프비티 통합 I/F 연동 가이드 v1.0

MFC

fbiti

목차

문서 이력

일자	버전	이력
2025.03.14	1.0.0	프비티 간편 통합 연동 가이드 - MFC 초안 작성
2025.04.23	1.1.0	거래 처리 요청 응답 전문에 거래처리 결과 추가

1 프비티 간편 통합 연동 개요

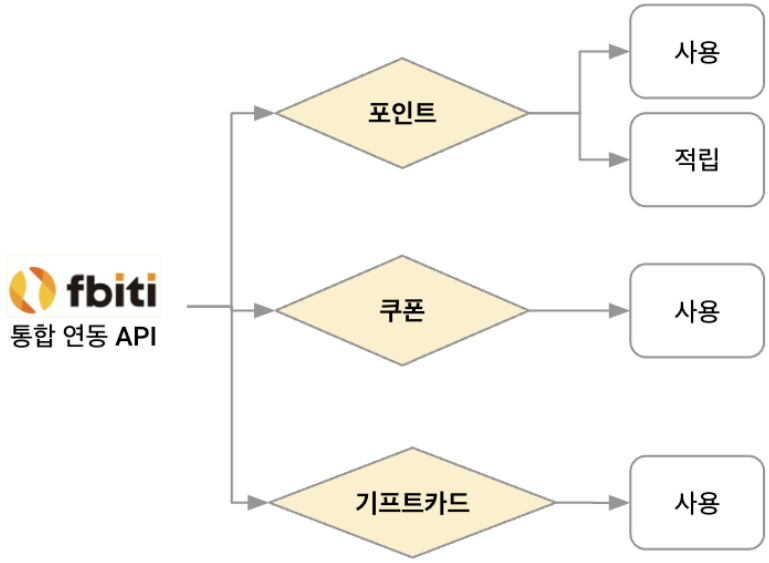
프비티의 간편 통합 연동을 통해 시스템에 쉽게 연동할 수 있으며, 별도의 개발 없이도 프비티의 멤버십 서비스를 바로 사용할 수 있습니다.

프비티 간편 통합 연동 가이드에서는 포인트, 쿠폰, 기프트카드와 관련된 거래 요청을 처리하고, 거래 확정, 거래 처리 요청, 주문 취소 요청 등의 세부적인 처리 방법을 확인할 수 있습니다.

간편 연동은 각 거래 요청에 맞는 통합 화면을 불러와, 고객이 포인트 적립이나 쿠폰 사용 등 멤버십 서비스를 팝업 형식의 화면에서 처리할 수 있게 합니다.

이를 통해 자사 프로그램에 추가 개발 없이 웹 기반 팝업 화면을 통해 멤버십 관련 기능을 손쉽게 제공할 수 있습니다.

사용 중 문의사항은 이메일로 보내주시면 신속히 답변드리겠습니다.



2 프비티 간편 통합 연동 공통 가이드

2.1 연동 인증 정보

프비티 간편 통합 연동 인증 정보는 다음과 같습니다.

- 개발 서버 : 개발사별 별도 제공
- 운영 서버 : 테넌트(브랜드)별 별도 제공

2.2 연동 결과 기본 응답 형식

프비티 간편 통합 연동 기본 응답 형식은 다음과 같습니다.

```
{
        "status": "200",
        "result": "OK",
        "timestamp": 1730962764,
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK"
```

```
        }
}
```

기본 응답 필드

필드명	타입	설명
status	String	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출
result	String	오류메시지 텍스트
timestamp	String	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	Object	
errorCode	Int	오류코드 (2.3 오류 코드 참고)
errorMsg	String	오류메시지 (2.3 오류 코드 참고)
errorMsgUser	String	사용자 오류메시지 (2.3 오류 코드 참고)

2.3 오류 코드

오류 구분	오류 코드	오류 메시지	가맹점 오류 메시지

ConnectionError	100010001	올바르지 않은 거래 요청 타입입니다	요청 정보가 올바르지 않습니다.
	100010002	올바르지 않은 application key입니다	요청 정보가 올바르지 않습니다.
	100010003	올바르지 않은 transactionID입니다	요청 정보가 올바르지 않습니다.
	100010004	올바르지 않은 포인트 사용 금액입니다	요청 정보가 올바르지 않습니다.
	100010005	올바르지 않은 포인트 적립 금액입니다	요청 정보가 올바르지 않습니다.
	100010006	올바르지 않은 기프트카드 결제 금액입니다	요청 정보가 올바르지 않습니다.
	100010007	요청 메세지의 형식이 올바르지 않습니다	요청 정보가 올바르지 않습니다.
	100010008	바코드 메세지의 형식이 올바르지 않습니다	요청 정보가 올바르지 않습니다.
	100010009	올바르지 않은 itemID입니다	상품 정보가 올바르지 않습니다.
	100010010	올바르지 않은 itemName입니다	상품 정보가 올바르지 않습니다.
	100010011	올바르지 않은 itemUnitPrice입니다	상품 정보가 올바르지 않습니다.
	100010012	올바르지 않은 itemCount입니다	상품 정보가 올바르지 않습니다.
	100010013	올바르지 않은 itemPrice입니다	상품 정보가 올바르지 않습니다.
	100010014	올바르지 않은 itemOptionId입니다	상품 정보가 올바르지 않습니다.
	100010015	올바르지 않은 itemOptionName입니다	상품 정보가 올바르지 않습니다.
	100010016	올바르지 않은 itemOptionUnitPrice입니다	상품 정보가 올바르지 않습니다.
	100010017	올바르지 않은 itemOptionCount입니다	상품 정보가 올바르지 않습니다.
	100010018	올바르지 않은 itemOptionPrice입니다	상품 정보가 올바르지 않습니다.

	100010019	올바르지 않은 요청 메세지 유형입니다	요청 정보가 올바르지 않습니다.
	100010020	올바르지 않은 포인트 사용 확정 여부입니다	요청 정보가 올바르지 않습니다.
	100010021	올바르지 않은 포인트 적립 확정 여부입니다	요청 정보가 올바르지 않습니다.
	100010022	올바르지 않은 쿠폰 확정 여부입니다	요청 정보가 올바르지 않습니다.
	100010023	올바르지 않은 기프트카드 확정 여부입니다	요청 정보가 올바르지 않습니다.
	100010024	올바르지 않은 전자영수증 발행 여부입니다	요청 정보가 올바르지 않습니다.
	100010025	올바르지 않은 결제수단입니다	요청 정보가 올바르지 않습니다.
	100010026	올바르지 않은 결제금액(적용금액)입니다	요청 정보가 올바르지 않습니다.
	100020001	passwd가 존재하지 않습니다	필수 정보가 누락되었습니다.
	100020002	accessType이 존재하지 않습니다	필수 정보가 누락되었습니다.
	100020003	인증 요청 정보가 존재하지 않습니다	필수 정보가 누락되었습니다.
	100020004	itemList가 존재하지 않습니다	필수 정보가 누락되었습니다.
	100020005	paymentList가 존재하지 않습니다	필수 정보가 누락되었습니다.
	100030001	제한 시간 내에 메세지 수신을 하지 못했습니다	요청 시간이 초과되었습니다. 다시 시도해 주세요.
	100040001	포인트 시스템을 지원하지 않는 단말기입니다	멤버십(포인트) 서비스를 사용할 수 없습니다. 본사 관리자에게 설정 지원을 요청하세요.
	100040002	쿠폰 시스템을 지원하지 않는 단말기입니다	멤버십(쿠폰) 서비스를 사용할 수 없습니다. 본사 관리자에게 설정 지원을 요청하세요.
	100040003	기프트카드 시스템을 지원하지 않는 단말기입니다	멤버십(기프트카드) 서비스를 사용할 수 없습니다. 본사 관리자에게 설정 지원을 요청하세요.
	100050001	사용자 취소입니다	요청이 사용자에 의해 취소되었습니다.

	100050002	사용자 입력 시간을 초과했습니다	사용자 입력 시간을 초과했습니다.
BusinessLogicError	200010001	알 수 없는 에러가 발생했습니다	알 수 없는 오류가 발생했습니다. 관리자에게 문의해 주세요.
	200010002	쿠폰 할인 적용 금액 계산 중 에러가 발생했습니다	요청 처리 중 오류가 발생했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	200010003	post message 처리 중 에러가 발생했습니다	요청 처리 중 오류가 발생했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	200010004	페이지 라우팅 에러가 발생했습니다	요청 처리 중 오류가 발생했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	200020001	message listner가 등록되지 않았습니다	요청 처리 중 오류가 발생했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
ApiError	300010001	잘못된 Tenant API URL/KEY 입니다	잘못된 가맹점 인증 정보입니다.
	300010002	잘못된 application key입니다	잘못된 가맹점 인증 정보입니다.
	300010003	단말기 ID 혹은 비밀번호가 올바르지 않습니다	아이디 혹은 비밀번호가 틀렸습니다. 다시 확인하시고 시도해 주세요.
	300010004	잘못된 access Token입니다	인증 정보가 잘못되었습니다. 다시 로그인해 주세요.
	300020001	토큰을 발급받는데 실패했습니다	인증 과정에서 문제가 발생했습니다. 관리자에게 문의해 주세요.
	300020002	포인트 환경 설정 조회에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020003	고객 정보 조회에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020004	포인트 사용 대기 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020005	포인트 적립 대기 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020006	포인트 정보 조회 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020007	기프트카드 조회에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020008	기프트카드 사용 대기 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.

	300020009	쿠폰 조회에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020010	쿠폰 사용 대기 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020011	주문 취소 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300020012	거래 처리 요청에 실패했습니다	요청 처리에 실패했습니다. 오류가 반복될 경우 관리자에게 문의해 주세요.
	300030001	통신 에러가 발생했습니다	통신 오류가 발생했습니다. 잠시 후 다시 시도해주세요.
	300030002	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030003	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030004	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030005	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030006	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030007	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030008	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030009	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030010	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.
	300030011	에러가 발생했습니다	시스템 오류가 발생했습니다. 관리자에게 문의해 주세요.

2.4 연동 요청

프비티 간편 통합 연동은 DLL을 통해 연동할 수 있습니다. MFC연동 예시는 다음과 같습니다.

```
#include "DllFbitiMembership.h"

// 포인트 적립 연동 예시
void CClientFbitiMembershipView::OnBnClickedPointEarn()
{
   // 부모 윈도우 핸들 가져오기
   HWND hWnd = this->GetSafeHwnd();

   // "3 프비티 간편 통합 연동 규격" 을 참고하여 TransactionInfo 생성
   STransactionInfo transationInfo;
   transationInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
   transationInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
   transationInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
   transationInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
   transationInfo.transactionID = "1234567890123456";
   transationInfo.payAmount = 1000;

   // SizeInfo 생성 (가로:세로 비율은 5:7로 하는 것을 권장)
   SFbitiMembershipSizeInfo sizeInfo;
   sizeInfo.dlgX = 0;
   sizeInfo.dlgY = 0;
   sizeInfo.dlgWidth = 400;
   sizeInfo.dlgHeight = 600;

   // DLL 동적 로드
   HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
   if (hDLL == NULL) {
       AfxMessageBox(L"Failed to load DLL");
```

```
    return;
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
    (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
if (DestroyFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 생성
DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
if (pFbitiMembership) {
    // PointEarn 메서드 호출
    this->m_FbitiWnd = pFbitiMembership->PointEarn(hWnd, transationInfo, sizeInfo);

    // 객체 삭제
    DestroyFbitiMembershipInstance(pFbitiMembership);
```

```
    }

    // DLL 해제
    FreeLibrary(hDLL);
}
```

2.5 연동 결과 수신

프비티 간편 통합 연동 결과 수신은 메세지 리스너를 통해 수신할 수 있습니다. MFC 연동 예시는 다음과 같습니다.

```
#include "FbitiMembershipMessages.h"
#include "CFbitiMembershipMessageSender.h"

BEGIN_MESSAGE_MAP(CClientFbitiMembershipView, CFormView)
    ON_MESSAGE(WM_USER_FBITI_MEMBERSHIP_MESSAGE_POINT_EARN, &CClientFbitiMembershipView::OnFbitiMembershipPointEarnMessage)
END_MESSAGE_MAP()

LRESULT CClientFbitiMembershipView::OnFbitiMembershipPointEarnMessage(WPARAM wParam, LPARAM lParam)
{
    CFbitiMembershipMessageSender::PointEarnMessage* pMessage =
reinterpret_cast<CFbitiMembershipMessageSender::PointEarnMessage*>(lParam);

    // 메시지 데이터를 사용하여 필요한 처리 수행
    CString info;
    info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\n적립된 포인트: %d"),
        CString(pMessage->status.c_str()),
        CString(pMessage->result.c_str()),
```

```
        CString(pMessage->data.errorCode.c_str()),
        pMessage->data.pointEarn
    );
    AfxMessageBox(info);

    if (this->m_FbitiWnd != nullptr)
    {
        // 프비티 멤버십 윈도우 닫기
        CWnd* fbitiWnd = CWnd::FromHandle(this->m_FbitiWnd);
        fbitiWnd->SendMessage(WM_CLOSE, 0, 0);
    }

    return 0;
}
```

2.6 바코드 스캐너 연동

바코드 스캐너는 USB 방식과 시리얼 통신 방식을 지원합니다. 다음과 같은 구조체를 전달하여 바코드 스캐너를 연동할 수 있습니다.

요청 값

```
#include "DllFbitiMembership.h"

// SFbitiMembershipBarcodeScanInfo 생성
SFbitiMembershipBarcodeScanInfo barcodeScanInfo;
barcodeScanInfo.mode = FBITI_MEMBERSHIP_SCANMODE_SERIAL;
barcodeScanInfo.comPort = "COM5";
```

SFbitiMembershipBarcodeScanInfo							
Name			type	length	Option	Value	설명
mode			int		필수	**FBITI_MEMBERSHIP_SCANMODE_KEYBOARD** : 키보드 **FBITI_MEMBERSHIP_SCANMODE_SERIAL** : 시리얼	바코드 스캔 모드
comPort			int		필수	예시 : COM5	COM 포트

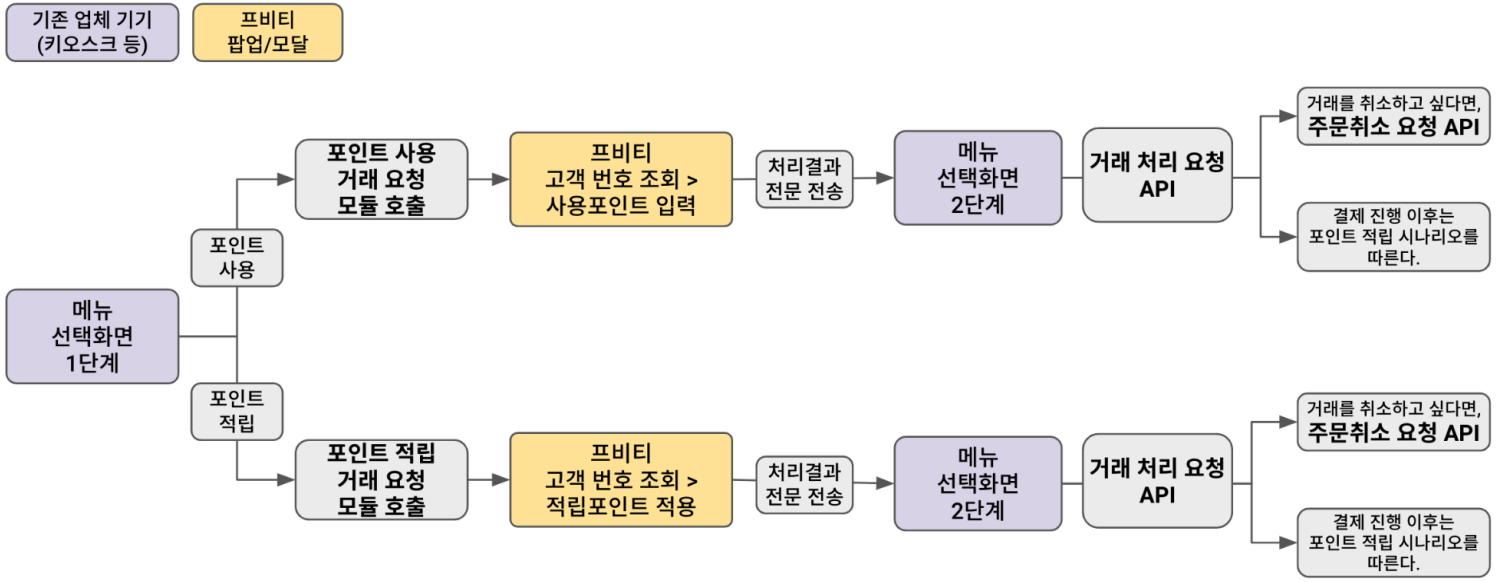
3 프비티 간편 통합 연동 규격 (거래 요청 - 거래 처리 - 주문 취소 순)

3.1 포인트 사용/적립 (포인트 거래 요청)

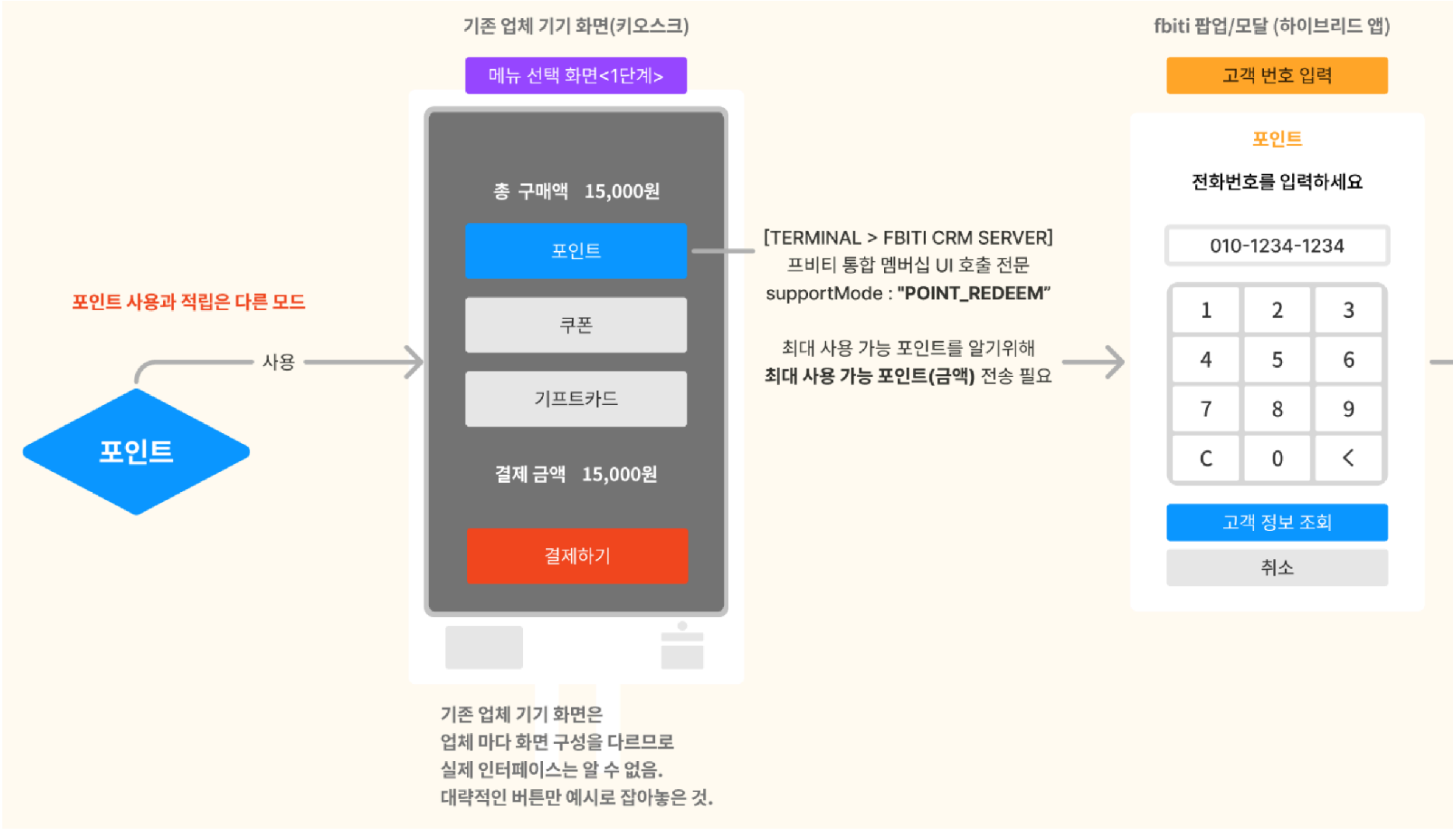
3.1.1 포인트 사용/적립 프로세스

프비티의 포인트는 가맹점 간에 자유롭게 사용할 수 있으며, 적립된 포인트는 다른 가맹점에서도 사용이 가능합니다. 포인트를 사용할 때는 자동으로 정산 처리가 이루어집니다.

포인트는 적립과 사용으로 구분되며, 고객의 전화번호로 조회한 후 적립하거나 사용할 포인트를 입력하면 됩니다. 포인트 적립 또는 사용을 요청하면, 연동된 모듈 화면이 나타나고 해당 화면에서 처리된 결과에 따라 포인트 적립이나 사용이 확정됩니다.



3.1.2 포인트 사용 거래 요청



PointRedeem - 포인트 사용

절차	설명	주요 송신 항목	주요 수신 항목
포인트 사용 요청	포인트 사용 API를 통해 요청 주시면 통합I/F 을 통해 처리한 처리결과 전문을 전송해드립니다.	transactionID : 거래 고유 Transaction ID payAmount : 포인트 사용 : 최대 사용 가능 금액	transactionID : 거래 고유 Transaction ID pointRedeem : 사용된 포인트

```
HWND PointRedeem(HWND hParentWnd, STransactionInfo transactionInfo, SFbitiMembershipSizeInfo sizeInfo);
```

Parameters

- HWND : 부모 윈도우 핸들

STransactionInfo							
Name			**type**	**length**	**Option**	**Value**	**설명**
auth	{						
	tenantID		String		필수		테넌트 ID
	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY
	passwd		String		필수		비밀번호

	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
payAmount			Int		필수		포인트 사용 : 최대 사용 가능 금액
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

SFbitiMembershipSizeInfo							
Name			**type**	**length**	**Option**	**Value**	**설명**
dlgX			int		필수		X 위치
dlgY			int		필수		Y 위치
dlgWidth			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	너비
dlgHeight			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	높이

Return

- HWND : 다이얼로그 윈도우 핸들

응답 메세지

CFbitiMembershipMessageSender::PointRedeemMessage							
Name			type	length		Value	설명
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출
result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						
	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	supportMode		String				거래 요청 타입
	transactionID		String				거래 고유 Transaction ID

	pointRedeem		Int				사용된 포인트
	}						

예시코드

- 고객이 결제해야 하는 금액이 7000원인 경우 payAmount 의 값으로 7000 을 보내면 화면이 나타납니다.

```
#include "DllFbitiMembership.h"

void CClientFbitiMembershipView::OnBnClickedPointRedeem()
{
   // 부모 윈도우 핸들 가져오기
   HWND hWnd = this->GetSafeHwnd();

   // TransactionInfo 생성
   STransactionInfo transactionInfo;
   transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
   transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
   transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
   transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
   transactionInfo.transactionID = "1234567890123456";
   transactionInfo.payAmount = 7000;

   // SizeInfo 생성
   SFbitiMembershipSizeInfo sizeInfo;
   sizeInfo.dlgX = 0;
   sizeInfo.dlgY = 0;
   sizeInfo.dlgWidth = 400;
```

```
sizeInfo.dlgHeight = 600;

// DLL 동적 로드
HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
if (hDLL == NULL) {
    AfxMessageBox(L"Failed to load DLL");
    return;
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
    (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
if (DestroyFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 생성
DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
```

```
    if (pFbitiMembership) {
        // PointRedeem 메서드 호출
        this->m_FbitiWnd = pFbitiMembership->PointRedeem(hWnd, transactionInfo, sizeInfo);

        // 객체 삭제
        DestroyFbitiMembershipInstance(pFbitiMembership);
    }

    // DLL 해제
    FreeLibrary(hDLL);
}
```

메세지 수신 리스너 설정 예시 코드

```
#include "FbitiMembershipMessages.h"
#include "CFbitiMembershipMessageSender.h"


BEGIN_MESSAGE_MAP(CClientFbitiMembershipView, CFormView)
    ON_MESSAGE(WM_USER_FBITI_MEMBERSHIP_MESSAGE_POINT_REDEEM, &CClientFbitiMembershipView::OnFbitiMembershipPointRedeemMessage)
END_MESSAGE_MAP()


LRESULT CClientFbitiMembershipView::OnFbitiMembershipPointRedeemMessage(WPARAM wParam, LPARAM lParam)
{
    CFbitiMembershipMessageSender::PointRedeemMessage* pMessage =
reinterpret_cast<CFbitiMembershipMessageSender::PointRedeemMessage*>(lParam);

    // 메시지 데이터를 사용하여 필요한 처리 수행
    CString info;
```

```
    info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\n사용된 포인트: %d"),
        CString(pMessage->status.c_str()),
        CString(pMessage->result.c_str()),
        CString(pMessage->data.errorCode.c_str()),
        pMessage->data.pointRedeem
    );
    AfxMessageBox(info);

    if (this->m_FbitiWnd != nullptr)
    {
        // 자식 윈도우 닫기
        CWnd* fbitiWnd = CWnd::FromHandle(this->m_FbitiWnd);
        fbitiWnd->SendMessage(WM_CLOSE, 0, 0);
    }

    return 0;
}
```

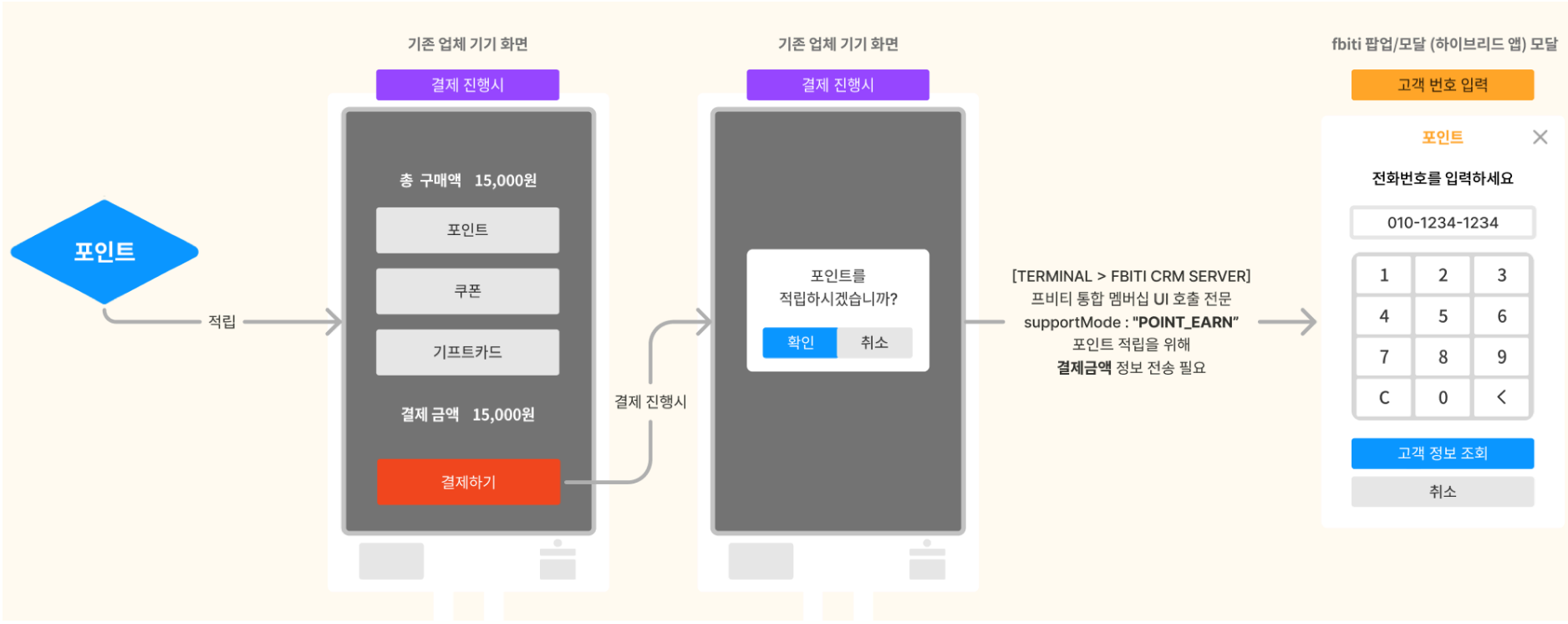
응답 값

- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.
- 고객이 3500 포인트를 사용한 경우 응답 값으로 pointRedeem 의 값이 3500 이다.

```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
```

```
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK",
                "supportMode": "POINT_REDEEM",
                "transactionID": "1c27375780c811ef9e03",
                "pointRedeem": 3500
        }
}
```

3.1.3 포인트 적립 거래 요청



PointEarn - 포인트 적립

절차	설명	주요 송신 항목	주요 수신 항목
포인트 적립 요청	거래 요청 포인트 적립 API를 통해 요청 주시면 통합I/F 을 통해 처리한 결과 전문을 전송해드립니다.	transactionID : 거래 고유 Transaction ID payAmount: 포인트 적립 : 결제 금액	transactionID : 거래 고유 Transaction ID pointEarn : 적립된 포인트

```
HWND PointRedeem(HWND hParentWnd, STransactionInfo transactionInfo, SFbitiMembershipSizeInfo sizeInfo);
```

Parameters

- HWND : 부모 윈도우 핸들

STransactionInfo							
Name			**type**	**length**	**Option**	**Value**	**설명**
auth	{						
	tenantID		String		필수		테넌트 ID
	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY

	passwd		String		필수		비밀번호
	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
payAmount			Int		필수		포인트 사용 : 최대 사용 가능 금액
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

SFbitiMembershipSizeInfo

Name			type	length	Option	Value	설명
dlgX			int		필수		X 위치
dlgY			int		필수		Y 위치
dlgWidth			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	너비
dlgHeight			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	높이

Return

- HWND : 다이얼로그 윈도우 핸들

응답 메세지

CFbitiMembershipMessageSender::PointEarnMessage							
Name			type	length		Value	설명
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출
result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						
	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	supportMode		String				거래 요청 타입
	transactionID		String				거래 고유 Transaction ID
	pointEarn		Int				적립된 포인트

	}						

예시코드

- 고객이 결제해야 하는 금액이 5000원인 경우 payAmount 의 값으로 5000 을 보낸다.

```
#include "DllFbitiMembership.h"

void CClientFbitiMembershipView::OnBnClickedPointEarn()
{
   // 부모 윈도우 핸들 가져오기
   HWND hWnd = this->GetSafeHwnd();

   // TransactionInfo 생성
   STransactionInfo transactionInfo;
   transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
   transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
   transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
   transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
   transactionInfo.transactionID = "1234567890123456";
   transactionInfo.payAmount = 5000;

   // SizeInfo 생성
   SFbitiMembershipSizeInfo sizeInfo;
   sizeInfo.dlgX = 0;
   sizeInfo.dlgY = 0;
   sizeInfo.dlgWidth = 400;
   sizeInfo.dlgHeight = 600;
```

```
// DLL 동적 로드
HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
if (hDLL == NULL) {
    AfxMessageBox(L"Failed to load DLL");
    return;
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
    (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
if (DestroyFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 생성
DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
if (pFbitiMembership) {
    // PointEarn 메서드 호출
```

```
        this->m_FbitiWnd = pFbitiMembership->PointEarn(hWnd, transactionInfo, sizeInfo);

        // 객체 삭제
        DestroyFbitiMembershipInstance(pFbitiMembership);
    }

    // DLL 해제
    FreeLibrary(hDLL);
}
```

메세지 수신 리스너 설정 예시 코드

```
#include "FbitiMembershipMessages.h"
#include "CFbitiMembershipMessageSender.h"

BEGIN_MESSAGE_MAP(CClientFbitiMembershipView, CFormView)
    ON_MESSAGE(WM_USER_FBITI_MEMBERSHIP_MESSAGE_POINT_EARN, &CClientFbitiMembershipView::OnFbitiMembershipPointEarnMessage)
END_MESSAGE_MAP()

LRESULT CClientFbitiMembershipView::OnFbitiMembershipPointEarnMessage(WPARAM wParam, LPARAM lParam)
{
    CFbitiMembershipMessageSender::PointEarnMessage* pMessage =
reinterpret_cast<CFbitiMembershipMessageSender::PointEarnMessage*>(lParam);

    // 메시지 데이터를 사용하여 필요한 처리 수행
    CString info;
    info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\n적립된 포인트: %d"),
        CString(pMessage->status.c_str()),
        CString(pMessage->result.c_str()),
        CString(pMessage->data.errorCode.c_str()),
```

```
        pMessage->data.pointEarn
    );
    AfxMessageBox(info);

    if (this->m_FbitiWnd != nullptr)
    {
        // 자식 윈도우 닫기
        CWnd* fbitiWnd = CWnd::FromHandle(this->m_FbitiWnd);
        fbitiWnd->SendMessage(WM_CLOSE, 0, 0);
    }

    return 0;
}
```

응답 값

- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.
- 5%의 적립률이 적용되어 고객이 250 포인트를 적립하였을 때 응답 값으로 pointEarn 의 값이 250 이다.


```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
```

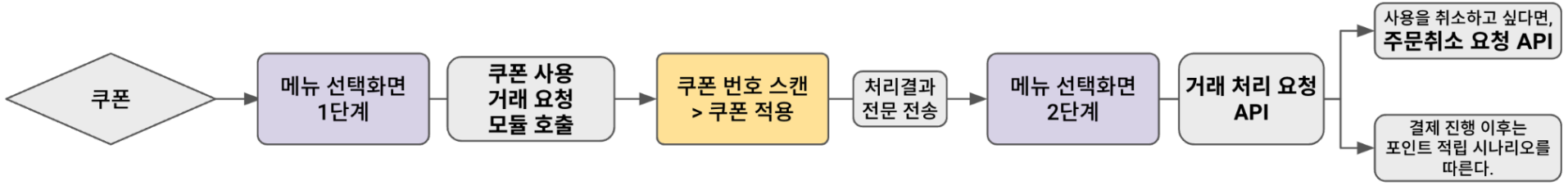
```
                "errorMsgUser": "OK",
                "supportMode": "POINT_EARN",
                "transactionID": "1c27375780c811ef9e03",
                "pointEarn": 250
        }
}
```

3.2 쿠폰 (쿠폰 거래 요청)

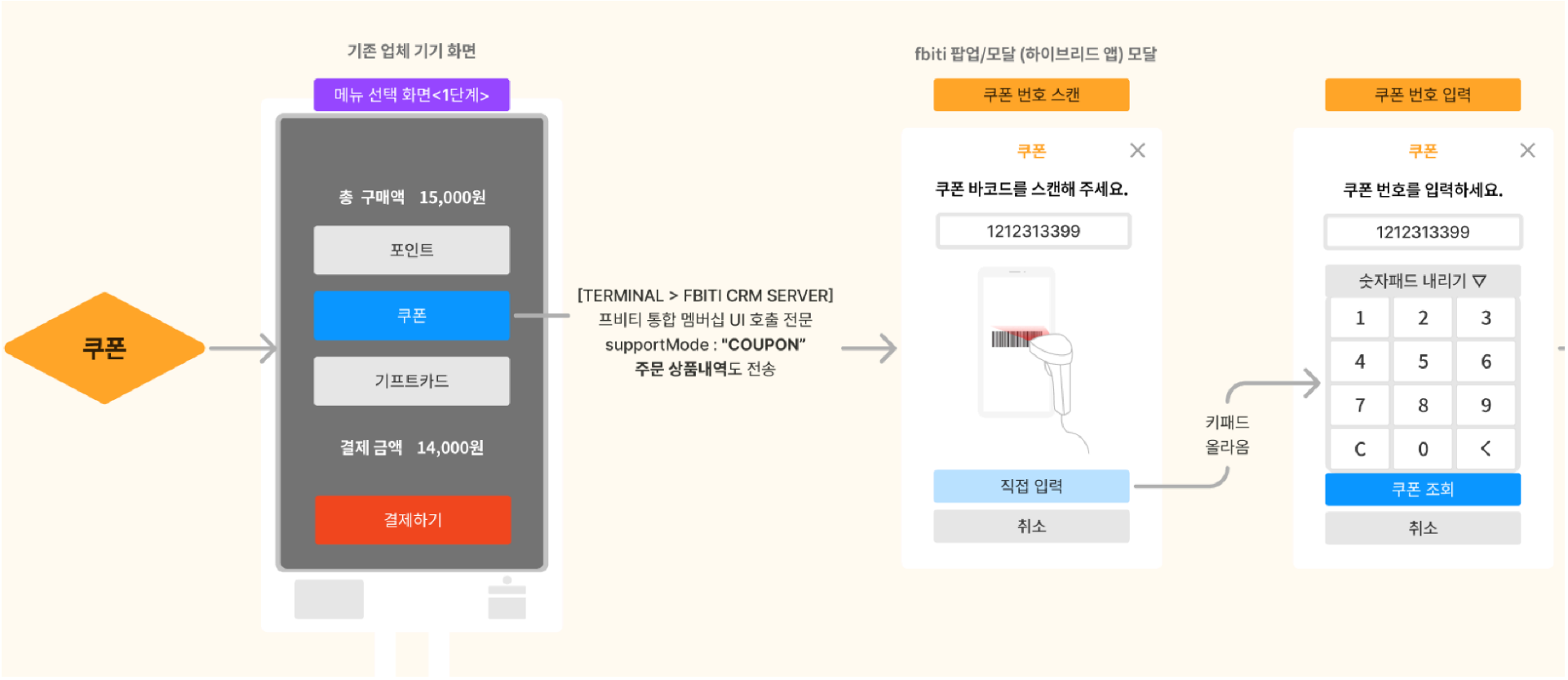
3.2.1 쿠폰 프로세스

쿠폰은 고객의 쿠폰 번호 또는 소유자 없이 생성된 쿠폰의 번호나 바코드를 스캔하여 조회할 수 있습니다. 바코드 스캔이 불가능할 경우, 16자리 쿠폰 번호를 직접 입력하여 조회할 수 있습니다.

쿠폰 사용을 요청하면 프비티 간편 연동 모듈이 호출됩니다. 모듈에서 고객이 사용할 쿠폰 번호를 스캔하거나 입력하고, 사용 처리를 진행하면 처리 결과를 전송해 드립니다. 이후 결제가 완료되면 쿠폰 사용이 확정됩니다.



3.2.2 쿠폰 사용 거래 요청



CouponUse - 쿠폰 사용

절차	설명	주요 송신 항목	주요 수신 항목
쿠폰 사용 요청	쿠폰 사용 API를 통해 요청 주시면 통합I/F 을 통해 처리한 처리결과 전문을 전송해드립니다.	transactionID : 거래 고유 Transaction ID itemList : 상품정보	transactionID : 거래 고유 Transaction ID

```
HWND CouponUse(HWND hParentWnd, STransactionInfo transactionInfo, SFbitiMembershipBarcodeScanInfo barcodeScanInfo, SFbitiMembershipSizeInfo sizeInfo);
```

Parameters

- HWND : 부모 윈도우 핸들

STransactionCouponInfo							
Name			type	length	Option	Value	설명
auth	{						
	tenantID		String		필수		테넌트 ID

	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY
	passwd		String		필수		비밀번호
	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
itemList	[{						
	itemID		String		필수	KD101111	상품ID
	itemName		String		필수	아메리카노	상품명
	itemUnitPrice		Int		필수	3000	상품 단가
	itemCount		Int		필수	3	상품 수량
	itemPrice		Int		필수	9000	상품 금액
	itemOptionList	[{			선택		
		itemOptionID	String			KD101111-01	상품옵션ID
		itemOptionName	String			샷추가	상품옵션명
		itemOptionUnitPrice	Int			500	상품옵션 단가

		itemOptionCount	Int			2	상품옵션 수량
		itemOptionPrice	Int			1000	상품옵션 금액
		}]					
	}]						
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

SFbitiMembershipBarcodeScanInfo

Name			type	length	Option	Value	설명
mode			int		필수	FBITI_MEMBERSHIP_SCANMODE_KEYBOARD : 키보드 FBITI_MEMBERSHIP_SCANMODE_SERIAL : 시리얼	바코드 스캔 모드
comPort			int		필수	예시 : COM5	COM 포트

SFbitiMembershipSizeInfo

Name			type	length	Option	Value	설명
dlgX			int		필수		X 위치
dlgY			int		필수		Y 위치
dlgWidth			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	너비
dlgHeight			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	높이

Return

- HWND : 다이얼로그 윈도우 핸들

응답 메세지

CFbitiMembershipMessageSender::CouponUseMessage							
Name			type	length		Value	설명
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출

result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						
	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	supportMode		String				거래 요청 타입
	transactionID		String				거래 고유 Transaction ID
	couponDiscountPrice		Int				쿠폰 적용 혜택 금액
	couponItemCodes		String[]				쿠폰 교환 가능 상품코드 리스트
	}						

예시코드

- 쿠폰 적용을 위해 고객이 주문한 상품 정보를 itemList 로 보낸다.
- 선택된 상품 옵션이 없는 경우 itemOptionList 는 보내지 않는다.

```
#include "DllFbitiMembership.h"
```

```
void CClientFbitiMembershipView::OnBnClickedCouponUse()
{
    // 부모 윈도우 핸들 가져오기
    HWND hWnd = this->GetSafeHwnd();

    // TransactionInfo 생성
    STransactionCouponInfo transactionInfo;
    transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
    transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
    transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
    transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
    transactionInfo.transactionID = "1234567890123456";

    // 첫 번째 상품 추가
    SItemInfo item1;
    item1.itemID = _T("KD101111");
    item1.itemName = _T("아메리카노");
    item1.itemUnitPrice = 3000;
    item1.itemCount = 3;
    item1.itemPrice = 9000;

    // 첫 번째 상품의 옵션 추가
    SItemOptionInfo option1;
    option1.itemOptionID = _T("KD101111-01");
    option1.itemOptionName = _T("샷추가");
    option1.itemOptionUnitPrice = 500;
    option1.itemOptionCount = 2;
    option1.itemOptionPrice = 1000;
```

```
// 옵션을 첫 번째 상품에 추가
item1.itemOptionList.push_back(option1);

// 첫 번째 상품을 transaction의 itemList에 추가
transactionInfo.itemList.push_back(item1);

// 두 번째 상품 추가
SItemInfo item2;
item2.itemID = _T("KD101112");
item2.itemName = _T("카페라떼");
item2.itemUnitPrice = 4000;
item2.itemCount = 2;
item2.itemPrice = 8000;

// 두 번째 상품의 옵션 추가
SItemOptionInfo option2;
option2.itemOptionID = _T("KD101112-01");
option2.itemOptionName = _T("바닐라 시럽 추가");
option2.itemOptionUnitPrice = 300;
option2.itemOptionCount = 1;
option2.itemOptionPrice = 300;

// 옵션을 두 번째 상품에 추가
item2.itemOptionList.push_back(option2);

// 두 번째 상품을 transaction의 itemList에 추가
transactionInfo.itemList.push_back(item2);

// SFbitiMembershipBarcodeScanInfo 생성
```

```
SFbitiMembershipBarcodeScanInfo barcodeScanInfo;
barcodeScanInfo.mode = FBITI_MEMBERSHIP_SCANMODE_SERIAL;
barcodeScanInfo.comPort = "COM5";

// SizeInfo 생성
SFbitiMembershipSizeInfo sizeInfo;
sizeInfo.dlgX = 0;
sizeInfo.dlgY = 0;
sizeInfo.dlgWidth = 400;
sizeInfo.dlgHeight = 600;

// DLL 동적 로드
HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
if (hDLL == NULL) {
    AfxMessageBox(L"Failed to load DLL");
    return;
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
```

```
        (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
    if (DestroyFbitiMembershipInstance == NULL) {
        AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
        FreeLibrary(hDLL);
        return;
    }

    // 객체 생성
    DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
    if (pFbitiMembership) {
        // CouponUse 메서드 호출
        this->m_FbitiWnd = pFbitiMembership->CouponUse(hWnd, transactionInfo, barcodeScanInfo, sizeInfo);

        // 객체 삭제
        DestroyFbitiMembershipInstance(pFbitiMembership);
    }

    // DLL 해제
    FreeLibrary(hDLL);
}
```

메세지 수신 리스너 설정 예시 코드

```
#include "FbitiMembershipMessages.h"
#include "CFbitiMembershipMessageSender.h"

BEGIN_MESSAGE_MAP(CClientFbitiMembershipView, CFormView)
    ON_MESSAGE(WM_USER_FBITI_MEMBERSHIP_MESSAGE_COUPON_USE, &CClientFbitiMembershipView::OnFbitiMembershipCouponUseMessage)
END_MESSAGE_MAP()
```

```
LRESULT CClientFbitiMembershipView::OnFbitiMembershipCouponUseMessage(WPARAM wParam, LPARAM lParam)
{
   CFbitiMembershipMessageSender::CouponUseMessage* pMessage =
reinterpret_cast<CFbitiMembershipMessageSender::CouponUseMessage*>(lParam);

   // 메시지 데이터를 사용하여 필요한 처리 수행
   CString info;

   if (pMessage->data.errorCode == "200") {
       info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\n사용된 금액: %d\ncouponItemCode: %s"),
           CString(pMessage->status.c_str()),
           CString(pMessage->result.c_str()),
           CString(pMessage->data.errorCode.c_str()),
           pMessage->data.couponDiscountPrice,
           CString(pMessage->data.couponItemCodes.c_str())
       );
   }
   else {
       info.Format(_T("상태: %s\n결과: %s\n오류코드: %s"),
           CString(pMessage->status.c_str()),
           CString(pMessage->result.c_str()),
           CString(pMessage->data.errorCode.c_str())
       );
   }
   AfxMessageBox(info);

   if (this->m_FbitiWnd != nullptr)
   {
```

```
        // 자식 윈도우 닫기
        CWnd* fbitiWnd = CWnd::FromHandle(this->m_FbitiWnd);
        fbitiWnd->SendMessage(WM_CLOSE, 0, 0);
    }

    return 0;
}
```

응답 값

- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.

```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK",
                "supportMode": "COUPON",
                "transactionID": "1c27375780c811ef9e03",
                "couponDiscountPrice": "2000",
                "couponItemCodes": ["KD101111", "KD108193"]
```

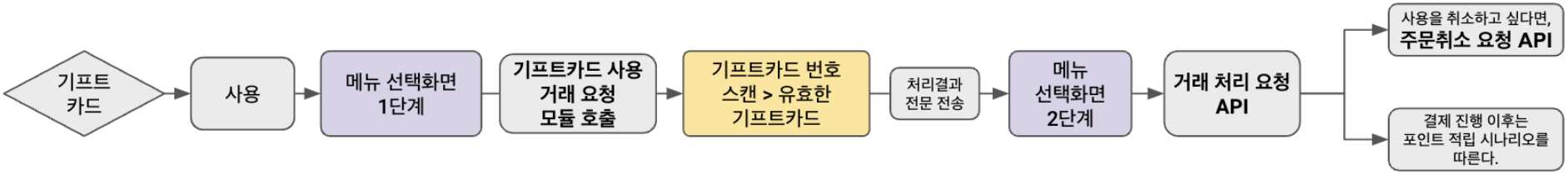
```
        }
}
```

3.3 기프트카드 (기프트카드 거래 요청)

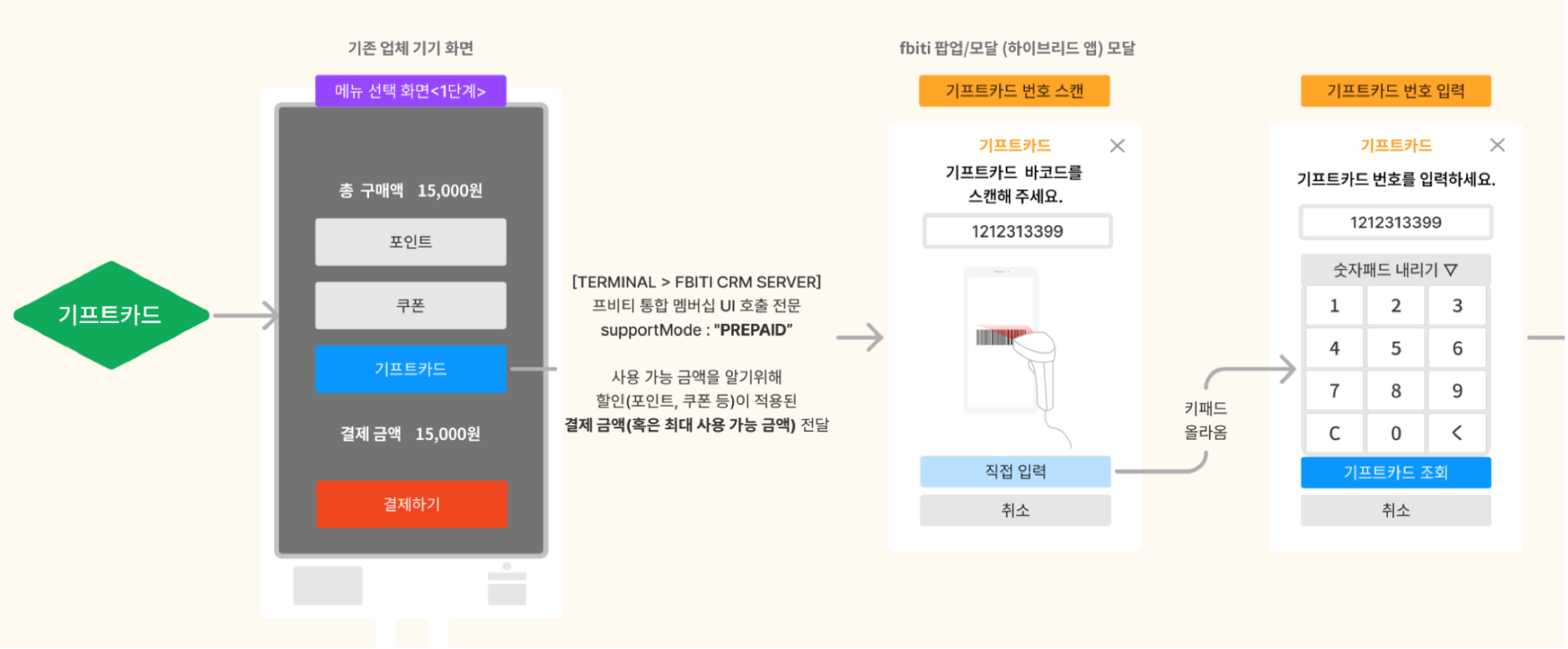
3.3.1 기프트카드 프로세스

기프트카드는 사용 시 가변 바코드를 스캔하거나 16자리 가변 번호를 입력하여 조회할 수 있습니다.

기프트카드를 사용할 때는 프비티 간편 연동 모듈이 호출되며, 모듈에서 가변 바코드의 유효성을 확인한 후 사용 처리 결과를 전송해 드립니다. 키오스크에서 처리가 완료되면 기프트카드 사용이 최종 확정됩니다.



3.3.2 기프트카드 사용 거래 요청



PrepaidUse - 기프트카드 사용

절차	설명	주요 송신 항목	주요 수신 항목
기프트카드 사용 요청	기프트카드 사용 API를 통해 요청 주시면 통합I/F 을 통해 처리한 처리결과 전문을 전송해드립니다.	transactionID : 거래 고유 Transaction ID itemList : 상품정보	transactionID : 거래 고유 Transaction ID

```
HWND PrepaidUse(HWND hParentWnd, STransactionInfo transactionInfo, SFbitiMembershipBarcodeScanInfo barcodeScanInfo, SFbitiMembershipSizeInfo sizeInfo);
```

Parameters

- HWND : 부모 윈도우 핸들

STransactionInfo							
Name			type	length	Option	Value	설명
auth	{						
	tenantID		String		필수		테넌트 ID
	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY

	passwd		String		필수		비밀번호
	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
payAmount			Int		필수		선불카드 : 결제 금액(혹은 최대 사용 가능 금액)
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

SFbitiMembershipBarcodeScanInfo							
Name			**type**	**length**	**Option**	**Value**	**설명**
mode			int		필수	FBITI_MEMBERSHIP_SCANMODE_KEYBOARD : 키보드 FBITI_MEMBERSHIP_SCANMODE_SERIAL : 시리얼	바코드 스캔 모드
comPort			int		필수	예시 : COM5	COM 포트

SFbitiMembershipSizeInfo							
Name			type	length	Option	Value	설명
dlgX			int		필수		X 위치
dlgY			int		필수		Y 위치
dlgWidth			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	너비
dlgHeight			int		필수	가로:세로 비율은 5:7로 하는 것을 권장	높이

Return

- HWND : 다이얼로그 윈도우 핸들

응답 메세지

CFbitiMembershipMessageSender::PrepaidUseMessage							
Name			type	length		Value	설명
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는

							result 텍스트를 송출
result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						
	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	supportMode		String				거래 요청 타입
	transactionID		String				거래 고유 Transaction ID
	prepaidPayPrice		Int				사용 승인 금액
	}						

예시코드

- 고객이 결제해야 하는 금액이 15000 원인 경우 payAmount 의 값으로 15000 을 보낸다.

```
#include "DllFbitiMembership.h"


void CClientFbitiMembershipView::OnBnClickedPrepaidUse()
{
```

```
// 부모 윈도우 핸들 가져오기
HWND hWnd = this->GetSafeHwnd();

// TransactionInfo 생성
STransactionInfo transactionInfo;
transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
transactionInfo.transactionID = "1234567890123456";
transactionInfo.payAmount = 15000;

// SFbitiMembershipBarcodeScanInfo 생성
SFbitiMembershipBarcodeScanInfo barcodeScanInfo;
barcodeScanInfo.mode = FBITI_MEMBERSHIP_SCANMODE_SERIAL;
barcodeScanInfo.comPort = "COM5";

// SizeInfo 생성
SFbitiMembershipSizeInfo sizeInfo;
sizeInfo.dlgX = 0;
sizeInfo.dlgY = 0;
sizeInfo.dlgWidth = 400;
sizeInfo.dlgHeight = 600;

// DLL 동적 로드
HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
if (hDLL == NULL) {
    AfxMessageBox(L"Failed to load DLL");
    return;
```

```
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
    (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
if (DestroyFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 생성
DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
if (pFbitiMembership) {
    // PrepaidUse 메서드 호출
    this->m_FbitiWnd = pFbitiMembership->PrepaidUse(hWnd, transactionInfo, barcodeScanInfo, sizeInfo);

    // 객체 삭제
    DestroyFbitiMembershipInstance(pFbitiMembership);
}
```

```
    // DLL 해제
    FreeLibrary(hDLL);
}
```

메세지 수신 리스너 설정 예시 코드

```
#include "FbitiMembershipMessages.h"
#include "CFbitiMembershipMessageSender.h"

BEGIN_MESSAGE_MAP(CClientFbitiMembershipView, CFormView)
    ON_MESSAGE(WM_USER_FBITI_MEMBERSHIP_MESSAGE_PREPAID_USE, &CClientFbitiMembershipView::OnFbitiMembershipPrepaidUseMessage)
END_MESSAGE_MAP()

LRESULT CClientFbitiMembershipView::OnFbitiMembershipPrepaidUseMessage(WPARAM wParam, LPARAM lParam)
{
    CFbitiMembershipMessageSender::PrepaidUseMessage* pMessage =
reinterpret_cast<CFbitiMembershipMessageSender::PrepaidUseMessage*>(lParam);

    // 메시지 데이터를 사용하여 필요한 처리 수행
    CString info;

    if (pMessage->data.errorCode == "200") {
        info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\n사용된 금액: %d"),
            CString(pMessage->status.c_str()),
            CString(pMessage->result.c_str()),
            CString(pMessage->data.errorCode.c_str()),
            pMessage->data.prepaidPayPrice
        );
```

```
    }
    else {
        info.Format(_T("상태: %s\n결과: %s\n오류코드: %s"),
            CString(pMessage->status.c_str()),
            CString(pMessage->result.c_str()),
            CString(pMessage->data.errorCode.c_str())
        );
    }
    AfxMessageBox(info);

    if (this->m_FbitiWnd != nullptr)
    {
        // 자식 윈도우 닫기
        CWnd* fbitiWnd = CWnd::FromHandle(this->m_FbitiWnd);
        fbitiWnd->SendMessage(WM_CLOSE, 0, 0);
    }

    return 0;
}
```

응답 값

- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.

```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
```

```
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK",
                "supportMode": "PREPAID",
                "transactionID": "1c27375780c811ef9e03",
                "prepaidPayPrice": "5000"
        }
}
```

3.4 거래 처리

거래 처리 요청은 포인트 적립, 포인트 사용, 쿠폰 사용, 기프트카드 사용에 대한 결제가 완료된 후 이를 확정하는 단계입니다. 포인트, 쿠폰, 기프트카드 각각의 사용이 확정되면, 처리할 항목에 따라 요청을 보내 확정 처리를 진행합니다. 처리할 항목에 Y 값을 입력해 주시면 됩니다.

3.4.1 거래 처리 요청

TransactionConfrim - 거래 확정

절차	설명	주요 송신 항목	주요 수신 항목
거래 처리 요청	API를 통해 거래 처리 요청을 보냅니다. transactionID를 기준으로 보내고 포인트, 쿠폰, 기프트카드 중 사용한 항목에 Y로 보내고 사용을 하지않거나 취소한 항목은 N으로 보내면 됩니다.	transactionID : 거래 고유 Transaction ID pointRedeemConfirmedYN : Y couponConfirmedYN : Y pointEarnConfirmedYN : N	transactionID : 거래 고유 Transaction ID
거래 처리 요청 시 전자영수증 발행	거래 처리 요청 시 전자영수증도 함께 발행을 요청할 수 있습니다.	transactionID : 거래 고유 Transaction ID receiptIssueYN : Y itemList : 전자영수증 발행 시 필수	

```
CFbitiMembershipMessageSender::TransactionConfirmMessage* TransactionConfrim(STransactionConfirmInfo transactionInfo);
```

Parameters

STransactionConfirmInfo					

Name			type	length	Option	Value	설명
auth	{						
	tenantID		String		필수		테넌트 ID
	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY
	passwd		String		필수		비밀번호
	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
pointRedeemConfirmedYN			String	1	필수	Y/N, Y : 확정	포인트 사용 확정 여부
pointEarnConfirmedYN			String	1	필수	Y/N, Y : 확정	포인트 적립 확정 여부
couponConfirmedYN			String	1	필수	Y/N, Y : 확정	쿠폰 확정 여부
prepaidConfirmedYN			String	1	필수	Y/N, Y : 확정	선불카드 확정 여부
eReceiptIssueYN			String	1	필수	Y/N, Y : 발행 N: 미발행	전자영수증 발행 여부
itemList	[{						
	itemID		String		필수	KD101111	상품ID

	itemName		String		필수	아메리카노	상품명
	itemUnitPrice		Int		필수	3000	상품 단가
	itemCount		Int		필수	3	상품 수량
	itemPrice		Int		필수	9000	상품 금액
	itemOptionList	[{			선택		
		itemOptionID	String			KD101111-01	상품옵션ID
		itemOptionName	String			샷추가	상품옵션명
		itemOptionUnitPrice	Int			500	상품옵션 단가
		itemOptionCount	Int			2	상품옵션 수량
		itemOptionPrice	Int			1000	상품옵션 금액
		}]					
	}]						
paymentList	[{						
	payment		String		필수	결제수단 입력	결제수단 (CARD/CASH/POINT/PREPAID/COUPON)

	paymentID		String		선택		결제수단 고유ID (카드번호, 선불카드번호, 쿠폰 번호 등)
	transactionID		String		선택		결제수단 거래 승인 트랜잭션 ID (카드의 경우 승인번호)
	paymentIssuer		String		선택		카드사 혹은 발급 / 지급 보증사
	cardPaymentPeriod		String		선택	일시불 / 할부	카드 결제시 할부여부
	cardMerchantID		String		선택	987654321	카드가맹점번호
	paymentPrice		Int		필수		결제금액 (적용금액)
	prepaidBalance		Int		선택		선불카드 잔액
	pointBalance		Int		선택		포인트 잔액
	paymentMemo		String		선택		결제수단 비고 (기타 메모)
	}]						
netAmount			Int		선택	2700	순매출
tax			Int		선택	300	부가세
discount			Int		선택	10000	할인금액
customerPaid			Int		선택	10000	결제금액
totalPaid			Int		선택	3000	총액
memo1			String		선택		안내1
memo2			String		선택		안내2

memo3			String		선택		안내3
memo4			String		선택		안내4
memo5			String		선택		안내5
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

Return

CFbitiMembershipMessageSender::TransactionConfirmMessage							
Name			**type**	**length**		**Value**	**설명**
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출
result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						

	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	transactionID		String				거래 고유 Transaction ID
	customerInfo	{					
		customerName		String			고객 성명
		customerNickName		String			고객 닉네임
		}					
	ponintInfo	{					
		pointRedeem		Int			사용된 포인트
		pointEarn		Int			적립된 포인트
		pointBalance		Int			포인트 잔여
		pointRedeemTransactionID		String			승인번호(사용)
		pointEarnTransactionID		String			승인번호(적립)
		}					
	couponInfo	{					
		couponName		String			쿠폰명

		couponNo		String			쿠폰번호
		discountAmount		Int			결제금액(쿠폰이 적용되어 받은 혜택 금액)
		couponTransactionID		String			승인번호
		}					
	prepaidInfo	{					
		prepaidName		String			카드명
		prepaidNo		String			카드번호
		prepaidNum		Int			사용금액
		prepaidBalance		Int			잔액
		prepaidTransactionID		String			승인번호
		}					
	}						

예시코드

- pointRedeemConfirmedYN 포인트를 사용하지 않아 값이 N 이다
- pointEarnConfirmedYN 포인트를 적립하여 값이 Y 이다
- couponConfirmedYN 쿠폰 사용을 요청했다가 취소하여 값이 N 이다
- prepaidConfirmedYN 기프트카드로 결제하여 값이 Y 이다

```
#include "DllFbitiMembership.h"

void CClientFbitiMembershipView::OnBnClickedTransactionConfirm()
{
   // TransactionInfo 생성
   STransactionConfirmInfo transactionInfo;
      transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
   transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
   transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
   transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
   transactionInfo.transactionID = "1234567890123456";

   transactionInfo.pointRedeemConfirmedYN = 'N';
   transactionInfo.pointEarnConfirmedYN = 'Y';
   transactionInfo.couponConfirmedYN = 'N';
   transactionInfo.prepaidConfirmedYN = 'Y';
   transactionInfo.eReceiptIssueYN = 'N';

   // 첫 번째 상품 추가
   SItemInfo item1;
   item1.itemID = _T("KD101111");
   item1.itemName = _T("아메리카노");
   item1.itemUnitPrice = 3000;
   item1.itemCount = 3;
   item1.itemPrice = 9000;

   // 첫 번째 상품의 옵션 추가
   SItemOptionInfo option1;
   option1.itemOptionID = _T("KD101111-01");
```

```
option1.itemOptionName = _T("샷추가");
option1.itemOptionUnitPrice = 500;
option1.itemOptionCount = 2;
option1.itemOptionPrice = 1000;

// 옵션을 첫 번째 상품에 추가
item1.itemOptionList.push_back(option1);

// 첫 번째 상품을 transaction의 itemList에 추가
transactionInfo.itemList.push_back(item1);

// 두 번째 상품 추가
SItemInfo item2;
item2.itemID = _T("KD101112");
item2.itemName = _T("카페라떼");
item2.itemUnitPrice = 4000;
item2.itemCount = 2;
item2.itemPrice = 8000;

// 두 번째 상품의 옵션 추가
SItemOptionInfo option2;
option2.itemOptionID = _T("KD101112-01");
option2.itemOptionName = _T("바닐라 시럽 추가");
option2.itemOptionUnitPrice = 300;
option2.itemOptionCount = 1;
option2.itemOptionPrice = 300;

// 옵션을 두 번째 상품에 추가
item2.itemOptionList.push_back(option2);
```

```
// 두 번째 상품을 transaction의 itemList에 추가
transactionInfo.itemList.push_back(item2);

SPaymentInfo payment1;
payment1.payment = "POINT";
payment1.paymentID = "";
payment1.transactionID = "";
payment1.paymentIssuer = "";
payment1.cardPaymentPeriod = "";
payment1.cardMerchantID = "";
payment1.paymentPrice = 8500;
payment1.prepaidBalance = 28000;
payment1.pointBalance = 5500;
payment1.paymentMemo = "";

SPaymentInfo payment2;
payment2.payment = "CASH";
payment2.paymentID = "";
payment2.transactionID = "";
payment2.paymentIssuer = "";
payment2.cardPaymentPeriod = "";
payment2.cardMerchantID = "";
payment2.paymentPrice = 8500;
payment2.prepaidBalance = 28000;
payment2.pointBalance = 5500;
payment2.paymentMemo = "";

transactionInfo.paymentList.push_back(payment1);
```

```
transactionInfo.paymentList.push_back(payment2);

// DLL 동적 로드
HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
if (hDLL == NULL) {
    AfxMessageBox(L"Failed to load DLL");
    return;
}

// 객체 생성 함수 로드
CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
    (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
if (CreateFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 삭제 함수 로드
DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
    (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
if (DestroyFbitiMembershipInstance == NULL) {
    AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
    FreeLibrary(hDLL);
    return;
}

// 객체 생성
DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
```

```
if (pFbitiMembership) {
    // TransactionConfrim 메서드 호출
    CFbitiMembershipMessageSender::TransactionConfirmMessage* pMessage = pFbitiMembership->TransactionConfrim(transactionInfo);

    // 메시지 데이터를 사용하여 필요한 처리 수행
    CString info;

    if (pMessage->data.errorCode == "200") {
        info.Format(_T("상태: %s\n결과: %s\n오류코드: %s"),
            CString(pMessage->status.c_str()),
            CString(pMessage->result.c_str()),
            CString(pMessage->data.errorCode.c_str())
        );
    }
    else {
        info.Format(_T("상태: %s\n결과: %s\n오류코드: %s"),
            CString(pMessage->status.c_str()),
            CString(pMessage->result.c_str()),
            CString(pMessage->data.errorCode.c_str())
        );
    }
    AfxMessageBox(info);

    // 객체 삭제
    DestroyFbitiMembershipInstance(pFbitiMembership);
}

// DLL 해제
FreeLibrary(hDLL);
```

```
}
```

응답 값

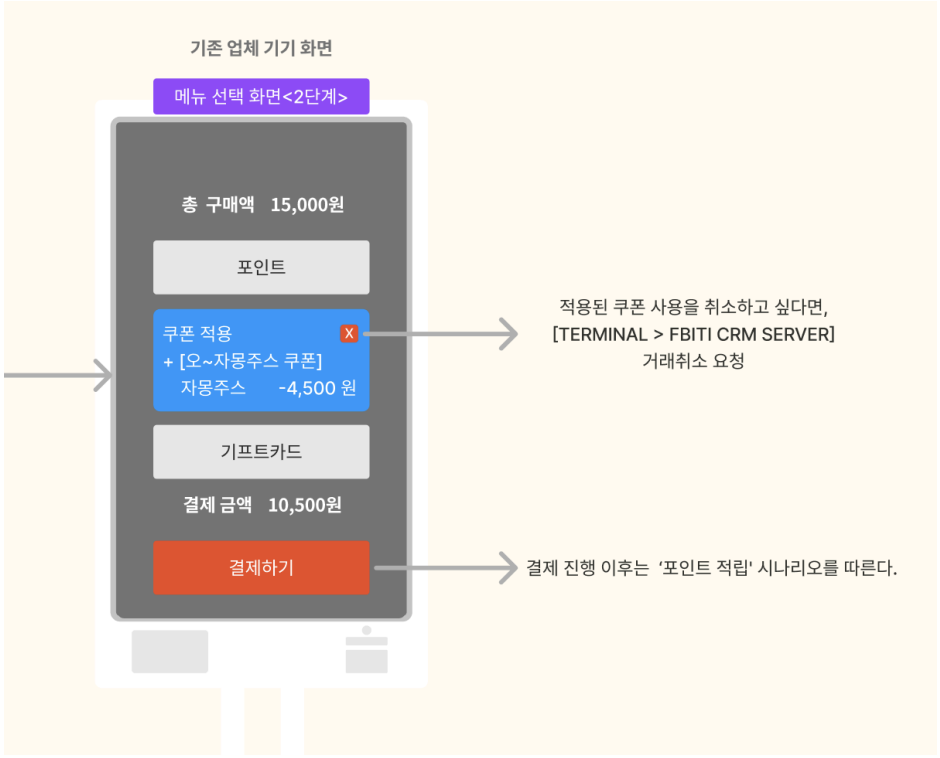
- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.

```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK",
                "transactionID": "1c27375780c811ef9e03"
        }
}
```

3.5 주문 취소

주문 취소는 고객이 키오스크나 포스에서 카드 결제를 취소한 후, 해당 거래에 대한 포인트 적립과 기프트카드 거래를 함께 취소하는 과정입니다. 주문 취소 요청은 거래의 고유 ID(transactionID)를 사용하여 진행되며, 동일한 transactionID에 해당하는 포인트, 쿠폰, 기프트카드 내역이 모두 취소 처리됩니다.

3.5.1 주문 취소 요청



TransactionCancel - 거래 취소

절차	설명	주요 송신 항목	주요 수신 항목
주문취소 요청	API를 통해 주문취소 요청 주시면 거래취소 요청을 보냅니다. transactionID를 통해 처리된 포인트, 쿠폰, 기프트카드 내역을 취소처리합니다.	transactionID : 거래 고유 Transaction ID	transactionID : 거래 고유 Transaction ID pointRedeemCanceledNum : 취소된 사용 포인트 pointEarnCanceledNum : 취소된 적립 포인트 couponCanceledID : 취소된 쿠폰 ID couponCanceledName : 취소된 쿠폰 이름 prepaidCanceledNum : 취소된 기프트카드 금액

```
CFbitiMembershipMessageSender::TransactionCancelMessage* TransactionCancel(STransactionCancelInfo transactionInfo);
```

Parameters

STransactionCancelInfo							
Name			type	length	Option	Value	설명
{							
auth	{						
	tenantID		String		필수		테넌트 ID

	applicationKey		String		필수		서버에 등록되어있는 인가된 APP_KEY
	passwd		String		필수		비밀번호
	accessType		String		필수	TERMINAL ID	접속요청 종류
	}						
transactionID			String		필수	최소 16자 ~ 최대 36자 유효 시간 30분	거래 고유 Transaction ID
}							
autoCloseTime			Int		선택	단위 : ms(밀리세컨드) 기본값 : 30000 1초일 경우 1000	자동 종료 시간

Return

CFbitiMembershipMessageSender::TransactionCancelMessage							
Name			**type**	**length**		**Value**	**설명**
status			String	3		200 : 정상 400 ~ : 오류	오류상태값. 오류는 모두 400번대로 응답 200번만 정상처리이며 오류는 result 텍스트를 송출

result			String	100		오류메시지 텍스트	200 : 정상일경우 OK 400 : 각 상황에 따른 에러메시지 텍스트 제공
timestamp			String	10		1242385883	UNIX TIME STAMP. 추가 정보요청시 함께 송신
data	{						
	errorCode		String				오류코드
	errorMsg		String				오류메시지
	errorMsgUser		String				사용자 오류메시지
	transactionID		String				거래 고유 Transaction ID
	pointRedeemCanceledNum		Int				취소된 사용 포인트
	pointEarnCanceledNum		Int				취소된 적립 포인트
	couponCanceledID		String				취소된 쿠폰 ID
	couponCanceledName		String				취소된 쿠폰 이름
	prepaidCanceledNum		Int				취소된 선불카드 금액
	}						

예시코드

- transactionID 값으로 거래 처리 취소 요청 시 사용한 트랜잭션 ID 를 보낸다.

```
#include "DllFbitiMembership.h"

void CClientFbitiMembershipView::OnBnClickedTransactionCancel()
{
   // 부모 윈도우 핸들 가져오기
   HWND hWnd = this->GetSafeHwnd();

   // TransactionInfo 생성
   STransactionCancelInfo transactionInfo;
   transactionInfo.auth.tenantID = "<프비티 간편 통합 연동 Tenant ID>";
   transactionInfo.auth.applicationKey = "<프비티 간편 통합 연동 Application Key>";
   transactionInfo.auth.passwd = "<프비티 간편 통합 연동 Password>";
   transactionInfo.auth.accessType = "<프비티 간편 통합 연동 Access Type>";
   transactionInfo.transactionID = "1234567890123456";
   transactionInfo.transactionID = "1234567890123456";

   // DLL 동적 로드
   HMODULE hDLL = LoadLibrary(L"DllFbitiMembership.dll");
   if (hDLL == NULL) {
       AfxMessageBox(L"Failed to load DLL");
       return;
   }

   // 객체 생성 함수 로드
   CreateFbitiMembershipInstanceFunc CreateFbitiMembershipInstance =
       (CreateFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "CreateFbitiMembershipInstance");
```

```
    if (CreateFbitiMembershipInstance == NULL) {
        AfxMessageBox(L"Failed to get CreateFbitiMembershipInstance function address");
        FreeLibrary(hDLL);
        return;
    }

    // 객체 삭제 함수 로드
    DestroyFbitiMembershipInstanceFunc DestroyFbitiMembershipInstance =
        (DestroyFbitiMembershipInstanceFunc)GetProcAddress(hDLL, "DestroyFbitiMembershipInstance");
    if (DestroyFbitiMembershipInstance == NULL) {
        AfxMessageBox(L"Failed to get DestroyFbitiMembershipInstance function address");
        FreeLibrary(hDLL);
        return;
    }

    // 객체 생성
    DllFbitiMembership* pFbitiMembership = CreateFbitiMembershipInstance();
    if (pFbitiMembership) {
        // TransactionCancel 메서드 호출
        CFbitiMembershipMessageSender::TransactionCancelMessage* pMessage = pFbitiMembership->TransactionCancel(transactionInfo);

        CString info;

        if (pMessage->data.errorCode == "200") {
            info.Format(_T("상태: %s\n결과: %s\n오류코드: %s\npointRedeemCanceledNum: %d\npointEarnCanceledNum: %d\ncouponCanceledID:
%s\ncouponCanceledName: %s\nprepaidCanceledNum: %d"),
                CString(pMessage->status.c_str()),
                CString(pMessage->result.c_str()),
                CString(pMessage->data.errorCode.c_str()),
```

```
                pMessage->data.pointRedeemCanceledNum,
                pMessage->data.pointEarnCanceledNum,
                CString(pMessage->data.couponCanceledID.c_str()),
                CString(pMessage->data.couponCanceledName.c_str()),
                pMessage->data.prepaidCanceledNum
            );
        }
        else {
            info.Format(_T("상태: %s\n결과: %s\n오류코드: %s"),
                CString(pMessage->status.c_str()),
                CString(pMessage->result.c_str()),
                CString(pMessage->data.errorCode.c_str())
            );
        }
        AfxMessageBox(info);

        // 객체 삭제
        DestroyFbitiMembershipInstance(pFbitiMembership);
    }

    // DLL 해제
    FreeLibrary(hDLL);
}
```

응답 값

- errorCode 의 값이 200 이 아닌 경우 요청 처리가 성공하지 못한 것으로, errorMsg 값으로 실패 사유를 보낸다.
- 고객이 해당 거래로 500 포인트를 적립한 경우 응답 값으로 pointEarnCanceledNum 의 값이 500 이다.

- 고객이 해당 거래에 기프트카드로 8500원을 결제한 경우 응답 값으로 prepaidCanceledNum 의 값이 8500 이다.
- 고객에 해당 거래에 쿠폰을 사용하지 않은 경우 couponCanceledID 와 couponCanceledName 의 값은 null 이다.


```
{
        "status": "200",
        "result": "OK",
        "timestamp": "1728025307",
        "data": {
                "errorCode": 200,
                "errorMsg": "OK",
                "errorMsgUser": "OK",
                "transactionID": "1c27375780c811ef9e03",
                "pointRedeemCanceledNum": 0,
                "pointEarnCanceledNum": 500,
                "couponCanceledID": null,
                "couponCanceledName": null,
                "prepaidCanceledNum": 8500
        }
}
```