네이버 체크아웃 가맹점 연동 가이드

저작권

문서 정보

문서 개요

이 문서는 네이버 체크아웃 가맹점이 구매 및 상품 정보, 찜 정보를 네이버 체크아웃과 연동하는 방법을 설명합니다.

독자

이 문서의 독자는 네이버 체크아웃 가맹점 사이트 개발자입니다.

문의처

이 문서의 내용에 오류가 있거나 내용과 관련한 의문 사항이 있으면 아래 연락처에 문의합니다.

기술지원 담당 : 김만복 (사내전화 : 031-600-6619, 메일 : mandory201@nhn.com)

기술지원 담당 : 김은정 (사내전화 : 031-600-6618, 메일 : eunjungkim@nhn.com)

문서 버전 및 이력

버전	일자	이력사항
1.0	2009-07-03	문서 배포
1.1	2009-08-05	문서 업데이트
1.2	2009-09-10	예제 오류 수정
1.3	2010-02-11	버튼 삽입 방식 변경, 배송 정보 연동 추가

표기 규칙

UI 메시지 표기

이 문서에서 메뉴 명과 창 이름, 버튼 이름 등 UI(User Interface) 메시지를 표현할 때는 대괄호를 사용해서 '[구매하기]'와 같이 표기합니다. 단, 소스 코드에 사용된 기호는 이 표기 규칙에 해당하지 않습니다.

소스 코드 표기

이 문서에서 소스 코드는 회색 바탕에 검정색 글씨로 표기합니다.

```
COPYDATASTRUCT st;
st.dwData = PURPLE_OUTBOUND_ENDING;
st.cbData = sizeof(pp);
st.lpData = &pp;
::SendMes(GetTargetHwnd(), WM_COPYDATA, (WPARAM)this->m_hWnd, (LPARAM)&st);
```

목차

표 및 그림 목록

표 목록

그림 목록

1. 개요

이 장에서는 네이버 체크아웃 구매 정보 및 상품 정보 연동의 기본 개념을 설명한다.

1.1 네이버 체크아웃 연동 개요

네이버 체크아웃 가맹점 가입 심사 후 승인 완료되면, 가맹점은 네이버에서 제공하는 가이드에 따라 구매 정보 및 상품 정보 연동 시스템을 개발하여, 이용자가 가맹점 사이트에서 네이버 체크아웃으로 구매할 수 있는 환경을 만들어야 한다.

네이버 체크아웃 버튼을 가맹점 사이트의 상품 상세 페이지와 장바구니 페이지에 삽입하고, 이용자가 네이버 체크아웃 버튼을 클릭하면, 가맹점은 구매 정보 또는 찜 정보를 네이버에 등록한다. 네이버 체크아웃에서 상품 정보를 요청하면 상품 정보를 전송한다.

가맹점이 사용하는 PG사와 네이버 담당자가 연동 시스템을 테스트하고, 연동 테스트가 완료되면 가맹점은 네이버 체크아웃센터 가입 완료와 동시에 네이버 체크아웃센터 서비스를 이용할 수 있다.

1.2 네이버 체크아웃 연동 과정

1.2.1 전체 연동 과정

가맹점은 다음과 같은 과정을 통해 구매 정보 및 상품 정보를 네이버 체크아웃과 연동시킨다.

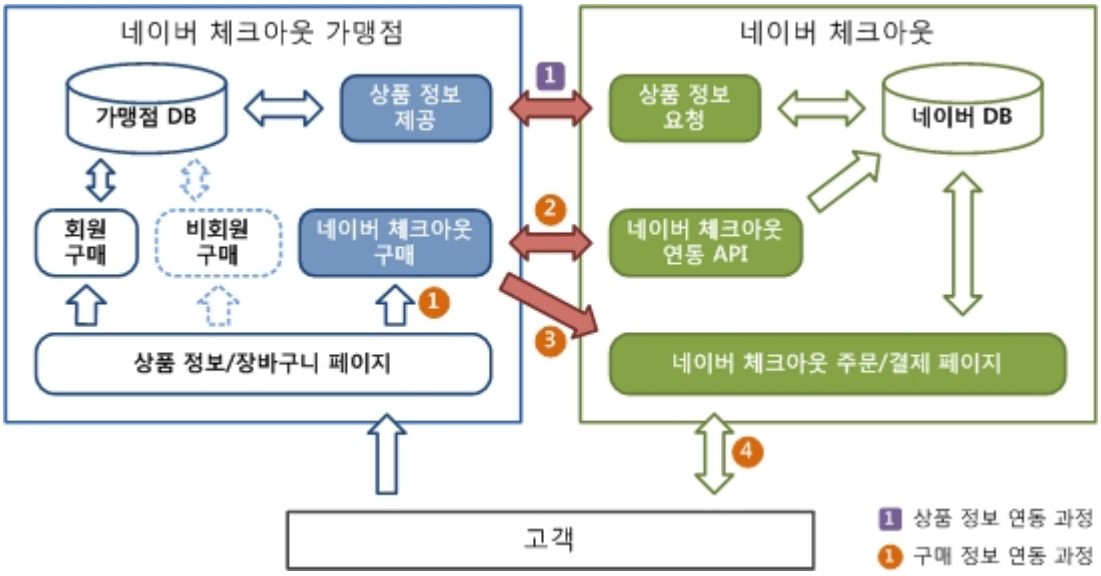


그림 1-1 네이버 체크아웃 가맹점 연동 구조도

1.2.2 구매 정보 연동

"그림 1-1"에서 주황색 원의 숫자는 구매 정보 연동 과정을 표시한 것이다. 구매 정보 연동 과정은 다음과 같다.

1. 이용자가 상품 정보 페이지나 장바구니 페이지에서 네이버 체크아웃 [구매하기] 버튼을 클릭하면, 해당 이벤트는 가맹점 사이트의 페이지로 전달된다.
2. 이벤트를 받은 페이지에서 네이버 체크아웃으로 주문 정보를 등록하고 주문 ID를 받는다.
3. 네이버 체크아웃 주문서 페이지 팝업 창을 생성하고 주문 ID와 상점 ID, 총 주문 금액을 전송한다.
4. 이용자는 네이버 체크아웃 주문/결제 페이지에서 결제한다.

1.2.3 상품 정보 연동

"그림 1-1"에서 보라색 사각형의 숫자는 상품 정보 연동 과정을 표시한 것이다. 상품 정보 연동 과정은 다음과 같다.

1. 네이버 체크아웃이 상품의 정보를 요청할 경우, 가맹점은 상품 정보를 전송한다.

2. 네이버 체크아웃 버튼

이 장에서는 네이버 체크아웃 버튼의 삽입 방법과 네이버 체크아웃 [구매하기] 버튼 클릭 시 이벤트 처리 방법을 설명한다.

2.1 네이버 체크아웃 버튼 삽입

네이버 체크아웃 가맹점 가입 승인이 완료되면, 네이버 체크아웃은 상점 ID, 연동 인증키와 함께 네이버 체크아웃 버튼을 위한 버튼 인증키를 제공한다.

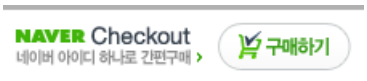
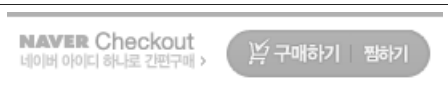
상품 상세 페이지에 삽입하는 네이버 체크아웃 버튼 모음에는 구매하기 버튼과 찜하기 버튼이 있고, 장바구니 페이지에 삽입하는 버튼 모음에는 구매하기 버튼만 있다. 버튼 모음은 종류와 색을 선택할 수 있다.

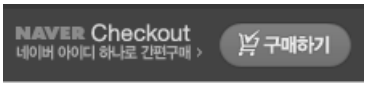
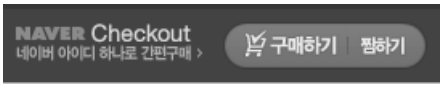
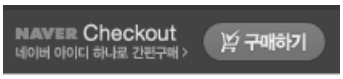
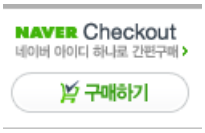
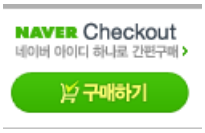
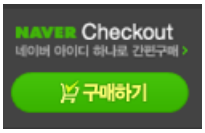
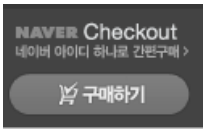
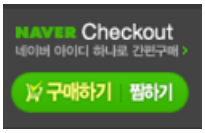
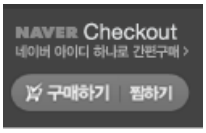
네이버 체크아웃으로 구매가 불가능한 취급 불가 상품(설치 상품, 주문 제작 상품, 해외 배송 상품 등)에는 비활성 버튼 모음을 삽입해야 한다.

다음의 경우에는 네이버 체크아웃 버튼 모음을 삽입하지 않을 수 있다.

- 이용자가 가맹점 사이트에 로그인 상태인 경우
- 상품 품절로 인해 판매가 불가능한 경우
  - 가맹점 사이트의 구매 버튼과 네이버 체크아웃 버튼 모음 모두 삽입하지 않는다. (권장)
  - 가맹점 사이트의 구매 버튼과 비활성 네이버 체크아웃 버튼 모음을 삽입한다.

표 2-1 네이버 체크아웃 버튼 모음 종류

종류	색	버튼 수 (크기)	활성	비활성
A	1	1 (236X46)		
		2 (285X46)		
B	1	1 (214X40)		
		2 (261X40)		
C	1	1 (225X49)		
		2 (272X49)		
	2	1 (225X49)		
		2 (272X49)		

	3	1 (225X49)		
		2 (272X49)		
D	1	1 (210X43)		
		2 (251X43)		
	2	1 (210X43)		
		2 (251X43)		
	3	1 (210X43)		
		2 (251X43)		
E	1	1 (124X77)		
		2 (124X77)		
	2	1 (124X77)		
		2 (124X77)		
	3	1 (124X77)		
		2 (124X77)		

네이버 체크아웃 버튼 모음 이미지 삽입을 위한 소스 코드는 다음과 같다.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">
<title>네이버 체크아웃 버튼 사용법</title>
<script type="text/javascript"
src="http://checkout.naver.com/customer/js/checkoutButton.js" charset="UTF-8"></script>
</head>
<body>

<script type="text/javascript" >//<![CDATA[
 nhn.CheckoutButton.apply({
        BUTTON_KEY: "버튼 인증 키", // 체크아웃에서 제공받은 버튼 인증 키 입력
        TYPE: "E", // 버튼 모음 종류 설정
        COLOR: 1, // 버튼 모음의 색 설정
        COUNT: 2, // 버튼 개수 설정. 구매하기 버튼만 있으면(장바구니 페이지) 1, 찜하기 버튼도
있으면(상품 상세 페이지) 2를 입력.
        ENABLE: "Y", // 품절 등의 이유로 버튼 모음을 비활성화할 때에는 "N" 입력
        "":""
 });
//]]></script>

</body>
</html>
```

2.2 네이버 체크아웃 버튼 동작

이용자가 네이버 체크아웃 [구매하기] 버튼을 클릭하면, 가맹점은 이용자가 옵션, 배송료 지불 방법 등의 선택 사항을 모두 선택했는지 확인하고 그렇지 않은 경우 팝업 경고를 표시한다. 이용자가 선택 사항을 모두 선택했거나 이용자 선택 사항이 없으면, 주문 정보 등록을 위한 페이지로 이동한다.

이용자가 비활성화된 네이버 체크아웃 [구매하기] 버튼을 클릭하면, 가맹점은 팝업 경고를 표시한다.

팝업 경고의 문구는 다음과 같다.

표 2-2 [구매하기] 버튼 클릭 시 오류 상황에 따른 팝업 경고 문구

오류 상황	팝업 경고 문구
옵션이 있는 상품의 옵션 미선택	상품 옵션을 선택해 주세요.
복수 개의 옵션이 있는 상품의 일부 옵션 미선택	상품 옵션을 모두 선택해 주세요.
배송료 지불 방법 미선택	배송비를 선택해 주세요.
기타 선택 사항 미선택	{선택 사항}을 선택한 후 주문을 완료해 주세요.
네이버 체크아웃 버튼 모음 비활성화	죄송합니다. NAVER Checkout 으로 구매가 불가한 상품입니다

다음은 상품 상세 페이지의 예이다.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=euc-kr">
<title>네이버 체크아웃 버튼 사용법</title>
<script type="text/javascript"
src="http://checkout.naver.com/customer/js/checkoutButton.js" charset="UTF-8"></script>
<script type="text/javascript" >//<![CDATA[
function checkOption(selectBox, nonSelectedIndex, optionName)
{
    if (selectBox.selectedIndex == nonSelectedIndex) {
        alert("상품 옵션을 선택해 주세요. (" + optionName + ")");
        return false;
    }
    return true;
}

function checkShippingPrice(radio)
{
    var len = radio.length;
    if (len == undefined) {
        if (radio.checked) {
            return true;
        }
    } else {
        for(var i = 0; i < len; i++) {
            if(radio[i].checked) {
                    return true;
            }
        }
    }
    alert("배송비를 선택해 주세요.");
    return false;
```

```
}

function buy_nc(url)
{
    var check = checkOption(document.getElementById("opt").color, 0, "색상") &&
    checkOption(document.getElementById("opt").size, 0, "크기") &&
    checkShippingPrice(document.getElementById("opt").spt);

    if ( check ) {
        //네이버 체크아웃으로 주문 정보를 등록하는 가맹점 페이지로 이동.
        //해당 페이지에서 주문 정보 등록 후 네이버 체크아웃 주문서 페이지로 이동.

        location.href=url;
    }

    return false;
}

function wishlist_nc(url)
{
    // 네이버 체크아웃으로 찜 정보를 등록하는 가맹점 페이지 팝업 창 생성.
    // 해당 페이지에서 찜 정보 등록 후 네이버 체크아웃 찜 페이지로 이동.
    window.open(url,"","scrollbars=yes,width=400,height=267");
    return false;
}

function not_buy_nc()
{
    alert("죄송합니다. NAVER Checkout으로 구매가 불가한 상품입니다.");
    return false;
}

//]]></script>
</head>

<body>

<div class="first">
<!-- 상품정보 -->
상품명 : 옷 <br/>
<form id="opt" action="" >
    색상 :
    <select id="color">
        <option value="no">-- 색상을 선택해 주세요 --</option>
        <option value="red">빨간색</option>
        <option value="red">파란색</option>
    </select><br/>
    크기 :
    <select id="size">
        <option value="no">-- 크기를 선택해 주세요 --</option>
        <option value="L">Large(L)</option>
        <option value="XL">Extra Large(XL)</option>
    </select><br/>
    배송비 :
    <input type="radio" name="spt" value="FREE"/>무료
    <input type="radio" name="spt" value="PAYED"/>선불
    <input type="radio" name="spt" value="ONDELIVERY"/>착불
</form>

<script type="text/javascript" >//<![CDATA[
 nhn.CheckoutButton.apply({
```

```
        BUTTON_KEY:"207994C7-2944-4D98-A24C-387FBE19452C", // 체크아웃에서 할당받은 버튼 KEY
를 입력하세요.
        TYPE: "E", // 템플릿을 확인하시고 원하는 타입의 버튼을 선택
        COLOR: 1, // 버튼의 색 설정
        COUNT: 2, // 버튼 개수 설정. 구매하기 버튼(장바구니 페이지)만 있으면 1, 찜하기 버튼(상품 상세
페이지)과 함께 있으면 2 를 입력한다.
        ENABLE: "Y", // 품절등과 같은 이유에 따라 버튼을 비활성화할 필요가 있을 경우
        BUY_BUTTON_HANDLER: buy_nc, // 구매하기 버튼 이벤트 Handler 함수 등록, 품절인 경우
not_buy_nc 함수 사용
        BUY_BUTTON_LINK_URL:"http://mydomain.com/buy/url/", // 링크 주소 (필요한 경우만 사용)
        WISHLIST_BUTTON_HANDLER:wishlist_nc, // 찜하기 버튼 이벤트 Handler 함수 등록
        WISHLIST_BUTTON_LINK_URL:"http://mydomain.com/wishlist/popup/url/", // 찜하기 팝업
링크 주소
        "":""
 });

//]]></script>

</div>

</body>
</html>
```

3. 네이버 체크아웃 연동

이 장에서는 가맹점이 네이버 체크아웃과 구매 정보, 상품 정보, 찜 정보를 연동하는 방법을 설명하고 PHP, Java, ASP로 작성된 예시를 제공한다.

3.1 구매 정보 연동

3.1.1 네이버 체크아웃 [구매하기] 버튼 클릭

이용자가 상품 정보 페이지나 장바구니 페이지에서 네이버 체크아웃 [구매하기] 버튼을 클릭하면, 가맹점은 이용자 선택 사항이 모두 선택되었는지 확인한다. 이용자 선택 사항이 모두 선택되었거나 이용자 선택 사항이 없으면 주문 정보 등록을 위한 페이지로 이동한다. 주문 정보 등록 페이지로 이동하는 방법은 가맹점이 선택할 수 있으며, 네이버 체크아웃은 Ajax, 페이지 전환, 새 창 띄우기의 순서로 권장한다.

3.1.2 주문 정보 등록

가맹점은 네이버 체크아웃에 HTTPS로 주문 정보를 등록하고 주문 ID를 받는다. 주문 정보 등록시 기본 문자 세트는 UTF-8이고, 다른 문자 세트를 사용하는 경우 UTF-8로 변환하거나, 다른 URL을 이용한다.

표 3-1 주문 정보 등록 URL

환경	문자 세트	URL
테스트	UTF-8	https://test-checkout.naver.com/customer/api/order.nhn
	EUC_KR	https://test-checkout.naver.com/customer/api/CP949/order.nhn
서비스	UTF-8	https://checkout.naver.com/customer/api/order.nhn
	EUC_KR	https://checkout.naver.com/customer/api/CP949/order.nhn

response로 받는 주문 ID는 영문과 숫자로 이루어지며 최대 19자리이다.

request 시 post로 아래와 같은 정보를 전달한다.

표 3-2 주문 정보 내용

항목	필수 여부	설명
SHOP_ID	Y	상점 ID. 네이버 체크아웃에 가입 승인될 때 정해진다.
CERTI_KEY	Y	인증키. 네이버 체크아웃에 가입 승인될 때 정해진다.
ITEM_ID	Y	상품 ID
ITEM_NAME	Y	상품 이름
ITEM_COUNT	Y	상품 주문 개수
ITEM_UPRICE	Y	개별 상품 단가. 0 보다 커야 한다.
ITEM_TPRICE	Y	해당 상품 총 가격. 상품 할인 행사가 있으면, ITEM_COUNT 와 ITEM_UPRICE 를 곱한 값보다 작은 값을 가질 수 있다.
ITEM_OPTION	Y	선택한 옵션 사항

SHIPPING_PRICE	Y	배송료. 무료이면 0, 선불 또는 착불이면 배송료(0 보다 커야 함).
SHIPPING_TYPE	Y	배송료 지불 방법. 무료이면 "FREE", 선불이면 "PAYED", 착불이면 "ONDELIVERY"
TOTAL_PRICE	Y	총 주문 금액. ITEM_TPRICE 의 합과 선불 배송료를 더한 값과 같아야 한다.
BACK_URL	Y	네이버 체크아웃 주문서 페이지에서 [이전페이지]를 클릭했을 때 이동하는 페이지 URL. 상점 메인 화면으로 돌아가거나, 주문을 따로 저장했다면 주문서 페이지로 돌아가게 할 수 있다.
SALES_CODE	N	경로별 매출 코드. 영문자 기준 최대 300 자. 매출 코드가 필요한 경우에 입력한다.
MALL_MANAGE_CODE	N	몰 내부 관리 코드. 영문자 기준 최대 300 자. 내부 관리 코드가 필요한 경우에 입력한다.
RESERVE1	N	예약 필드 1
RESERVE2	N	예약 필드 2
RESERVE3	N	예약 필드 3
RESERVE4	N	예약 필드 4
RESERVE5	N	예약 필드 5

ITEM_ID, ITEM_NAME, ITEM_COUNT, ITEM_UPRICE, ITEM_TPRICE, ITEM_OPTION은 여러 개일 수 있다. 예를 들어, 한 주문에 상품 종류가 두 개라면 각 파라미터는 두 개씩 존재한다. 이 때 각 파라미터에서 상품의 순서를 지켜야 한다.

고유 키(unique key)는 ITEM_ID가 아니라 (ITEM_ID, ITEM_OPTION)의 쌍이다. 따라서 동일한 ITEM_ID 의 상품의 ITEM_OPTION이 다르면 다른 파라미터로 전송해야 한다.

ITEM_OPTION의 값은 실제 주문서 페이지에 그대로 표시된다. 각 쇼핑몰은 이용자가 선택한 옵션 사항을 텍스트로 표시할 수 있다. HTML 태그는 사용할 수 없다. ITEM_OPTION이 여러 종류일 경우 슬래시(/)로 구분하는 것을 권장한다. 예를 들어, 색상은 노랑이고 사이즈는 XL이면 ITEM_OPTION은 "노랑/XL" 또는 "색상:노랑/사이즈:XL"로 표기하는 것을 권장한다.

TOTAL_PRICE는 이용자가 최종 지불하는 총 금액으로서, ITEM_TPRICE의 총합과 같아야 한다. 배송료가 선불("PAYED")이면, ITEM_TPRICE의 총합에 SHIPPING_PRICE를 더한 값과 반드시 같아야 한다.

SALES_CODE는 유입경로 별 매출을 확인하기 위해 각 가맹점이 별도의 매출 코드 관리가 필요한 경우 입력한다.. 네이버 체크아웃의 각 주문에 대한 매출 코드를 출력하는 기능만을 담당하고 별도의 통계 기능 등은 제공하지 않는다.

MALL_MANAGE_CODE는 각 가맹점에서 체크아웃 주문서 호출 시 해당 정보를 자사 DB에 보관할 필요가 있는 경우, 해당 정보를 DB에 보관하고 보관에 대한 KEY 값을 입력한다. 체크아웃 주문 연동 정보가 실제 자사 주문 처리를 위해 필요한 DB와 다르거나 부족할 경우에 사용할 수 있다.

예를 들어 사진 인화 사이트의 경우, 구매자가 업로드한 파일 정보는 네이버 체크아웃 주문서 정보에 입력할 수 없다. 이때, 구매자가 업로드한 파일 정보를 각 가맹점이 내부적으로 저장한 후 그 정보에 대한 KEY 값을 체크아웃 주문 정보에 추가하면, 구매자의 결제 이후 각 주문에 대해 출력된 KEY 값을 이용해 자사 어드민에 저장된 DB와 매핑해서 주문 처리에 사용할 수 있다.

RESERVEn 은 추후 확장을 위한 예약 공간이다.

다음은 PHP로 주문 정보를 등록하는 예이다.

```
<?
//item data를 생성한다.
class ItemStack {
	var $id;
	var $name;
	var $tprice;
	var $uprice;
	var $option;
	var $count;

	//option이 여러 종류라면, 선택된 옵션을 슬래시(/)로 구분해서 표시하는 것을 권장한다.
	function ItemStack($_id, $_name, $_tprice, $_uprice, $_option, $_count) {
		$this->id = $_id;
		$this->name = $_name;
		$this->tprice = $_tprice;
		$this->uprice = $_uprice;
		$this->option = $_option;
		$this->count = $_count;
	   }

	function makeQueryString() {
		$ret .= 'ITEM_ID=' . urlencode($this->id);
		$ret .= '&ITEM_NAME=' . urlencode($this->name);
		$ret .= '&ITEM_COUNT=' . $this->count;
		$ret .= '&ITEM_OPTION=' . urlencode($this->option);
		$ret .= '&ITEM_TPRICE=' . $this->tprice;
		$ret .= '&ITEM_UPRICE=' . $this->uprice;
		return $ret;
	}
};

$shopId = 'naver_checkout';
$certiKey = 'naver_checkout';
$shippingType = 'PAYED';

if ($shippingType == 'PAYED') {
   $shippingPrice = 2500;
} else {
   $shippingPrice = 0;
}

$backUrl = "http://www.naver.com";

$queryString = 'SHOP_ID='.urlencode($shopId);
$queryString .= '&CERTI_KEY='.urlencode($certiKey);
$queryString .= '&SHIPPING_TYPE='.$shippingType;
$queryString .= '&SHIPPING_PRICE='.$shippingPrice;
$queryString .= '&RESERVE1=&RESERVE2=&RESERVE3=&RESERVE4=&RESERVE5=';
$queryString .= '&BACK_URL='.$backUrl;

$totalMoney = 0;
```

```
//DB와 장바구니에서 상품 정보를 얻어 온다.
//while(...) {
   $id = "item1";
   $name = "상품1";
   $uprice = 1000;
   $count = 1;
   $tprice = $uprice * $count;
   $option = "";
   $item = new ItemStack($id, $name, $tprice, $uprice, $option, $count);
   $totalMoney += $tprice;
   $queryString .= '&'.$item->makeQueryString();
//}

$totalPrice = (int)$totalMoney + (int)$shippingPrice;
$queryString .= '&TOTAL_PRICE='.$totalPrice;

echo($queryString."<br>\n");

$req_addr = 'ssl://test-checkout.naver.com';
$req_url = 'POST /customer/api/order.nhn HTTP/1.1'; // utf-8
// $req_url = 'POST /customer/api/CP949/order.nhn HTTP/1.1'; // euc-kr
$req_host = 'test-checkout.naver.com';
$req_port = 443;
$nc_sock = @fsockopen($req_addr, $req_port, $errno, $errstr);
if ($nc_sock) {
      fwrite($nc_sock, $req_url."\r\n" );
      fwrite($nc_sock, "Host: ".$req_host.":".$req_port."\r\n" );
      fwrite($nc_sock, "Content-type: application/x-www-form-urlencoded; charset=utf-
8\r\n");
      //fwrite($nc_sock, "Content-type: application/x-www-form-urlencoded;
charset=CP949\r\n");
      fwrite($nc_sock, "Content-length: ".strlen($queryString)."\r\n");
      fwrite($nc_sock, "Accept: */*\r\n");
      fwrite($nc_sock, "\r\n");
      fwrite($nc_sock, $queryString."\r\n");
      fwrite($nc_sock, "\r\n");

      // get header
      while(!feof($nc_sock)){
            $header=fgets($nc_sock,4096);
            if($header=="\r\n"){
                  break;
            } else {
                $headers .= $header;
            }
      }

      // get body
      while(!feof($nc_sock)){
            $bodys.=fgets($nc_sock,4096);
      }

      fclose($nc_sock);

      $resultCode = substr($headers,9,3);

      if ($resultCode == 200) {
         // success
         $orderId = $bodys;
      } else {
         // fail
         echo $bodys;
      }
}
else {
      echo "$errstr ($errno)<br>\n";
      exit(-1);
      //에러처리
}

//리턴받은 order_id로 주문서 page를 호출한다.
```

```
echo ($orderId."<br>\n");

$orderUrl = "https://test-checkout.naver.com/customer/order.nhn";

?>
<html>
<body>
<form name="frm" method="get" action="<?=$orderUrl?>">
<input type="hidden" name="ORDER_ID" value="<?=$orderId?>">
<input type="hidden" name="SHOP_ID" value="<?=$shopId?>">
<input type="hidden" name="TOTAL_PRICE" value="<?=$totalPrice?>">
</form>
</body>
<script>

<? if ($resultCode == 200) { ?>
document.frm.target = "_top";
document.frm.submit();
<? } ?>
</script>
</html>
```

다음은 Java로 주문 정보를 등록하는 예이다.

```
package test.com.nhncorp.bp.external.mall;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.ArrayList;
import java.util.List;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.SSLSocketFactory;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

public class JavaOrderSample
{
    private static final String ENCODING = "UTF-8";
    private static final String NCKEY_ITEMID = "ITEM_ID";
    private static final String NCKEY_ITEMNAME = "ITEM_NAME";
    private static final String NCKEY_COUNT = "ITEM_COUNT";
    private static final String NCKEY_TPRICE = "ITEM_TPRICE";
    private static final String NCKEY_UPRICE = "ITEM_UPRICE";
    private static final String NCKEY_OPTION = "ITEM_OPTION";
    private static final String NCKEY_SHIPPINGTYPE = "SHIPPING_TYPE";
    private static final String NCKEY_SHIPPINGPRICE = "SHIPPING_PRICE";
    private static final String NCKEY_TOTALPRICE = "TOTAL_PRICE";
    private static final String NCKEY_SHOPID = "SHOP_ID";
    private static final String NCKEY_CERTIKEY = "CERTI_KEY";
    private static final String NCKEY_BACKURL = "BACK_URL";
    private static final String NCKEY_RESERVE1 = "RESERVE1";
    private static final String NCKEY_RESERVE2 = "RESERVE2";
    private static final String NCKEY_RESERVE3 = "RESERVE3";
    private static final String NCKEY_RESERVE4 = "RESERVE4";
    private static final String NCKEY_RESERVE5 = "RESERVE5";

    public URL _url;
    private SSLSocketFactory _sslSockFactory;
```

```
   public JavaOrderSample()
   {
      _url = null;
   }

   public JavaOrderSample(URL url)
   {
      _url = url;
      _initHttps();
   }

   public JavaOrderSample(String url) throws MalformedURLException
   {
      this(new URL(url));
   }

   public void setUrl(String url) throws MalformedURLException
   {
      _url = new URL(url);
   }

   public static class ItemStack
   {
      private String _itemId;
      private String _itemName;
      private int _itemTPrice;
      private int _itemUPrice;
      private String _selectedOption;
      private int _count;

      /**
       * @param itemId Mall Item Code
       * @param itemName 상품
       * @param itemPrice 상품 개별 가격
       * @param selectedOption 선택된 옵션. - 여러 옵션이 선택되었을 경우 '/'로 구분하는 것을 권장
       * @param count 해당 상품 구매 갯수.
       */
      public ItemStack(String itemId, String itemName, int itemTPrice, int itemUPrice,
String selectedOption, int count)
      {
         _itemId = itemId;
         _itemName = itemName;
         _itemTPrice = itemTPrice;
         _itemUPrice = itemUPrice;
         _selectedOption = selectedOption;
         _count = count;
      }

      public String getItemId()
      {
         return _itemId;
      }

      public String getItemName()
      {
         return _itemName;
      }

      public int getItemTotalPrice()
      {
         return _itemTPrice;
      }

      public int getItemUnitPrice()
      {
         return _itemUPrice;
      }

      public String getSelectedOption()
      {
```

```
            if (_selectedOption == null) return "";
            return _selectedOption;
        }

        public int getCount()
        {
            return _count;
        }
    }

    private void _urlEncode(StringBuffer sb, String key, String value)
    {
        try {
            sb.append(URLEncoder.encode(key, ENCODING));
            sb.append('=');
            sb.append(URLEncoder.encode(value, ENCODING));
        }
        catch(UnsupportedEncodingException e) {
            //일어나지 않음
            throw new Error(e);
        }
    }

    private String _makeQueryString(String shopId, String certificationKey, ItemStack[]
items, int shippingPrice, String shippingType, String backURL)
    {
        //주문금액은 각 상품 금액 + 배송비이다. (단 선불일 경우)
        int totalPrice = shippingPrice>0?shippingPrice:0;
        StringBuffer sb = new StringBuffer();
        _urlEncode(sb, NCKEY_SHOPID, shopId);
        sb.append('&');
        _urlEncode(sb, NCKEY_CERTIKEY, certificationKey);
        sb.append('&');
        for (ItemStack is : items) {
            totalPrice += is.getItemTotalPrice();
            _urlEncode(sb, NCKEY_ITEMID, is.getItemId());
            sb.append('&');
            _urlEncode(sb, NCKEY_ITEMNAME, is.getItemName());
            sb.append('&');
            _urlEncode(sb, NCKEY_TPRICE, String.valueOf(is.getItemTotalPrice()));
            sb.append('&');
            _urlEncode(sb, NCKEY_UPRICE, String.valueOf(is.getItemUnitPrice()));
            sb.append('&');
            _urlEncode(sb, NCKEY_COUNT, String.valueOf(is.getCount()));
            sb.append('&');
            _urlEncode(sb, NCKEY_OPTION, is.getSelectedOption());
            sb.append('&');
        }

        _urlEncode(sb, NCKEY_SHIPPINGTYPE, shippingType);
        sb.append('&');
        _urlEncode(sb, NCKEY_SHIPPINGPRICE, String.valueOf(shippingPrice));
        sb.append('&');
        _urlEncode(sb, NCKEY_TOTALPRICE, String.valueOf(totalPrice));
        sb.append('&');
        _urlEncode(sb, NCKEY_BACKURL, backURL);
        sb.append('&');
        _urlEncode(sb, NCKEY_RESERVE1, "");
        sb.append('&');
        _urlEncode(sb, NCKEY_RESERVE2, "");
        sb.append('&');
        _urlEncode(sb, NCKEY_RESERVE3, "");
        sb.append('&');
        _urlEncode(sb, NCKEY_RESERVE4, "");
        sb.append('&');
        _urlEncode(sb, NCKEY_RESERVE5, "");
        System.out.println(sb.toString());

        return sb.toString();
    }
```

```
   /* test 환경에서는 인증서 오류가 날 수도 있다. 이 코드를 이용해 인증서 오류를 회피한다. */
   private void _initHttps()
   {
      TrustManager[] trustAllCerts = new TrustManager[] {
            new X509TrustManager() {
               public void checkClientTrusted(X509Certificate[] chain, String authType)
throws CertificateException {}
               public void checkServerTrusted(X509Certificate[] chain, String authType)
throws CertificateException {}
               public X509Certificate[] getAcceptedIssuers()
               {
                  return new X509Certificate[0];
               }
            }
      };
      try {
         SSLContext sslContext = SSLContext.getInstance("SSL");
         sslContext.init(null, trustAllCerts, new SecureRandom());
         _sslSockFactory = sslContext.getSocketFactory();
      }
      catch(Exception e) {
         RuntimeException re = new RuntimeException(e);
         re.setStackTrace(e.getStackTrace());
         throw re;
      }
   }

   /**
    * @param items 주문 상품 목록.
    * @param shippingPrice 배송비.
    * @param shippingType 배송비결제 구분. "FREE": 무료. "PAYED": 선불. "ONDELIVERY": 착불
    * @return 주문키
    * @throws IOException
    */
   public String sendOrderInfoToNC(String shopId, String certificationKey, ItemStack[]
items, int shippingPrice, String shippingType, String backURL) throws IOException
   {
      HttpURLConnection conn = (HttpURLConnection)_url.openConnection();

      /* test 환경에서는 인증서 오류가 날 수도 있다. 이 코드를 이용해 인증서 오류를 회피한다. */
      if (conn instanceof HttpsURLConnection) {
         ((HttpsURLConnection)conn).setSSLSocketFactory(_sslSockFactory);
         ((HttpsURLConnection)conn).setHostnameVerifier(
               new HostnameVerifier() {
                  @Override
                  public boolean verify(String hostname, SSLSession session)
                  {
                     return true;
                  }
               }
         );
      }
      conn.setDoInput(true);
      conn.setDoOutput(true);
      conn.setUseCaches(false);
      conn.setRequestMethod("POST");
      conn.addRequestProperty("Content-Type", "application/x-www-form-urlencoded;
charset=UTF-8");

      Writer writer = new OutputStreamWriter(conn.getOutputStream(), ENCODING);
      writer.write(_makeQueryString(shopId, certificationKey, items, shippingPrice,
shippingType, backURL));
      writer.flush();
      writer.close();

      int respCode = conn.getResponseCode();
      if (respCode != 200) {
         throw new RuntimeException(String.format("NC Response fail : %d %s", respCode,
conn.getResponseMessage()));
      }
```

```
      BufferedReader reader = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
      String orderKey = reader.readLine();
      return orderKey;
   }

   public static void main(String[] args) throws IOException
   {
      //주문상품 내역으로 items 데이터를 생성한다.
      List<ItemStack> items = new ArrayList<ItemStack>();
      items.add(new ItemStack("a1", "아이템1", 2000, 1000, "", 3));
      items.add(new ItemStack("a2", "아이템2", 2000, 1000, "XL/빨강", 2));

      int shippingPrice = 2500;
      String shippingType = "PAYED";
      String backURL = "http://www.naver.com";

      JavaOrderSample sample = new JavaOrderSample("https://test-
checkout.naver.com/customer/api/order.nhn");
      String orderKey = sample.sendOrderInfoToNC("naver_checkout", "naver_checkout",
items.toArray(new ItemStack[0]), shippingPrice, shippingType, backURL);
      //여기서 얻은 orderKey로 NC 결제창에 넘겨 결제를 진행한다.
      System.out.println(String.format("OrderKey = %s", orderKey));
   }
}
```

다음은 ASP로 주문 정보를 등록하는 예이다.

```
<%
Class Itemstack
      Private mId
      Private mName
      Private mCount
      Private mUnitPrice
      Private mTotalPrice
      Private mItemOption

      Public Property Get Id()
            Id = mId
      End Property

      Public Property Let Id(val)
            mId = val
      End Property

      Public Property Get Name()
            Name = mName
      End Property

      Public Property Let Name(val)
            mName = val
      End Property

      Public Property Get Count()
            Count = mCount
      End Property

      Public Property Let Count(val)
            mCount = val
      End Property

      Public Property Get UnitPrice()
            UnitPrice = mUnitPrice
      End Property

      Public Property Let UnitPrice(val)
            mUnitPrice = val
      End Property

      Public Property Get TotalPrice()
```

```
		TotalPrice = mTotalPrice
	End Property

	Public Property Let TotalPrice(val)
		mTotalPrice = val
	End Property

	Public Property Get ItemOption()
		ItemOption = mItemOption
	End Property

	Public Property Let ItemOption(val)
		mItemOption = val
	End Property

	Public Property Get QueryString()
		QueryString = "ITEM_ID=" + Server.URLEncode(mId)
		QueryString = QueryString + "&ITEM_NAME=" + Server.URLEncode(mName)
		QueryString = QueryString + "&ITEM_COUNT=" + Server.URLEncode(mCount)
		QueryString = QueryString + "&ITEM_UPRICE=" + Server.URLEncode(mUnitPrice)
		QueryString = QueryString + "&ITEM_TPRICE=" + Server.URLEncode(mTotalPrice)
		QueryString = QueryString + "&ITEM_OPTION=" + Server.URLEncode(mItemOption)
	End Property
End Class

Class Order
	Private mShopId
	Private mCertiKey
	Private mShippingPrice
	Private mShippingType
	Private mTotalPrice
	Private mBackURL

	Public Property Get ShopId()
		ShopId = mShopId
	End Property

	Public Property Let ShopId(value)
		mShopId = value
	End Property

	Public Property Get CertiKey()
		CertiKey = mCertiKey
	End Property

	Public Property Let CertiKey(value)
		mCertiKey = value
	End Property

	Public Property Get ShippingPrice()
		ShippingPrice = mShippingPrice
	End Property

	Public Property Let ShippingPrice(value)
		mShippingPrice = value
	End Property

	Public Property Get ShippingType()
		ShippingType = mShippingType
	End Property

	Public Property Let ShippingType(value)
		mShippingType = value
	End Property

	Public Property Get TotalPrice()
		TotalPrice = mTotalPrice
	End Property

	Public Property Let TotalPrice(value)
		mTotalPrice = value
	End Property
```

```
	Public Property Get BackURL()
		BackURL = mBackURL
	End Property

	Public Property Let BackURL(value)
		mBackURL = value
	End Property

	Public Property Get QueryString()
		QueryString = "SHOP_ID=" + Server.URLEncode(mShopId)
		QueryString = QueryString + "&CERTI_KEY=" + Server.URLEncode(mCertiKey)
		QueryString = QueryString + "&SHIPPING_PRICE=" +
Server.URLEncode(mShippingPrice)
		QueryString = QueryString + "&SHIPPING_TYPE=" +
Server.URLEncode(mShippingType)
		QueryString = QueryString + "&TOTAL_PRICE=" + Server.URLEncode(mTotalPrice)
		QueryString = QueryString + "&BACK_URL=" + Server.URLEncode(mBackURL)
		QueryString = QueryString + "&RESERVE1="
		QueryString = QueryString + "&RESERVE2="
		QueryString = QueryString + "&RESERVE3="
		QueryString = QueryString + "&RESERVE4="
		QueryString = QueryString + "&RESERVE5="
	End Property

End Class

Dim myOrder, items(0), queryString

' loop
Set items(0) = new ItemStack
items(0).Id = "items2"
items(0).Name = "아이템2"
items(0).Count = 1
items(0).UnitPrice = "100"
items(0).TotalPrice = "100"
items(0).ItemOption = ""
' end loop

Set myOrder = new Order
myOrder.ShopId = "naver_checkout"
myOrder.CertiKey = "naver_checkout"
myOrder.ShippingPrice = 1500
myOrder.ShippingType = "PAYED"
myOrder.TotalPrice = 1600
myOrder.BackURL = "http://www.naver.com"

queryString = myOrder.QueryString
For Each item In items
	queryString = queryString + "&" + item.QueryString
Next

Response.Write(queryString+"<br>"+chr(10))

Dim objXMLHTTP, orderId
Set objXMLHTTP = server.CreateObject("MSXML2.ServerXMLHTTP")
objXMLHTTP.setOption 2,13056 'ignore SSL errors
objXMLHTTP.Open "POST", "https://test-checkout.naver.com/customer/api/order.nhn", false
'objXMLHTTP.Open "POST", "https://test-checkout.naver.com/customer/api/CP949/order.nhn",
false
objXMLHTTP.setRequestHeader "Content-Type", "application/x-www-form-
urlencoded;charset=UTF-8"
'objXMLHTTP.setRequestHeader "Content-Type", "application/x-www-form-urlencoded;charset=
CP949 "

objXMLHTTP.Send queryString

If objXMLHTTP.status = 200 Then
	orderId = objXMLHTTP.responseText
Else
	'에러처리
End if
```

```
Response.Write(orderId)

orderUrl = "https://test-checkout.naver.com/customer/order.nhn"
%>
<html>
<body>
<form name="frm" method="get" action="<%=orderUrl%>">
<input type="hidden" name="ORDER_ID" value="<%=orderId%>">
<input type="hidden" name="SHOP_ID" value="<%=myOrder.ShopId%>">
<input type="hidden" name="TOTAL_PRICE" value="<%=myOrder.TotalPrice%>">
</form>
</body>
<script>

<% If objXMLHTTP.status = 200 Then %>
document.frm.target = "_top";
document.frm.submit();
<% End if

Set objXMLHTTP = Nothing
%>
</script>
</html>
```

3.1.3 주문서 전송

가맹점은 네이버 체크아웃 주문서 팝업 창을 생성하여 GET 인자로 아래의 정보를 전달한다.

표 3-3 주문서 팝업 창 URL

환경	URL
테스트	https://test-checkout.naver.com/customer/order.nhn
서비스	https://checkout.naver.com/customer/order.nhn

표 3-4 주문서 내용

항목	필수 여부	설명
ORDER_ID	Y	주문 ID. 주문 정보 등록 결과 네이버 체크아웃에서 받는다.
SHOP_ID	Y	상점 ID. 네이버 체크아웃에 가입 승인될 때 정해진다.
TOTAL_PRICE	Y	총 주문 금액

3.1.4 네이버 체크아웃 결제

결제 과정은 네이버 체크아웃 주문/결제 페이지에서 이루어진다. 이후, 네이버 체크아웃에서 상품 정보를 업데이트하기 위해 상품 정보를 요청할 수 있다.

3.2 상품 정보 연동

3.2.1 상품 정보 요청

네이버 체크아웃은 비주기적으로 상품 정보를 업데이트하기 위해, 가맹점 사이트에 HTTP로 상품 정보 요청을 보낸다. 가맹점은 요청받은 상품 정보를 XML로 전송하는 페이지를 생성해야 한다.

네이버 체크아웃이 상품 정보를 요청하는 형식은 다음과 같다.

```
http://가맹점도메인/가맹점페이지?ITEM_ID=XXX&ITEM_ID=XXX&ITEM_ID=XXX
```

표 3-5 상품 정보 요청 내용

항목	필수 여부	설명
ITEM_ID	Y	주문 정보 등록 시 네이버에 등록한 ITEM_ID. ITEM_ID 는 여러 개일 수 있다.

3.2.2 상품 정보 제공

가맹점은 XML로 response에 각각의 ITEM_ID에 대해 다음의 상품 정보를 전송한다.

필수 항목은 모두 element로 명시해야 하며, URL 값은 반드시 유효해야 한다.

표 3-6 상품 정보 제공 내용

항목	필수 여부	설명
item@id	Y	상품 ID
item/name	Y	상품 이름
item/url	Y	상품 정보 URL
item/description	Y	상품 설명
item/image	Y	상품 사진 URL
item/thumb	Y	상품 썸네일 URL
item/price	Y	상품의 정상 가격
item/quantity	Y	상품의 재고량
item/options	N	선택한 옵션 사항
item/category	Y	가맹점 사이트에서 상품의 카테고리

문자 세트는 각 환경에 맞게 설정하고, content-type은 application/xml로 한다.

```
<?xml version="1.0" encoding="UTF-8">
<response>
    <item id="1"> <!-- 상품 ID. -->
        <name><![CDATA[상품 이름]]></name>
```

```
      <url>http://www.mymall.com/show_item?itemId=itemId</url> <!-- 상품 설명 페이지 url -->
      <description><![CDATA[상품 설명]]></description>
      <image>http://www.mymall.com/images/itemId.jpg</image> <!-- 상품 image url -->
      <thumb>http://www.mymall.com/images/itemId.jpg</thumb> <!-- 상품 썸네일 image url -->
      <price>10000</price> <!-- 정상 가격 -->
      <quantity>10</quantity> <!-- 재고량 -->
      <options> <!-- 상품의 옵션이 없으면 이 내용은 없어도 된다. -->
         <option name="색상"> <!-- 옵션 종류 이름 -->
            <select><![CDATA[빨강]]></select> <!-- 해당 옵션의 선택 사항 -->
            <select><![CDATA[파랑]]></select>
            <select><![CDATA[노랑]]></select>
         </option>
         <option name="크기">
            <select><![CDATA[L]]></select>
            <select><![CDATA[XL]]></select>
         </option>
      </options>
      <category> <!-- 카테고리 -->
         <first>...</first> <!-- 최상위 카테고리 이름 -->
         <second>...</second> <!-- 차상위 카테고리 이름 -->
         ... <!-- 최대 4단계 카테고리까지 표시 가능하다. (first ~ fourth) -->
      </category>
   </item>
   <!-- 여러 개의 상품 정보를 요청하면 그 개수만큼의 item element가 필요하다. -->
</response>
```

다음은 PHP로 상품 정보를 전송하는 예이다.

```
<?
$query = $_SERVER['QUERY_STRING'];
$vars = array();
foreach(explode('&', $query) as $pair) {
      list($key, $value) = explode('=', $pair);
      $key = urldecode($key);
      $value = urldecode($value);
      $vars[$key][] = $value;
}

$itemIds = $vars['ITEM_ID'];

if (count($itemIds) < 1) {
      exit('ITEM_ID 는 필수입니다.');
}

header('Content-Type: application/xml;charset=euc-kr');
echo ('<?xml version="1.0" encoding="euc-kr"?>');
?>

<response>
<?

//while(...):
      $id = "itemid";
      $name = "아이템명";
      $description = "간지나는아이템";
      $price = 1000;
      $quantity = 1;
?>
      <item id="<?=$id?>">
            <name><![CDATA[<?=$name?>]]></name>
            <url>http://<?=$_SERVER["HTTP_HOST"]?>/test.html</url>
            <description><![CDATA[<?=$description?>]]></description>
```

```
            <image>http://<?=$_SERVER["HTTP_HOST"]?>/test.jpg</image>
            <thumb>http://<?=$_SERVER["HTTP_HOST"]?>/test.jpg</thumb>
            <price><?=$price?></price>
            <quantity><?=$quantity?></quantity>
            <category>
                  <first id="MJ01">대분류</first>
                  <second id="ML01">중분류</second>
                  <third id="MN01">소분류</third>
            </category>

            <options>
                  <option name="크기">
                     <select> <![CDATA[ X ]]> </select>
                     <select> <![CDATA[ XL ]]> </select>
                  </option>
            </options>
   </item>
<?
//end while;

echo('</response>');
?>
```

다음은 Java로 상품 정보를 전송하는 예이다.

```
package test.com.nhncorp.bp.external.mall;

import java.io.IOException;
import java.io.Writer;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@SuppressWarnings("serial")
public class JavaItemInfoServlet extends HttpServlet
{
   public void _writeItemInfo(String itemId, Writer writer) throws IOException
   {
      //itemId로 DB에서 정보 조회
      String description = "";
      String url = "http://www.mymall.com/show_item?itemId=" + itemId;
      String itemName = "아이템이름";
String itemImage = "http://www.mymall.com/images/" + itemId + ".jpg";
      String thumbImage = "http://www.mymall.com/thumbs/" + itemId + ".jpg";
      String[] options = new String[2];
      options[0] = "색상";
      options[1] = "크기";
      String[][] optionList = new String[2][];
      optionList[0] = new String[3];
      optionList[0][0] = "빨강";
      optionList[0][1] = "파랑";
      optionList[0][2] = "노랑";
      optionList[1] = new String[2];
      optionList[1][0] = "L";
      optionList[1][1] = "XL";

      int price = 10000;
      int quantity = 10;

      String majorCategoryId = "MJ01";
      String majorCategoryName = "대분류";

      String middleCategoryId = "ML01";
      String middleCategoryName = "중분류";
```

```
      String minorCategoryId = "MN01";
      String minorCategoryName = "소분류";
      // 여기까지 DB에서 조회한 정보.

      // ! 각 String에 xml의 attribute나 body 혹은 CDATA block안에 들어갈 수 없는 문자가 있다면
적절하게 escaping 해야 한다.
      // Write구조 대신 javax.xml.transform.TransformerFactory 과 org.w3c.dom.Document 를
이용해 DOM구조를 XML로 출력하는 방법을 사용할 수도 있다.
      writer.write(String.format("<item id=\"%s\">\r\n", itemId));
      writer.write(String.format("<name><![CDATA[%s]]></name>\r\n", itemName));
      writer.write(String.format("<url><![CDATA[%s]]></url>\r\n", url));
      writer.write(String.format("<description><![CDATA[%s]]></description>\r\n",
description));
      writer.write(String.format("<image><![CDATA[%s]]></image>\r\n", itemImage));
      writer.write(String.format("<thumb><![CDATA[%s]]></thumb>\r\n", thumbImage));

      if (options != null) {
         writer.write("<options>\r\n");
         for (int i=0;i<options.length;++i) {
            writer.write(String.format("<option name=\"%s\">\r\n", options[i]));
            for (int j=0;j<optionList[i].length;++j) {
               writer.write(String.format("<select><![CDATA[%s]]></select>\r\n",
optionList[i][j]));
            }
            writer.write("</option>\r\n");
         }
         writer.write("</options>\r\n");
      }
      writer.write(String.format("<price>%d</price>\r\n", price));
      writer.write(String.format("<quantity>%d</quantity>\r\n", quantity));

      writer.write("<category>\r\n");
      writer.write(String.format("<first id=\"%s\"><![CDATA[%s]]></first>\r\n",
majorCategoryId, majorCategoryName));
      writer.write(String.format("<second id=\"%s\"><![CDATA[%s]]></second>\r\n",
middleCategoryId, middleCategoryName));
      writer.write(String.format("<third id=\"%s\"><![CDATA[%s]]></third>\r\n",
minorCategoryId, minorCategoryName));
      writer.write("</category>\r\n");
      writer.write("</item>\r\n");
   }

   @Override
   public void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, java.io.IOException
   {
      String[] itemIds = req.getParameterValues("ITEM_ID");
resp.setContentType("application/xml;charset=UTF-8");
      Writer writer = resp.getWriter();
      writer.write("<?xml version=\"1.0\" encoding=\"UTF-8\"?>\r\n");
      writer.write("<response>\r\n");
      for (String itemId : itemIds) {
         _writeItemInfo(itemId, writer);
      }
      writer.write("</response>");
      writer.flush();
      writer.close();
   }

   @Override
   public void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, java.io.IOException
   {
      doPost(req, resp);
   }
}
```

다음은 ASP로 상품 정보를 전송하는 예이다.

```
<% Response.ContentType = "text/xml;charset=euc-kr" %><?xml version="1.0" encoding="euc-
kr"?>
<response>

<%

Dim itemId, url, name, description, image, thumb, options(), price, quantity

    for i=1 to Request.QueryString("ITEM_ID").Count

        itemId = Request.QueryString("ITEM_ID")(i)
        url = "http://www.mymall.com/show_item?itemid=" + itemId
        name = "아이템이름"
        description = ""
        image = "http://www.mymall.com/images/" + itemId + ".jpg"
        thumb = "http://www.mymall.com/thumbs/" + itemId + ".jpg"
        price = 10000
        quantity = 10
        majorId = "MJ01"
        majorName = "대분류"
        middleId = "ML01"
        middleName = "중분류"
        minorId = "MN01"
        minorName = "소분류"
%>
    <item id="<%=itemId%>">
        <name> <![CDATA[ <%=name%> ]]> </name>
        <url><%=url%></url>
        <description> <![CDATA[ <%=description%> ]]> </description>
        <image><%=image%></image>
        <thumb><%=thumb%></thumb>
        <price><%=price%></price>
        <quantity><%=quantity%></quantity>
        <category>
            <first id="<%=majorId%>"><![CDATA[ <%=majorName%> ]]> </first>
            <second id="<%=middleId%>"><![CDATA[ <%=middlenorName%> ]]> </second>
            <third id="<%=minorId%>"><![CDATA[ <%=minorName%> ]]> </third>
        </category>
        <options>
            <option name="크기">
                <select> <![CDATA[ X ]]> </select>
                <select> <![CDATA[ XL ]]> </select>
            </option>
        </options>

    </item>
<%
    next
%>
</response>
```

3.3 찜 정보 연동

3.3.1 네이버 체크아웃 [찜하기] 버튼 클릭

이용자가 상품 정보 페이지에서 네이버 체크아웃 [찜하기] 버튼을 클릭하면, 가맹점은 팝업 창을 생성하여 찜 정보 등록을 위한 페이지로 이동한다.

3.3.2 찜 정보 등록

가맹점은 네이버 체크아웃에 HTTPS로 찜한 상품 정보를 등록하고 네이버 체크아웃 상품 ID를 받는다.

찜 정보 등록 시 기본 문자 세트는 UTF-8이고, 다른 문자셋을 사용하는 경우 UTF-8로 변환하거나, 다른 URL을 이용한다.

표 3-7 찜 정보 등록 URL

환경	문자셋	URL
테스트	UTF-8	https://test-checkout.naver.com/customer/api/wishlist.nhn
	EUC-KR	https://test-checkout.naver.com/customer/api/CP949/wishlist.nhn
서비스	UTF-8	https://checkout.naver.com/customer/api/wishlist.nhn
	EUC-KR	https://checkout.naver.com/customer/api/CP949/wishlist.nhn

response로 받는 네이버 체크아웃 상품 ID는 영문과 숫자로 이루어지며 최대 19자리이다.

request 시 post로 아래와 같은 정보를 전달한다. URL 값은 반드시 유효해야 한다.

표 3-8 찜 정보 등록 내용

항목	필수 여부	설명
SHOP_ID	Y	상점 ID. 네이버 체크아웃에 가입 승인될 때 정해진다.
CERTI_KEY	Y	인증키. 네이버 체크아웃에 가입 승인될 때 정해진다.
ITEM_ID	Y	상품 ID
ITEM_NAME	Y	상품 이름
ITEM_DESC	Y	상품 설명
ITEM_UPRICE	Y	개별 상품 단가. 0 보다 커야 한다.
ITEM_IMAGE	Y	상품 사진 URL.
ITEM_THUMB	Y	상품 썸네일 URL.
ITEM_URL	Y	상품 정보 URL.
RESERVE1	N	예약 필드 1

RESERVE2	N	예약 필드 2
RESERVE3	N	예약 필드 3
RESERVE4	N	예약 필드 4
RESERVE5	N	예약 필드 5

ITEM_ID, ITEM_NAME, ITEM_UPRICE는 여러 개일 수 있다. 예를 들어, 한 번에 찜한 상품 종류가 두 개라면 각 파라미터는 두 개씩 존재한다. 이 때 각 파라미터에서 상품의 순서를 지켜야 한다.

여러 개의 상품을 찜하면, response로 받는 네이버 체크아웃 상품 ID 또한 여러 개이며, ITEM_ID는 순서대로 공백 없이 쉼표(,)로 구분된다.

다음은 PHP로 찜 정보를 등록하는 예이다.

```
<?
//item data를 생성한다.
class ItemStack {
	var $id;
	var $name;
	var $uprice;
	var $image;
	var $thumb;
	var $url;

	function ItemStack($_id, $_name, $_uprice, $_image, $_thumb, $_url) {
		$this->id = $_id;
		$this->name = $_name;
		$this->uprice = $_uprice;
		$this->image = $_image;
		$this->thumb = $_thumb;
		$this->url = $_url;
	}

	function makeQueryString() {
		$ret .= 'ITEM_ID=' . urlencode($this->id);
		$ret .= '&ITEM_NAME=' . urlencode($this->name);
		$ret .= '&ITEM_UPRICE=' . $this->uprice;
		$ret .= '&ITEM_IMAGE=' . urlencode($this->image);
		$ret .= '&ITEM_THUMB=' . urlencode($this->thumb);
		$ret .= '&ITEM_URL=' . urlencode($this->url);
		return $ret;
	}
};

$shopId = 'naver_checkout';
$certiKey = 'naver_checkout';


$queryString = 'SHOP_ID='.urlencode($shopId);
$queryString .= '&CERTI_KEY='.urlencode($certiKey);
$queryString .= '&RESERVE1=&RESERVE2=&RESERVE3=&RESERVE4=&RESERVE5=';

//DB 에서 상품 정보를 얻어온다.
//while(...) {
   $uid = "item1";
   $name = "상품1";
   $uprice = 1000;
   $image = "http://mymall.com/image/item1.jpg";
   $thumb = "http://mymall.com/image/item1.jpg";
   $url = "http://mymall.com/item?id=item1";
   $item = new ItemStack($uid, $name, $uprice, $image, $thumb, $url);
$queryString .= '&'.$item->makeQueryString();
//}
```

```
echo($queryString."<br>\n");

$req_addr = 'ssl://test-checkout.naver.com';
$req_url = 'POST /customer/api/wishlist.nhn HTTP/1.1'; // utf-8
// $req_url = 'POST /customer/api/CP949/wishlist.nhn HTTP/1.1'; // euc-kr
$req_host = 'test-checkout.naver.com';
$req_port = 443;
$nc_sock = @fsockopen($req_addr, $req_port, $errno, $errstr);
if ($nc_sock) {
	fwrite($nc_sock, $req_url."\r\n" );
	fwrite($nc_sock, "Host: ".$req_host.":".$req_port."\r\n" );
	fwrite($nc_sock, "Content-type: application/x-www-form-urlencoded; charset=utf-
8\r\n"); // utf-8
	//fwrite($nc_sock, "Content-type: application/x-www-form-urlencoded;
charset=CP949\r\n");  // euc-kr
	fwrite($nc_sock, "Content-length: ".strlen($queryString)."\r\n");
	fwrite($nc_sock, "Accept: */*\r\n");
	fwrite($nc_sock, "\r\n");
	fwrite($nc_sock, $queryString."\r\n");
	fwrite($nc_sock, "\r\n");

	// get header
	while(!feof($nc_sock)){
		$header=fgets($nc_sock,4096);
		if($header=="\r\n"){
			break;
		} else {
		    $headers .= $header;
		}
	}

	// get body
	while(!feof($nc_sock)){
		$bodys.=fgets($nc_sock,4096);
	}

	fclose($nc_sock);

	$resultCode = substr($headers,9,3);

	if ($resultCode == 200) {
	   // success

	   // 한개일경우
	   $itemId = $bodys;

	   // 여러개일경우
	   //$itemIds = trim($bodys);
	   //$itemIdList = split(",",$itemIds);

	} else {
	   // fail
	   echo $bodys;
	}
}
else {
	echo "$errstr ($errno)<br>\n";
	exit(-1);
	//에러처리
}

//리턴받은 itemId로 주문서 page를 호출한다.
echo ($itemId."<br>\n");

$wishlistPopupUrl = "https://test-checkout.naver.com/customer/wishlistPopup.nhn";

?>
<html>
<body>
```

```
<form name="frm" method="get" action="<?=$wishlistPopupUrl?>">
<input type="hidden" name="SHOP_ID" value="<?=$shopId?>">

<!-- 한 개일 경우 -->
<input type="hidden" name="ITEM_ID" value="<?=$itemId?>">

<!-- 여러 개일 경우
<? for($i=0; $i < count($itemIdList); $i++) { ?>
<input type="hidden" name="ITEM_ID" value="<?=$itemIdList[$i]?>">
<? } ?>
-->

</form>
</body>
<script>

<? if ($resultCode == 200) { ?>
document.frm.target = "_top";
document.frm.submit();
<? } ?>
</script>
</html>
```

다음은 Java로 찜 정보를 등록하는 예이다.

```
package test.com.nhncorp.bp.external.mall;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.io.UnsupportedEncodingException;
import java.io.Writer;
import java.net.HttpURLConnection;
import java.net.MalformedURLException;
import java.net.URL;
import java.net.URLEncoder;
import java.security.SecureRandom;
import java.security.cert.CertificateException;
import java.security.cert.X509Certificate;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;

import javax.net.ssl.HostnameVerifier;
import javax.net.ssl.HttpsURLConnection;
import javax.net.ssl.SSLContext;
import javax.net.ssl.SSLSession;
import javax.net.ssl.SSLSocketFactory;
import javax.net.ssl.TrustManager;
import javax.net.ssl.X509TrustManager;

public class JavaZzimSample
{
    private static final String ENCODING = "UTF-8";
    private static final String NCKEY_ITEMID = "ITEM_ID";
    private static final String NCKEY_ITEMNAME = "ITEM_NAME";
    private static final String NCKEY_UPRICE = "ITEM_UPRICE";
    private static final String NCKEY_ITEMIMAGE = "ITEM_IMAGE";
    private static final String NCKEY_ITEMTHUMB = "ITEM_THUMB";
    private static final String NCKEY_ITEMURL = "ITEM_URL";
    private static final String NCKEY_SHOPID = "SHOP_ID";
    private static final String NCKEY_CERTIKEY = "CERTI_KEY";
    private static final String NCKEY_RESERVE1 = "RESERVE1";
    private static final String NCKEY_RESERVE2 = "RESERVE2";
    private static final String NCKEY_RESERVE3 = "RESERVE3";
    private static final String NCKEY_RESERVE4 = "RESERVE4";
    private static final String NCKEY_RESERVE5 = "RESERVE5";

    public URL _url;
    private SSLSocketFactory _sslSockFactory;
```

```
   public JavaZzimSample()
   {
      _url = null;
   }

   public JavaZzimSample(URL url)
   {
      _url = url;
      _initHttps();
   }

   public JavaZzimSample(String url) throws MalformedURLException
   {
      this(new URL(url));
   }

   public void setUrl(String url) throws MalformedURLException
   {
      _url = new URL(url);
   }

   public static class ItemStack
   {
      private String _itemId;
      private String _itemName;
      private int _itemUPrice;
      private String _itemImage;
      private String _itemThumb;
      private String _itemUrl;

      /**
       * @param itemId Mall Item Code
       * @param itemName 상품
       * @param itemPrice 상품 개별 가격
       */
      public ItemStack(String itemId, String itemName, int itemUPrice, String itemImage,
String itemThumb, String itemUrl)
      {
         _itemId = itemId;
         _itemName = itemName;
         _itemUPrice = itemUPrice;
         _itemImage = itemImage;
         _itemThumb = itemThumb;
         _itemUrl = itemUrl;
      }

      public String getItemId()
      {
         return _itemId;
      }

      public String getItemName()
      {
         return _itemName;
      }

      public int getItemUnitPrice()
      {
         return _itemUPrice;
      }

      public String getItemImage()
      {
         return _itemImage;
      }

      public String getItemThumb()
      {
         return _itemThumb;
      }

      public String getItemUrl()
```

```
      {
         return _itemUrl;
      }
   }

   private void _urlEncode(StringBuffer sb, String key, String value)
   {
      try {
         sb.append(URLEncoder.encode(key, ENCODING));
         sb.append('=');
         sb.append(URLEncoder.encode(value, ENCODING));
      }
      catch(UnsupportedEncodingException e) {
         //일어나지 않음
         throw new Error(e);
      }
   }

   private String _makeQueryString(String shopId, String certificationKey, ItemStack[]
items)
   {
      StringBuffer sb = new StringBuffer();
      _urlEncode(sb, NCKEY_SHOPID, shopId);
      sb.append('&');
      _urlEncode(sb, NCKEY_CERTIKEY, certificationKey);
      sb.append('&');
      for (ItemStack is : items) {
         _urlEncode(sb, NCKEY_ITEMID, is.getItemId());
         sb.append('&');
         _urlEncode(sb, NCKEY_ITEMNAME, is.getItemName());
         sb.append('&');
         _urlEncode(sb, NCKEY_UPRICE, String.valueOf(is.getItemUnitPrice()));
         sb.append('&');
         _urlEncode(sb, NCKEY_ITEMIMAGE, String.valueOf(is.getItemImage()));
         sb.append('&');
         _urlEncode(sb, NCKEY_ITEMTHUMB, String.valueOf(is.getItemThumb()));
         sb.append('&');
         _urlEncode(sb, NCKEY_ITEMURL, String.valueOf(is.getItemUrl()));
         sb.append('&');
      }
      _urlEncode(sb, NCKEY_RESERVE1, "");
      sb.append('&');
      _urlEncode(sb, NCKEY_RESERVE2, "");
      sb.append('&');
      _urlEncode(sb, NCKEY_RESERVE3, "");
      sb.append('&');
      _urlEncode(sb, NCKEY_RESERVE4, "");
      sb.append('&');
      _urlEncode(sb, NCKEY_RESERVE5, "");
      System.out.println(sb.toString());

      return sb.toString();
   }

   /* test 환경에서는 인증서 오류가 날 수도 있다. 이 코드를 이용해 인증서 오류를 회피한다. */
   private void _initHttps()
   {
      TrustManager[] trustAllCerts = new TrustManager[] {
            new X509TrustManager() {
               public void checkClientTrusted(X509Certificate[] chain, String authType)
throws CertificateException {}
               public void checkServerTrusted(X509Certificate[] chain, String authType)
throws CertificateException {}
               public X509Certificate[] getAcceptedIssuers()
               {
                  return new X509Certificate[0];
               }
            }
      };
      try {
         SSLContext sslContext = SSLContext.getInstance("SSL");
         sslContext.init(null, trustAllCerts, new SecureRandom());
```

```
            _sslSockFactory = sslContext.getSocketFactory();
        }
        catch(Exception e) {
            RuntimeException re = new RuntimeException(e);
            re.setStackTrace(e.getStackTrace());
            throw re;
        }
    }

    /**
     * @param items 주문 상품 목록.
     * @return 주문키
     * @throws IOException
     */
    public String[] sendZzimToNC(String shopId, String certificationKey, ItemStack[]
items) throws IOException
    {
        HttpURLConnection conn = (HttpURLConnection)_url.openConnection();

        /* test 환경에서는 인증서 오류가 날 수도 있다. 이 코드를 이용해 인증서 오류를 회피한다. */
        if (conn instanceof HttpsURLConnection) {
            ((HttpsURLConnection)conn).setSSLSocketFactory(_sslSockFactory);
            ((HttpsURLConnection)conn).setHostnameVerifier(
                    new HostnameVerifier() {
                        @Override
                        public boolean verify(String hostname, SSLSession session)
                        {
                            return true;
                        }
                    }
            );
        }
        conn.setDoInput(true);
        conn.setDoOutput(true);
        conn.setUseCaches(false);
        conn.setRequestMethod("POST");
        conn.addRequestProperty("Content-Type", "application/x-www-form-urlencoded;
charset=UTF-8");

        Writer writer = new OutputStreamWriter(conn.getOutputStream(), ENCODING);
        writer.write(_makeQueryString(shopId, certificationKey, items));
        writer.flush();
        writer.close();

        int respCode = conn.getResponseCode();
        if (respCode != 200) {
            throw new RuntimeException(String.format("NC Response fail : %d %s", respCode,
conn.getResponseMessage()));
        }

        BufferedReader reader = new BufferedReader(new
InputStreamReader(conn.getInputStream()));
        String retStr = reader.readLine();
        return retStr.split(",");
    }

    public static void main(String[] args) throws IOException
    {
        //주문상품 내역으로 items 데이터를 생성한다.
        List<ItemStack> items = new ArrayList<ItemStack>();
        items.add(new ItemStack("a1", "아이템1", 1000, "http://mymall.com/images/a1.jpg",
"http://mymall.com/thumbs/a1.jpg", "http://mymall.com/iteminfo?item_id=a1"));
        items.add(new ItemStack("a2", "아이템2", 1000, "http://mymall.com/images/a2.jpg",
"http://mymall.com/thumbs/a2.jpg", "http://mymall.com/iteminfo?item_id=a2"));

        JavaZzimSample sample = new JavaZzimSample("https://test-
checkout.naver.com/customer/api/wishlist.nhn");
        String[] prodSeqs = sample.sendZzimToNC("naver_checkout", "naver_checkout",
items.toArray(new ItemStack[0]));
```

```
        //여기서 얻은prodSeqs로 zzim popup을 띄운다.
        System.out.println(Arrays.toString(prodSeqs));
    }
}
```

다음은 ASP로 찜 정보를 등록하는 예이다.

```
<%
Class Itemstack
    Private mId
    Private mName
    Private mUnitPrice
    Private mItemImage
    Private mItemThumb
    Private mItemUrl

    Public Property Get Id()
        Id = mId
    End Property

    Public Property Let Id(val)
        mId = val
    End Property

    Public Property Get Name()
        Name = mName
    End Property

    Public Property Let Name(val)
        mName = val
    End Property

    Public Property Get UnitPrice()
        UnitPrice = mUnitPrice
    End Property

    Public Property Let UnitPrice(val)
        mUnitPrice = val
    End Property

    Public Property Get ItemImage()
        ItemImage = mItemImage
    End Property

    Public Property Let ItemImage(val)
        mItemImage = val
    End Property

    Public Property Get ItemThumb()
        ItemThumb = mItemThumb
    End Property

    Public Property Let ItemThumb(val)
        mItemThumb = val
    End Property

    Public Property Get ItemUrl()
        ItemUrl = mItemUrl
    End Property

    Public Property Let ItemUrl(val)
        mItemUrl = val
    End Property

    Public Property Get QueryString()
        QueryString = "ITEM_ID=" + Server.URLEncode(mId)
        QueryString = QueryString + "&ITEM_NAME=" + Server.URLEncode(mName)
        QueryString = QueryString + "&ITEM_UPRICE=" + Server.URLEncode(mUnitPrice)
        QueryString = QueryString + "&ITEM_IMAGE=" + Server.URLEncode(mItemImage)
        QueryString = QueryString + "&ITEM_THUMB=" + Server.URLEncode(mItemThumb)
        QueryString = QueryString + "&ITEM_URL=" + Server.URLEncode(mItemUrl)
    End Property
```

```
End Class

Class Zzim
    Private mShopId
    Private mCertiKey

    Public Property Get ShopId()
        ShopId = mShopId
    End Property

    Public Property Let ShopId(value)
        mShopId = value
    End Property

    Public Property Get CertiKey()
        CertiKey = mCertiKey
    End Property

    Public Property Let CertiKey(value)
        mCertiKey = value
    End Property

    Public Property Get QueryString()
        QueryString = "SHOP_ID=" + Server.URLEncode(mShopId)
        QueryString = QueryString + "&CERTI_KEY=" + Server.URLEncode(mCertiKey)
        QueryString = QueryString + "&RESERVE1="
        QueryString = QueryString + "&RESERVE2="
        QueryString = QueryString + "&RESERVE3="
        QueryString = QueryString + "&RESERVE4="
        QueryString = QueryString + "&RESERVE5="
    End Property

End Class

Dim myZzim, items(0), queryString
Set myZzim = new Zzim
myZzim.ShopId = "naver_checkout"
myZzim.CertiKey = "naver_checkout"

Set items(0) = new ItemStack
items(0).Id = "items2"
items(0).Name = "아이템2"
items(0).UnitPrice = "100"
items(0).ItemImage = "http://mymall.com/image/items2.jpg"
items(0).ItemThumb = "http://mymall.com/thumb/items2.jpg"
items(0).ItemUrl = "http://mymall.com/item?itemid=items2"

queryString = myZzim.QueryString
For Each item In items
    queryString = queryString + "&" + item.QueryString
Next

Dim objXMLHTTP, sXML
set objXMLHTTP = server.CreateObject("MSXML2.ServerXMLHTTP")
objXMLHTTP.setOption 2,13056 'ignore SSL errors
objXMLHTTP.Open "POST", "https://test-checkout.naver.com/customer/api/wishlist.nhn",
false
'objXMLHTTP.Open "POST", "https://test-
checkout.naver.com/customer/api/CP949/wishlist.nhn", false
objXMLHTTP.setRequestHeader "Content-Type", "application/x-www-form-
urlencoded;charset=utf-8"
'objXMLHTTP.setRequestHeader "Content-Type", "application/x-www-form-
urlencoded;charset=CP949"
objXMlHTTP.Send queryString

sXML = objXMLHTTP.responseText

If objXMLHTTP.status = 200 Then
       '상품이 하나일 때
         itemId = objXMLHTTP.responseText
```

```
      '상품이 여러 개일 때
        'itemIds = objXMLHTTP.responseText
        'itemIdList = split(trim(itemIds),",")

Else
    '에러처리
End if


Response.Write(itemId)

wishlistPopupUrl = "https://test-checkout.naver.com/customer/wishlistPopup.nhn"
%>
<html>
<body>
<form name="frm" method="get" action="<%=wishlistPopupUrl%>">
<input type="hidden" name="SHOP_ID" value="<%=myZzim.ShopId%>">
<!-- 상품이 하나일 때 -->
<input type="hidden" name="ITEM_ID" value="<%=itemId%>">

<!-- 상품이 여러 개일 때
<% For i = 0 To UBound(itemIdList) %>
   <input type="hidden" name="ITEM_ID" value="<%=itemIdList(i)%>">
<% Next %>
-->

</form>
</body>
<script>

<% If objXMLHTTP.status = 200 Then %>
document.frm.target = "_top";
document.frm.submit();
<% End if

Set objXMLHTTP = Nothing
%>
</script>
</html>
```

3.3.3 찜 목록 전송

네이버 체크아웃 찜 목록 전송 팝업 창을 생성하여 GET 인자로 아래의 정보를 전송한다. ITEM_ID는 여러 개일 수 있다.

표 3-9 찜 목록 전송 팝업 창 URL

환경	URL
테스트	https://test-checkout.naver.com/customer/wishlistPopup.nhn
서비스	https://checkout.naver.com/customer/wishlistPopup.nhn

표 3-10 찜 목록 전송 내용

항목	필수 여부	설명
SHOP_ID	Y	상점 ID. 네이버 체크아웃에 가입 승인될 때 정해진다.
ITEM_ID	Y	네이버 체크아웃 상품 ID. 찜 정보를 등록하고 response 로 받는다.

3.4 배송 정보 연동(옵션 개발 사항)

3.4.1 배송 정보 연동

배송 정보 연동이란 가맹점 담당자가 주문 처리 시 배송 방법을 '업체별 배송' 으로 설정한 주문에 대해서, 네이버 체크아웃은 해당 주문 번호의 상품에 대한 배송 정보를 각 가맹점으로 요청하고, 가맹점은 네이버 체크아웃의 요청에 따라 필요한 배송 정보를 송신하는 과정을 말한다.

- 몰인몰 형태 가맹점(하나의 주문에 대해 둘 이상의 배송 업체가 존재하는 가맹점)은 반드시 배송 정보 연동을 개발해야 한다.
- 몰인몰이 아닌 일반 독립몰은 각 가맹점의 판단에 따라 개발 여부를 결정한다.

개발 시에는 반드시 사전에 네이버 체크아웃 담당자와의 협의 및 확인을 거쳐야 한다.

3.4.2 배송 정보 요청

네이버 체크아웃은 몰인몰 업체 주문의 배송을 확인하기 위해 HTTP로 배송 정보 요청을 보낸다.
가맹점은 요청받은 배송 정보를 XML로 전송하는 페이지를 생성해야 한다.

네이버 체크아웃이 배송 정보를 요청하는 형식은 다음과 같다.

```
http://배송조회URL?ORDER_ID=XXX&ITEM_ID=YYY&ITEM_ID=ZZZ
```

표 3-11 배송 정보 요청 내용

항목	필수 여부	설명
ORDER_ID	Y	주문 번호
ITEM_ID	Y	상품 ID

위의 주문 번호는 구매자가 네이버 체크아웃으로 구매했을 때 네이버 체크아웃이 발행한 주문 번호이다. 가맹점이 관리하는 주문 번호가 아니므로 주의한다.

3.4.3 배송 정보 제공

가맹점은 XML로 배송 정보 목록을 전송한다.

필수 항목은 모두 element로 명시해야 하며, URL 값은 반드시 유효해야 한다.

표 3-12 배송 정보 제공 내용

항목	필수 여부	설명
deliveries	Y	배송 정보 목록
deliveries/delivery*	Y	배송 정보 단위
deliveries/delivery/itemID	Y	상품 번호

deliveries/delivery/companyName	Y	배송 회사
deliveries/delivery/trackingNumber	N	송장 번호
deliveries/delivery/status	Y	배송 상태
deliveries/delivery/sellerName	Y	판매자 이름
deliveries/delivery/senderAddress	Y	발송지 주소
deliveries/delivery/senderName	Y	발송자 이름
deliveries/delivery/returnAddress	Y	반품지 주소
deliveries/delivery/returnName	Y	반품 수령인 이름
deliveries/delivery/statusURL	N	배송 상세 조회 URL
deliveries/delivery/statusURL@type	N	배송 상세 조회 오픈 방식. popup 이면 팝업 창, new 이면 새 창. 기본은 새 창.
deliveries/delivery/statusURL@width	N	type 이 popup 인 경우 팝업 창의 너비
deliveries/delivery/statusURL@height	N	type 이 popup 인 경우 팝업 창의 높이

배송 회사에는 배송 서비스를 이용하는 택배사 이름을 입력한다. 각 택배사 이름은 다음과 같이 표기하는 것을 권장한다.

```
대한통운 | 한진택배 | 로젠택배 | 현대택배 | KG옐로우캡택배 | KGB택배 | EMS | DHL | 한덱스 | Fedex | 동부익스프레스 | CJGLS | SC로지스 | UPS | 하나로택배 | 대신택배
```

배송 방법으로 택배가 아닌 퀵 서비스, 화물 배달 등을 이용한다면, 사용하는 배송 수단을 입력해야 한다. 각 배송 수단은 다음과 같이 표기하는 것을 권장한다.

```
퀵 서비스 | 직배송 | 방문수령
```

배송 상태에는 각 가맹점의 기준에 따른 배송 상태를 입력한다. 각 배송 상태는 다음과 같이 표기하는 것을 권장한다.

```
집화 예정 | 배송 중 | 배송 완료
```

송장 번호(trackingNumber)와 배송 상세 조회 URL(statusURL), 배송 상세 조회 오픈 방식(statusURL@type)은 택배사를 이용하지 않고 퀵 서비스, 화물 배달 등을 이용하는 경우에 한해서 입력하지 않아도 된다.

판매자 이름에는 해당 상품의 판매자 이름 또는 상호를 입력한다.

배송 상세 조회 URL에는 해당 배송의 상세 추적을 위해 가맹점이 제공하는 페이지의 URL을 입력한다. (자체 페이지가 존재하면 자체 페이지의 URL, 택배사 페이지를 이용하면 택배사의 해당 배송 조회 페이지 URL)

배송 상세 조회 오픈 방식은 해당 배송의 상세 추적을 위해 제공하는 페이지를 오픈하는 방식을 의미한다. 오픈 시 새 창이 나타나게 하려면 new를 입력한다. 지정된 크기의 팝업 창이 나타나게 하려면 popup을 입력하고, 팝업 창의 너비(@width)와 높이(@height) 값을 입력한다.

문자 세트는 각 환경에 맞게 설정하고, content-type은 application/xml로 한다.

```
<?xml version="1.0" encoding="UTF-8">
<deliveries>
   <delivery> <!-- 배송 정보 -->
      <itemID><![CDATA[ID001]]></itemID> <!-- 상품 번호 -->
      <companyName><![CDATA[배송 회사]]></companyName>
      <trackingNumber><![CDATA[1234567890]]></trackingNumber> <!-- 송장번호 -->
      <status><![CDATA[배송중]]></status> <!-- 배송상태 -->
      <sellerName><![CDATA[홍길동]]></sellerName> <!-- 판매자 명-->
      <senderAddress><![CDATA[경기도 성남시 분당구 서현동 XXX-XX]]></senderAddress> <!-- 발송지
주소-->
      <senderName><![CDATA[홍길동]]></senderName> <!-- 발송자 명-->
      <returnAddress><![CDATA[경기도 성남시 분당구 서현동 XXX-XX]]></returnAddress> <!-- 반품지
주소-->
      <returnName><![CDATA[홍길동]]></returnName> <!-- 반품 수령인 명-->
      <statusURL type="popup" width="100" height="200"
><![CDATA[http://testmall.com/deliveryPopup.php?]]></statusURL> <!-- 배송상세 조회 URL-->
   </delivery>
   <!-- 배송이 여러 개 있는 경우 여러 개의 delivery 를 순차적으로 나열한다. -->
</deliveries>
```