Contents

[Toy Vending Machine 2](#_Toc413932447)

[Introduction 2](#_Toc413932448)

[Use Cases 2](#_Toc413932449)

[Database Tables 3](#_Toc413932450)

[Class Diagrams 4](#_Toc413932451)

[Overview 4](#_Toc413932452)

[Person, Customer, and Owner 5](#_Toc413932453)

[Database 6](#_Toc413932454)

[Source Codes 7](#_Toc413932455)

[App.java 7](#_Toc413932456)

[Person.java 9](#_Toc413932457)

[Owner.java 10](#_Toc413932458)

[Customer.java 12](#_Toc413932459)

[VendingMachine.java 14](#_Toc413932460)

[CapsuleVendingMachine.java 15](#_Toc413932461)

[CapsuleFactory.java 16](#_Toc413932462)

[MiniFigurineCapsuleFactory.java 17](#_Toc413932463)

[StickerCapsuleFactory.java 18](#_Toc413932464)

[DAO.java 19](#_Toc413932465)

[HibernateDAO.java 20](#_Toc413932466)

[MiniFigurineCapsuleDAO.java 22](#_Toc413932467)

[StickerCapsuleDAO.java 23](#_Toc413932468)

[Capsule.java 24](#_Toc413932469)

[ColoredToyCapsule.java 25](#_Toc413932470)

[MiniFigurineCapsule.java 26](#_Toc413932471)

[StickerCapsule.java 27](#_Toc413932472)

[Content.java 28](#_Toc413932473)

[Describable.java 29](#_Toc413932474)

[MiniFigurine.java 30](#_Toc413932475)

[Sticker.java 31](#_Toc413932476)

[Resources 32](#_Toc413932477)

[beans.xml 32](#_Toc413932478)

[MiniFigurine.hbm 35](#_Toc413932479)

[MiniFigurineCapsule.hbm 36](#_Toc413932480)

[Sticker.hbm 37](#_Toc413932481)

[StickerCapsule.hbm 38](#_Toc413932482)

# Toy Vending Machine

## Introduction

The project is about a capsule vending machine. You put a quarter into the slot, and a capsule rolls out. A database will be used to store the capsules, and each time a user turns the knob, the vending machine will remove a capsule from

the database and return the capsule to the user.

User can now choose between mini figurines or stickers. Unlike figurines, each capsule contains two stickers.

### Use Cases

1. Anyone can interact with the toy vending machine. They can turn the knob, and the machine will crank out a toy.
2. Anyone can interact with the toy itself. They can open the capsule, and inside is a mini figurine or stickers
3. Anyone can choose between mini figurines or stickers
4. Only the owner can refill the machine.

## Database Tables

create table MiniFigurine (

FigurineId int primary key,

name varchar(255)

);

create table MiniFigurineCapsule (

CapsuleId int primary key,

Color varchar(10),

FigurineId int

);

create table Sticker (

StickerId int primary key,

name varchar(255),

CapsuleId int

);

create table StickerCapsule (

CapsuleId int primary key,

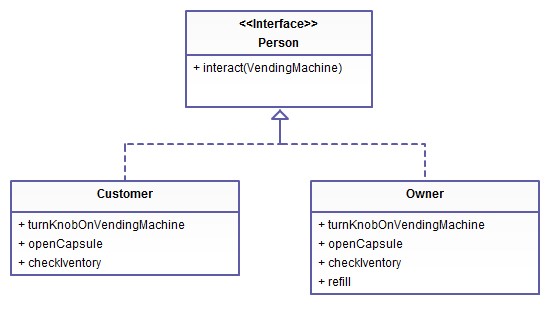
Color varchar(10)

);

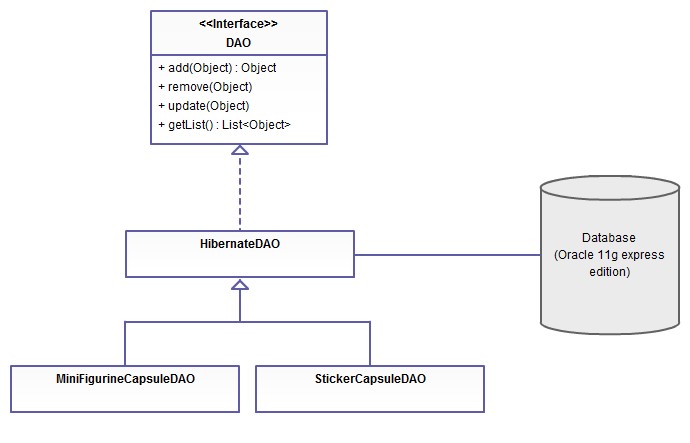
## Class Diagrams

### Overview

### Person, Customer, and Owner



### Database



## Source Codes

### App.java

package com.tom.VendingMachine;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import org.springframework.context.ApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

/\*\*

\* Hello world!

\*

\*/

public class App

{

public static void main( String[] args )

{

ApplicationContext context = new ClassPathXmlApplicationContext("beans.xml");

// display greeting

System.out.println( "Hello! Welcome to the store!" );

try {

// prompt user: ask for 'customer or owner' status

System.out.println("Are you a customer or the owner?[customer, owner]");

// read user input

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String prompt1 = br.readLine();

// prompt user

System.out.println("Which capsule vending machine to use?[mini figurine, sticker]");

// read user input

BufferedReader br2 = new BufferedReader(new InputStreamReader(System.in));

String prompt2 = br.readLine();

VendingMachine vendingMachine = null;

// The user interacts with the machine.

if (new String("sticker").contains(prompt2.toLowerCase())) {

vendingMachine = (VendingMachine) context.getBean("sticker vending machine");

} else if (new String("mini figurine").contains(prompt2.toLowerCase())) {

vendingMachine = (VendingMachine) context.getBean("figurine vending machine");

} else {

throw new Exception("unsupported vending machine");

}

// The user interacts with the machine.

if (new String("owner").contains(prompt1.toLowerCase())) {

Person person = new Owner();

person.interact(vendingMachine);

} else if (new String("customer").contains(prompt1.toLowerCase())) {

Person person = new Customer();

person.interact(vendingMachine);

} else {

throw new Exception("unknown person category");

}

} catch (Exception e) {

e.printStackTrace();

}

}

}

### Person.java

package com.tom.VendingMachine;

public interface Person {

void interact(VendingMachine vm) throws Exception;

}

### Owner.java

package com.tom.VendingMachine;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.List;

public class Owner implements Person {

private Customer customer = new Customer();

public Object turnKnobOnVendingMachine(List<Describable> inventory, VendingMachine vendingMachine) {

return customer.turnKnobOnVendingMachine(inventory, vendingMachine);

}

public void openCapsule(List<Describable> inventory, List<Capsule> unopenedCapsules) {

customer.openCapsule(inventory, unopenedCapsules);

}

public void checkInventory(List<Describable> inventory) {

customer.checkInventory(inventory);

}

public void refill(VendingMachine vendingMachine) {

vendingMachine.refill();

System.out.println("\nYou refill the vending machine.");

}

public void interact(VendingMachine vm) throws Exception {

List<Describable> inventory = new ArrayList<Describable>();

List<Capsule> unopenedCapsules = new ArrayList<Capsule>();

System.out.println("You are the owner.");

while (true) {

System.out.println();

System.out.println("What would you like to do?");

System.out.println("1. Turn knob on vending machine");

System.out.println("2. Open a capsule");

System.out.println("3. Check inventory");

System.out.println("4. Refill vending machine");

System.out.println("5. Exit");

System.out.print("Choose an option: ");

// read user input

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String prompt = br.readLine();

if (prompt.contains("1")) {

Object o = turnKnobOnVendingMachine(inventory, vm);

if (o != null) {

unopenedCapsules.add((Capsule) o);

}

} else if (prompt.contains("2")) {

openCapsule(inventory, unopenedCapsules);

} else if (prompt.contains("3")) {

checkInventory(inventory);

} else if (prompt.contains("4")) {

refill(vm);

} else if (prompt.contains("5")) {

System.out.println("Good bye!");

break;

}

}

}

}

### Customer.java

package com.tom.VendingMachine;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.List;

public class Customer implements Person {

public Customer() { }

public Object turnKnobOnVendingMachine(List<Describable> inventory, VendingMachine vendingMachine) {

Object o = vendingMachine.turnKnob();

if (o != null) {

Describable d = (Describable) o;

System.out.println("\nYou turn the knob. The machine cranks out a "

+ d.getDescription());

inventory.add(d);

} else {

System.out.println("\nYou turn the knob. The vending machine is empty!");

System.out.println("The owner needs to refill it.");

}

return o;

}

public void openCapsule(List<Describable> inventory, List<Capsule> unopenedCapsules) {

if (unopenedCapsules.isEmpty()) {

System.out.println("\nThere are no unopened capsules");

} else {

Capsule capsule = unopenedCapsules.remove(0);

Describable capsuleDesc = (Describable) capsule;

Describable content = capsule.open();

System.out.println("\nYou open a "+capsuleDesc.getDescription()

+". Inside is a "+ content.getDescription());

inventory.remove(capsuleDesc);

if (content instanceof Content) {

for (Describable d : (Content) content) {

inventory.add(d);

}

} else {

inventory.add(content);

}

}

}

public void checkInventory(List<Describable> inventory) {

System.out.println("\nYour inventory:");

for (Describable d : inventory) {

System.out.println(" "+d.getDescription());

}

}

public void interact(VendingMachine vm) throws Exception {

List<Describable> inventory = new ArrayList<Describable>();

List<Capsule> unopenedCapsules = new ArrayList<Capsule>();

System.out.println("You are a customer.");

while (true) {

System.out.println();

System.out.println("What would you like to do?");

System.out.println("1. Turn knob on vending machine");

System.out.println("2. Open a capsule");

System.out.println("3. Check inventory");

System.out.println("4. Exit");

System.out.print("Choose an option: ");

// read user input

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

String prompt = br.readLine();

if (prompt.contains("1")) {

Object o = turnKnobOnVendingMachine(inventory, vm);

if (o != null) {

unopenedCapsules.add((Capsule) o);

}

} else if (prompt.contains("2")) {

openCapsule(inventory, unopenedCapsules);

} else if (prompt.contains("3")) {

checkInventory(inventory);

} else if (prompt.contains("4")) {

System.out.println("Good bye!");

break;

}

}

}

}

### VendingMachine.java

package com.tom.VendingMachine;

public interface VendingMachine {

Object turnKnob();

void refill();

}

### CapsuleVendingMachine.java

package com.tom.VendingMachine;

import java.util.List;

public class CapsuleVendingMachine implements VendingMachine {

private DAO dao;

private CapsuleFactory capsuleFactory;

public void setDao(DAO dao) {

this.dao = dao;

}

public void setCapsuleFactory(CapsuleFactory capsuleFactory) {

this.capsuleFactory = capsuleFactory;

}

public Object turnKnob() {

List<Object> lo = dao.getlist();

if (lo.isEmpty())

return null;

Object o = lo.remove(0);

dao.remove(o);

return o;

}

public void refill() {

List<Object> lo = dao.getlist();

for (int i=lo.size(); i<10; i++) {

Object o = capsuleFactory.makeRandom();

dao.add(o);

}

}

}

### CapsuleFactory.java

package com.tom.VendingMachine;

public interface CapsuleFactory {

Object makeRandom();

}

### MiniFigurineCapsuleFactory.java

package com.tom.VendingMachine;

import java.util.HashMap;

import java.util.Map;

import org.springframework.beans.BeansException;

import org.springframework.context.ApplicationContext;

import org.springframework.context.ApplicationContextAware;

public class MiniFigurineCapsuleFactory implements CapsuleFactory, ApplicationContextAware {

private ApplicationContext context;

public Object makeRandom() {

Map<Integer, String> map = new HashMap<Integer, String>();

map.put(1, "red capsule");

map.put(2, "green capsule");

map.put(3, "blue capsule");

int i = (int) (Math.random() \* 3 + 1);

return context.getBean(map.get(i));

}

public void setApplicationContext(ApplicationContext arg0)

throws BeansException {

context=arg0;

}

}

### StickerCapsuleFactory.java

package com.tom.VendingMachine;

import java.util.HashMap;

import java.util.Map;

import org.springframework.beans.BeansException;

import org.springframework.context.ApplicationContext;

import org.springframework.context.ApplicationContextAware;

public class StickerCapsuleFactory implements CapsuleFactory, ApplicationContextAware {

private ApplicationContext context;

public Object makeRandom() {

Map<Integer, String> map = new HashMap<Integer, String>();

map.put(1, "purple capsule");

map.put(2, "orange capsule");

map.put(3, "yellow capsule");

int i = (int) (Math.random() \* 3 + 1);

return context.getBean(map.get(i));

}

public void setApplicationContext(ApplicationContext arg0)

throws BeansException {

context=arg0;

}

}

### DAO.java

package com.tom.VendingMachine;

import java.util.List;

public interface DAO {

Object add(Object t);

void remove(Object t);

void update(Object t);

List<Object> getlist();

}

### HibernateDAO.java

package com.tom.VendingMachine;

import java.util.List;

import org.hibernate.HibernateException;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

public abstract class HibernateDAO implements DAO {

private SessionFactory sessionFactory;

private Class type;

public HibernateDAO(Class type) {

this.type = type;

}

public void setSessionFactory(SessionFactory sessionFactory) {

this.sessionFactory = sessionFactory;

}

public SessionFactory getSessionFactory() {

return this.sessionFactory;

}

public Object add(Object t) {

Session session = sessionFactory.openSession();

Transaction tx = session.beginTransaction();

try

{

session.save(t);

tx.commit();

} catch (Exception e) {

tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

return t;

}

public void remove(Object t) {

Session session = sessionFactory.openSession();

Transaction tx = session.beginTransaction();

try

{

session.delete(t);

tx.commit();

} catch (HibernateException e) {

tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

}

public void update(Object t) {

Session session = sessionFactory.openSession();

Transaction tx = session.beginTransaction();

try

{

session.update(t);

tx.commit();

} catch (Exception e) {

tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

}

public List<Object> getlist() {

Session session = getSessionFactory().openSession();

Transaction tx = session.beginTransaction();

List<Object> lt = null;

try {

lt = session.createCriteria(type).list();

tx.commit();

} catch (Exception e) {

tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

return lt;

}

}

### MiniFigurineCapsuleDAO.java

package com.tom.VendingMachine;

public class MiniFigurineCapsuleDAO extends HibernateDAO {

public MiniFigurineCapsuleDAO() {

super(MiniFigurineCapsule.class);

}

}

### StickerCapsuleDAO.java

package com.tom.VendingMachine;

public class StickerCapsuleDAO extends HibernateDAO {

public StickerCapsuleDAO( ) {

super(StickerCapsule.class);

}

}

### Capsule.java

package com.tom.VendingMachine;

public interface Capsule {

Describable open();

}

### ColoredToyCapsule.java

package com.tom.VendingMachine;

public abstract class ColoredToyCapsule implements Describable, Capsule {

private int capsuleId;

private String color;

public void setCapsuleId (int id) {

capsuleId = id;

}

public int getCapsuleId () {

return capsuleId;

}

public void setColor(String color) {

this.color = color;

}

public String getColor() {

return this.color;

}

public abstract Describable open();

public String getDescription() {

return getColor() + " capsule";

}

}

### MiniFigurineCapsule.java

package com.tom.VendingMachine;

public class MiniFigurineCapsule extends ColoredToyCapsule {

private MiniFigurine figurine;

public void setFigurine(MiniFigurine figurine) {

this.figurine = figurine;

}

public MiniFigurine getFigurine() {

return figurine;

}

@Override

public Describable open() {

return figurine;

}

}

### StickerCapsule.java

package com.tom.VendingMachine;

import java.util.List;

public class StickerCapsule extends ColoredToyCapsule {

private List<Sticker> stickers;

public void setStickers(List<Sticker> stickers) {

this.stickers = stickers;

}

public List<Sticker> getStickers () {

return this.stickers;

}

public Describable open() {

Content content = new Content();

content.setDescription("few stickers");

content.addAll(stickers);

return content;

}

}

### Content.java

package com.tom.VendingMachine;

import java.util.ArrayList;

public class Content extends ArrayList<Describable> implements Describable {

private String description;

public void setDescription(String description) {

this.description = description;

}

public String getDescription() {

return this.description;

}

}

### Describable.java

package com.tom.VendingMachine;

public interface Describable {

String getDescription();

}

### MiniFigurine.java

package com.tom.VendingMachine;

public class MiniFigurine implements Describable {

private int figurineId;

private String name;

public void setFigurineId (int id) {

figurineId = id;

}

public int getFigurineId () {

return figurineId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getDescription() {

return getName() + " mini figurine";

}

}

### Sticker.java

package com.tom.VendingMachine;

public class Sticker implements Describable {

private int stickerId;

private String name;

public void setName (String name) {

this.name = name;

}

public String getName () {

return this.name;

}

public void setStickerId (int id) {

stickerId = id;

}

public int getStickerId () {

return this.stickerId;

}

public String getDescription() {

return name + " sticker";

}

}

## Resources

### beans.xml

<?xml version="1.0" encoding="UTF-8"?>

<beans xmlns="http://www.springframework.org/schema/beans"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

<bean id="dataSource"

class="org.springframework.jdbc.datasource.DriverManagerDataSource">

<property name="driverClassName" value="oracle.jdbc.driver.OracleDriver" />

<property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />

<property name="username" value="tom" />

<property name="password" value="123" />

</bean>

<bean id="sessionFactory" class="org.springframework.orm.hibernate4.LocalSessionFactoryBean">

<property name="dataSource" ref="dataSource"/>

<property name="mappingResources">

<list>

<value>/MiniFigurine.hbm.xml</value>

<value>/MiniFigurineCapsule.hbm.xml</value>

<value>/Sticker.hbm.xml</value>

<value>/StickerCapsule.hbm.xml</value>

</list>

</property>

<property name="hibernateProperties">

<props>

<prop key="hibernate.dialect">org.hibernate.dialect.Oracle10gDialect</prop>

<prop key="hibernate.show\_sql">true</prop>

</props>

</property>

</bean>

<bean id="red capsule" scope="prototype" class="com.tom.VendingMachine.MiniFigurineCapsule">

<property name="color" value="red" />

<property name="figurine" ref="batman figurine" />

</bean>

<bean id="batman figurine" scope="prototype" class="com.tom.VendingMachine.MiniFigurine">

<property name="name" value="batman" />

</bean>

<bean id="blue capsule" scope="prototype" class="com.tom.VendingMachine.MiniFigurineCapsule">

<property name="color" value="blue" />

<property name="figurine" ref="superman figurine" />

</bean>

<bean id="superman figurine" scope="prototype" class="com.tom.VendingMachine.MiniFigurine">

<property name="name" value="superman" />

</bean>

<bean id="green capsule" scope="prototype" class="com.tom.VendingMachine.MiniFigurineCapsule">

<property name="color" value="green" />

<property name="figurine" ref="spiderman figurine" />

</bean>

<bean id="spiderman figurine" scope="prototype" class="com.tom.VendingMachine.MiniFigurine">

<property name="name" value="spiderman" />

</bean>

<bean id="figurine vending machine" class="com.tom.VendingMachine.CapsuleVendingMachine">

<property name="dao" ref="figurine capsule dao" />

<property name="capsuleFactory" ref="MiniFigurineCapsuleFactory" />

</bean>

<bean id="figurine capsule dao" class="com.tom.VendingMachine.MiniFigurineCapsuleDAO">

<property name="sessionFactory" ref="sessionFactory" />

</bean>

<bean id="MiniFigurineCapsuleFactory" class="com.tom.VendingMachine.MiniFigurineCapsuleFactory">

</bean>

<bean id="mickey mouse sticker" class="com.tom.VendingMachine.Sticker" scope="prototype">

<property name="name" value="mickey mouse" />

</bean>

<bean id="donald duck sticker" class="com.tom.VendingMachine.Sticker" scope="prototype">

<property name="name" value="donald duck" />

</bean>

<bean id="goofy sticker" class="com.tom.VendingMachine.Sticker" scope="prototype">

<property name="name" value="goofy" />

</bean>

<bean id="purple capsule" scope="prototype" class="com.tom.VendingMachine.StickerCapsule">

<property name="color" value="purple" />

<property name="stickers">

<list value-type="com.tom.VendingMachine.Sticker">

<ref bean="mickey mouse sticker"/>

<ref bean="donald duck sticker" />

</list>

</property>

</bean>

<bean id="yellow capsule" scope="prototype" class="com.tom.VendingMachine.StickerCapsule">

<property name="color" value="yellow" />

<property name="stickers">

<list value-type="com.tom.VendingMachine.Sticker">

<ref bean="mickey mouse sticker"/>

<ref bean="goofy sticker" />

</list>

</property>

</bean>

<bean id="orange capsule" scope="prototype" class="com.tom.VendingMachine.StickerCapsule">

<property name="color" value="orange" />

<property name="stickers">

<list value-type="com.tom.VendingMachine.Sticker">

<ref bean="goofy sticker"/>

<ref bean="donald duck sticker" />

</list>

</property>

</bean>

<bean id="sticker vending machine" class="com.tom.VendingMachine.CapsuleVendingMachine">

<property name="dao" ref="sticker capsule dao" />

<property name="capsuleFactory" ref="StickerCapsuleFactory" />

</bean>

<bean id="sticker capsule dao" class="com.tom.VendingMachine.StickerCapsuleDAO">

<property name="sessionFactory" ref="sessionFactory" />

</bean>

<bean id="StickerCapsuleFactory" class="com.tom.VendingMachine.StickerCapsuleFactory">

</bean>

</beans>

### MiniFigurine.hbm

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<!-- Generated Feb 7, 2014 10:47:36 PM by Hibernate Tools 3.4.0.CR1 -->

<hibernate-mapping>

<class name="com.tom.VendingMachine.MiniFigurine" table="MINIFIGURINE">

<id name="figurineId" type="int">

<column name="FIGURINEID" />

<generator class="increment" />

</id>

<property name="name" column="NAME" />

</class>

</hibernate-mapping>

### MiniFigurineCapsule.hbm

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<!-- Generated Feb 7, 2014 10:47:36 PM by Hibernate Tools 3.4.0.CR1 -->

<hibernate-mapping>

<class name="com.tom.VendingMachine.MiniFigurineCapsule" table="MINIFIGURINECAPSULE">

<id name="capsuleId" type="int">

<column name="CAPSULEID" />

<generator class="increment" />

</id>

<many-to-one name="figurine" class="com.tom.VendingMachine.MiniFigurine"

fetch="join" cascade="all">

<column name="FIGURINEID" />

</many-to-one>

<property name="color" column="COLOR" />

</class>

</hibernate-mapping>

### Sticker.hbm

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<!-- Generated Feb 7, 2014 10:47:36 PM by Hibernate Tools 3.4.0.CR1 -->

<hibernate-mapping>

<class name="com.tom.VendingMachine.Sticker" table="STICKER">

<id name="stickerId" type="int">

<column name="STICKERID" />

<generator class="increment" />

</id>

<property name="name" column="NAME" />

</class>

</hibernate-mapping>

### StickerCapsule.hbm

<?xml version="1.0"?>

<!DOCTYPE hibernate-mapping PUBLIC "-//Hibernate/Hibernate Mapping DTD 3.0//EN"

"http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<!-- Generated Feb 7, 2014 10:47:36 PM by Hibernate Tools 3.4.0.CR1 -->

<hibernate-mapping>

<class name="com.tom.VendingMachine.StickerCapsule" table="STICKERCAPSULE">

<id name="capsuleId" type="int">

<column name="CAPSULEID" />

<generator class="increment" />

</id>

<bag name="stickers" cascade="all">

<key column="CAPSULEID" />

<one-to-many class="com.tom.VendingMachine.Sticker"/>

</bag>

<property name="color" column="COLOR" />

</class>

</hibernate-mapping>