Spring Day 8

Address.java

```java
package com.carshop.domain;

import java.io.Serializable;

import lombok.AllArgsConstructor;
import lombok.Data;

@SuppressWarnings("serial")
@Data
@AllArgsConstructor
public class Address implements Serializable {

  private String detailName;
  private String addressName;
  private String country;
  private String zipCode;

  @Override
  public boolean equals(Object obj) {
    if (this == obj)
      return true;
    if (obj == null)
      return false;
    if (getClass() != obj.getClass())
      return false;
    Address other = (Address) obj;
    if (addressName == null) {
      if (other.addressName != null)
        return false;
    } else if (!addressName.equals(other.addressName))
      return false;
    if (country == null) {
      if (other.country != null)
        return false;
    } else if (!country.equals(other.country))
      return false;
    if (detailName == null) {
      if (other.detailName != null)
        return false;
    } else if (!detailName.equals(other.detailName))
      return false;
    if (zipCode == null) {
      if (other.zipCode != null)
        return false;
    } else if (!zipCode.equals(other.zipCode))
      return false;
    return true;
  }

  @Override
  public int hashCode() {
    final int prime = 31;
    int result = 1;
    result = prime * result + ((addressName == null) ? 0 : addressName.hashCode());
    result = prime * result + ((country == null) ? 0 : country.hashCode());
    result = prime * result + ((detailName == null) ? 0 : detailName.hashCode());
    result = prime * result + ((zipCode == null) ? 0 : zipCode.hashCode());
    return result;
  }
}
```

Customer.java

```java
package com.carshop.domain;

import java.io.Serializable;

import lombok.Getter;
import lombok.Setter;

@SuppressWarnings("serial")
public class Customer implements Serializable {
  @Getter@Setter
  private String customerId;

  @Getter@Setter
  private String name;

  private Address address; // 고객주소 객체
  private String phone;

  public Customer() { // 기본 생성자
    this.address = new Address();
  }

  public Customer(String customerId, String name) {
    this();
    this.customerId = customerId;
    this.name = name;
  }

  @Override
  public boolean equals(Object obj) {
    if (this == obj)
      return true;
    if (obj == null)
      return false;
    if (getClass() != obj.getClass())
      return false;
    Customer other = (Customer) obj;
    if (customerId == null) {
      if (other.customerId != null)
        return false;
    } else if (!customerId.equals(other.customerId))
      return false;
    return true;
  }

  @Override
  public int hashCode() {
    final int prime = 31;
    int result = 1;
    result = prime * result + ((customerId == null) ? 0 : customerId.hashCode());
    return result;
  }


}
```

Shipping.java

```
package com.carshop.domain;

import java.io.Serializable;
import java.util.Date;

import org.springframework.format.annotation.DateTimeFormat;

import lombok.Data;

@SuppressWarnings("serial")
@Data
public class Shipping implements Serializable {

  private String name;

  @DateTimeFormat(pattern="yyyy/MM/dd")
  private Date date;

  private Address address;

  public Shipping() {
    this.address = new Address();
  }
}
```

Order.java

```
package com.carshop.domain;

import java.io.Serializable;
import com.carshop.controller.Cart;

import lombok.Data;

@SuppressWarnings("serial")
@Data
public class Order implements Serializable {

  private Long orderId;
  private Cart cart; // 장바구니 객체
  private Customer customer; // 고객 객체
  private Shipping shipping; // 배송 객체

  public Order() {
    this.customer = new Customer();
    this.shipping = new Shipping();
  }
}
```

OrderRepository.java

```
package com.carshop.domain;

public interface OrderRepository {
  Long saveOrder(Order order);
}
```

OrderRepositoryImpl.java

```
package com.carshop.domain;

import java.util.HashMap;
import java.util.Map;

import org.springframework.stereotype.Repository;

@Repository
public class OrderRepositoryImpl implements OrderRepository {

  private Map<Long, Order> listOfOrders;
  private long nextOrderId;

  public OrderRepositoryImpl() {
    listOfOrders = new HashMap<Long, Order>();
    nextOrderId = 2000;
  }

  @Override
  public Long saveOrder(Order order) {
    order.setOrderId(getNextOrderId());
    listOfOrders.put(order.getOrderId(), order);
    return order.getOrderId();
  }

  private synchronized long getNextOrderId() {
    return nextOrderId++;
  }
}
```

OrderService.java

```
package com.carshop.domain;

public interface OrderService {
  void confirmOrder(String cid, long quantity);
  Long saveOrder(Order order);
}
```

OrderServiceImpl.java

```
package com.carshop.domain;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.carshop.controller.CartService;

@Service
public class OrderServiceImpl implements OrderService {

  @Autowired
  private OrderRepository orderRepository;

  @Autowired
  private CartService cartService;

  @Override
  public void confirmOrder(String cid, long quantity) {
    // 추후에는 판매가 이루어지면 재고량에서 1개를 빼주는 계산 처리

  }

  @Override
  public Long saveOrder(Order order) {
    Long orderId = orderRepository.saveOrder(order);
    cartService.delete(order.getCart().getCartId());
    return orderId;
    //주문이 처리가 되어 기존 장바구니는 비워준다
  }

}
```

OrderController.java

```
package com.carshop.domain;

import org.springframework.stereotype.Controller;

@Controller
public class OrderController {
  //사용X
}
```

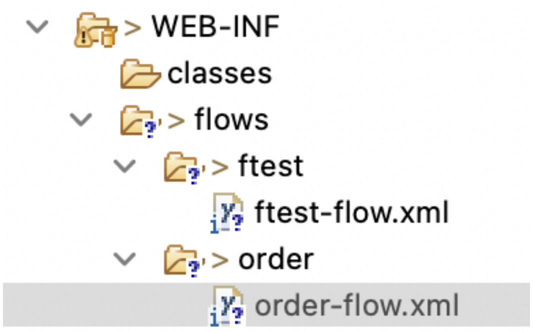
pom.xml

```
<!-- spring-webflow -->
<dependency>
    <groupId>org.springframework.webflow</groupId>
    <artifactId>spring-webflow</artifactId>
    <version>2.5.1.RELEASE</version>
</dependency>
```

servlet-context.xml

```
xmlns:webflow="http://www.springframework.org/schema/webflow-config"

http://www.springframework.org/schema/webflow-config http://www.springframework.org/schema/webflow-config/spring-webflow-config.xsd

<webflow:flow-registry id="flowRegistry">
  <webflow:flow-location
    path="/WEB-INF/flows/order/order-flow.xml" id="order" />
  <webflow:flow-location
    path="/WEB-INF/flows/ftest/ftest-flow.xml" id="ftest" />
</webflow:flow-registry>

<webflow:flow-executor id="flowExecutor"
  flow-registry="flowRegistry" />

<beans:bean id="flowHandlerMapping"
  class="org.springframework.webflow.mvc.servlet.FlowHandlerMapping">
  <beans:property name="flowRegistry" ref="flowRegistry" />
</beans:bean>

<beans:bean id="flowHandlerAdapter"
```

```
  class="org.springframework.webflow.mvc.servlet.FlowHandlerAdapter">
  <beans:property name="flowExecutor" ref="flowExecutor" />
</beans:bean>
```


WEB-INF
classes
flows
ftest
ftest-flow.xml
order
order-flow.xml

Ftest.java

```
package com.carshop.domain;

import java.io.Serializable;

@SuppressWarnings("serial")
public class Ftest implements Serializable{
  //명목상 지정하는 클래스
}
```

ftest-flow.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<flow xmlns="http://www.springframework.org/schema/webflow"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/webflow
      http://www.springframework.org/schema/webflow/spring-webflow.xsd">
  <var name="ftest" class="com.carshop.domain.Ftest" />
  <view-state id="ftest1">
    <transition to="ftest2" />
  </view-state>
</flow>
```

xml 추가 기능

```
<end-state id="">

</end-state>

<action-state id="">
  <evaluate expression=""/>
</action-state>

<global-transitions>
  <transition>

  </transition>
</global-transitions>
```

이동 버튼

```
<div>
  <form:form>
    <input type="submit" value="이동" name="_eventId_ftest2" />
    <button name="_eventId_cancel">취소</button>
  </form:form>
</div>
```

CarShop webflow 적용

order-flow.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<flow xmlns="http://www.springframework.org/schema/webflow"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/webflow
      http://www.springframework.org/schema/webflow/spring-webflow.xsd">

  <var name="order" class="com.carshop.domain.Order" />

  <!-- 카트에 담긴 정보 -->
  <action-state id="addCartToOrder">
    <evaluate
      expression="cartServiceImpl.validateCart(requestParameters.cartId)"
      result="order.cart" />
    <transition to="orderCartWarning"
      on-exception="com.carshop.exception.CartException" />
    <transition to="orderCustomerInfo" />
  </action-state>

  <!-- 주문자 정보 -->
  <view-state id="orderCustomerInfo" model="order">
    <transition on="customerInfo" to="orderShippingInfo" />
  </view-state>

  <!-- 배송 받는 사람 정보 -->
  <view-state id="orderShippingInfo" model="order">
    <transition on="shippingInfo" to="orderConfirmation" />
    <transition on="backToCustomerInfo" to="orderCustomerInfo" />
  </view-state>

  <!-- 승인 -->
  <view-state id="orderConfirmation">
    <transition on="orderConfirmed" to="confirmOrder" />
    <transition on="backToShippingInfo" to="orderShippingInfo" />
  </view-state>

  <action-state id="confirmOrder">
    <evaluate expression="orderServiceImpl.saveOrder(order)"
      result="order.orderId" />
    <transition to="orderFinished" />
  </action-state>

  <view-state id="orderFinished" model="order">
    <transition to="endState" />
  </view-state>

  <end-state id="endState"></end-state>

  <view-state id="orderCartWarning">
    <transition to="endState" />
  </view-state>

  <global-transitions>
    <transition on="cancel" to="endState" />
  </global-transitions>

</flow>
```