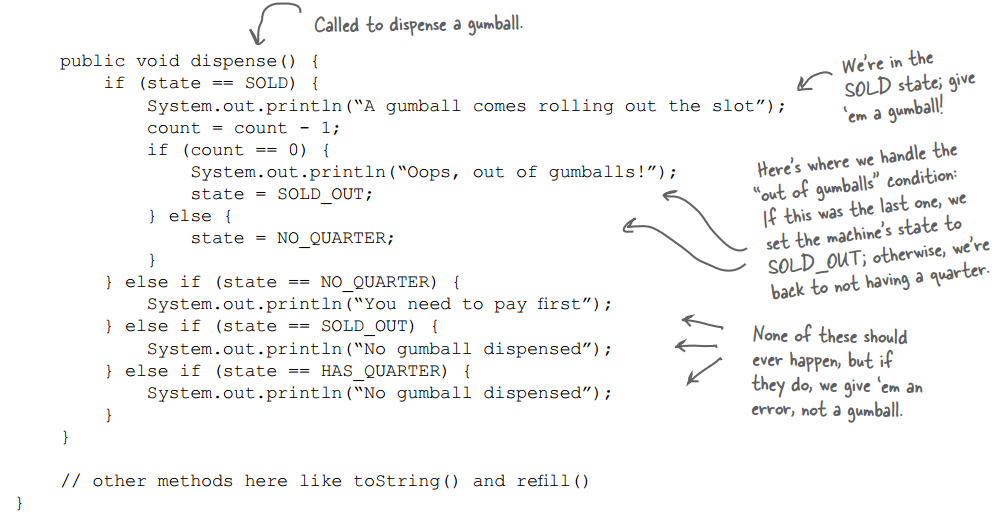
# La primera versión de la aplicación







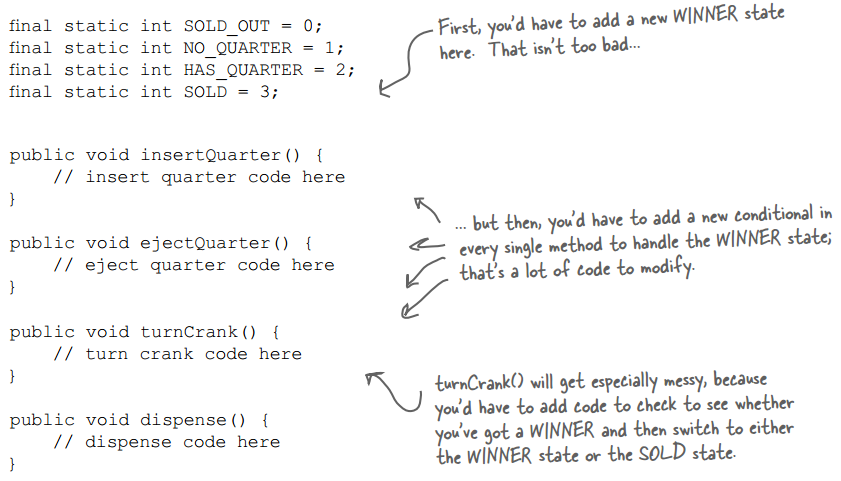




# Sabíamos que pasaría…un requerimiento de cambio

El CEO de Mighty Gumball, Inc. piensa que convertir “comprar un goma” en un juego incrementaría significativamente las ventas. Para esto, desea que agreguemos la siguiente lógica: “el 10% de las veces que se gira la manivela, el cliente recibe 2 gomas en vez de 1”.

Analicemos cómo implementar este cambio…….



¿Cuáles de los siguientes puntos describen el estado de nuestra implementación?

|  |  |
| --- | --- |
| 1. El código definitivamente no se adhiere al “Open Closed Principle”. 2. Este código haría que un programador FORTRAN se sienta orgulloso. 3. El diseño no es para nada orientado a objetos. | 1. La transición de estados no es explícita, está dispersa en el medio de varias sentencias condicionales. 2. El código no se adhiere al “Encapsulate What Varies Principle”. 3. Es probable que las nuevas adiciones causen errores en el código que ya funciona. |

# Class Diagram

