

NICEPAY 스마트폰 매뉴얼

문서번호
Ver.3.0.4

[illegible]

목 차

1. 제품의 개요 및 특징	5
1.1. 제품의 개요	5
1.2. 결제 흐름 구성도	5
1.2.1. 스마트 폰 결제 Flow	5
1.2.2. 신용카드 결제 화면 흐름도	6
1.2.3. 신용카드 결제 화면 기타	6
1.2.4. 계좌이체 결제 화면 흐름도	7
1.2.5. 휴대폰 결제 화면 흐름도	7
1.2.6. 가상계좌 결제 화면 흐름도	8
1.3. 제품의 특징	8
2. 결제 요청 페이지 연동	9
2.1. JSP 결제 요청 페이지 개발	10
2.2. ASP 결제 요청 페이지 개발	11
2.3. PHP 결제 요청 페이지 개발	13
2.4. .NET 결제 요청 페이지 개발	15
2.5. 결제 수단 설정 안내	16
2.6. 나이스 간편 결제 연동 안내	17
2.7. 결제 요청 안내 및 네이버 앱등 팝업 차단에 대한 안내	17
2.8. UTF-8 캐릭터셋 사용시 안내	17
2.8.1. PHP TX 사용시	18
2.8.2. JSP TX 사용시	18
2.9. 결제 요청 페이지 전문	19
3. 결제 결과 처리 연동	20
3.1. JSP 결제 결과 처리 연동	20
3.2. ASP 결제 결과 처리 연동	22
3.3. PHP 결제 결과 처리 연동	24
3.4. .NET 결제 결과 처리 연동	26
3.5. 결제 승인 전문 내역 (공통)	29
3.5.1. 카드 결제 승인 전문	29
3.5.2. 계좌이체 및 SSG은행계좌 결제 승인 전문	30
3.5.3. 가상계좌 결제 승인 전문	30
3.5.4. 휴대폰 결과 파라미터	30
4. 안드로이드 앱 연동 가이드	30
4.1. 링크 상수 및 변수 선언	30

4.2.	웹뷰의 속성 설정	31
4.3.	안드로이드 targetSDK에 따른 쿠키 및 기타 설정 처리	31
4.4.	웹뷰의 캐시사용 설정 변경 금지에 대한 안내	31
4.5.	웹뷰 onCreate url 처리	32
4.6.	웹뷰에서의 override 처리	32
5.5	계좌이체 이벤트 설정	36
4.7.	스키마 설정	37
5.	iPhone APP 개발 가이드	38
5.1.	웹뷰의 설정	38
5.1.1.	Mobile ISP APP 호출하는 경우	38
5.1.2.	BankPay App 을 호출하는 경우	38
5.1.3.	Mobile ISP , BankPay(계좌이체) APP 설치	39
5.2.	스키마 설정	39
5.3.	인증 결과 처리	39
5.3.1.	Mobile ISP APP 에서 호출 되는 경우	39
5.3.2.	BankPay 에서 호출 되는 경우	40
5.4.	Whitelist 설정	41
6.	가맹점 ID, KEY 확인	44
7.	가맹점 거래취소 비밀번호 등록	44
8.	기타	45
8.1.	결제수단별 결과코드	45
8.1.1.	신용카드	45
8.1.2.	계좌이체	46
8.1.3.	가상계좌	46
8.1.4.	휴대폰결제	46
8.1.5.	SSG은행계좌 결제	47
8.1.6.	결제취소	47
8.2.	신용카드 코드	48
8.3.	계좌이체 은행 코드	49

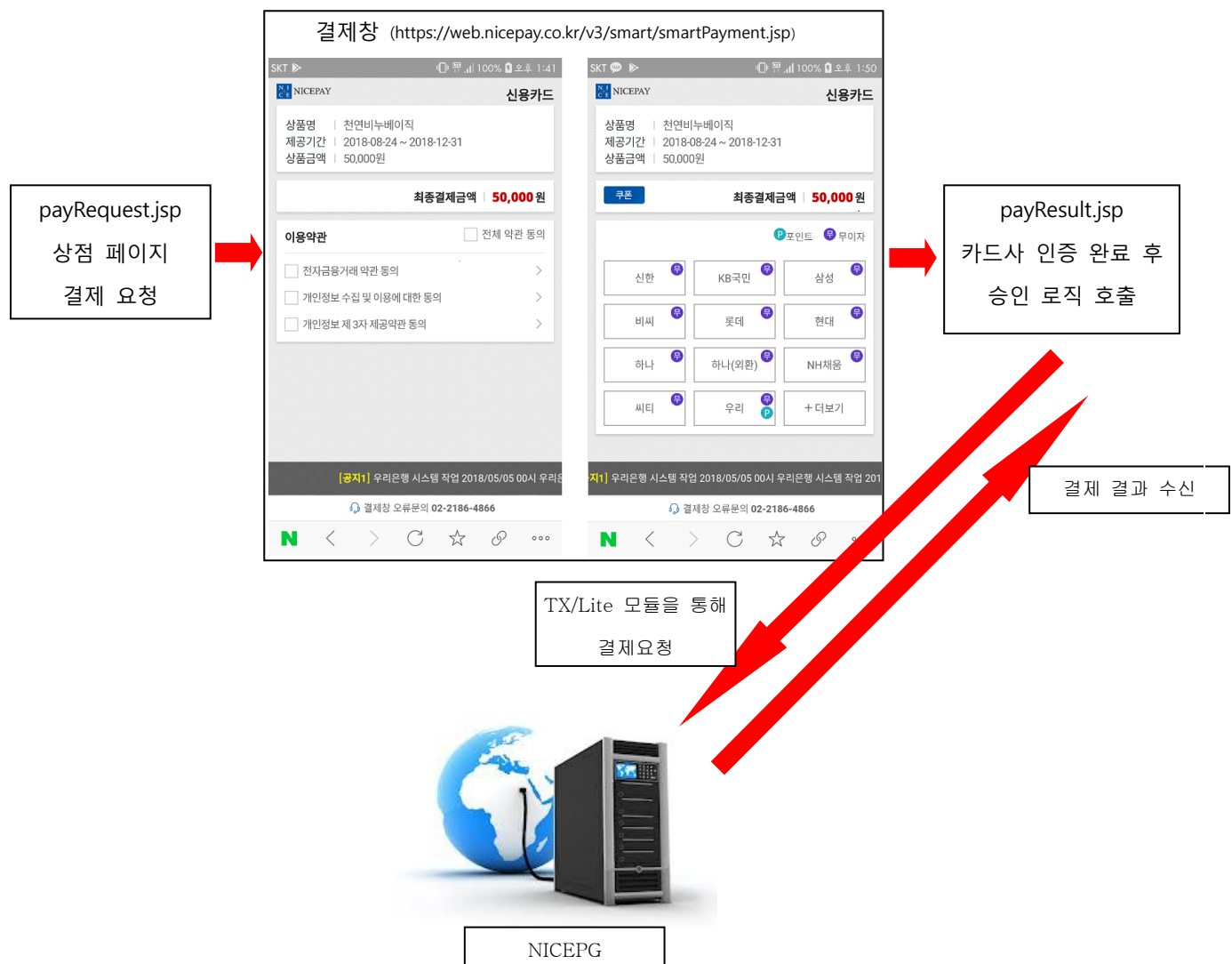
1. 제품의 개요 및 특징

1.1. 제품의 개요

NICEPAY 스마트폰 지불 시스템은 스마트폰 환경에서 간단히 결제를 연동하여 구성 할 수 있도록 웹페이지 링크방식의 결제 연동 시스템을 제공합니다.

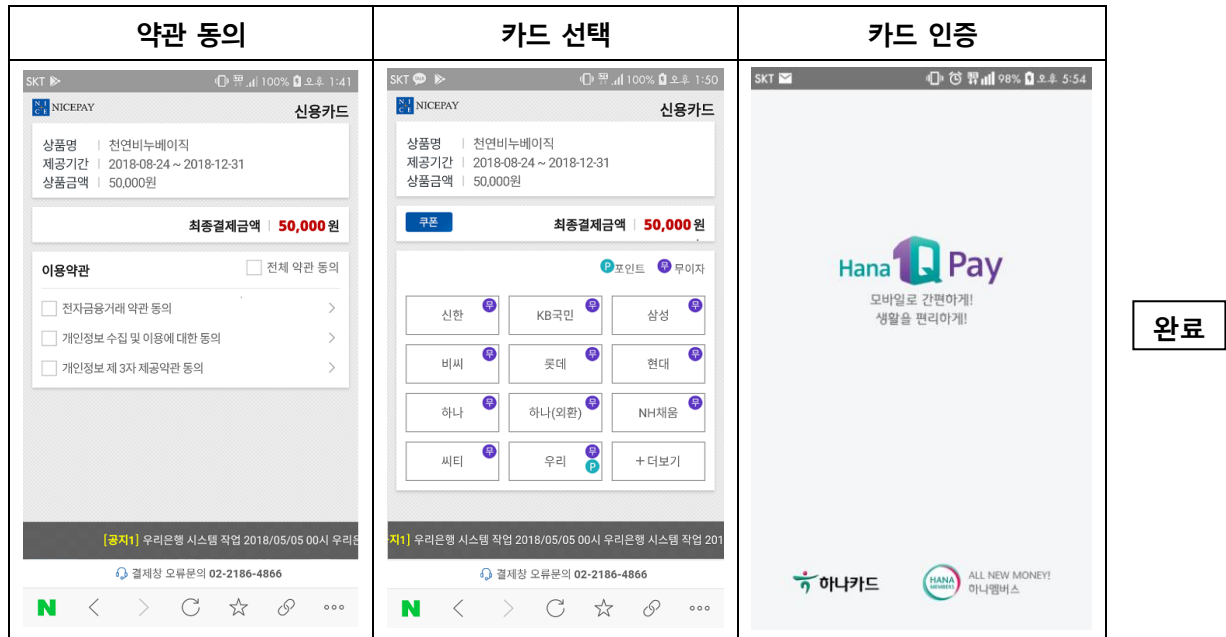
1.2. 결제 흐름 구성도

1.2.1. 스마트 폰 결제 Flow



1. 결제요청 : 상점 페이지에서 결제를 진행하는 단계입니다. (구매)
2. NicePay Smart 결제창이 호출되게 됩니다.
3. 결제 수단에 맞도록 카드 인증 또는 계좌이체등의 정보를 통해 인증을 하는 단계입니다.
4. 인증후 가맹점 returnUrl로 고유인증키를 요청후, 나이스 모듈에서 결과를 송수신하여 페이지에 처리합니다.

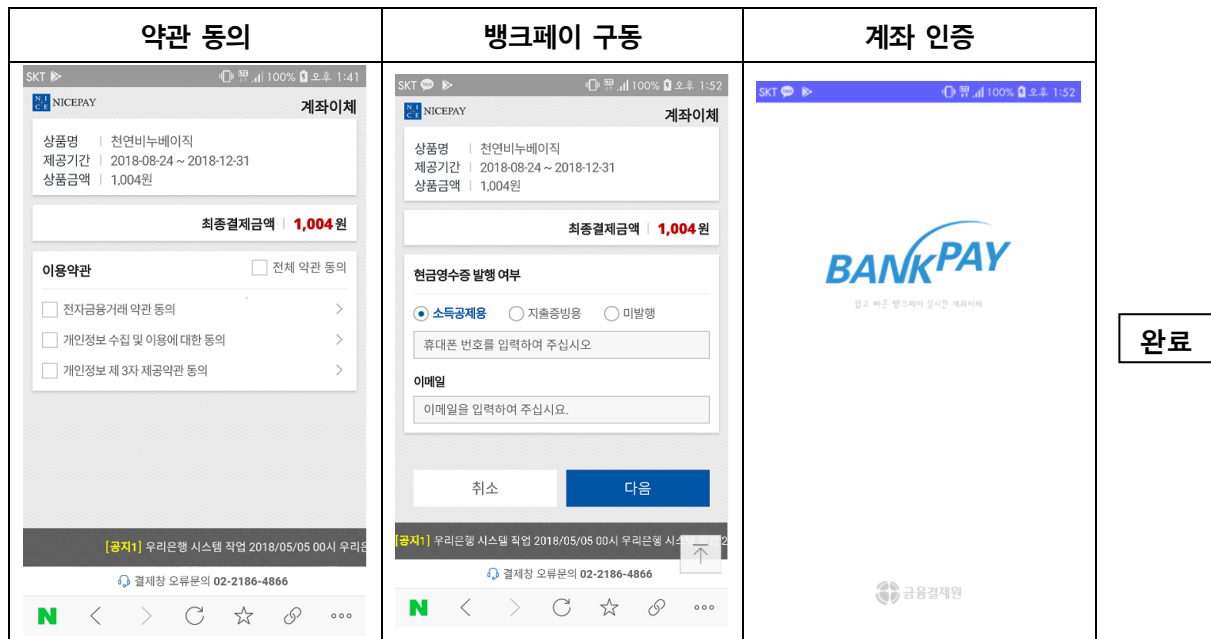
1.2.2. 신용카드 결제 화면 흐름도



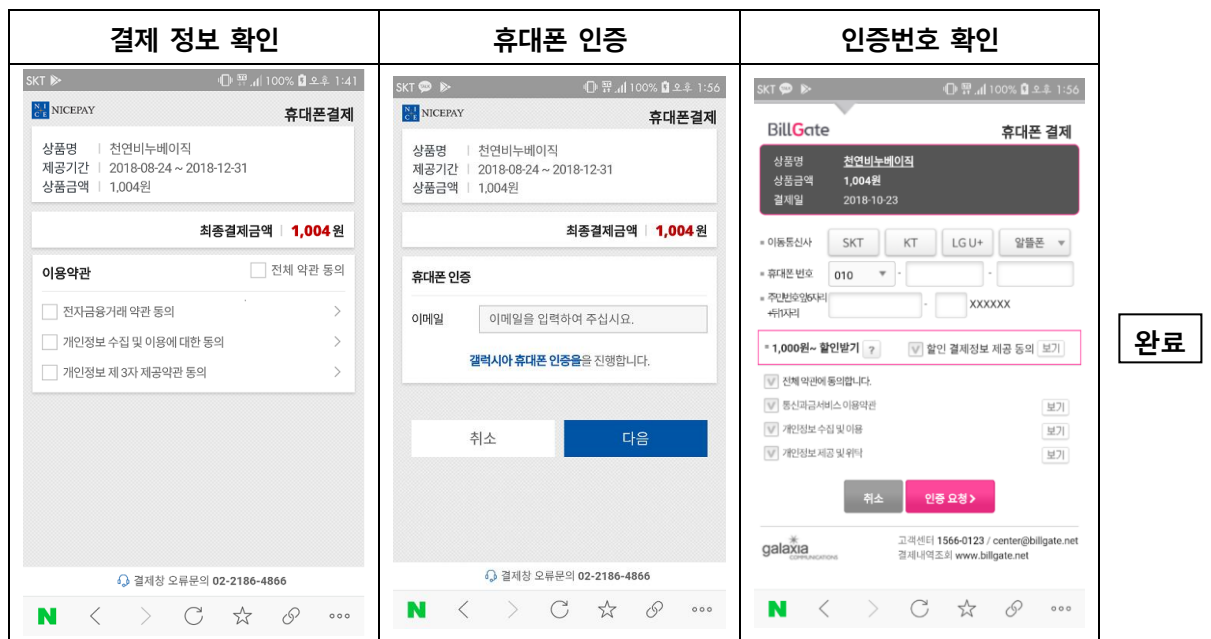
1.2.3. 신용카드 결제 화면 기타



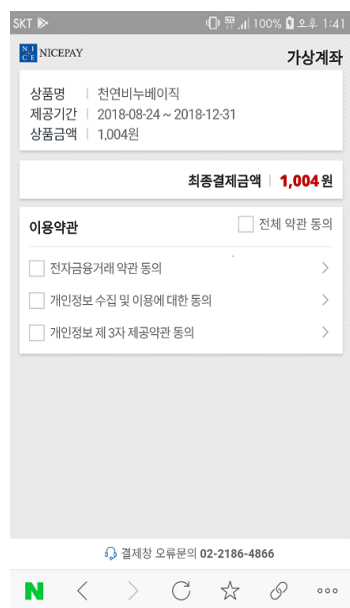

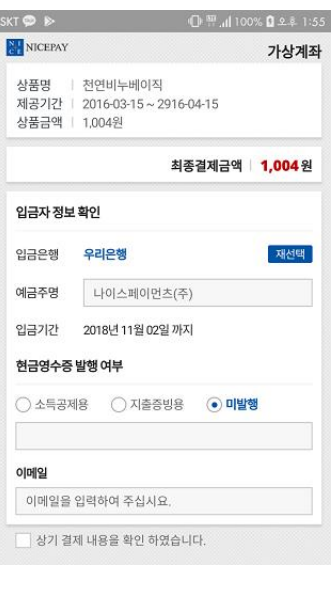
1.2.4. 계좌이체 결제 화면 흐름도



1.2.5. 휴대폰 결제 화면 흐름도



1.2.6. 가상계좌 결제 화면 흐름도

약관 동의	은행 선택	현금영수증확인
 <p>SKT > NICEPAY 가상계좌</p> <p>상품명 천연비누배이직 제공기간 2018-08-24 ~ 2018-12-31 상품금액 1,004원</p> <p>최종결제금액 1,004 원</p> <p>이용약관 <input type="checkbox"/> 전체 약관 동의</p> <p><input type="checkbox"/> 전자금융거래 약관 동의 > <input type="checkbox"/> 개인정보 수집 및 이용에 대한 동의 > <input type="checkbox"/> 개인정보 제 3자 제공약관 동의 ></p> <p>결제창 오류문의 02-2186-4866</p>	 <p>SKT > NICEPAY 가상계좌</p> <p>상품명 천연비누배이직 제공기간 2018-08-24 ~ 2018-12-31 상품금액 1,004원</p> <p>최종결제금액 1,004 원</p> <p>기업은행 (00:30~23:30) 국민은행 (00:30~23:30) 외환은행 (01:00~22:45) 농협중앙회 (00:30~23:30) 우리은행 (00:30~23:30) 하나은행 (00:30~23:30) 신한은행 (00:30~23:30) 대구은행 (08:00~23:00) 부산은행 (00:30~23:30) 우체국 (00:30~23:30)</p> <p>결제창 오류문의 02-2186-4866</p>	 <p>SKT > NICEPAY 가상계좌</p> <p>상품명 천연비누배이직 제공기간 2016-03-15 ~ 2916-04-15 상품금액 1,004원</p> <p>최종결제금액 1,004 원</p> <p>입금자 정보 확인</p> <p>입금은행 우리은행 재선택</p> <p>예금주명 나이스페이먼츠(주)</p> <p>입금기간 2018년 11월 02일 까지</p> <p>현금영수증 발행 여부</p> <p><input type="radio"/> 소특공제용 <input type="radio"/> 지출증빙용 <input checked="" type="radio"/> 미발행</p> <p>이메일 <input type="text"/></p> <p>이메일을 입력하여 주십시오.</p> <p><input type="checkbox"/> 상기 결제 내용을 확인 하였습니다.</p>

완료

1.3. 제품의 특징

NICEPAY 스마트폰 전자 결제 시스템은 부가적인 보안장치 없이 독자적인 위·변조방지를 위한 암호화를 적용함으로써 안전하게 결제 처리를 할 수 있습니다.

NICEPAY 스마트 전자 결제 시스템은 다음과 같은 결제 수단을 제공합니다.

결제수단	설명
신용카드	국내 또는 해외에서 발급된 카드를 이용한 전자 결제
ISP안전결제	카드사에 발급받은 인증서를 이용한 전자 결제 (비씨카드, 국민카드)
실시간 계좌이체	인터넷뱅킹용 공인인증서를 이용하여 실시간 계좌 이체
가상계좌이체	대금입금을 위해 계좌번호(예금주: 나이스정보통신) 를 부여 받는 전자 결제
휴대폰결제	휴대폰으로 물품 구입을 위한 금액을 지불하고, 휴대폰 요금으로 과금되는 전자 결제

2. 결제 요청 페이지 연동

※ 중요

1) 환경설정 파일

- 환경설정 파일의 경우 외부에서 접근이 가능한 경로에 두시면 안됩니다.
- 환경파일이 외부에 노출이 되는 경우 해킹의 위험이 존재하므로 반드시 외부에서 접근이 불가능한 경로에 두시기 바랍니다.

예) [Window 계열] C:\inetpub\wwwroot\wnicepay 외부 접근 절대불가(웹 디렉토리)

2) 해시검증

- 결제요청 시에 결제창이 호출되면 해시데이터 위변조 체크하는 로직이 있습니다. 샘플 소스를 참고하여 연동 부탁 드립니다.

- 또한, 금액의 경우 인증 완료 시에 "Amt" 필드에 리턴됩니다. 주문번호 필드인 "Moid"를 기준으로 하여 원상품가격과 결제결과금액과 비교하여 금액이 동일하지 않다면 결제 금액의 위변조가 의심됨으로 정상적인 처리가 되지 않도록 처리 부탁 드립니다.

- 해당부분은 결제 결과 페이지에 해당 부분 추가하시면 됩니다.

2.1. JSP 결제 요청 페이지 개발

상점 페이지에서 결제 UI와 연동하기 위해서는 다음과 같은 작업이 필요합니다.
(payRequest.jsp 참고)

- 상점키 및 상점 ID 설정

: 상점(회원사) ID에 맞는 고유의 키 및 상점 ID를 설정합니다.
상점키는 계약시 발급됩니다.

```
String merchantKey = "{상점키}"
String merchantID = "{상점ID}";
...
<input type="hidden" name="MID" value="<%=merchantID%>" />
*merchantID와 MID의 value는 동일하게 설정해야 합니다.
```

- 상품 가격 설정

```
String price = "1000";
....
<input type="hidden" name="Amt" value="<%= price%>" />
*price 와 Amt의 value는 동일하게 설정해야 합니다.
```

- 상품명 / 주문 번호 설정

```
<!-- 주문번호 -->
<input type="hidden" name="Moid" value="merchant_15690" />
<!-- 상품명 -->
<input type="hidden" name="GoodsName" value="테스트상품" />
```

- 구매자 정보 설정

```
<!-- 구매자 명 -->
<input type="hidden" name="BuyerName" value="홍길동" />
<!-- 구매자 전화번호 -->
<input type="hidden" name="BuyerTel" value="00000000000" />
<!-- 구매자 이메일 주소 -->
<input type="hidden" name="BuyerEmail" value="test@abc.com" />
<!-- 구매자 주소 -->
<input type="hidden" name="BuyerAddr" value="서울시 마포구 아현동 689" />
```

- 결제 결과 페이지 및 처리 페이지 설정

```
<!-- 결과를 처리할 url을 지정하십시오. -->
<input type="hidden" name="ReturnURL"
value="http://aaa.com/smart/payResult.jsp">
```

- 상점 파라미터 전달

```
<!-- 상점에서 여분으로 사용할 값을 지정하여 주십시오. 그대로 전달됩니다 -->
<input type="hidden" name="MallReserved" value=""/>
```

- 고정 파라미터 사용

```
<input type="hidden" name="EncryptData" value="<%=hash_String%"/>
<input type="hidden" name="ediDate" value="<%=ediDate%"/>
<input type="hidden" name="TrKey" value=""/>
<input type="hidden" name="MallIP" value="<%=inet.getHostAddress()%"/>
<input type="hidden" name="AcsNoIframe" value="Y"/>
```

2.2. ASP 결제 요청 페이지 개발

상점 페이지에서 결제 UI와 연동하기 위해서는 다음과 같은 작업이 필요합니다.
(payRequest.asp 참고)

- 상점키 및 상점 ID 설정
: 상점(회원사) ID에 맞는 고유의 키 및 상점 ID를 설정합니다.
상점키는 계약시 발급됩니다.

```
' 상점서명키 (꼭 해당 상점키로 바꿔주세요)'
merchantKey = "{해당상점키}"
' 상점 ID
merchantID = "{해당상점아이디}" ...
<input type="hidden" name="MID" value="<%=merchantID%"/>
```

- 상품 가격 설정

```
' 가격
price = "1004"....
<input type="hidden" name="Amt" value="<%= price%"/>
*price 와 Amt의 value는 동일하게 설정해야 합니다.
```

- 상품명 / 주문 번호 설정

```
<!-- 주문번호 -->
<input type="hidden" name="Moid" value="merchant_15690"/>
<!-- 상품명 -->
<input type="hidden" name="GoodsName" value="테스트상품"/>
```

- 구매자 정보 설정

```
<!-- 구매자 명 -->
<input type="hidden" name="BuyerName" value="홍길동"/>
<!-- 구매자 전화번호 -->
<input type="hidden" name="BuyerTel" value="000000000000"/>
<!-- 구매자 이메일 주소 -->
<input type="hidden" name="BuyerEmail" value="test@abc.com"/>
<!-- 구매자 주소 -->
<input type="hidden" name="BuyerAddr" value="서울시 마포구 아현동 689"/>
```

- 결제 결과 페이지 및 처리 페이지 설정

```
<!-- 결과를 처리할 url을 지정하십시오. -->
<input type="hidden" name="ReturnURL"
value="http://aaa.com/smart/payResult.jsp">
```

- 상점 파라미터 전달

```
<!-- 상점에서 여분으로 사용할 값을 지정하여 주십시오. 그대로 전달됩니다 -->
<input type="hidden" name="MallReserved" value=""/>
```

- 고정 파라미터 사용

```
<input type="hidden" name="EncryptData" value="<%=hash_String%>"/>
<input type="hidden" name="ediDate" value="<%=ediDate%>"/>
<input type="hidden" name="TrKey" value=""/>
<input type="hidden" name="AcsNoIframe" value="Y"/>
```

2.3. PHP 결제 요청 페이지 개발

상점 페이지에서 결제 UI와 연동하기 위해서는 다음과 같은 작업이 필요합니다.
(payRequest.php 참고)

- 상점키 및 상점 ID 설정
: 상점(회원사) ID에 맞는 고유의 키 및 상점 ID를 설정합니다.
상점키는 계약시 발급됩니다.

```
<?
// 전문생성일시
$ediDate = date("YmdHis");

// 상점서명키 (꼭 해당 상점키로 바꿔주세요)
$merchantKey =
"33F49GnCMS1mFYlGXisbUDzVf2ATWCI9k3R++d5hDd3Frmuos/XLx8XhXpe+LDYAbpG
KZYSwtlyyLOtS/8aD7A==";

// hash 처리
$MerchantID = "nicetest00m";
$price = "1004";
```

- 상품 가격 설정

```
$price = "1004";
<input type="hidden" name="Amt" value="<?=$price?>">
*price 와 Amt의 value는 동일하게 설정해야 합니다.
```

- 상품명 / 주문 번호 설정

```
<!-- 주문번호 -->
<input type="hidden" name="Moid" value="merchant_15690"/>
<!-- 상품명 -->
<input type="hidden" name="GoodsName" value="테스트상품"/>
```

- 구매자 정보 설정

```
<!-- 구매자 명 -->
<input type="hidden" name="BuyerName" value="홍길동"/>
<!-- 구매자 전화번호 -->
<input type="hidden" name="BuyerTel" value="000000000000"/>
<!-- 구매자 이메일 주소 -->
<input type="hidden" name="BuyerEmail" value="test@abc.com"/>
<!-- 구매자 주소 -->
<input type="hidden" name="BuyerAddr" value="서울시 마포구 아현동
689"/>
```

- 결제 결과 페이지 및 처리 페이지 설정

```
<!-- 결과를 처리할 url을 지정하십시오. -->
<input type="hidden" name="ReturnURL"
value="http://aaa.com/smart/payResult.php">
```

- 상점 파라미터 전달

```
<!-- 상점에서 여분으로 사용할 값을 지정하여 주십시오. 그대로 전달됩니다 -->
<input type="hidden" name="MallReserved" value=""/>
```

- 고정 파라미터 사용

```
<input type="hidden" name="EncryptData" value="<?=$hash_String?>"/>
<input type="hidden" name="ediDate" value="<?=$ediDate?>"/>
<input type="hidden" name="TrKey" value=""/>
<input type="hidden" name="AcsNoIframe" value="Y"/>
```

2.4. .NET 결제 요청 페이지 개발

상점 페이지에서 결제 UI와 연동하기 위해서는 다음과 같은 작업이 필요합니다.

(payRequest.aspx / payRequest.aspx.cs)

- 상점키 및 상점 ID 설정

: 상점(회원사) ID에 맞는 고유의 키 및 상점 ID를 설정합니다.

상점키는 계약시 발급됩니다.

```
// aspx 페이지 참고, 상점아이디에 맞는 상점 키를 반드시 확인 후 적용 바랍니다.  
merchantKey = "{상점 키}";  
merchantID = "{상점아이디}";  
  
<!--aspx.cs 참고, merchantID와 MID의 value는 동일하게 설정해야 합니다. -->  
<input type="hidden" name="MID" value="<%=merchantID%>">
```

- 상품 가격 설정

```
// aspx 참고  
price = "1004";  
  
<!--aspx.cs 참고, price 와 Amt의 value는 동일하게 설정해야 합니다. -->  
<input type="hidden" name="Amt" value="<%=price%>">
```

- 상품명 / 주문 번호 설정

```
<!-- 주문번호, aspx 페이지에서 설정한 값을 가져옵니다. -->  
<input type="hidden" name="Moid" value="<%=moid%>">  
  
<!-- 상품명, aspx 페이지에서 설정한 값을 가져옵니다. -->  
<input type="hidden" name="GoodsName" value="<%=goodsName%>">
```

- 구매자 정보 설정

```
<!-- 구매자 명, aspx 페이지에서 설정한 값을 가져옵니다. -->  
<input type="hidden" name="BuyerName" value="<%=buyerName%>">  
  
<!-- 구매자 전화번호, aspx 페이지에서 설정한 값을 가져옵니다. -->  
<input type="hidden" name="BuyerTel" value="<%=buyerTel%>">  
  
<!-- 구매자 이메일 주소, aspx 페이지에서 설정한 값을 가져옵니다. -->  
<input type="hidden" name="BuyerEmail" value="<%=buyerEmail%>">
```

- 결제 결과 페이지 및 처리 페이지 설정

```
// aspx.cs 참고.
returnURL = "{상점 결제 결과 페이지}";
<!-- aspx.cs 참고, 결과 처리 URL -->
<input type="hidden" name="ReturnURL" value="<%=returnURL%>">
```

- 상점 여분 파라미터 전달

```
<!-- 500byte 이내로 상점여분필드가 제공됩니다. -->
<input type="hidden" name="MallReserved" value=""/>
```

- 고정 파라미터 사용(변경 불가)

```
<!-- aspx 페이지에서 설정한 값을 가져옵니다. -->
<input type="hidden" name="EncryptData" value="<%=hash_String%>" />
<input type="hidden" name="ediDate" value="<%=ediDate%>" />
<input type="hidden" name="TrKey" value=""/>
<input type="hidden" name="AcsNoIframe" value="Y"/>
```

2.5. 결제 수단 설정 안내

결제 수단 설정은 단일 결제로 구성할 수 있습니다.

- 1) 신용카드로 단일 결제 시 설정

```
<input type="hidden" name="PayMethod" value="CARD">
```

* 결제수단 설정표

결제 수단	설정명
신용카드	CARD
계좌이체	BANK
가상계좌	VBANK
휴대폰	CELLPHONE
SSG 은행계좌	SSG_BANK

2.6. 나이스 간편 결제 연동 안내

나이스 간편 결제 앱을 바로 구동시키는 방법은 결제 요청 페이지에 아래 파라미터를 추가로 설정합니다.

*단 결제수단은 신용카드로 설정해야 합니다.

<!--나이스 간편 결제사용 -->

<input type="hidden" name="SelectCardCode" value="NICEAPP">

2.7. 결제 요청 안내 및 네이버 앱등 팝업 차단에 대한 안내

스마트폰 결제 요청은 아래와 같이 합니다.

```
function goPay(form) {  
    form.target = "_self";  
    form.method = "post";  
    form.action = "https://web.nicepay.co.kr/v3/smart/smartPayment.jsp"  
    form.submit();  
}
```

만약 네이버 앱등에서 구동할 경우 form.target 을 _self로 변경하여 주십시오.

2.8. UTF-8 캐릭터셋 사용시 안내

utf-8 사용시에는 goPay 함수에 document.charset = 'euc-kr' 을 추가합니다.

```
function goPay(form) {  
    document.charset = 'euc-kr';  
    form.method = "post";  
    form.action = "https://web.nicepay.co.kr/v3/smart/smartPayment.jsp"  
    form.submit();  
}
```

Form 태그에 accept-charset="euc-kr" 구문을 추가하여 주십시오.

Ex)

<form name="tranMgr" accept-charset="euc-kr">

결과 처리 페이지에서 utf-8로 결과를 받고자 할 경우 아래와 같이 설정합니다.

<input type="hidden" name="CharSet" value="utf-8"/>

2.8.1. PHP TX 사용시

- 결과 페이지 아래 구문을 추가하여 utf-8 처리를 합니다.

```
$nicepayWEB->setParam("CHARSET", "UTF8");           // utf-8 사용시 옵션 설정  
$resultMsg = $responseDTO->getParameterUTF("ResultMsg"); // utf-8 사용시 한글 메시지는  
다음과 같이 받으실 수 있습니다.
```

2.8.2. JSP TX 사용시

- 결과 페이지는 utf-8타입으로 저장하시고 아래 구문을 추가하여 주십시오.

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>  
<%  
    request.setCharacterEncoding("UTF-8");//한글이 깨지지 않도록 UTF-8 으로 받음.
```

2.9. 결제 요청 페이지 전문

영문명	한글명	필수여부	크기	상세내용
GoodsCnt	상품개수	Y	1 byte	1
Moid	상점 주문번호	Y	64 byte	상점에서 사용하는 주문번호
MallUserID	상점 사용자 아이디	N	20 byte	상점에서 사용하는 사용자 아이디
BuyerName	구매자명	Y	30 byte	구매자 이름
BuyerTel	구매자 전화번호	Y	40 byte	구매자 전화번호
BuyerEmail	구매자 이메일	Y	60 byte	구매자 이메일
BuyerAddr	구매자 주소	N	100 byte	구매자 주소
MallReserved	상점 예약필드	N	500 byte	상점 정보 전달용 예비필드
ReturnURL	결제 처리 URL	Y		결제 처리 후 forwarding 되는 url
EncryptData	암호처리 예비필드	N		내부 사용하는 필드로 항목은 존재해야함
ediDate	승인 일자	Y	30 byte	승인일자로 반드시 설정해야함
WapUrl	App scheme 값	N		APP 내 WebView 로 연동시, ISP 와 계좌이체 인증 후 다시 가맹점 APP 으로 돌아오기 위한 가맹점 APP Schemes 를 설정. 예) nicepaysample://
IspCancelUrl	ISP 취소 시 이동 URL (APP 연동인 경우만 사용)	N		ISP APP 에서 취소 버튼 터치 시에만 리턴 되는 URL APP Scheme 설정 예) nicepaysample://ISPCancel
PayMethod	결제수단	Y	10 byte	CARD BANK VBANK CELLPHONE SSG_BANK 중에서 사용
Amt	결제금액	Y	12 byte	1000원 이상 처리가능
GoodsName	상품명	Y	40 byte	상품명
MID	상점아이디	Y	10 byte	상점아이디
TrKey	거래 고유키	Y		
MallIP	가맹점 IP	Y		
GoodsCl	휴대폰 상품 구분	Y	1 byte	1: 실물 0: 컨텐츠

3. 결제 결과 처리 연동

- 결제 결과는 PC용 샘플과 동일하게 처리합니다.
- 첨부된 샘플의 결과 페이지(Result)를 참조하면 됩니다.
- 모바일에서 가맹점의 추가 필드는 **반드시 request로 MallReserved 예비필드로 받으셔야 합니다.** (Charset을 UTF8로 하셨을 경우, UTF-8인코딩된 예비필드는 **MallReservedEncode**로 리턴됩니다.)
- 3에서 설명하였듯이 금액의 경우 **"Amt"** 필드에 리턴됩니다. 주문번호 필드인 **"Moid"**를 기준으로 하여 **원상품가격과 결제결과금액과 비교하여 금액이 동일하지 않다면 결제 금액의 위변조가 의심**됨으로 정상적인 처리가 되지 않도록 처리 부탁드립니다.

3.1. JSP 결제 결과 처리 연동

- 클래스 Import

```
<%@ page import="kr.co.nicevan.nicepay.web.adapter.NicePayHttpServletRequestWrapper"%>
<%@ page import="kr.co.nicevan.nicepay.web.adapter.web.NicePayWEB"%>
<%@ page import="kr.co.nicevan.nicepay.web.adapter.web.dto.WebMessageDTO"%>
```

- 인증결과값 확인

```
if("0000".equals(request.getParameter("AuthResultCode")) {
    // 인증 성공했을 때만 0000으로 리턴됩니다. 해당 내용 꼭 확인하세요.
} else {
    // 인증 실패에 대한 처리를 해주세요.
}
```

- Request Wrapper 등록

```
NicePayHttpServletRequestWrapper httpRequestWrapper =
    new NicePayHttpServletRequestWrapper(request);
```

- 소켓 어댑터 객체 생성

```
NicePayWEB nicePayWEB = new NicePayWEB();
```

- 로그 디렉토리 설정

```
nicePayWEB.setParam("NICEPAY_LOG_HOME","/wwwroot/ipg_adaptor_log/log");
```

이벤트로그, 어플리케이션 로그 파일이 생성되는 위치, 설정한 디렉터리는 **파일 쓰기 권한이 반드시 필요.**

- 로그 디렉토리 설정 주의 사항

설정한 로그 디렉토리 경로는 절대 경로로 설정하고 반드시 외부 웹(Web) 에서 접근이 불가 하도록 접근 권한을 설정 합니다.

- 어플리케이션로그 모드 설정(8. 로그 항목 참고)

```
nicePayWEB.setParam("APP_LOG","1"); // 0: DISABLE, 1: ENABLE
```

- 암호화 플래그 설정(라이브러리<->NICEPAY 구간 암호화 여부)

```
// N: 평문, S:암호화
nicePayWEB.setParam("EncFlag","S");
```

- 서비스코드 설정

```
// 결제 서비스 : PY0 , 취소 서비스 : CL0
nicePayWEB.setParam("SERVICE_MODE", "PY0");
```

- 통화구분 설정

```
nicePayWEB.setParam("Currency", "KRW"); // 현재 원화만 지원
```

- 결제요청

```
WebMessageDTO responseDTO =
    nicePayWEB.doService(httpRequestWrapper,response);
```

- 결제결과출력

```
// 결과코드 (정상 :3001 , 그 외 에러)
String resultCode = responseDTO.getParameter("ResultCode");

// 결과메시지
String resultMsg = responseDTO.getParameter("ResultMsg");

// 승인일시 YYYYMMDDHH24mmss
String authDate = responseDTO.getParameter("AuthDate");

// 승인번호
String authCode = responseDTO.getParameter("AuthCode");

// 결제카드사코드
String cardCode = responseDTO.getParameter("CardCode");

// 결제카드사명
String cardName = responseDTO.getParameter("CardName");

// 가맹점 예비필드
String reserved = request.getParameter("MallReserved");
```

3.2. ASP 결제 결과 처리 연동

- 객체생성

```
Set NICEpay = Server.CreateObject("NICE.NICETX2.1")
```

- 인스턴스 초기화 및 거래유형 설정

```
PInst = NICEpay.Initialize("")
NICEpay.SetActionType CLng(PInst), "SECUREPAY"
```

- 인증결과값 확인

```
If Request.Form("AuthResultCode") = "0000" Then
    '인증결과코드 (정상 :0000 , 그 외 에러)
    '해당 부분에 나이스페이 라이브러리를 호출하여 승인하도록 해주세요.
Else
    '실패에 대한 처리
End If
```

- 지불 정보 설정

NICEpay.SetField CLng(PInst, "tid", Request("TID"))	'거래 아이디'
NICEpay.SetField CLng(PInst, "paymethod", Request("PayMethod"))	'지불수단'
NICEpay.SetField CLng(PInst, "mid", Request("MID"))	'상점 ID'
NICEpay.SetField CLng(PInst, "amt", Request("Amt"))	'결제금액'
NICEpay.SetField CLng(PInst, "moid", Request("Moid"))	'상점 주문번호'
NICEpay.SetField CLng(PInst, "GoodsName", Request("GoodsName"))	'상품명'
NICEpay.SetField CLng(PInst, "currency", "KRW")	'통화구분'
NICEpay.SetField CLng(PInst, "buyername", Request("BuyerName"))	'성명'
NICEpay.SetField CLng(PInst, "malluserid", Request("malluserid"))	'회원사고객ID'
NICEpay.SetField CLng(PInst, "buyertel", Request("BuyerTel"))	'이동전화'
NICEpay.SetField CLng(PInst, "buyeremail", Request("BuyerEmail"))	'이메일'
NICEpay.SetField CLng(PInst, "LicenseKey", '가맹점라이센스 키')	
NICEpay.SetField CLng(PInst, "debug", "true" '로그모드("true"로 설정하면 상세한 로그를 남김)	
NICEpay.SetField CLng(PInst, "CancelPwd", "123456" '망취소시에 필요한 취소 패스워드	
NICEpay.SetField CLng(PInst, "trkey", Request("TrKey"))	
NICEpay.SetField CLng(PInst, "goodscl", Request("GoodsCl"))	'휴대폰컨텐츠 구분'
NICEpay.SetField CLng(PInst, "TransType", Request("TransType"))	'거래형태'

- 승인 요청 처리

NICEpay.StartAction(CLng(PInst))

- 결제 결과값

m_tid = NICEpay.GetResult(CLng(PInst), "tid")	'거래 번호'
m_moid = NICEpay.GetResult(CLng(PInst), "moid")	'상점거래번호'
m_resultCode = NICEpay.GetResult(CLng(PInst), "resultcode")	'결과코드'
m_resultMsg = NICEpay.GetResult(CLng(PInst), "resultmsg")	'결과메시지'
m_authDate = NICEpay.GetResult((PInst), "AuthDate")	'승인일시YYMMDDHH24mmss'
m_authCode = NICEpay.GetResult((PInst), "authcode")	'승인번호(가상 계좌 결제 수단인 경우 승인번호 없음)'
m_amt = NICEpay.GetResult((PInst), "amt")	'승인금액'
m_payMethod = NICEpay.GetResult((PInst), "PayMethod")	'결제수단'
m_payMethod = Request("MallReserved")	'가맹점 추가 필드(예비)'

- 인스턴스 해제

```
NICEpay.Destroy CLng(PInst)
```

- 결제결과출력

```
// 결과코드 (정상 :3001 , 그 외 에러)
String resultCode = responseDTO.getParameter("ResultCode");
// 결과메시지
String resultMsg = responseDTO.getParameter("ResultMsg");
// 승인일시 YYMMDDHH24mmss
String authDate = responseDTO.getParameter("AuthDate");
// 승인번호
String authCode = responseDTO.getParameter("AuthCode");
// 결제카드사코드
String cardCode = responseDTO.getParameter("CardCode");
// 결제카드사명
String cardName = responseDTO.getParameter("CardName");
// 가맹점 예비필드
String reserved = request.getParameter("MallReserved");
```

3.3. PHP 결제 결과 처리 연동

샘플 페이지 **payResult.php**를 참고하여 결제결과 페이지를 구성한다.

- 클래스 Import

```
require_once dirname(__FILE__).'/lib/nicepay/web/NicePayWEB.php';
require_once dirname(__FILE__).'/lib/nicepay/core/Constants.php';
require_once dirname(__FILE__).'/lib/nicepay/web/NicePayHttpServletRequestWrapper.php';
```

- 인증결과값 확인

```
if($_POST["AuthResultCode"] == "0000"){
    // 인증 성공 시에 승인 처리하세요. (정상 :0000 , 그 외 에러)
}else{
    // 인증 실패 처리하세요.
}
```


- Request Wrapper 등록

```
$httpRequestWrapper = new NicePayHttpServletRequestWrapper($_REQUEST);  
$_REQUEST = $httpRequestWrapper->getHttpRequestMap();
```

- 소켓 어댑터 객체 생성

```
$nicePayWEB = new NicePayWEB();
```

- 로그 디렉토리 설정

```
$nicePayWEB->setParam("NICEPAY_LOG_HOME","{로그디렉토리}");
```

이벤트로그, 어플리케이션 로그 파일이 생성되는 위치, 설정한 디렉터리는 **파일 쓰기 권한이 반드시 필요.**

- 로그 디렉토리 설정 주의 사항

설정한 로그 디렉토리 경로는 절대 경로로 설정하고 반드시 외부 웹(Web) 에서 접근이 불가하도록 접근 권한을 설정 합니다.

- 이벤트로그 모드 설정 (8. 로그 항목 참고)

```
$nicePayWEB->setParam("EVENT_LOG","1"); // 0: DISABLE, 1: ENABLE
```

- 어플리케이션로그 모드 설정(8. 로그 항목 참고)

```
$nicePayWEB->setParam("APP_LOG","1"); // 0: DISABLE, 1: ENABLE
```

- 암호화 플래그 설정(라이브러리<->NICEPAY 구간 암호화 여부)

```
// N: 평문, S:암호화  
$nicePayWEB->setParam("EncFlag","S");
```

- 서비스코드 설정

```
// 결제 서비스 : PY0 , 취소 서비스 : CL0  
$nicePayWEB->setParam("SERVICE_MODE", "PY0");
```

- 통화구분 설정

```
$nicepayWEB->setParam("Currency", "KRW");
```

- 결제요청

```
$responseDTO = $nicePayWEB->doService($_REQUEST);
```

- 결제결과출력

```
$resultCode = $responseDTO->getParameter("ResultCode"); // 결과코드
$resultMsg = $responseDTO->getParameter("ResultMsg"); // 결과메시지
$authDate = $responseDTO->getParameter("AuthDate"); // 승인일시 YYYYMMDDHH24mmss
$authCode = $responseDTO->getParameter("AuthCode"); // 승인번호
$cardCode = $responseDTO->getParameter("CardCode"); // 결제카드사코드
$cardName = $responseDTO->getParameter("CardName"); // 결제카드사명

$mallReserved = $_REQUEST['MallReserved']; // 예비필드
```

※ BIZ전문의 응답 항목명으로 결과 데이터 출력 가능.

- Utf-8 사용시

```
$nicepayWEB->setParam("CHARSET", "UTF8"); // utf-8 사용시 옵션 설정
$resultMsg = $responseDTO->getParameterUTF("ResultMsg"); // utf-8 사용시 한글 메시지는
다음과 같이 받으실 수 있습니다.
```

3.4. .NET 결제 결과 처리 연동

- 객체생성

```
protected System.Web.UI.WebControls.Literal PayMethod;
...
```

- 거래유형 설정

```
NiceLite lite = new NiceLite(@"SECUREPAY");
```

- 로그 경로 설정

```
LogPath = @"C:\log"; //로그 디렉토리
```

- 로그 디렉토리 설정 주의 사항

설정한 로그 디렉토리 경로는 절대 경로로 설정하고 반드시 외부 웹(Web) 에서 접근이 불가 하도록 접근 권한을 설정 합니다.

- 인증결과값 확인

```
protected void Page_Load(object sender, EventArgs e){
    authResultCode = Request.Params["AuthResultCode"];           // 인증결과 : 0000(성공)
    authResultMsg = Request.Params["AuthResultMsg"];             // 인증결과 메시지
    if (authResultCode == "0000"){
        if (!Page.IsPostBack){
            resultData();
        }else{
            ResultCode.Text = authResultCode;
            ResultMsg.Text = authResultMsg;
        }
    }
}
```

- 지불 정보 설정

```
lite.SetField(@"LogPath",LogPath);                               //로그폴더 설정
lite.SetField(@"type",@"SECUREPAY");                           //타입설정(고정)
lite.SetField(@"MID",Request.Params["MID"]);                     //상점 아이디
lite.SetField(@"Amt",Request.Params["Amt"]);                     //지불금액
lite.SetField(@"EncodeKey",Request.Params["MerchantKey"]);       //상점 키(상점 아이디별 상이)
lite.SetField(@"CancelPwd",CancelPwd);                           //취소 패스워드(상점 관리자페이지에서 설정)
lite.SetField(@"PayMethod",Request.Params["PayMethod"]);         //지불수단
lite.SetField(@"GoodsName",Request.Params["GoodsName"]);         //상품명
lite.SetField(@"GoodsCnt",Request.Params["GoodsCnt"]);           //상품갯수
lite.SetField(@"BuyerName",Request.Params["BuyerName"]);         //구매자 명
lite.SetField(@"BuyerTel",Request.Params["BuyerTel"]);           //구매자 연락처
lite.SetField(@"BuyerEmail",Request.Params["BuyerEmail"]);       //구매자 이메일
lite.SetField(@"GoodsCl",Request.Params["GoodsCl"]);             //휴대폰 컨텐츠 구분
lite.SetField(@"TrKey",Request.Params["TrKey"]);                 //인증key
lite.SetField(@"TransType",Request.Params["TransType"]);         //요청 타입(일반(0) 에스스로(1))
lite.SetField(@"MallUserID",Request.Params["MallUserID"]);       //회원사 고객ID
lite.SetField(@"debug",@"true");                                //로그모드(실 서비스 "false" 로 설정)
lite.SetField(@"Moid",Request.Params["Moid"]);                   //가맹점 주문번호
```

- 승인 요청 처리

```
lite.DoPay();
```

- 결제 결과값

```

ResultCode.Text    = lite.GetValue("ResultCode");           //결제결과코드
ResultMsg.Text     = lite.GetValue("ResultMsg");           //결제결과메시지
AuthDate.Text      = lite.GetValue("AuthDate");           //결제승인일시
AuthCode.Text      = lite.GetValue("AuthCode");           //결제승인번호
BuyerName.Text     = lite.GetValue("BuyerName");           //구매자명
GoodsName.Text     = lite.GetValue("GoodsName");           //결제상품명
MID.Text           = lite.GetValue("MID");                 //상점ID
TID.Text           = lite.GetValue("TID");                 //거래ID
Moid.Text          = lite.GetValue("Moid");                //주문번호
Amt.Text           = lite.GetValue("Amt");                 //결제금액
PayMethod.Text     = lite.GetValue("PayMethod");           //결제지불수단
.....

```

- 결제결과출력

```

bool paySuccess = false;
if (PayMethod.Text.Equals("CARD"))
    if (ResultCode.Text.Equals("3001")) paySuccess = true; //신용카드(정상 결과코드:3001)
        cardPanel.Visible = true;
    }else if (PayMethod.Text.Equals("BANK")){
        if (ResultCode.Text.Equals("4000")) paySuccess = true; //계좌이체(정상 결과코드:4000)
            bankPanel.Visible = true;
    }else if (PayMethod.Text.Equals("CELLPHONE")){
        if (ResultCode.Text.Equals("A000")) paySuccess = true; //휴대폰(정상 결과코드:A000)
            cellphonePanel.Visible = true;
    }else if (PayMethod.Text.Equals("VBANK")){
        if (ResultCode.Text.Equals("4100")) paySuccess = true; //가상계좌(정상 결과코드:4100)
            vbankPanel.Visible = true;
    }
}

```

3.5. 결제 승인 전문 내역 (공통)

파라미터명	파라미터설명	MaxSize (byte)	내용
ResultCode	결제 결과 코드	4 byte	결제 수단별 결제 성공 코드 값 신용카드 - 3001 계좌이체 - 4000 가상계좌 - 4100 휴대폰 - A000 현금영수증 - 7001
ResultMsg	결제 결과 메시지	100 byte	예) 카드 결제 성공
TID	거래번호	30 byte	거래를 구분하는 transaction ID 예) nicetest00m01011104191651325596
Moid	상점 주문번호	64 byte	상점에서 부여한 거래 주문번호 예) mnoid1234567890
MID	상점 ID	10 byte	상점 ID 예) nicetest00m
PayMethod	결제 수단 코드	10 byte	신용카드 : CARD 계좌이체 : BANK 가상계좌 : VBANK 핸드폰 : CELLPHONE SSG은행계좌 : SSG_BANK
Amt	결제상품금액	12 byte	12 byte, 반드시 숫자로만 입력 예) 1000원인 경우 -> 000000001000
AuthDate	승인 날짜 (공통)	12 byte	표시 타입 : YYMMDDHHMMSS 예)11041916520
AuthCode	승인 코드 (공통)	30byte	승인 번호 (신용카드, 계좌이체, 휴대폰)

3.5.1. 카드 결제 승인 전문

파라미터명	파라미터설명	길이	내용
CardCode	결제카드사코드	3 byte	신용카드사별 코드 (기타참조)
CardName	결제카드사명	20 byte	예) 비씨
CardQuota	할부개월	2 byte	예) 00 (일시불) 03 (3개월)
CardNo	카드 번호	16byte	예) 40051123****4573
CcPartCl	부분취소가능여부	1 byte	1: 가능, 0: 불가능

CouponAmt	쿠폰금액	12 byte	12 byte, 반드시 숫자로만 입력 예) 1000원인 경우 -> 000000001000
CouponMinAmt	쿠폰최소기준금액	12 byte	12 byte, 반드시 숫자로만 입력 예) 5000원인 경우 -> 000000005000

3.5.2. 계좌이체 및 SSG은행계좌 결제 승인 전문

파라미터명	파라미터설명	길이	내용
BankCode	결제은행코드	3 byte	은행별 코드(기타 참조)
BankName	결제은행명	20 byte	예) sc 제일은행
RcptType	현금영수증 타입	1 byte	예) 0:발행안함,1:소득공제,2:지출증빙
RcptAuthCode	현금영수증 승인번호		

3.5.3. 가상계좌 결제 승인 전문

파라미터명	파라미터설명	길이	내용
VbankBankCode	결제은행코드	3 byte	은행별 코드(기타 참조)
VbankBankName	결제은행명	20 byte	예) sc 제일은행
VbankExpDate	가상계좌 입금예정일		
VbankNum	가상계좌번호	20 byte	30011073537

3.5.4. 휴대폰 결과 파라미터

파라미터명	파라미터설명	비고
DstAddr	결제 휴대폰 번호	

4. 안드로이드 앱 연동 가이드

*WebViewActivity.java를 참조하십시오.

4.1. 링크 상수 및 변수 선언

```
final String ISP_LINK = "market://details?id=kvp.jjy.MispAndroid320"; // ISP 설치 링크
final String KFTC_LINK = "market://details?id=com.kftc.bankpay.android"; //금융결제원 설치 링크
final String MERCHANT_URL = "http://web.nicepay.co.kr/smart/mainPay.jsp"; // 가맹점의 결제 요청
// 페이지 URL
private String NICE_BANK_URL = ""; // 계좌이체 인증후 거래 요청 URL
private WebView mWebView; // 웹뷰 인스턴스
// AndroidManifest.xml에 명시된 값과 동일한 값을 설정하십시오.
// 스키마 뒤에 ://를 붙여주십시오.
```

```
private String WAP_URL = "nicepaysample" + "://";
private String BANK_TID = "";
```

4.2. 웹뷰의 속성 설정

onCreate 내 안드로이드 웹뷰를 설정합니다.

```
mWebView = (WebView) findViewById(R.id.webview);
mWebView.setWebChromeClient(new WebChromeClient());
mWebView.setWebViewClient(new WebViewClientClass());
// Webview 설정
mWebView.getSettings().setJavaScriptEnabled(true);
```

4.3. 안드로이드 targetSDK에 따른 쿠키 및 기타 설정 처리

안드로이드 5.0 이상(API 21)으로 targetSDK를 설정하여 빌드한 경우 아래 구문을 추가하여 주십시오.

```
if(android.os.Build.VERSION.SDK_INT >= Build.VERSION_CODES.LOLLIPOP){
    mWebView.getSettings().setMixedContentMode(WebSettings.MIXED_CONTENT_ALWAYS_ALLOW);
    CookieManager cookieManager = CookieManager.getInstance();
    cookieManager.setAcceptCookie(true);
    cookieManager.setAcceptThirdPartyCookies(mWebView, true);
}
```

안드로이드 5.0 이전 버전으로 targetSDK를 설정하여 빌드한 경우 아래 구문을 추가하여 주십시오.

```
CookieManager cookieManager = CookieManager.getInstance();
cookieManager.setAcceptCookie(true);
```

4.4. 웹뷰의 캐시사용 설정 변경 금지에 대한 안내

WebView에서 캐시사용 관련 Default 설정은 WebSettings.LOAD_DEFAULT 입니다.

ex) mWebView.getSettings().setCacheMode(WebSettings.LOAD_DEFAULT);

가급적 캐시 사용 설정을 변경하지 않을 것을 권고 드립니다.

ex) 'WebSettings.LOAD_CACHE_ELSE_NETWORK' 로 변경금지.

4.5. 웹뷰 onCreate url 처리

```
Uri uri = intent.getData();
if (uri != null){
    String url = uri.toString();
    Log.i("NICE", "NicePay onCreate url : " + url);
    if (url.startsWith(WAP_URL)) {
        // 결제결과 url을 설정한다.
        urlString = url;
        urlString = url.substring(WAP_URL.length());
        mWebView.loadUrl(urlString);
    }
} else{
    // 결제 요청할 가맹점의 웹페이지를 호출합니다.
    // 가맹점에 맞는 URL로 변경하십시오. 해당 웹페이지는 NICEPAY_SMART_WEB을 참조합니다.
    mWebView.postUrl(MERCHANT_URL, null);
}
```

4.6. 웹뷰에서의 override 처리

shouldOverrideUrlLoading 를 오버라이딩 하여 처리합니다.

```
@Override
@Override
public boolean shouldOverrideUrlLoading(WebView view, String url) {

    Log.i("NICE", "NicePay OverrideUrlLoading : " + url);
    Intent intent = null;
    // 웹뷰에서 ispmobile 실행한 경우...
    if(url.startsWith("ispmobile")){
        if(Utility.isPackageInstalled(getApplicationContext(), "kvp.jjy.MispAndroid320")){
            intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
            startActivity(intent);
            return true;
        } else{
            installISP();
            return true;
        }
    }
}
```



```

// 웹뷰에서 계좌이체를 실행한 경우...
}else if(url.startsWith("kftc-bankpay")){
    if(Utility.isPackageInstalled(getApplicationContext(),
"com.kftc.bankpay.android")){

        String sub_str_param = "kftc-bankpay://eftpay?";
        String reqParam = url.substring(sub_str_param.length());

        try {

            reqParam = URLDecoder.decode(reqParam,"utf-8");
        } catch (UnsupportedEncodingException e) {
            e.printStackTrace();
        }

        reqParam = makeBankPayData(reqParam);

        intent = new Intent(Intent.ACTION_MAIN);
        intent.setComponent(new
ComponentName("com.kftc.bankpay.android", "com.kftc.bankpay.android.activity.MainActivity"));
        intent.putExtra("requestInfo",reqParam);
        startActivityForResult(intent,1);

        return true;
    }else{

        installKFTC();
        return true;
    }

// 웹뷰에서 안심클릭을 실행한 경우...
}else if (url != null && (url.contains("vguard")
|| url.contains("droidxantivirus")
|| url.contains("lottesmartpay")
|| url.contains("smshinhancardusim://")
|| url.contains("shinhan-sr-ansimclick")
|| url.contains("v3mobile")
|| url.endsWith(".apk")
|| url.contains("smartwall://")
|| url.contains("appfree://")
|| url.contains("market://")

```

```

|| url.contains("ansimclick://")
|| url.contains("ansimclickscard")
|| url.contains("ansim://")
|| url.contains("mpocket")
|| url.contains("mvaccine")
|| url.contains("market.android.com")
|| url.startsWith("intent://")
|| url.contains("samsungpay")
|| url.contains("droidx3web://")
|| url.contains("kakaopay")
|| url.contains("callonlinepay")
|| url.contains("http://m.ahnlab.com/kr/site/download")) {

    try{

        try {

            intent = Intent.parseUri(url, Intent.URI_INTENT_SCHEME);
            Log.i("NICE", "intent getDataString +++==>" + intent.getDataString());

        } catch (URISyntaxException ex) {
            Log.e("Browser", "Bad URI " + url + ":" + ex.getMessage());
            return false;
        }

        if (url.startsWith("intent")) { //chrome 버전 방식

            if

(getPackageManager().resolveActivity(intent,0)==null){

                String packagename=intent.getPackage();

                if (packagename !=null){

                    Uri uri = Uri.parse("market://search?q=pname:"+packagename);
                    intent = new Intent(Intent.ACTION_VIEW,uri);
                    startActivity(intent);
                    return true;
                }
            }

            Uri uri = Uri.parse(intent.getDataString());
            intent = new Intent(Intent.ACTION_VIEW, uri);

```

```

startActivity(intent);

        return true;
    } else { //구 방식
        intent = new Intent(Intent.ACTION_VIEW, Uri.parse(url));
        startActivity(intent);
        //return true;
    }
} catch (Exception e) {
    Log.i("NICE", e.getMessage());
    return false;
}
}

// ispmobile에서 결제 완료후 스마트주문 앱을 호출하여 결제결과를 전달하는 경우
else if (url.startsWith(WAP_URL))
{
    String thisurl = url.substring(WAP_URL.length());
    view.loadUrl(thisurl);
    return true;
} else {
    view.loadUrl(url);
    return false;
}

return true;
}

```

5.5 계좌이체 이벤트 설정

```
/**
 *
 *   계좌이체 결과값을 받아와 오류시 해당 메시지를, 성공시에는 결과 페이지를 호출한다.
 *
 */
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data)
{
    super.onActivityResult(requestCode, resultCode, data);
    String resVal = data.getExtras().getString("bankpay_value");
    String resCode = data.getExtras().getString("bankpay_code");
    Log.i("NICE", "resCode : " + resCode);
    Log.i("NICE", "resVal : " + resVal);

    if("091".equals(resCode)){//계좌이체 결제를 취소한 경우
        Utility.AlertDialog("인증 오류", "계좌이체 결제를 취소하였습니다.",
WebViewActivity.this);
        mWebView.postUrl(MERCHANT_URL, null);
    }else if("060".equals(resCode)){
        Utility.AlertDialog("인증 오류", "타임아웃", WebViewActivity.this);
        mWebView.postUrl(MERCHANT_URL, null);
    }else if("050".equals(resCode)){
        Utility.AlertDialog("인증 오류", "전자서명 실패", WebViewActivity.this);
        mWebView.postUrl(MERCHANT_URL, null);
    }else if("040".equals(resCode)){
        Utility.AlertDialog("인증 오류", "OTP/보안카드 처리 실패", WebViewActivity.this);
        mWebView.postUrl(MERCHANT_URL, null);
    }else if("030".equals(resCode)){
        Utility.AlertDialog("인증 오류", "인증모듈 초기화 오류", WebViewActivity.this);
        mWebView.postUrl(MERCHANT_URL, null);
    }else if("000".equals(resCode)){ // 성공일 경우
        String postData =
"callbackparam2="+BANK_TID+"&bankpay_code="+resCode+"&bankpay_value="+resVal;
        mWebView.postUrl(NICE_BANK_URL,EncodingUtils.getBytes(postData,"euc-kr"));
```

```

    }
}

/**
 *
 * 계좌이체 데이터를 파싱한다.
 *
 * @param str
 * @return
 */
private String makeBankPayData(String str)
{
    String[] arr = str.split("&");
    String[] parse_temp;
    HashMap<String, String> tempMap = new HashMap<String,String> ();

    for(int i=0;i<arr.length;i++){
        try{
            parse_temp = arr[i].split("=");
            tempMap.put(parse_temp[0], parse_temp[1]);
        }catch(Exception e){

        }
    }

    BANK_TID = tempMap.get("user_key");
    NICE_BANK_URL = tempMap.get("callbackparam1");
    return str;
}

```

4.7. 스키마 설정

ISP 결제 호출 후 다시 어플을 호출하기 위하여 결제 앱의 스키마를 설정합니다.

* 결제 요청 페이지(webview에서 결제 호출시) 에서 WapUrl에 스키마 이름을 명시

```
<input type="hidden" name="WapUrl" value="앱스키마이름"/>
```

* 안드로이드 앱의 ApplicationManifest.xml에서 스키마 설정 추가

```
<data android:scheme="앱스키마이름"/>
```

5. iPhone APP 개발 가이드

5.1. 웹뷰의 설정

카드 ISP 결제 와 계좌이체 결제 의 별도의 APP 을 통해 인증 합니다.

(제공 된 샘플 상 PaymentWebViewController.m 페이지를 참고)

APP 을 호출 하기 위해 WebView 의 shouldStartLoadWithRequest 함수를 위임 받습니다.

아래에는 이해를 돕기 위한 일부 코드만 삽입 되어 있으니 Sample 상 전체 소스를 확인 하시기 바랍니다.

5.1.1. Mobile ISP APP 호출하는 경우

```
//isp App을 호출하는 경우
if([URLString hasPrefix:@"ismobile://"]){
//앱이 설치 되어 있는 확인
if([[UIApplication sharedApplication] canOpenURL:request.URL]) { //설치 되어 있을 경우 isp App 호출
    [[UIApplication sharedApplication] openURL:request.URL];
}
else { //설치 되어 있지 않다면 app store 연결
    [self showAlertViewWithMessage:@"모바일 ISP가 설치되어 있지 않아\nApp Store로 이동합니다."
    tagNum:app_link_isp];
    return NO;
}
}
```

5.1.2. BankPay App 을 호출하는 경우

```
if([URLString hasPrefix:@"kftc-bankpay://"]){
//앱이 설치 되어 있는 확인
if([[UIApplication sharedApplication] canOpenURL:request.URL]) {
    NSRange range = [URLString rangeOfString:@"callbackparam1="];
    if(range.location != NSNotFound) {
        int cutIdx = range.location + [@"callbackparam1=" length];
        self.bankPayUrlString = [URLString substringFromIndex:cutIdx];
        NSLog(@"bankPayUrlString: %@",bankPayUrlString);
    }
    [[UIApplication sharedApplication] openURL:request.URL]; //설치 되어 있을 경우 App 호출
}
else {
//설치 되어 있지 않다면 app store 연결
```

```

        [self showAlertViewWithMessage:@"Bank Pay가 설치되어 있지 않아WnApp Store로 이동합니다."
        tagNum:app_link_bank];

        return NO;

    }

}

```

5.1.3. Mobile ISP , BankPay(계좌이체) APP 설치

App 설치되어 있는 애플의 경우 아래의 URL을 [[UIApplication sharedApplication] openURL] 함수를 이용해 APP Store 로 이동 합니다.

* Mobile ISP AppStore URL :

<http://itunes.apple.com/kr/app/id369125087?mt=8>

* BankPay(계좌이체) AppStore URL :

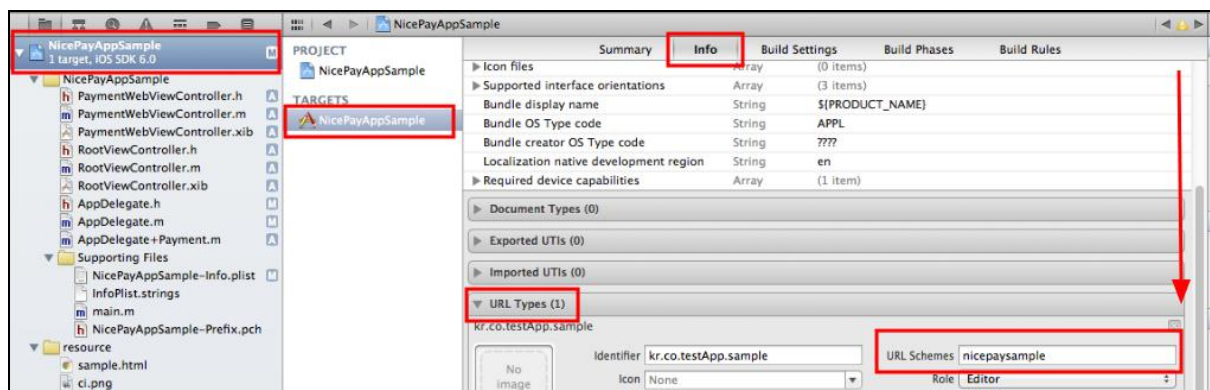
<http://itunes.apple.com/us/app/id398456030?mt=8>

5.2. 스키마 설정

ISP결제, 계좌이체는 APP 이용해 인증 처리가 되기 때문에 가맹점에서 APP 내 WebView 를 이용해 결제 하는 경우 필수로 설정 되어야 합니다.

: 프로젝트 프로퍼티 > TARGETS > Info < URL Types > URL Schemes > 스키마 입력

예) nicepaysample" 로 Schemes 를 입력한 한 경우



5.3. 인증 결과 처리

Mobile ISP 또는 BankPay APP 에서 인증 후 가맹점 APP을 호출 할 때 가맹점에서 처리해야 할 방법을 설명합니다

5.3.1. Mobile ISP APP 에서 호출 되는 경우

WEB 에서는 APP을 호출 할 수 있도록 설정, APP 에서는 ISP 통해 호출 되는 경우를 구현 해야 합니다.

1) 결제 요청 WEB 에서의 설정 방법 (샘플 상 "payRequestForiPhone.asp" 참고)

결제 요청 시 Mobile ISP 에서 가맹점 APP 을 호출 할 수 있도록 결제 요청 페이지에 APP Scheme 명을 설정 합니다.

*ISP 인증이 완료되는 경우 호출 되는 App Scheme

```
<input type="hidden" name="WapUrl" value="nicepaysample://" />
```

*ISP 인증이 취소되는 경우 호출 되는 App Schemes

'하단 ISPCancel 문구는 변경 해도 관계 없으나 가맹점 APP 에서 취소임을 식별 할 수 있어야 함

```
<input type="hidden" name="IspCancelUrl" value="nicepaysample://ISPCancel" />
```

2) APP 의 처리 방법(샘플 상 "AppDelegate+Payment.m" 참고)

* ISP 인증 완료 처리

```
NSRange range = [URLkeyString rangeOfString:@"ispResult.jsp"];
if(range.location != NSNotFound) { //ISP 인증 후 결제 진행
    URLkeyString = [URLkeyString substringFromIndex:[MY_APP_URL_KEY length]+3];
    [self.webViewController requestIspPayResult:URLkeyString];
    return;
}
```

* ISP 취소 처리

```
NSRange range = [URLkeyString rangeOfString:@"ISPCancel"];
UIAlertView* alertView = [[[UIAlertView alloc] initWithTitle:@"알림"
    message:@"결제를 취소하셨습니다."
    delegate:self cancelButtonTitle:@"확인"
    otherButtonTitles:nil]autorelease];

alertView.tag = 900;
[alertView show];

return;
}

- (void)alertView:(UIAlertView *)alertView clickedButtonAtIndex:(NSInteger)buttonIndex
{
    if(alertView.tag == 900){
        [self.webViewController close]; // ISP 결제 취소로 인해 결제 화면을 닫습니다.
    }
}
```

5.3.2. BankPay 에서 호출 되는 경우

계좌이체 인증처리가 완료 또는 취소 되는 경우 호출 되며 아래의 방법에 따라 처리 합니다.

1) 결제 요청 WEB 에서의 설정 방법 (샘플 상 payRequestForiPhone.asp 참고)

결제 요청 시 BankPay 에서 가맹점 APP 을 호출 할 수 있도록 결제 요청 페이지에 APP Schemes 명을

설정 합니다. (Mobile ISP 동일한 필드명 사용)

*계좌이체 인증이 완료 또는 취소하는 경우 호출 되는 App Scheme

```
<input type="hidden" name="WapUrl" value="nicepaysample://" />
```

2) APP 에서의 처리 방법 (샘플 상 "AppDelegate+Payment.m" 참고)

BankPay APP 의 경우 별도로 인증 완료와 취소 부분이 없으므로 가맹점 APP 에서는 WebView 로 인증 결과를 Post 하여 NICE Web 페이지에서 처리 결과를 확인 하도록 합니다.

* ISP 인증 완료 처리

```
NSRange range = [URLkeyString rangeOfString:@"ispResult.jsp"];
if(range.location != NSNotFound) { //ISP 인증 후 결제 진행
    URLkeyString = [URLkeyString substringFromIndex:[MY_APP_URL_KEY length]+3];
    [self.webViewController requestIsPayResult:URLkeyString];
}return;
}
```

5.4. Whitelist 설정

IOS 9버전에서는 권고사항이었던 정책이 IOS 10에서 필수로 바뀜에 따라 APP 내 보안정책에 대한 canOpenUrl 함수 사용 시, info.plist 파일에 LSApplicationQueriesSchemes 배열을 정의하여 호출할 App Scheme List를 등록해 주셔야 합니다.

아래 App스키마는 결제에 필요한 카드사 또는 금결원 App 스키마 명입니다.

info.plist에 아래와 같이 키와 해당 배열을 추가합니다.

```
<key>LSApplicationQueriesSchemes</key>
<array>
    <string>kftc-bankpay</string>
    <string>ispmobile</string>
    <string>shinhan-sr-ansimclick</string>
    <string>smshinhanansimclick</string>
    <string>hdcappcardansimclick</string>
    <string>smhyundaiansimclick</string>
    <string>mpocket.online.ansimclick</string>
    <string>scardcertiapp</string>
    <string>cloudpay</string>
    <string>nhapppcardansimclick</string>
    <string>nonghyupcardansimclick</string>
    <string>kb-acp</string>
    <string>lotteappcard</string>
    <string>lottesmartpay</string>
```

```

<string>citispay</string>
<string>shinsegaeasypayment</string>
<string>tswansimclick</string>
<string>nhallonepayansimclick</string>
<string>citimobileapp</string>
<string>payco</string>
<string>kakaotalk</string>
<string>hanaskcardmobileportal</string>
<string>lpayapp</string>
<string>wooripay</string>
</array>

```

- 카드사별 스키마명

지불수단	APP	SCHEME
신용카드	신한 APP카드	shinhan-sr-ansimclick://
	신한 공인인증 APP (일반결제)	smshinhanansimclick://
	현대 APP카드	hdcardappcardansimclick://
	현대 공인인증 APP (일반결제)	smhyundaiansimclick://
	삼성 APP카드	mpocket.online.ansimclick://
	삼성 공인인증 APP (일반결제)	scardcertiapp://
	하나 APP카드	cloudpay://
	농협 APP카드	nhappcardansimclick://
	농협 공인인증 APP (일반결제)	nonghyupcardansimclick://
	국민 APP카드	kb-acp://
	롯데 APP카드	lotteappcard://
	롯데 스마트페이	lottesmartpay://
	롯데 엘페이	lpayapp://
	씨티카드 APP카드	citispay://
	ISP	ispmobile://
	SSG-PAY	shinsegaeasypayment://
	시럽 APP카드	tswansimclick://
	NH 올원페이	nhallonepayansimclick://
	시티카드 APP카드	citimobileapp://

	PAYCO	payco://
	카카오페이	kakaotalk://
	하나 공인인증 APP (일반결제)	hanaskcardmobileportal://
	우리 앱카드 (우리페이)	wooripay://
계좌이체	금결원	kftc-bankpay://

6. 가맹점 ID, KEY 확인

결제 요청 시 EncodeKey 필드에 사용되는 상점 키 조회 방법을 설명합니다.

1. NICEPAY 가맹점 관리자 페이지 로그인

<https://npg.nicepay.co.kr/>

2. 라이선스 키 등록

가맹점 정보 > KEY 관리 > KEY 복사 (Ctrl + C)

The screenshot shows the NICEPAY merchant management interface. The top navigation bar includes links for Quick, 거래내역조회, 정산내역, 세금계산서조회, 공지사항, 기본정보, 신용카드심사현황, and a user profile section. The main menu on the left lists 가맹점정보, 기본정보, KEY관리 (highlighted with a red box), 등록현황, 사용자관리, and 비밀번호관리. The main content area is titled 'KEY관리' and displays a table with columns for MID (niceday00m), 상 호, and a license key (나이스정보통신(주)). A note below the table states: '* 아래의 Key를 복사하여 결제시 암호화 Key로 적용하시면 됩니다.' The license key is shown as a long alphanumeric string.

7. 가맹점 거래취소 비밀번호 등록

결제 취소 요청 시 CancelPwd 필드에 사용되는 취소 패스워드 등록방법을 설명합니다.

1. NICEPAY 가맹점 관리자 페이지 로그인

<https://npg.nicepay.co.kr/>

2. 거래취소 비밀번호 등록

가맹점 정보 > 거래취소 비밀번호 등록

The screenshot shows the NICEPAY merchant management interface for the '거래취소비밀번호' (Transaction Cancellation Password) registration. The top navigation bar is the same as in the previous screenshot. The main menu on the left lists 가맹점정보, 등록현황, 사용자관리, 비밀번호관리, and 거래취소비밀번호 (highlighted with a red box). The main content area is titled '거래취소비밀번호' and contains a form with the following fields: 아이디 (niceday00m), 비밀번호 (password), and 비밀번호 재입력 (password confirmation). A note above the form states: '1) 거래 취소시, 해당 등록된 비밀번호를 입력해야 합니다. 2) 10자리이하의 숫자로 설정해 주시기 바랍니다.' There is a '저장' (Save) button at the bottom right of the form.

8. 기타

8.1. 결제수단별 결과코드

8.1.1. 신용카드

번호	결과코드	메시지
1	3001	카드 결제 성공
2	3011	카드번호 오류
3	3012	카드가맹점 정보 미확인
4	3013	카드 가맹점 개시 안됨
5	3014	카드가맹점 정보 오류
6	3021	유효기간 오류
7	3022	할부 개월 오류
8	3023	할부 개월 한도 초과
9	3031	무이자할부 카드 아님
10	3032	무이자할부 불가 개월 수
11	3033	무이자할부 가맹점 아님
12	3034	무이자할부 구분 미 설정
13	3041	금액 오류(1000원 미만 신용카드 승인 불가)
14	3051	해외카드 미등록 가맹점
15	3052	통화코드 오류
16	3053	확인 불가 해외카드
17	3054	환율전환오류
18	3055	인증 시 달러승인 불가
19	3056	국내카드 달러승인불가
20	3057	인증 불가카드
21	3061	국민카드 인터넷안전결제 적용 가맹점
22	3062	신용카드 승인번호 오류
23	3071	매입요청 가맹점 아님
24	3072	매입요청 TID 정보 불일치
25	3073	기 매입 거래
26	3081	카드 잔액 값 오류
27	3091	제휴카드 사용불가 가맹점
28	3095	카드사 실패 응답

8.1.2. 계좌이체

번호	결과코드	메시지
1	4000	계좌이체 결제 성공
2	4001	회원 사 서비스 불가 은행
3	4002	출금일자 불일치
4	4003	출금요청금액 불일치
5	4004	거래번호(TID) 불일치
6	4005	회신 정보 불일치
7	4006	계좌이체 승인번호 오류
8	4007	은행 시스템 서비스 중단

8.1.3. 가상계좌

번호	결과코드	메시지
1	4100	가상계좌 발급 성공
2	4110	가상계좌 입금 성공
3	4101	가상계좌 최대거래금액 초과
4	4102	가상계좌 입금예정일 오류
5	4103	가상계좌 입금예정시간 오류
6	4104	가상계좌 정보 오류

8.1.4. 휴대폰결제

번호	결과코드	메시지
1	A000	휴대폰결제 처리 성공
2	A001	휴대폰결제 처리 실패
3	A002	필수입력 값(거래키) 누락
4	A003	필수입력 값(이 통사구분) 누락
5	A004	필수입력 값(SMS승인번호) 누락
6	A005	필수입력 값(업체TID) 누락
7	A006	필수입력 값(휴대폰번호) 누락
8	A041	결제금액 오류
9	A564	휴대폰결제ID설정 오류
10	A565	휴대폰결제ID 미 설정 오류
11	A566	휴대폰결제사 설정 오류
12	A567	상품구분코드 설정 오류
13	A568	서비스구분코드 설정 오류

8.1.5. SSG은행계좌 결제

번호	결과코드	메시지
1	0000	결제성공
2	2000	DB오류
3	2011	CINO미존재
4	2012	주문번호없음
5	2032	가맹점주문번호길이이상
6	2151	거래정지 가맹점
7	2152	미등록가맹점
8	2154	제휴사상태미확인
9	2156	중복등록된거래요청
10	2157	허용되지않는입력방식
11	2158	중복등록된입력방식
12	2159	해당은행장애
13	2201	기승인존재

8.1.6. 결제취소

결과코드	메시지
2001	취소 성공
2211	환불 성공 (2001과 함께 취소 성공 처리 할 것)
2003	취소 실패
2010	취소 요청금액 0원 이하
2011	취소 금액 불일치
2012	취소 해당거래 없음
2014	취소 불가능 거래
2015	기 취소 요청
2016	취소 기한 초과
2017	취소 불가 회원 사
2018	신용카드 매 입후 취소 불가능 가맹점
2019	타 회원 사 거래 취소 불가
2023	취소 한도 초과
2024	취소패스워드 불일치
2025	취소패스워드 미 입력
2026	입금액보다 취소금액이 큼니다.
2027	에스크로 거래는 구매 또는 구매거절 시 취소가능.

2028	부분취소 불가능 가맹점.
2029	부분취소 불가능 결제수단.
2030	해당결제수단 부분취소 불가.
2031	전체금액취소 불가.
2032	취소금액이 취소가능금액보다 큼.
2033	부분취소 불가능금액. 전체취소 이용바람.

8.2. 신용카드 코드

신용카드사	코드
비씨	01
국민	02
외환	03
삼성	04
신한	06
현대	07
롯데	08
한미	09
신세계한미	10
시티	11
농협	12
수협	13
평화	14
우리	15
하나	16
동남(주택)	17
주택	18
조흥(강원)	19
축협(농협)	20
광주	21
전북	22
제주	23
산은	24
해외비자	25
해외마스터	26
해외다이너스	27

해외AMX	28
해외JCB	29
카카오뱅크	37
케이뱅크	38

8.3. 계좌이체 은행 코드

은행(증권사)명	코드
한국은행	001
산업은행	002
기업은행	003
국민은행	004
수협은행	007
수출입은행	008
농협은행	011
지역 농축협	012
우리은행	020
SC제일은행	023
한국씨티은행	027
대구은행	031
부산은행	032
광주은행	034
제주은행	035
전북은행	037
경남은행	039
새마을금고중앙회	045
신협	048
저축은행	050
기타 외국계은행	051
모건스탠리은행	052
HSBC은행	054
도이치은행	055
제이피모간체이스은행	057
미즈호은행	058
엠유에프지은행	059

BOA은행	060
비엔피파리은행	061
중국공상은행	062
중국은행	063
산림조합중앙회	064
대화은행	065
교통은행	066
중국건설은행	067
우체국	071
신용보증기금	076
기술보증기금	077
하나은행	081
신한은행	088
케이뱅크	089
카카오뱅크	090
국세청	091
한국주택금융공사	093
서울보증보험	094
경찰청	095
한국전자금융(주)	096
금융결제원	099
한국신용정보원	101
대신저축은행	102
에스비아이저축은행	103
에이치케이저축은행	104
웰컴저축은행	105
유안타증권 (동양종합금융증권)	209
KB증권	218
골든브릿지투자증권	221
한양증권	222
리딩투자증권	223
BNK투자증권	224
IBK투자증권	225
KTB투자증권	227
미래에셋대우	238
삼성증권	240

한국투자증권	243
NH투자증권	247
교보증권	261
하이투자증권	262
현대차증권	263
키움증권	264
이베스트투자증권	265
SK증권	266
대신증권	267
한화투자증권	269
하나금융투자	270
신한금융투자	278
DB금융투자	279
유진투자증권	280
메리츠종합금융증권	287
NH투자증권	289
부국증권	290
신영증권	291
케이프투자증권	292
한국증권금융	293
펀드온라인코리아	294
우리종합금융	295
아주캐피탈	299