
메라크 메일서버

개발자용 설명서

Version 9



SoftMail®

(주)소프트메일

목 차

제 1 장 개요	4
1.1 관리 인터페이스	4
1.2 계정 정보	5
1.3 API.....	6
제 2 장 ODBC	8
2.1 개요	8
2.2 ODBC 드라이버 등록	9
2.3 데이터베이스 추가	10
2.4 ODBC 데이터베이스 등록	13
2.5 메라크 메일서버에서 ODBC 설정 및 변환	16
2.6 데이터베이스 구조	18
제 3 장 IceWarpServer API	24
3.1 개요	24
3.2 TOOL 명령어	26
3.3 TOOL 명령어 예제	27
3.4 로컬 서버 – 사용자 계정	30
3.5 로컬 서버 – 외부 계정	34
3.6 로컬 서버 – 메일링 리스트	39
3.7 외부 서버 – 외부 API	43
3.8 IceWarpCOM – 메일 발송 전용 API.....	49
제 4 장 IceWarpCOM API – 대량 발송용	50
4.1 개요	50
4.2 IceWarpCOM API 등록	50
4.3 IceWarpCOM.Mailer	51
4.4 IceWarpCOM.IMMessage.....	52
제 5 장 DCOM 등록	54
5.1 개요	54

5.2 DCOM 등록	54
5.3 AD 및 문제 해결	57
제 6 장 규칙(Rule, B&W 리스트 필터) 관리	58
6.1 개요	58
6.2 규칙에 관련된 API	59
6.3 규칙 파일의 저장 위치	59
6.4 규칙 파일 형식	60



제 1 장 개요

1.1 관리 인터페이스

관리자와 개발자 입장에서 메라크 메일 서버를 관리하는 방법은 다음과 같습니다.

- 메라크 관리도구(Merak Administration)
- 메라크 원격 관리 도구(Merak Remote Administration)
- 메라크 웹 관리 도구
- TOOL 명령어
- API

메라크 관리도구는 관리자가 서버 컴퓨터에 접속(원격 접속)하여 시작 -> 프로그램 -> 메라크 메일서버 -> 메라크 관리도구를 실행하는 방식으로 일반적으로 이 방법을 많이 사용합니다.

메라크 원격 관리도구는 마치 FTP 클라이언트가 서버에 접속하여 제어하듯이 관리자의 PC에 설치된 메라크 원격관리도구를 이용하여 메라크 서버에 접속하여 원격에서 관리하는 방법으로 별도의 프로그램이 필요합니다. 이 프로그램이 필요한 관리자는 당사로 문의하여 주십시오. 원격 관리도구 프로그램은 서버에 설치된 메라크 메일서버의 버전과 동일해야 합니다.

관리자가 사무실에 있는 경우에는 앞서 언급한 2 가지 방법으로 관리할 수 있지만, 외근을 나가거나 기타 이유로 인해 서버 컴퓨터에 원격으로 접속할 수 없거나 자기 PC를 사용할 수 없는 경우에는 사용하기가 어렵습니다. **메라크 웹 관리 도구**는 웹 메일 인터페이스로 제공되기 때문에 관리도구가 제공하는 기능을 거의 모두 지원합니다. 아래와 같이 웹메일 접속 경로 마지막 부분을 admin으로 변경하여 접속할 수 있습니다. 접속할 때에는 관리자 권한이 필요하므로 먼저 메라크 관리도구에서 해당 사용자의 권한을 설정해야 합니다.

<http://hostname:32000/admin>

TOOL 명령어는 도스 명령어로 사용자, 도메인 또는 메라크 관리도구 내의 특정 설정 값을 보거나 변경할 수 있습니다. 이 프로그램은 메라크 메일서버가 설치된 컴퓨터에서 실행해야 합니다. 또한 와일드카드(*)를 지원하므로 배치 작업을 할 때 아주 유용합니다. 예를 들어, 사용자 계정의 형태를 IMAP에서 IMAP/POP3로 변경해야 한다고 할 때에 앞서 언급한 3가지 관리 방법에서는 사용자의 수가 많은 경우에는 시간이 많이 걸릴 수 있으며 실수로 일부 계정을 누락시킬 수도 있습니다. 이러한 경우에 TOOL 명령어를 사용하면 쉽게 변경할 수 있습니다. TOOL 프로그램은 메라크 메일서버 설치 폴더(기본 경로 - C:\WProgram Files\WMerak)에 저장되어 있습니다.

`tool modify account *@merakdemo.com U_AccountType 1`

위의 예에서 U_AccountType은 계정의 형식을 나타내는 변수로 일반적으로 API의 속성(Property)라고 부르기도 합니다.

지금까지 설명한 관리 방법은 보통 관리자가 직접 다루어야 하는 방식입니다.

이제 설명할 API는 ASP, PHP, C와 같은 프로그래밍 언어를 이용하여 메라크 메일 서버를 관리하는 방법입니다.

보통 그룹웨어나 ERP를 운영하는 환경에서 메라크 메일서버를 동일한 서버 또는 다른 서버에서 운영하는 경우에 사용자 또는 관리자의 편의성을 위해 사용자 계정을 동기화하거나 SSO 기능을 지원하고자 합니다. 또한, 사용자가 그룹웨어/ERP에서 사용하는 비밀번호를 변경한 경우에 이 변경된 값을 메라크 메일서버도 함께 변경되는 단방향 동기화(Sync)를 필요로 합니다.

본 설명서는 이러한 목적에 맞게 특별하게 제작된 것으로 메라크 메일 서버의 동작 원리, API 사용 방법 및 예제 등에 대해 설명합니다.

1.2 계정 정보

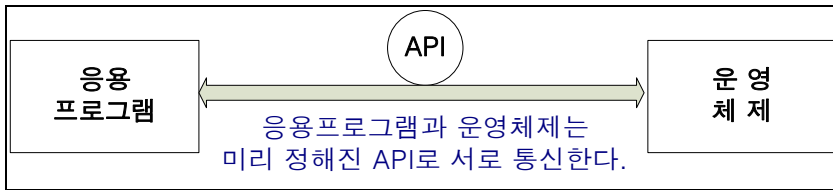
메라크 메일서버는 도메인과 계정 정보를 내부적으로 암호화하여 저장합니다. 하지만, ODBC를 지원함으로써 개발자가 프로그래밍을 통해 계정 정보를 직접 조작할

수 있습니다.

ODBC에 저장하는 방법에 대해서는 **2장. ODBC**에서 자세히 다룹니다.

1.3 API

API는 Application Programming Interface(응용 프로그램 인터페이스)의 약자로 응용 프로그램의 기능을 사용할 수 있도록 미리 정의한 약속을 말합니다.



예를 들어, 사용자가 윈도우 운영체제를 사용하는 경우에는 키보드나 마우스를 이용하여 원하는 프로그램을 실행합니다. 응용 프로그램은 윈도우 운영체제에서 미리 지정된 틀 안에서 동작하며 사용자는 응용 프로그램이 운영체제와 어떻게 동작하는지 알 필요가 없습니다.

이와 마찬가지로 어떤 응용 프로그램의 API를 알고 이용할 수 있으면 실제 이 프로그램이 어떻게 동작하는지 몰라도 필요한 기능을 모두 사용할 수 있습니다.

메라크 메일서버에서는 Direct API와 COM 객체를 이용하는 API 두 가지를 지원합니다. 하지만 COM 객체를 이용하는 방법이 널리 보편화되어 있으며 사용법 또한 쉽습니다. COM 객체는 ASP, VB, PHP와 같이 프로그래밍 언어뿐만 아니라 웹 애플리케이션을 개발하는데 이용할 수 있습니다.

본 설명서에서는 PHP 언어를 이용하여 COM 객체를 다루는 방법에 대해 설명합니다.

메라크 메일서버에서 제공하는 API는 다음과 같습니다.

- api.dll - IceWarpServer
- icewarpcom.dll - IceWarpCOM

IceWarpServer COM 객체에 대해서 중점적으로 다루며, 자세한 내용은 **3장.**

IceWarpServer API에서 설명합니다. 그리고, **IceWarpCOM**은 메일 발송 전용 컴포넌트입니다. 컴포넌트에 대한 자세한 사항은 **4장 IceWarpCOM API - 대량 발송용**에서 자세하게 설명합니다.

알림: Merak v8.91 이하 버전에서는 객체 이름이 **MerakCOM** 입니다.

제 2 장 ODBC

2.1 개요

ODBC(Open Database Connectivity)는 데이터를 액세스하기 위한 표준 개방형 응용 프로그램 인터페이스를 의미합니다. 프로그램 또는 웹 사이트에서 ODBC 쿼리 문장을 사용하여 MS 액세스, MS SQL, MySQL, Oracle과 같은 데이터베이스 엔진에서 제공하는 데이터베이스를 액세스할 수 있습니다.

ODBC를 사용하기 위해서는 해당 데이터베이스와 연동할 수 있는 별도의 모듈 즉, 드라이버가 설치되어 있어야 합니다. 초기에는 마이크로소프트에서 제공하는 윈도우 제품용 ODBC 드라이버가 출시되었지만 현재에는 리눅스, 유닉스 등 다양한 운영 체제에서 사용할 수 있는 ODBC 드라이버가 제공되고 있습니다.

메라크 메일서버에서는 다음과 같은 정보를 데이터베이스에 저장할 수 있습니다. 이러한 데이터베이스에 저장된 데이터를 검색하거나 조작하려면 ODBC를 이용합니다.

- 계정 정보
- 로그
- 안티 스팸
- 그룹웨어

이 중에서 계정 정보는 주로 회사 내의 인트라넷(또는 그룹웨어, ERP 등등)과 상호 연동하여 SSO(Single Sign-On)을 구현할 때 이용합니다.

즉, 인트라넷에 로그인하는 순간 이미 메라크 웹메일 또는 자체적으로 구현한 웹메일에서 자동으로 로그인하여 메일을 처리할 수 있게 합니다. 또한 개발적인 측면에서 볼 때 인트라넷의 관리자가 새로운 직원을 추가하면 자동으로 메라크 메일서버에 메일 사용자 계정이 추가되고, 비밀번호를 바꾸면 함께 바뀌고, 직원이 퇴사하면 자동으로 메일 사용을 하지 못하도록 계정을 삭제하는 방식으로 개발할 수 있습니다.

참고로, 안티 스팸 및 그룹웨어 데이터베이스는 메라크 메일서버 엔진에서 자체적으로 사용하는 부분입니다.

본 설명서에서는 MS SQL 데이터베이스 엔진을 기준으로 설명합니다.

메라크 메일서버에서는 MS 액세스, MS SQL, MySQL, InterBase, Oracle, FireBird 와 같이 ODBC를 지원하는 대부분의 데이터베이스 엔진에서 사용할 수 있습니다.

지원하는 드라이버는 ODBC, MySQL, SQLite 등입니다. 특히, MySQL에서는 PDO 를 지원하지만 본 설명서의 범위를 벗어나므로 설명하지는 않습니다.

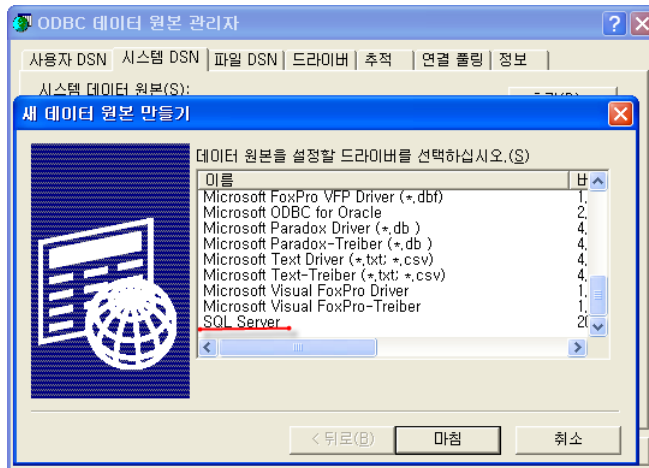
이 장에서는 다음의 사항에 대해 설명합니다.

- ODBC 드라이버 등록
- 데이터베이스 추가
- ODBC 데이터베이스 등록
- 메라크 메일서버에서 ODBC 설정 및 변환

2.2 ODBC 드라이버 등록

MS 액세스(MDB)와 MS SQL 데이터베이스를 이용할 수 있는 ODBC 드라이버는 윈도우에서 기본적으로 제공하기 때문에 추가적으로 드라이버를 등록할 필요는 없습니다.

기본적으로 제공하는 드라이버의 목록을 보기 위해서는 **시작 -> 모든 프로그램 -> 관리 도구 -> 데이터 원본(ODBC)**을 선택합니다. **시스템 DSN** 탭을 클릭하고 **추가** 버튼을 클릭합니다.



MySQL과 Oracle과 같은 마이크로소프트 이외의 데이터베이스 엔진을 이용하려면 해당 드라이버를 설치해야 합니다.

MySQL용 ODBC 드라이버는 아래 링크에서 다운로드 받아 설치할 수 있습니다. 설치 과정은 쉬우므로 별도로 설명하지 않습니다.

<http://www.mysql.com>

Oracle에서 사용하는 ODBC는 DBA의 도움을 받으시기 바랍니다.

참고로, 메라크 메일서버는 MS SQL 7, MySQL 4, Oracle 8.17 이상의 버전에서 사용할 수 있습니다. 하지만, 한글을 완벽하게 지원하기 위해서는 UTF-8을 지원하는 데이터베이스 버전을 사용하는 것이 좋습니다.

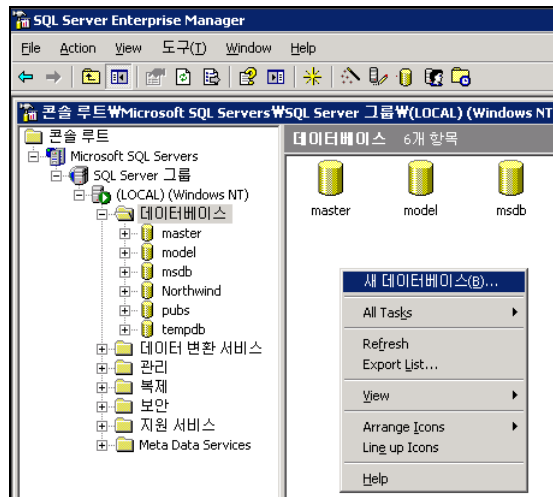
2.3 데이터베이스 추가

MS SQL에서 메라크 메일서버의 계정 정보를 보관할 데이터베이스를 관리하는 방법에 대해 설명합니다. 다음과 같이 두 가지 과정으로 분류할 수 있습니다.

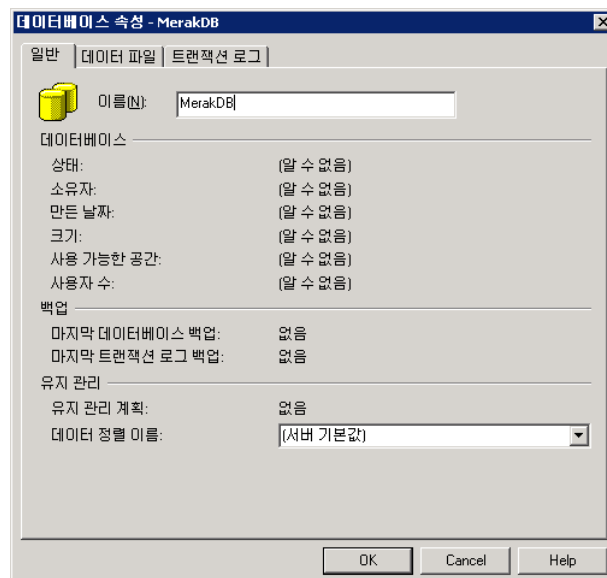
- 데이터베이스 생성
- 관리 계정 추가 및 권한 부여

2.3.1 데이터베이스 생성

MS SQL의 데이터베이스 관리 프로그램인 **Enterprise Manager**(또는 MS SQL Management Studio)에서 **Microsoft SQL Server -> SQL Server 그룹 -> 서버 이름 -> 데이터베이스**로 이동합니다. 오른쪽 세부 창에서 마우스 오른쪽 버튼을 클릭하고 **새 데이터베이스**를 선택합니다.

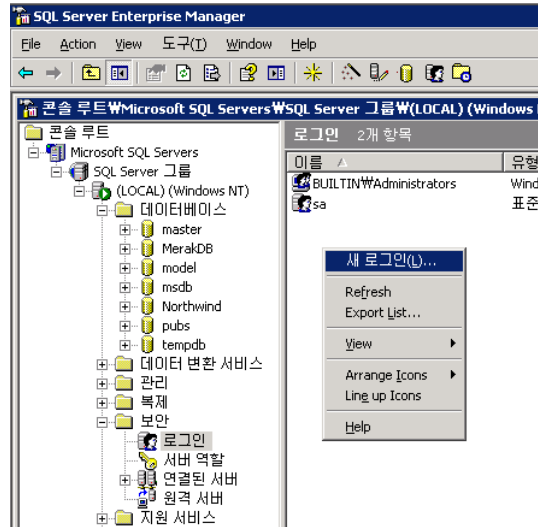


데이터베이스 속성 대화상자의 일반 탭에서 이름을 입력하고 **확인** 버튼을 클릭합니다.

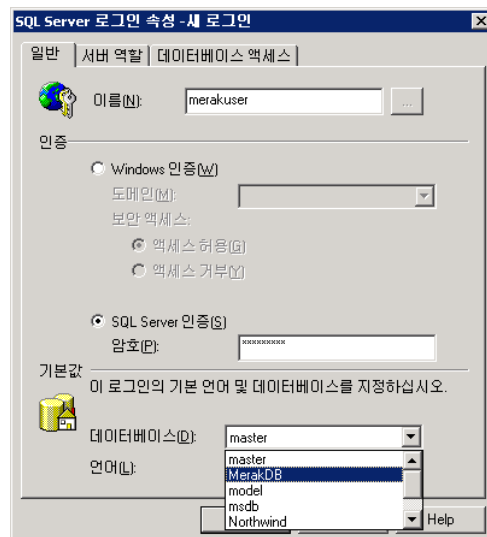


2.3.2 관리 계정 추가 및 권한 부여

MS SQL의 데이터베이스 관리 프로그램인 **Enterprise Manager**(또는 MS SQL Management Studio)에서 **Microsoft SQL Server -> SQL Server 그룹 -> 서버 이름 -> 보안 -> 로그인**으로 이동합니다. 오른쪽 세부 창에서 마우스 오른쪽 버튼을 클릭하고 **새 로그인**을 선택합니다.

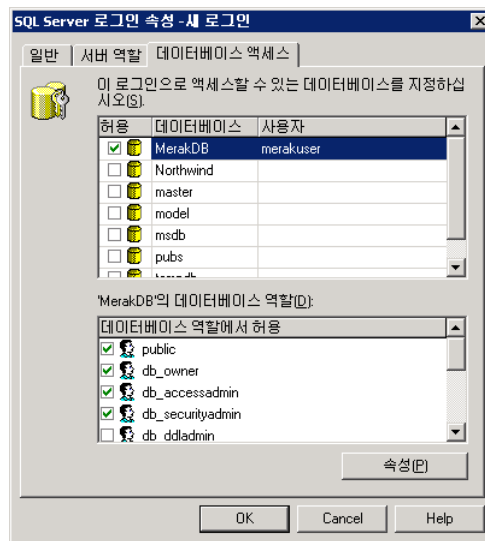


SQL Server 로그인 속성 - 새 로그인 대화상자에서 관리 계정의 이름을 입력합니다. 마지막으로 **데이터베이스** 항목에서 앞서 생성한 데이터베이스를 선택합니다.



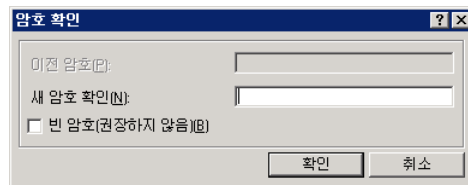
데이터베이스 액세스 탭을 클릭합니다. 앞서 선택한 데이터베이스를 선택하고, 하단의 데이터베이스 역할 항목에서 다음과 같이 선택하고 확인 버튼을 클릭합니다.

- db_owner
- db_accessadmin
- db_securityadmin
- db_datareader
- db_datawriter



알림: 일반적으로 db_owner 권한이 다른 권한보다 높기 때문에 이 권한만 설정해도 무방합니다.

아래와 같이 비밀번호를 한번 더 확인하는 **암호 확인** 대화상자가 나타날 수도 있습니다. 정확히 다시 한번 입력하고 **확인** 버튼을 클릭합니다.



2.4 ODBC 데이터베이스 등록

ODBC에서 등록하려는 데이터베이스의 드라이버를 설치한 이후에 ODBC 데이터베이스를 등록할 수 있습니다.

ODBC 데이터베이스는 메라크 메일 서버가 설치된 컴퓨터에서 등록합니다.

데이터베이스를 등록하기 위해서는 데이터베이스 이름, 데이터베이스가 설치되어 있는 서버의 호스트 이름(또는 IP 주소), 해당 데이터베이스를 액세스할 수 있는 계정 정보(사용자 ID, 비밀번호)가 필요합니다.

데이터베이스 엔진은 로컬 컴퓨터 또는 외부에 있는 다른 서버에 위치할 수도 있습니다. 외부에 있는 경우에는 방화벽에서 적절한 포트를 개방하는 과정이 추가적으로 필요합니다. 이에 대한 자세한 사항은 네트워크 관리자 또는 방화벽 엔지니어에게 문의하시기 바랍니다.

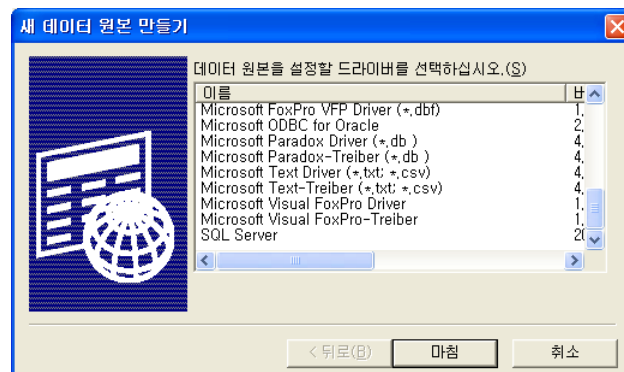
ODBD 데이터베이스를 등록하는 과정은 다음과 같이 두 가지로 나뉩니다.

- ODBC 드라이버 선택
- 데이터 원본 정보 입력

MS SQL을 기준으로 설명합니다.

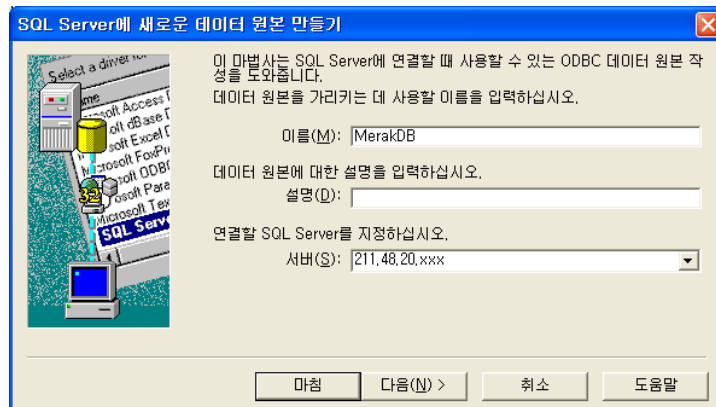
2.4.1 ODBC 드라이버 선택

시작 -> 모든 프로그램 -> 관리 도구 -> 데이터 원본(ODBC)을 선택합니다. 시스템 DSN 탭을 클릭하고 추가 버튼을 클릭합니다. SQL Server를 선택하고 확인 버튼을 클릭합니다.

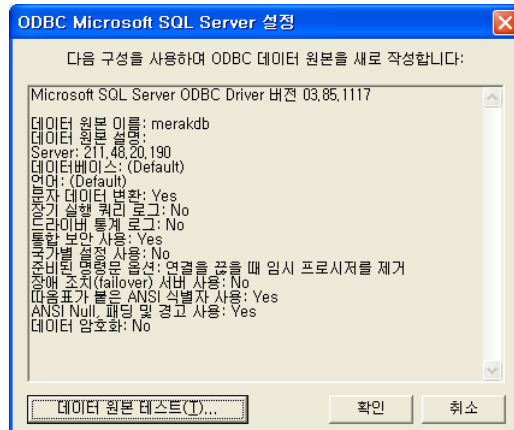


2.4.2 데이터 원본 정보 입력

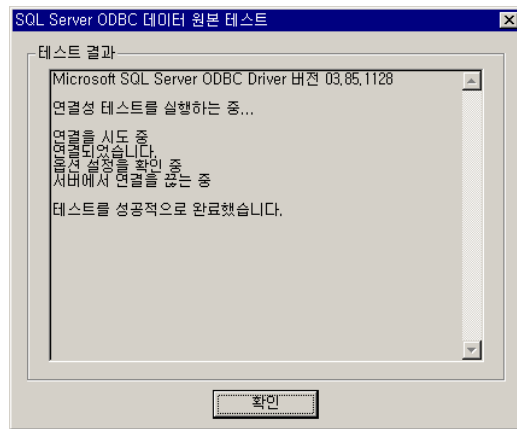
SQL Server에 새로운 데이터 원본 만들기 대화상자의 이름(ODBC에서 식별하기 위한 이름)과 서버의 호스트 이름(또는 IP 주소)을 입력하고 다음 버튼을 클릭합니다.



로그인 정보를 입력하는 대화상자가 나타납니다. 일반적으로 사용자가 입력한 로그인 ID 및 암호를 사용하는 SQL Server 인증 사용을 선택하고 로그인 ID와 암호를 정확히 입력합니다. 그리고 다음 버튼을 클릭합니다.



설정이 정확한지 확인하기 위해 데이터 원본 테스트 버튼을 클릭합니다. 아래와 같은 화면이 나타나면 ODBC에 데이터베이스 등록이 완료됩니다.



주의: Windows XP, 2003, 2008 64비트 운영체제에서는 64비트용 ODBC와 32비트 ODBC를 모두 제공합니다. 하지만, 메라크 메일서버에서는 32비트 ODBC만을 지원하므로 **Windows\SysWOW64** 폴더에 위치한 **odbcad32.exe** 프로그램을 실행하여 ODBC 등록 작업을 진행해야 합니다.

2.5 메라크 메일서버에서 ODBC 설정 및 변환

이제 메라크 메일서버에서 계정 정보를 ODBC에서 등록한 데이터베이스로 저장하도록 설정하는 방법과 계정 정보를 데이터베이스로 변환하는 방법을 설명합니다.

참고로 메라크 메일 서버에서 계정 정보를 저장하고 이용하는 방식은 아래와 같이 모두 세 가지입니다.

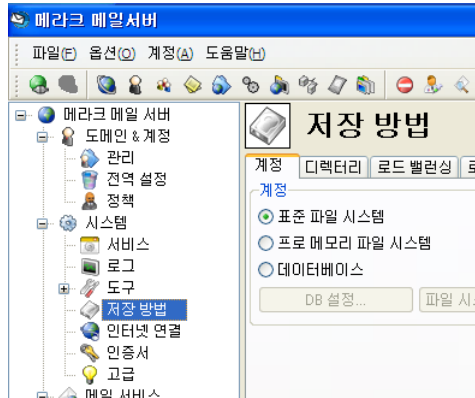
- 파일 시스템
- 프로 메모리 파일 시스템
- 데이터 베이스

프로 메모리 파일 시스템은 기본적으로 파일 시스템과 유사하지만 계정에 관련된 정보를 메모리에 캐시 하여 저장함으로써 보다 빠른 액세스를 제공해 줍니다. 보통 500 명 미만의 사용자인 경우에는 파일 시스템을 사용하고 그 이상인 경우에는 프로 메모리 파일 시스템을 사용하는 것이 좋습니다.

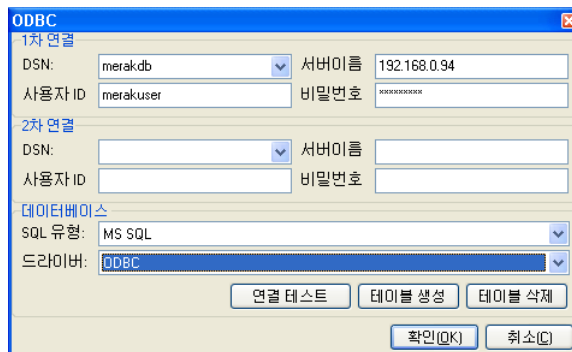
메라크 메일 서버에서 계정 정보를 데이터베이스로 저장하는 방법은 다음과 같습니다.

다.

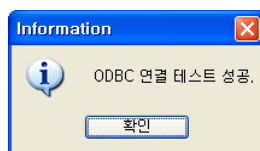
메라크 관리도구를 열고, 시스템 -> 저장 방법 -> 계정 탭으로 이동합니다.



데이터베이스를 선택하고 DB 설정... 버튼을 클릭합니다. 아래와 같이 DSN을 선택하고 서버 이름, 사용자 ID, 비밀번호를 입력합니다. 그리고, SQL 유형과 드라이버를 정확히 선택합니다.

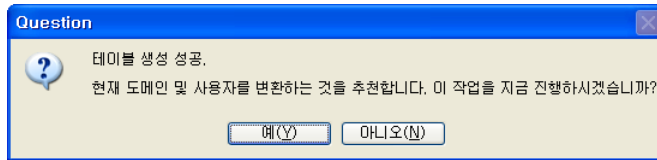


연결 테스트 버튼을 클릭하여 ODBC가 성공적으로 연결되었는지 반드시 확인합니다. 오류가 나타날 경우에는 처음부터 하나하나 다시 확인하면서 진행합니다.

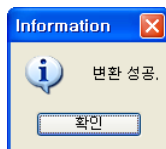


이제 테이블을 생성합니다. 참고로 메라크 메일서버의 계정 정보는 Users, Aliases, Domains 등 세 개이며 자동으로 생성할 수 있습니다. 테이블에 대한 자세한 사항은 2.6 데이터베이스 구조 단원을 참고하시기 바랍니다.

테이블 생성 버튼을 클릭합니다.



테이블이 정상적으로 생성되면 아래와 같이 계정 정보를 변환할지 묻습니다. **예**를 클릭합니다.



마지막으로 메라크 관리도구 오른쪽 하단에 있는 **적용** 버튼을 클릭하여 변경 사항을 반영합니다.

2.6 데이터베이스 구조

데이터베이스에 저장되는 계정 정보는 다음과 같이 모두 3가지로 구성됩니다.

- Domains
- Users
- Aliases

참고로 테이블 구조에 대해 언급하는 이유는 API와 함께 사용할 수 있는 방법을 미리 소개하기 위함입니다. 앞서 API를 통해 메라크 메일서버의 데이터를 조작할 수 있지만, 데이터베이스에 저장된 데이터는 프로그래밍 언어(SQL 쿼리 문장)를 통해 직접 레코드를 수정할 수도 있기 때문입니다.

2.6.1 Domains 테이블

메라크 메일서버의 관리도구에서 등록한 도메인에 대한 정보가 저장됩니다. 보통 도메인의 등록이나 삭제, 이름 변경은 관리도구에서 직접 처리하는 경우가 많습니다. 또한 Domains 테이블의 레코드를 직접 변경하는 경우에는 문제점이 발생할 수 있는 소지가 있으므로 가급적 직접 조작하지 않도록 주의하여야 합니다.

Domains 테이블의 구조는 다음과 같습니다. 진하게 표시한 부분이 도메인 이름을 저장하는 레코드입니다.

```
CREATE TABLE [Domains] (
    [D_ID] [int] NOT NULL,
    [D_Domain] [varchar] (95) NULL,
    [D_IP] [varchar] (16) NULL,
    [D_Set] [varchar] (1) NULL,
    [D_Description] [varchar] (56) NULL,
    [D_DiskQuota] [int] NULL,
    [D_AVScan] [varchar] (1) NULL,
    [D_AccountNumber] [int] NULL,
    [D_DomainType] [int] NULL,
    [D_DomainTo] [varchar] (63) NULL,
    [D_PostMaster] [varchar] (127) NULL,
    [D_AdminEmail] [varchar] (127) NULL,
    [D_UnknownUsersForward] [int] NULL,
    [D_UnknownUsersForwardTo] [varchar] (127) NULL,
    [D_IM] [varchar] (1) NULL,
    [D_Calendar] [varchar] (1) NULL,
    [D_ChallengeResponse] [varchar] (1) NULL,
    [D_MIAS] [varchar] (1) NULL,
    [D_DisableAllUsers] [varchar] (1) NULL,
    [D_UserMailbox] [int] NULL,
    [D_UserMB] [int] NULL,
    [D_UserNumber] [int] NULL,
    [D_UserMsg] [int] NULL,
```

```

[D_InfoToAdmin] [varchar](1) NULL,
[D_NumberLimit] [int] NULL,
[D_VolumeLimit] [int] NULL,
[D_Expires] [varchar](1) NULL,
[D_ExpiresOn] [int] NULL,
[D_Notify] [varchar](1) NULL,
[D_NotifyBefore] [int] NULL,
[D_DeleteExpired] [varchar](1) NULL,
[D_Hostname] [varchar](63) NULL,
[D_FolderPath] [varchar](63) NULL,
[D_VerifyType] [varchar](1) NULL,
PRIMARY KEY CLUSTERED
(
    [D_ID] ASC <- 이 값을 고유한 일련번호로 사용합니다.
) ON [PRIMARY]

```

2.6.2 Users 테이블

메라크 메일서버에서는 아래와 같이 모두 9가지의 계정 형태를 지원합니다.

- 사용자 계정
- 그룹 계정
- 메일링 리스트 계정
- 리스트 서버
- 실행 계정
- 외부 계정
- 고정 라우팅
- 통지 계정
- 카탈로그 계정

여기서 중요한 점은 바로 **외부 계정(Remote Account)**은 데이터베이스에 저장되지 않고 메라크 관리도구 자체의 데이터 영역에 저장됩니다.

따라서, 앞서 설명한 것과 같이 데이터베이스를 조작하는 방식으로는 외부

계정을 처리할 수 없습니다. 외부 계정은 오직 API를 통해서만 조작할 수 있습니다.

아래는 계정의 형태(U_Type)로 정렬한 화면입니다.

	U_Type	U_ID	U_Alias	U_Domain
1	0	2	user	merakdemo2.com
2	1	4	mailinglist	merakdemo2.com
3	2	6	executable	merakdemo2.com
4	3	8	notification	merakdemo2.com
5	4	7	staticrouting	merakdemo2.com
6	5	9	catalog	merakdemo2.com
7	6	5	listserver	merakdemo2.com
8	7	3	group	merakdemo2.com

Users 테이블의 구조는 다음과 같습니다. 가장 중요한 부분은 진하게 표시하였습니다.

```
CREATE TABLE [Users] (
    [U_Type] [int] NULL, <- 계정 형식
    [U_ID] [int] NOT NULL,
    [U_Alias] [varchar] (255) NULL, <- 사용자 별명
    [U_Domain] [varchar] (95) NULL,
    [U_AntiSpamIndex] [int] NULL,
    [U_Name] [varchar] (127) NULL, <- 사용자 이름
    [U_Mailbox] [varchar] (127) NULL, <- 사용자 ID
    [U_AccountDisabled] [varchar] (1) NULL,
    [U_AccountValid] [varchar] (1) NULL,
    [U_AccountValidTill] [int] NULL,
    [U_AllowRemote] [varchar] (1) NULL,
    [U_ValidityReport] [varchar] (1) NULL,
    [U_ValidityReportDays] [int] NULL,
    [U_AuthMode] [varchar] (1) NULL,
    [U_AccountType] [varchar] (1) NULL,
    [U_IMAPMailbox] [varchar] (63) NULL,
    [U_DontShowMessagesSize] [int] NULL,
    [U_AVScan] [varchar] (1) NULL,
    [U_AnyPassword] [varchar] (1) NULL,
    [U_ETRN] [varchar] (1) NULL,
    [U_DeleteExpire] [varchar] (1) NULL,
```

```

[U_NULL] [varchar](1) NULL,
[U_Password] [varchar](63) NULL, <- 사용자 비밀번호
[U_AuthModeValue] [varchar](127) NULL,
[U_IASCustom] [int] NULL,
[U_DomainAdmin] [varchar](1) NULL,
[U_MailboxPath] [varchar](127) NULL,
[U_Admin] [varchar](1) NULL,
[U_MaxBox] [varchar](1) NULL,
[U_MaxBoxSize] [int] NULL,
[U_RespondOnlyIfToMe] [varchar](1) NULL,
[U_Respond] [int] NULL,
[U_ServicesAccess] [int] NULL,
[U_UseRemoteAddress] [varchar](1) NULL,
[U_ForwardTo] [varchar](255) NULL,
[U_RespondWith] [varchar](127) NULL,
[U_MailIn] [varchar](127) NULL,
[U_MailOut] [varchar](127) NULL,
[U_ValidReport] [varchar](127) NULL,
[U_DeleteOlder] [varchar](1) NULL,
[U_DeleteOlderDays] [int] NULL,
[U_ForwardOlder] [varchar](1) NULL,
[U_ForwardOlderDays] [int] NULL,
[U_ForceFromAddress] [varchar](127) NULL,
[U_ForwardOlderTo] [varchar](127) NULL,
[U_RemoteAddress] [varchar](127) NULL,
[U_NoMailList] [varchar](1) NULL,
[U_NumberSendLimit] [int] NULL,
[U_MegabyteSendLimit] [int] NULL,
PRIMARY KEY CLUSTERED
(
    [U_ID] ASC <- 이 값을 고유한 일련번호로 사용합니다.
) ON [PRIMARY]

```

2.6.3 Aliases 테이블

사용자 계정(ID)은 여러 개의 별명(Alias)를 보유할 수 있습니다. 따라서, Aliases 테이블에서는 사용자 ID와 별명만 관리합니다.

Aliases 테이블의 구조는 다음과 같습니다.

```
CREATE TABLE [Aliases] (  
    [A_Alias] [varchar] (63) NULL,  
    [A_Domain] [varchar] (95) NULL,  
    [A_UserID] [int] NOT NULL  
) ON [PRIMARY]
```

제 3 장 IceWarpServer API

3.1 개요

메라크 API에서는 프로그램이나 스크립트 언어를 이용하여 메라크 메일서버의 도메인, 사용자 등의 계정 정보와 환경 설정 정보를 추가, 변경, 삭제할 수 있는 기능을 제공합니다. 일반적으로 그룹웨어와 연동하는 경우가 많으므로 대부분 ASP와 같은 웹 개발 또는 스크립트 언어를 이용합니다.



메라크 메일서버에서는 Direct API와 COM 객체를 이용한 API를 지원합니다. 일반적으로 COM 객체를 이용하는 방법을 선호하므로 이 방법을 중심으로 설명합니다.

또한 원격에 위치한 COM 객체를 액세스하기 위해 RPC(Remote Procedure Call)를 지원합니다.

COM 객체를 지원하는 API는 메라크 설치 폴더 내에 저장되어 있는 api.dll 파일이 담당합니다.

메라크 메일 서버가 제공하는 API의 클래스는 다음과 같습니다.

- IceWarpServer.APIObject
APIObject 클래스는 메라크 시스템의 환경 설정 백업, 복원, 전역 설정 등에 대한 객체를 제어합니다.
- IceWarpServer.DomainObject
도메인 추가, 변경, 삭제 등과 같이 도메인에 관련된 객체를 제어합니다.
- IceWarpServer.AccountObject
계정 추가, 변경 삭제 등과 같이 계정에 관련된 객체를 제어합니다.

- IceWarpServer.RemoteAccountObject
외부 계정을 추가, 변경 삭제하는 등 외부 계정 객체를 제어합니다.
- IceWarpServer.ScheduleObject
일정한 기간마다 실행하는 예약 기능을 제어합니다.
- IceWarpServer.StatisticsObject
통계 정보를 보거나 변경합니다.
- IceWarpServer.TokenObject
외부 API를 이용할 수 있도록 토큰 객체를 제어합니다.

클래스 중에서는 AccountObject, RemoteAccountObject, ScheduleObject, TokenObject가 가장 많이 사용됩니다.

클래스는 함수(Function), 프로시저(Method)와 변수(Variable)로 구성됩니다. 함수 및 프로시저는 **classes.txt**(메라크 설치 폴더 / api), 변수는 **apiconst.pas**(메라크 설치 폴더 / api / delphi)에 자세히 설명되어 있습니다.

3 장에서는 TOOL 명령어 및 사용방법에 대해 간략히 살펴 봅니다. 그리고, 가장 중요한 부분인 사용자 계정 추가 및 비밀번호 변경, 메일링 리스트 추가 및 비밀번호 변경, 외부 계정 추가 및 별도의 테이블 관리에 대해 설명합니다.

주의: 메라크 메일서버에서 사용하는 계정 정보는 모두 데이터베이스(ODBC)로 저장해야 하며, 프로그램이나 스크립트에서는 API를 이용해야 합니다. 외부 계정은 ODBC로 저장할 수 없으며 오직 API로만 제어할 수 있습니다. 또한 메일링 리스트 계정에 속한 구성원은 파일이나 ODBC 쿼리 문장으로 지정할 수 있습니다. 하지만 메라크 메일서버가 설치되지 않은 다른 WAS(웹 애플리케이션 서버)에서는 원격으로 파일을 조작하기가 수월하지 않기 때문에 특정한 테이블을 생성하여 이 부분에서 관리하는 방식을 소개합니다.

3.4 단원부터는 PHP 언어를 이용하여 메라크 메일서버의 사용자 계정, 메일링 리스트 계정, 외부 계정을 처리하는 방법을 소개합니다. 아울러 외부 WAS 서버에서 제어하는 외부 API에 대해서도 설명합니다.

본 설명서는 MS SQL 데이터베이스 엔진을 기준으로 설명합니다.

3.2 TOOL 명령어

메라크 메일 서버에서 지원하는 객체를 제어하기 위해 프로그램을 작성하거나 스크립트를 만들어야 하는 수고를 덜 수 있는 TOOL 명령어를 제공합니다.

TOOL 명령어는 도스 프롬프트에서 관리자가 직접 입력하거나 미리 만들어 놓은 배치 파일로 실행하는 방법으로 사용할 수 있습니다.

TOOL 명령어를 이용하여 메라크 메일서버가 제공하는 모든 API를 제어할 수 있습니다.

TOOL 명령어의 형식은 다음과 같습니다.

TOOL.EXE <명령어> <객체> <변수> <옵션>
--

지원하는 명령어(함수)는 다음과 같습니다. 자주 사용하는 명령어는 진하게 표시하였습니다.

- **create** <객체> - 새로운 객체를 생성합니다.
- **delete** <객체> - 객체를 삭제합니다.
- **modify** <객체> - 객체의 변수 값을 변경합니다.
- **display** <객체> - 객체의 변수 값을 보여 줍니다.
- **check** <객체> - 객체 데이터가 올바른지 확인합니다.
- **export** <객체> - 객체 변수 값을 CSV 형식으로 내보냅니다.
- **import** <객체> - CSV 형식의 파일로부터 객체 변수 값을 가져옵니다.
- **upgrade** <객체> - 기존 버전에서 업그레이드 함수를 호출합니다.
- **search** <객체> - 입력한 문자열에 관련된 API를 검색합니다.

지원하는 객체는 다음과 같습니다. 변수 이름은 앞에서 언급했었던 메라크 API의 클래스 이름과 유사합니다. 자주 사용하는 객체는 진하게 표시하였습니다.

- **system** - 시스템에 관련된 객체입니다.
- **domain** - 도메인에 관련된 객체입니다.
- **account** - 계정에 관련된 객체입니다.

- remoteaccount - 외부 계정에 관련된 객체입니다.
- service - 서비스에 관련된 객체입니다.
- userstatistics - 사용자 통계에 관련된 객체입니다.
- tables - 연결 문자열에 관련된 객체입니다.
- backup - 환경 설정 백업에 관련된 객체입니다.
- storage - 저장 방법에 관련된 객체입니다.

세 번째 입력해야 할 부분은 바로 변수입니다. 객체가 지원하는 변수는 종류도 다양하기 때문에 실제 외워서 사용하기가 어렵습니다. 변수에 대한 상세한 정보는 apiconst.pas 파일을 참고하십시오. 변수는 프로그래밍 언어에서도 그대로 사용하므로 가장 핵심적으로 필요한 변수를 따로 정리하여 두면 편리합니다.

네 번째 옵션은 다음과 같은 기능을 제공합니다. 자주 사용하는 옵션은 진하게 표시하였습니다.

- -m <템플릿 이름> - 객체 템플릿을 적용합니다.
- -r <외부 접속 정보> - 원격에 위치한 메라크 메일서버를 제어하기 위한 접속 정보를 지정합니다.
- -p <서버 경로> - 서버 디렉터리의 경로를 지정합니다.
- -f - 계정 정보를 보는 과정에서 필터링 기능을 제공합니다.
- -q - 실행하는 동안 화면에 아무런 표시도 하지 않습니다. 배치 파일에서 주로 사용하는 옵션입니다.
- -h - 도움말을 보여 줍니다.
- -t - TOOL 명령어의 사용 예제를 보여 줍니다.
- -v - 메라크 메일서버의 버전 정보를 보여 줍니다.

다음 단원에서는 자주 사용하는 TOOL 명령어의 예를 다룹니다.

3.3 TOOL 명령어 예제

예제에서는 반드시 표시하는 명령어는 대문자로 입력하고 관리자가 입력하는 부분은 소문자로 표시합니다. 그리고, 변수는 대소문자로 표시합니다.

3.3.1 도메인 추가

도메인을 추가할 때 반드시 필요한 변수는 다음과 같습니다.

- D_Type - 도메인의 형식을 지정합니다. 표준 도메인인 경우에는 생략해도 무방합니다.
- D_AdminEmail - 도메인 관리자의 메일주소를 지정합니다.

```
TOOL CREATE DOMAIN merakdemo.com D_AdminEmail "support@merakdemo.com"
```

3.3.2 도메인 삭제

```
TOOL DELETE DOMAIN merakdemo.com
```

3.3.3 사용자 추가

사용자를 추가할 때에는 다음과 같은 변수가 필요합니다.

- U_Name - 계정 이름으로 관리도구에서 관리자가 식별 목적으로 사용합니다.
- U-MailBox - 사용자 이름(ID)를 지정합니다.
- U_Password - 사용자 이름에 해당하는 비밀번호를 지정합니다.

```
TOOL CREATE ACCOUNT admin@merakdemo.com U_Name "관리자" U-MailBox  
"admin" U_Password "adminpassword"
```

주의: U-Mailbox 값을 입력하지 않는 경우에는 메일 주소 @ 앞에 있는 문자열을 자동으로 지정합니다.

3.3.4 사용자 비밀번호 변경

사용자의 비밀번호를 변경하려면 U_Password 변수를 이용합니다.

```
TOOL MODIFY ACCOUNT admin@merakdemo.com U_Password "newpassword"
```

3.3.5 사용자 삭제

```
TOOL DELETE ACCOUNT admin@merakdemo.com
```

3.3.6 도메인에 속한 모든 메일 계정 정보 보기

하나가 아닌 여러 사용자의 정보를 한 번에 처리할 수 있도록 **와일드카드(*)** 문자를 지원합니다. 즉, *@domain.co.kr은 domain.co.kr에 속한 모든 메일 주소를 의미합니다.

아래 예제는 도메인에 속한 모든 사용자의 메일주소, 이름을 화면에 표시합니다.

```
TOOL DISPAY ACCOUNT \*@merakdemo.com U_Name
```

3.3.7 도메인에 속한 모든 메일 계정의 형식 변경하기

사용자 계정은 POP3, IMAP/POP3, IMAP 형식으로 지정할 수 있습니다. 다만 형식에 해당하는 숫자 상수 값을 입력해야 하므로 apiconst.pas 파일을 참조하는 것이 좋습니다.

- U_AccountType: 0 - POP3 / 1 - IMAP & POP3 / 2 - IMAP

아래 예제는 도메인에 속한 사용자의 형식을 POP3에서 IMAP으로 변경합니다.

```
TOOL MODIFY ACCOUNT \*@merakdemo.com U_AccountType "2"
```

3.4 로컬 서버 – 사용자 계정

로컬 서버는 메라크 메일 서버가 설치된 컴퓨터에 PHP와 같은 스크립트 언어를 함께 사용하는 경우를 의미합니다. 외부 서버는 메라크 메일 서버가 설치된 컴퓨터가 아닌 다른 서버 컴퓨터에서 제어하는 경우를 의미합니다.

로컬 서버에서 사용자 계정을 추가, 삭제 그리고 비밀번호 변경과 같은 작업은 API를 이용하거나 SQL 쿼리 문장을 이용할 수 있습니다. 일관성을 가지기 위해 사용자 계정을 API로 추가하는 PHP 스크립트 예제를 소개합니다. 또한, 입력한 값이 올바른지 확인하는 과정 등은 생략합니다.

3.4.1 사용자 계정 추가

사용자 계정을 추가하기 위해 필요한 변수는 다음과 같습니다. 그 외 옵션은 apiconst.pas 파일을 참고하십시오.

- U_Name - 계정 이름.
- U-Mailbox - 사용자 이름(사용자 ID)
- U_EmailAlias - 문자열, 별명. 첫 번째 입력 값은 U-Mailbox와 동일해야 합니다. 여러 개의 별명을 입력하는 경우에는 세미콜론 (;)으로 구분합니다.
- U_Password - 문자열, 사용자 비밀번호.

아래는 메라크 메일서버 관리도구에서 사용자 계정을 추가하는 화면입니다.

The screenshot shows a web-based administration interface for Merak Mail Server. The user is logged in as 'admin@merakdemo.com'. The interface has a navigation bar with tabs for '사용자' (Users), '그룹' (Groups), '메일박스' (Mailboxes), '제한' (Limits), '옵션' (Options), '자동 응답' (Autoresponders), and '규칙' (Rules). The '사용자' tab is active, and the 'Add User' form is displayed. The form contains the following fields and values:

- 별명 (@앞): admin:admin1:admin2
- 내선번호: (empty)
- 사용자 ID: admin
- 전체 이름: 관리자
- 비밀번호: ***** (with a '확인' button)
- 인증 모드: 표준 (with a dropdown arrow)
- 계정 형태: IMAP & POP3 (with a dropdown arrow)
- 권한: 시스템 관리자 (with a dropdown arrow and '도메인 권한..' link)
- 전달 주소: (empty)

사용자 계정을 추가하는 PHP 소스는 다음과 같습니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->New("user@merakdemo.com"); // id@domain.co.kr 즉, 메일주소 형태
$account->SetProperty("u_name", "사용자");
$account->SetProperty("u_emailalias", "user;user1;user2");
$account->SetProperty("u_mailbox", "user");
$account->SetProperty("u_password", "userpass");
$account->Save();

echo $account->LastError; // 0이면 성공, 그 외의 값이면 오류임.
?>
```

알림: `LastError` 변수 값에는 직전에 실행한 프로시저(함수)의 결과값을 가지고 있습니다. 반환하는 값의 유형은 다음과 같습니다.

- 0 - 실행 성공
- -1 - 실패.
- -2 - 라이선스 오류
- -3 - 매개변수 오류
- -4 - `Init()` 함수 내의 경로 오류
- -5 - 환경 설정 오류

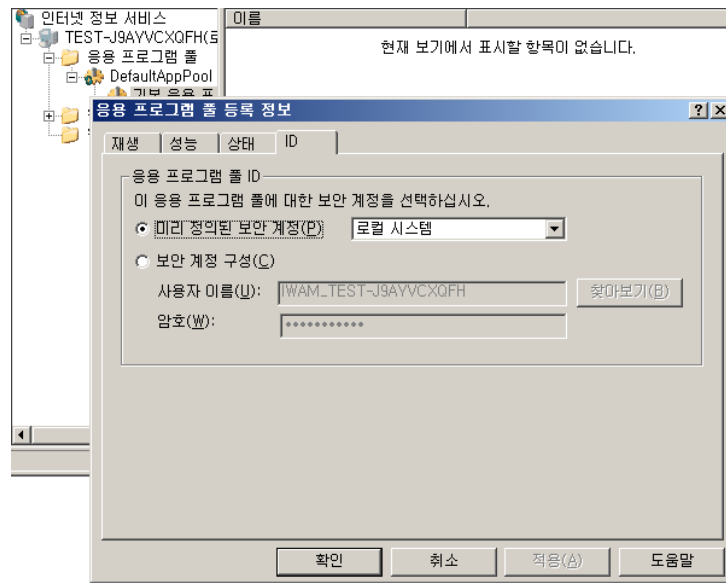
Windows 2003 IIS 환경에서는 IIS 서비스가 기본적으로 네트워크 서비스로 동작하도록 설정되어 있어 다음과 같은 오류가 발생합니다.

```
IceWarpServer.AccountObject 오류 '8000ffff'
```

```
Property u_name not found
```

```
/user.asp, 줄 13
```

다음과 같이 IIS(인터넷 정보 서비스)의 응용 프로그램 풀에서 ID를 '로컬 시스템'으로 변경하고 IIS 서비스를 재시작합니다.



ASP 소스는 다음과 같습니다.

```
<%
  Dim api
  Dim api_Account
  Dim api_Domain

  Set api = Server.CreateObject("IceWarpServer.APIObject")
  Set api_Account = Server.CreateObject("IceWarpServer.AccountObject")
  Set api_Domain = Server.CreateObject("IceWarpServer.DomainObject")

  api.Init("c:\Program Files\IceWarp\") // 반드시 포함되어야 함. 메라크 설치 경로

  api_Account.new("user@icewarp.com")
  api_Account.SetProperty "u_name", "FullName"
  api_Account.SetProperty "u_password", "UserPassword"
  api_Account.SetProperty "u_mailbox", "user"
  api_Account.Save()
  Set api = Nothing
  Set api_Account = Nothing
  Set api_Domain = Nothing

%>
```

3.4.2 사용자 비밀번호 변경

사용자의 비밀번호를 변경하는 방법은 API를 이용하거나 SQL 쿼리 문장을 이용할 수 있습니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->Open("admin@merakdemo2.com"); // id@domain.co.kr 메일 주소 입력.
$account->SetProperty("u_password", "adminpass"); // 새로운 비밀번호 입력.
$account->Save();

echo $account->LastError; // 0이면 성공, 그 외의 값이면 오류임
?>
```

```
UPDATE [MerakDB].*[Users]
    SET [U_Password] = 'NewPassword'
WHERE U_Mailbox = 'admin' AND U_Domain = 'merakdemo2.com'
```

3.4.3 사용자 계정 삭제

사용자 계정을 삭제하는 방법은 비밀번호를 바꾸는 소스와 거의 유사합니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->Open("user@merakdemo2.com"); // id@domain.co.kr 메일 주소 입력.
$account->Delete();
```

```
echo $account->LastError; // 0이면 성공, 그 외의 값이면 오류임
?>
```

3.5 로컬 서버 – 외부 계정

외부 계정을 추가하기 위해서는 RemoteAccount 객체와 ScheduleObject 객체를 이용합니다. ScheduleObject 객체를 사용하는 이유는 지정한 시간 간격마다 외부 메일 서버에 접속하여 메일을 가져오기 때문입니다.

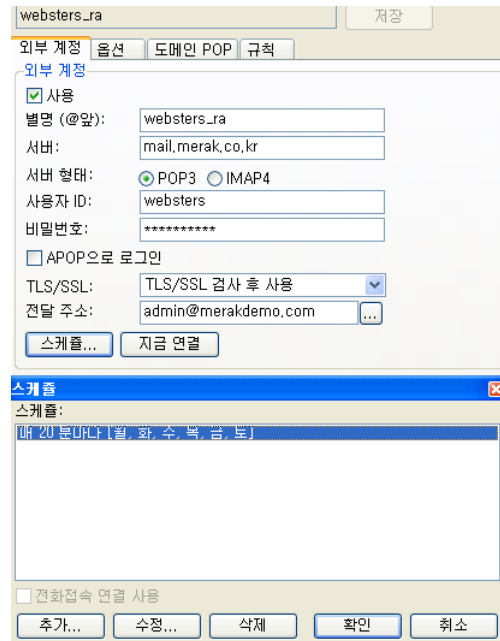
외부 계정은 데이터베이스에 저장되지 않고 메라크 자체 데이터 영역에 저장됩니다. 따라서, 사용자들이 외부 계정을 등록하거나 조회하기 위한 별도의 테이블을 추가하여 개발하는 것이 좋습니다.

3.5.1 외부 계정 추가

외부 계정을 추가하기 위해 필요한 변수는 다음과 같습니다. 그 외 옵션은 apiconst.pas 파일을 참고하십시오.

- RA_Name – 외부 계정 이름
- RA_Server – 외부 메일 서버 주소
- RA_UserName – 사용자 ID
- RA_Password – 사용자 비밀번호
- RA_ForwardTo – 메일을 전달받을 주소
- RA_DomainString – 외부 계정이 속한 도메인. 어느 도메인에 외부 계정을 추가할지 지정하는 것으로 보통 전달받을 주소의 도메인 부분으로 한정하는 것이 좋습니다.
- S_every – ScheduleObject에 속한 변수 값으로 지정한 기한마다 메일 서버에 접속하여 메일을 가져 옵니다. 초 단위입니다.

아래는 메라크 메일서버 관리도구에서 외부 계정을 추가하는 화면입니다.



사용자 계정을 추가하는 PHP 소스는 다음과 같습니다.

```
<?php
$ra = new COM("IceWarpServer.RemoteAccountObject");
$schedule = new COM("IceWarpServer.ScheduleObject");

$ra->New();
$ra->SetProperty("RA_DomainString", "merakdemo.com"); // 전달받는 사용자의 도메인
이름 부분.
$ra->SetProperty("RA_Name", "websters_ra2"); // 외부 계정 이름
$ra->SetProperty("RA_Server", "remote.mailserver.co.kr"); // 접속 정보
$ra->SetProperty("RA_UserName", "loginin"); // 사용자 ID
$ra->SetProperty("RA_Password", "loginpass"); // 사용자 비밀번호
$ra->SetProperty("RA_ForwardTo", "user@merakdemo.com"); // 전달받을 주소

$schedule = $ra->GetSchedule('ra_schedule');
$schedule->add; // 스케줄 추가

$schedule->SetProperty("s_weekdays_su", true);
$schedule->SetProperty("s_weekdays_mo", true);
```

```

$schedule->SetProperty("s_weekdays_tu", true);
$schedule->SetProperty("s_weekdays_we", true);
$schedule->SetProperty("s_weekdays_th", true);
$schedule->SetProperty("s_weekdays_fr", true);
$schedule->SetProperty("s_weekdays_sa", true);

$schedule->SetProperty("s_scheduletype", 0); // 매 분마다 접속
$schedule->SetProperty("s_every", 1200); // 20분.
$schedule->SetProperty("s_wholeday", true);

$ra->SetSchedule('ra_schedule', $schedule);
$ra->save();

echo $ra->LastError; // 0이면 성공, 그 외의 값이면 오류임
?>

```

3.5.2 외부 계정 설정 변경

사용자 계정은 Open() 함수 내에 메일 주소를 매개변수로 입력하여 사용자 계정에 속한 정보를 변경하는 방식입니다. 하지만, 외부 계정은 Index 값을 내부적으로 관리하기 때문에 다음과 같은 방식으로 설정을 변경해야 하는 외부 계정을 찾아 내어야 합니다.

Index 값은 외부 계정을 추가하는 순서대로 1씩 증가합니다. 가장 처음에 생성한 외부 계정의 Index 값은 0 입니다.

아래 소스를 간단히 설명하면, 먼저 메라크 메일서버에 등록된 외부 계정의 개수를 count()함수를 이용하여 모두 구합니다. 그리고, for 반복문을 이용하여 index 값을 0부터 1씩 증가하면서 찾고자 하는 외부 계정 이름을 비교하여 일치하는 경우에는 원하는 변수 값을 변경합니다.

```

<?php
$ra = new COM("IceWarpServer.RemoteAccountObject");
$schedule = new COM("IceWarpServer.ScheduleObject");

```

```

$ramaxcount = $ra->count(); // 외부 계정 전체 개수를 구함.
$rname = "ra2"; // 비밀번호를 변경할 외부 계정 이름.

for ($i = 0; $i < $ramaxcount; $i++) // index를 0부터 시작하여 전체 개수까지 반복.
{
    $ra->Open($i);

    if ( $ra->GetProperty("RA_Name") == $rname )
    {
        $ra->SetProperty("RA_Password", "newpass");
        // 중략
        // 다른 속성도 변경 가능.
        $ra-> save();
    }
}

?>

```

이러한 프로그램을 작성하는 경우에는 사용자가 웹메일 상에서 직접 등록한 사용자 계정을 보거나 설정을 변경하는데 있어 불편함을 가져올 수 있으며 또한 관리자 측면에서 어느 사용자가 외부 계정을 등록해서 사용하는지 알아내기가 어렵습니다. 따라서, 다음과 같은 테이블을 메라크 계정 데이터베이스 내에 추가하여 SQL 쿼리 문장을 이용하여 보다 쉽게 다룰 수 있도록 프로그래밍하는 과정을 추천합니다.

테이블에 포함되는 컬럼은 기본적으로 Index와 외부 계정을 생성할 때 반드시 필요한 항목 그리고, 외부 계정을 등록한 사용자의 ID입니다. 테이블의 이름은 RemoteAccounts라 칭합니다.

열 이름	데이터 형식	Null 허용
RA_Index	int	<input type="checkbox"/>
RA_DomainString	varchar(95)	<input type="checkbox"/>
RA_MailBox	varchar(127)	<input checked="" type="checkbox"/>
RA_Name	varchar(127)	<input type="checkbox"/>
RA_Server	varchar(127)	<input type="checkbox"/>
RA_UserName	varchar(127)	<input type="checkbox"/>
RA_Password	varchar(63)	<input type="checkbox"/>
RA_ForwardTo	varchar(127)	<input type="checkbox"/>
▶ RA_s_everly	int	<input checked="" type="checkbox"/>

- RA_Index - Index 값.
- RA_MailBox - 외부 계정을 등록한 사용자 계정(ID)
NULL 값을 허용하는 이유는 사용자가 등록한 경우 이외, 즉 관리자가 그룹웨어에서 직접 등록하는 경우를 식별하기 위한 것으로 반드시 필요한 부분은 아닙니다.
- RA_Name - 외부 계정의 이름(별명)으로 사용자 계정과 같이 계정과 중복될 수 없습니다. 따라서, 사용자 이름 뒤에 임의의 숫자를 붙이는 과정이 필요합니다.

주의: 앞서 설명한 테이블은 가장 최소한으로 필요한 부분만을 구성한 것입니다. 필요에 따라 컬럼을 추가하여 사용자의 편의를 추가로 제공할 수 있습니다.

3.5.3 외부 계정 삭제

외부 계정을 삭제하는 루틴 자체도 외부 계정의 설정을 변경하는 방식과 거의 유사합니다.

```
<?php
$ra = new COM("IceWarpServer.RemoteAccountObject");
$schedule = new COM("IceWarpServer.ScheduleObject");

$ramaxcount = $ra->count(); // 외부 계정 전체 개수.
$ra_name = "ra2"; // 삭제할 외부 계정 이름

for ($i = 0; $i < $ramaxcount; $i++) // index를 0부터 시작하여 전체 개수까지 반복.
{
    $ra->Open($i);

    if ( $ra->GetProperty("RA_Name") == $ra_name )
    {
        $ra->delete();
    }
}
}
```

?>

주의: 외부 계정에서 open() 함수로 알아내는 index 값과 RemoteAccounts 테이블의 RA_Index 값은 일치하지 않을 수 있습니다. 외부 계정을 조작하려면 반복문(FOR)와 조건문(IF)을 이용하여 index 값을 찾아야 합니다.

3.6 로컬 서버 – 메일링 리스트

메일링 리스트 계정은 사용자 계정과 외부 계정을 조작하는 방법을 복합적으로 사용합니다. 먼저 메일링 리스트 계정 자체는 ODBC에 저장할 수 있습니다. 그리고, 구성원(메일의 사본을 받는 사용자 집합) 목록은 텍스트 파일에 저장하거나, ODBC 쿼리 문장을 통해 지정할 수 있습니다.

로컬 서버인 경우에는 텍스트 파일을 직접 조작할 수 있지만, 외부 서버인 경우에는 이러한 방법을 구현하기가 까다롭습니다. 따라서, 메일링 리스트 계정도 또한 외부 계정과 같이 별도의 테이블을 메라크 계정 데이터베이스 내에 추가하여 SQL 쿼리 문장을 이용하여 보다 쉽게 다룰 수 있도록 프로그래밍하는 과정을 추천합니다.

3.6.1 메일링 리스트 추가

메일링 리스트를 추가하기 위해 필요한 변수는 다음과 같습니다. 그 외 변수의 값에 대한 자세한 사항은 apiconst.pas 파일을 참고하십시오.

- U_Type - 계정 유형. 메일링 리스트는 1 값을 가짐.
- M_Alias - 메일링 리스트 이름
- M_Name - 설명
- M_OwnerAddress - 소유자 메일 주소(보통 메일링 리스트를 추가한 사용자 ID)
- M_SendAllLists - 구성원 저장 방식. ODBC는 5 값을 가짐.
- M_ODBC - ODBC 연결 정보

DSN 이름;사용자 ID;비밀번호;서버 IP주소;SQL 유형; 드라이버
Ex) MerakDB;MerakAdmin;AdminPass;;2;1

주의: 서버의 IP주소는 ODBC에 저장되어 있으므로 입력하지 않습니다.

- M_SQL - SQL 쿼리 문장

아래는 메라크 메일서버 관리도구에서 메일링 리스트를 추가하는 화면입니다.

메일링 리스트를 추가하는 PHP 소스는 다음과 같습니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->new(" mailinglist@merakdemo.com");

$account->SetProperty("U_Type", 1);
$account->SetProperty("M_Alias", " mailinglist");
$account->SetProperty("M_Name", " 간단한 설명");
$account->SetProperty("M_OwnerAddress", " admin@merakdemo.com");
$account->SetProperty("M_SendAllLists", 5);
```

```

$account->SetProperty("M_ODBC", "MerakDB;merakdbuser;merakdbpass;;2;1");
$account->SetProperty("M_SQL", "select ML_Member from Members where ML_Alias like
'user");

$account->save();

?>

```

외부 계정과 마찬가지로 사용자가 직접 등록한 메일링 리스트를 관리하는 테이블을 추가하고, 메일링 리스트에 속한 구성원을 저장하기 위한 테이블이 필요합니다.

<메일링 리스트를 관리하기 위한 테이블>

열 이름	데이터 형식	Null 허용
ML_Alias	nvarchar(255)	<input type="checkbox"/>
ML_Memo	nvarchar(127)	<input checked="" type="checkbox"/>
ML_Owner	nvarchar(127)	<input type="checkbox"/>
ML_ODBC	nvarchar(255)	<input type="checkbox"/>
ML_SQL	nvarchar(255)	<input type="checkbox"/>

<구성원을 관리하기 위한 테이블>

열 이름	데이터 형식	Null 허용
ML_Alias	nvarchar(127)	<input type="checkbox"/>
ML_Owner	nvarchar(127)	<input type="checkbox"/>
ML_Member	nvarchar(127)	<input type="checkbox"/>

주의: 앞서 설명한 테이블은 가장 최소한으로 필요한 부분만을 구성한 것입니다. 필요에 따라 컬럼을 추가하여 사용자의 편의를 추가로 제공할 수 있습니다.

3.6.2 메일링 리스트 설정 변경

메일링 리스트 설정 부분에서 변경될 부분은 설명(M_User) 정도 밖에는 없습니다. 그리고 구성원 정보가 가장 많이 바뀔 수 있는 부분으로 이는 별도의 테이블로 구성되므로 관리자가 레코드를 추가, 변경, 삭제할 수 있는 인터페이스를 제공해야 합니다. 아래 소스는 소유자(m_OwnerAddress)를 변경하는 예제입니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->open("mailinglist@merakdemo.com");
$account->SetProperty("M_OwnerAddress", "newadmin@merakdemo.com");
// 중략...
// 다른 속성도 변경 가능.

$account->save();

?>
```

3.6.3 메일링 리스트 삭제

메일링 리스트를 삭제하는 소스는 다음과 같습니다.

```
<?php
$account = new COM("IceWarpServer.AccountObject");

$account->open("mailinglist@merakdemo.com");
$account->delete();

?>
```

메일링 리스트 추가, 변경, 삭제에 대한 소스는 매우 간단합니다. 하지만, 사용자의 편리성을 위해 추가한 테이블(MailingLists, Members)을 추가로 조작해야 하는 어려움이 있으므로 메일링 리스트의 기능에 대해 면밀히 검토한 후에 개발을 진행하는 것이 좋습니다.

3.7 외부 서버 - 외부 API

그룹웨어/ERP 서버가 메라크 메일서버와 함께 설치된 환경을 **로컬 서버**라고 부른다. 하지만, 최근에는 서버의 부하나 장애에 대한 대비를 위해 그룹웨어/ERP 서버와 메라크 메일서버를 따로 분리하는 경우가 많습니다.

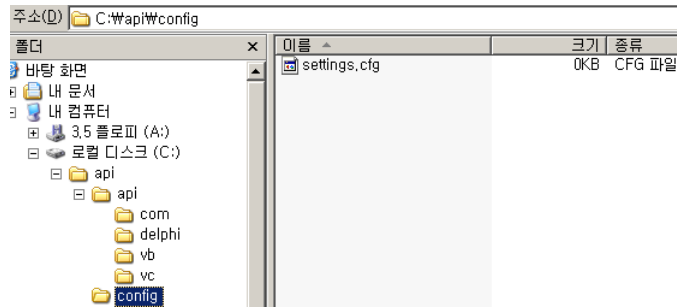
이러한 요구 사항을 만족시키기 위해 메라크 메일 서버에서는 다음과 같은 인터페이스를 제공합니다.

- 외부 API(Remote API) - 지금까지 사용자 계정 등을 생성하는 방식은 COM API를 이용하는 것이었습니다. 이러한 COM API를 다른 컴퓨터에서 제어하는 방식입니다.
- XML- 메라크 메일서버가 제공하는 COM API를 사용하지 않고 윈도우에서 기본적으로 제공하는 RPC(Remote Procedure Call) 호출을 이용하는 방식입니다. 이 방식을 사용하려면 RPC에 대한 기본적인 지식을 필요로 하므로 본 설명서에서는 설명하지 않습니다.

3.7.1 IceWarpServer API 등록

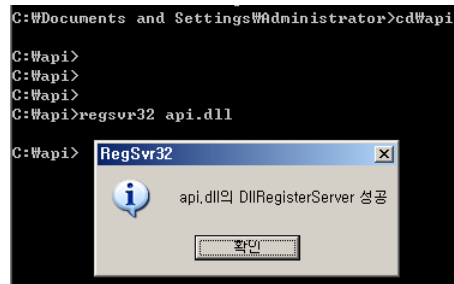
메라크 메일서버가 아닌 다른 WAS 서버에서 메라크 API를 이용하려면 먼저 DLL을 등록해야 합니다. 이 파일은 메라크 메일서버가 설치된 폴더에 위치한 api.dll 파일과 하위 폴더를 다른 WAS 서버에 복사하여 등록합니다.

- 1) 외부 WAS 서버에 관리자 권한으로 로그인합니다.
- 2) api.dll 파일 및 하위 폴더를 특정한 위치로 복사합니다.

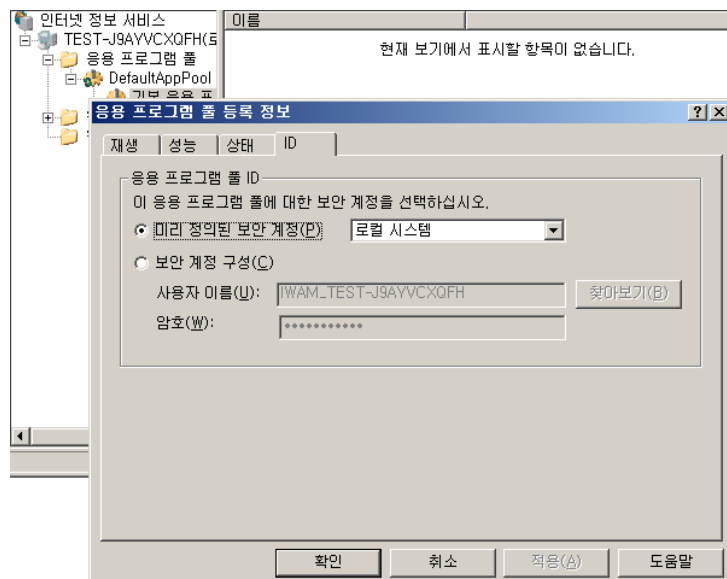


그리고, api 폴더 아래에 **config** 폴더를 생성하고, 그 안에 **settings.cfg** 파일을 생성합니다.

- 3) **Regsvr32 api.dll** 명령어를 실행하여 DLL을 레지스트리에 등록합니다.



- 4) 관리 도구 -> 인터넷 정보 서비스(IIS) 관리를 실행하고 아래 화면과 같이 **응용 프로그램 풀**에서 ID를 **로컬 시스템**으로 변경하고 IIS 서비스를 재시작합니다.



Windows 2003 ASP.NET에서는 identity impersonate 옵션을 이용하여 권한을 상승시킵니다. 즉, 메라크 메일 서버에 관리자 권한을 가진 사용자 계정을 생성한 후에 이 계정 정보를 아래와 같이 **web.config** 파일에 저장합니다.

```
<!-- 외부 API를 사용하기 위해 관리자 계정 정보를 추가함 -->
<identity impersonate="true" userName="merakadmin" password="password" />
<identity impersonate="true" />
```

ASP.NET의 보안 모델 및 Impersonation에 대한 자세한 정보는 아래 링크를 참고하십시오.

[http://msdn.microsoft.com/en-us/library/aa292118\(VS.71\).aspx](http://msdn.microsoft.com/en-us/library/aa292118(VS.71).aspx)

Windows 2000 .NET v1.1.4322 환경에서는 아래와 같이 변경합니다.

```
경로: C:\WinNT\Microsoft.NET\Framework\v1.1.4322\CONFIG
파일: machine.cfg
위치: <processModel (441 줄 근처)
변경: username="machine" -> username="system"
```

5) 메라크 메일 서버 컴퓨터에서 메라크 관리도구를 실행하여 관리자 계정을 추가합니다. 추가하는 관리자 계정의 비밀번호는 가급적 복잡하게 정하는 것이 좋습니다.

아래 화면에서 **권한을 시스템 관리자로 변경**합니다.

3.7.2 TokenObject

TokenObject는 RPC를 이용하여 API를 원격에서 실행할 수 있도록 도와주는 클래스입니다. 다른 클래스를 사용하기 전에 먼저 TokenObject 클래스를 초기화하고 URL을 연결합니다.

URL은 “*administrator:password@서버 IP 주소:포트번호*” 형식입니다. Web/Control 서비스 포트 번호가 HTTP(TCP 80)인 경우에는 생략해도 무방합니다.

아래는 외부에 위치한 메라크 메일 서버의 버전 번호를 확인하는 PHP 소스입니다.

```
<?php

$token = new COM("IceWarpServer.TokenObject");
$token->URL = "admin:password@remote.server.com:32000"; // 반드시 IP 주소를 입력.

$api = new COM("IceWarpServer.APIObject");
$api->TokenHandle = $token->TokenHandle;

$api->GetProperty("c_version");

?>
```

3.7.3 사용자 계정 추가

```
<?php

$token = new COM("IceWarpServer.TokenObject");
$token->URL = "AdminId:AdminPass@192.168.0.200:32000"; // 반드시 IP 주소를 입력.

$account = new COM("IceWarpServer.AccountObject");
$account->TokenHandle = $token->TokenHandle;
```

```
$account->New("user1111@merakdemo.com"); // id@domain.co.kr 즉, 메일주소 형태
$account->SetProperty("u_name", "사용자");
$account->SetProperty("u_emailalias", "user1111;user1112;user1113");
$account->SetProperty("u_mailbox", "user1111");
$account->SetProperty("u_password", "userpass");
$account->Save();

echo $account->LastError; // 0이면 성공, 그 외의 값이면 오류임.

?>
```

ASP 소스는 다음과 같습니다.

```
<%
Set token = Server.CreateObject("IceWarpServer.TokenObject")
token.URL = "admin:admin@192.168.0.22:32000"

Set api = Server.CreateObject("IceWarpServer.APIObject")
Set account = Server.CreateObject("IceWarpServer.AccountObject")
account.TokenHandle = token.TokenHandle

api.Init("C:\Program Files\Merak\")

account.new("user2@merakdemo.com")
account.SetProperty "U_Name", "user2"
account.SetProperty "U-Mailbox", "user2"
account.SetProperty "U-Password", "user2"

account.Save()

Set token = nothing
Set account = nothing

api.Done()
%>
```

3.7.4 외부 계정 추가

```
<?php

$token = new COM("IceWarpServer.TokenObject");
$token->URL = "AdminID:AdminPass@192.168.0.200:32000"; // 반드시 IP 주소를 입력.

$ra = new COM("IceWarpServer.RemoteAccountObject");
$ra->TokenHandle = $token->TokenHandle;

$schedule = new COM("IceWarpServer.ScheduleObject");

$ra->New();
  $ra->SetProperty("RA_DomainString", "merakdemo.com"); // 전달받는 사용자의 도메인
이름 부분.
  $ra->SetProperty("RA_Name", "websters_ra"); // 외부 계정 이름
  $ra->SetProperty("RA_Server", "remote.mailserver.co.kr"); // 접속 정보
  $ra->SetProperty("RA_UserName", "loginid");
  $ra->SetProperty("RA_Password", "loginpass");
  $ra->SetProperty("RA_ForwardTo", "admin@merakdemo.com"); // 전달받을 주소

$schedule = $ra->GetSchedule('ra_schedule');
$schedule->add;           // 스케줄 추가

$schedule->SetProperty("s_weekdays_su", false);
$schedule->SetProperty("s_weekdays_mo", true);
$schedule->SetProperty("s_weekdays_tu", true);
$schedule->SetProperty("s_weekdays_we", true);
$schedule->SetProperty("s_weekdays_th", true);
$schedule->SetProperty("s_weekdays_fr", true);
$schedule->SetProperty("s_weekdays_sa", true);

$schedule->SetProperty("s_scheduledtype", 0); // 매 분마다 접속
```

```
$schedule->SetProperty("s_every", 1200); // 20분.  
$schedule->SetProperty("s_wholeday", true);  
  
$ra->SetSchedule('ra_schedule', $schedule);  
  
$ra-> save();  
echo $ra->LastError;  
  
?>
```

3.8 IceWarpCOM – 메일 발송 전용 API

지금까지 살펴 본 IceWarpServer API는 메라크 메일 서버를 관리하는 방식에 대한 부분이었습니다. 메일 서버의 역할에 맞게 메일을 발송하는 SMTP API를 소개합니다.

메라크 메일서버에서는 외부로 메일을 손쉽게 발송할 수 있도록 IceWarpCOM API를 제공합니다. 메라크 메일서버를 설치하는 도중에 레지스트리에 자동으로 등록됩니다. 하지만 다른 WAS 서버에서 사용하려면 레지스트리에 등록하는 과정이 필요합니다.

제 4 장 IceWarpCOM API – 대량 발송용

4.1 개요

지금까지 살펴 본 IceWarpServer API는 메라크 메일 서버를 관리하는 방식에 대한 부분이었습니다. 메일 서버의 역할에 맞게 메일을 발송하는 SMTP 발송 전용 API를 소개합니다.

IceWarpCOM API는 메라크 메일서버를 구입한 고객에게 무상으로 제공되는 SMTP 컴포넌트입니다. 메일을 외부로 보내는 기능 이외에도 메라크 메신저에서도 메시지를 발송할 수 있습니다.

4.2 IceWarpCOM API 등록

메라크 메일서버가 아닌 다른 WAS 서버에서 메라크 API를 이용하려면 먼저 DLL을 등록해야 합니다. 이 파일은 메라크 메일서버가 설치된 폴더에 위치한 icewarpcom.dll으로 동일한 버전의 파일을 다른 WAS 서버에 복사하여 등록합니다.

- 1) 외부 WAS 서버에 관리자 권한으로 로그인합니다.
- 2) icewarpcom.dll 파일을 WWinNT\System32(Windows NT/2000) 또는 WWindows\System32(윈도우 2003/2008)으로 복사합니다.
- 3) **Regsvr32 icewarpcom.dll** 명령어를 실행하여 DLL을 레지스트리에 등록합니다.

IceWarpCOM API에 대한 자세한 내용은 <메라크 설치 폴더> / api 폴더에 위치한 icewarpcom.txt 파일을 참고하십시오.

주의: 메라크 메신저 사용자에게 메시지를 보내기 위해서는 동일한 서버 내에서 전송해야 합니다. 즉, 외부에 있는 WAS 서버에서는 메신저로 메시지를 보낼 수 없습니다.

알림: 메라크 메신저를 사용하기 위해서는 메신저 라이선스를 별도로 구매해야 합니다.

4.3 IceWarpCOM.Mailer

IceWarpCOM.Mailer 클래스는 메일 발송에 관련되어 있습니다. 발신자의 메일 주소가 메라크 메일 서버에 등록되어 있어야 한다는 점을 주의하십시오.

클래스의 구조는 다음과 같습니다. 자주 사용되는 변수와 함수에는 설명을 추가하였습니다.

```
IceWarpCOM.Mailer ['{AFE68541-8496-11D7-BE4E-00055DDED8D2}']
property RemoteHost // 메일 발송 서버의 주소 또는 호스트 이름.
Property FromAddress // 발송자 메일 주소
property FromName // 발송자 이름
property BodyText // 본문
property Subject: // 제목
property Recipients: WideString dispid 206;
property MailFrom: WideString dispid 207;
    property ContentTransferEncoding: WideString dispid 208;
    property Charset: WideString dispid 209;
    property IsHTML: WordBool dispid 210;
    property Result: Integer readonly dispid 211;
    property Response: WideString readonly dispid 212;
    procedure Reset; dispid 213;
    property Helo // SMTP HELO 명령어 값
procedure AddRecipient(const Email: WideString; const Name: WideString) //받는사람 추가
procedure AddCc(const Email: WideString; const Name: WideString) // 참조 추가
procedure AddBcc(const Email: WideString; const Name: WideString) // 숨은참조 추가
function SendMail // 메일 발송 함수
    procedure AddCustomHeader(const HeaderValue: WideString); dispid 219;
    procedure AddAddress(const Email: WideString; const Name: WideString); dispid 220;
    procedure AddReplyTo(const Email: WideString; const Name: WideString); dispid 221;
procedure AddAttachment(const Attachment: WideString // 첨부 추가
```

```
function EncodeHeader(const Header: WideString): WideString; dispid 223;
procedure AddEmbeddedImage(const FileName: WideString; const CID: WideString);
procedure AppendBodyFromFile(const FileName: WideString); dispid 225;
procedure ResetRecipients; dispid 226;
property UserName: WideString dispid 227;
property Password: WideString dispid 228;
end;
```

메일을 발송하는 PHP 소스는 다음과 같습니다.

```
<?PHP
$com = new COM("IceWarpCOM.Mailer");

$com->RemoteHost = "localhost";
$com->Helo = "localhost";
$com->FromName = "Mr.Employee";
$com->FromAddress = "admin@localhost";
$com->AddRecipient("recipient@domain.com", "Mr.Boss");
$com->Subject = "Hello";
$com->BodyText = "Body";
$com->AddAttachment("c:Wattach.zip");

if (!$com->SendMail()) {
echo $com->Response; // 오류 처리
}
?>
```

4.4 IceWarpCOM.IMMessage

IceWarpCOM.IMMessage 클래스를 이용하여 메라크 사용자끼리 메시지를 주고 받을 수 있습니다. 발신자와 수신자 모두 메라크 메일 서버에 등록되어 있어야 하며 메라크 그룹웨어 라이선스도 보유해야 합니다.

```
IceWarpCOM.IMMessage
property MessageBody // 메시지 본문
property MessageFrom // 발송자
property MessageSubject // 메시지 제목
property MessageTo // 수신자
property MessageType
property MessageTag
procedure Reset
function SendMessage // 메시지 발송
procedure AddAttribute(const Attribute: WideString; const Value: WideString)
procedure AddTag(const Tag: WideString)
function IsOnline(const Email: WideString)
end;
```

메시지를 발송하는 PHP 소스는 다음과 같습니다.

```
<?
$com = new COM("IceWarpCOM.IMMessage");

$com->MessageFrom = "sender@merakdemo.com";
$com->MessageTo = "receiver@merakdemo.com";
$com->MessageSubject = "Subject Hello";
$com->MessageBody = "Message Body Hello";

if (!$com->SendMessage()) echo "Error";

?>
```

제 5 장 DCOM 등록

5.1 개요

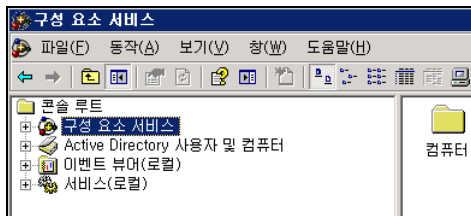
메라크 메일서버에서는 C, C#, ASP, PHP 등의 다양한 언어를 사용하여 API를 제어함으로써 메라크 관리도구의 항목들을 관리할 수 있습니다.

API(api.dll)는 메라크 메일서버가 설치되는 동안에 자동으로 DLL이 등록됩니다. 하지만, 일부 시스템(특히 윈도우 2000, 2003 서버)에서는 권한 등의 문제로 인하여 API가 제대로 수행되지 못하는 현상이 나타나기도 합니다.

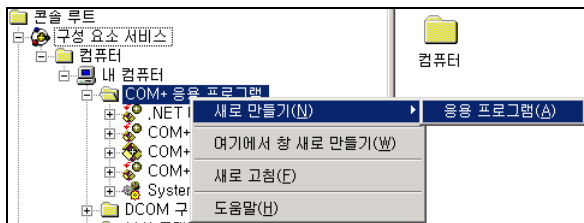
이러한 경우에는 **윈도우 구성요소 서비스**에서 DCOM을 직접 등록하여 줌으로써 문제를 해결할 수 있습니다.

5.2 DCOM 등록

윈도우 관리 도구 -> 구성요소 서비스를 실행합니다.

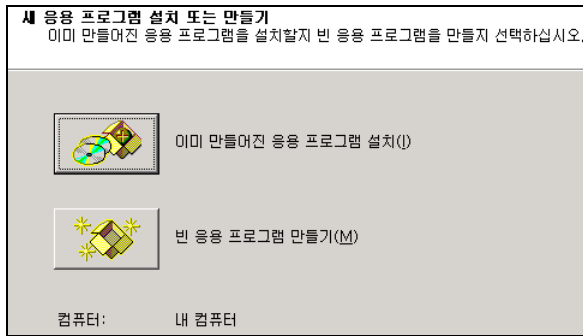


구성 요소 서비스 -> 컴퓨터 -> 내 컴퓨터 -> COM+ 응용 프로그램을 확장합니다. 그리고 COM+ 응용 프로그램을 마우스 오른쪽 버튼으로 클릭하고, 새로 만들기 -> 응용 프로그램을 선택합니다.

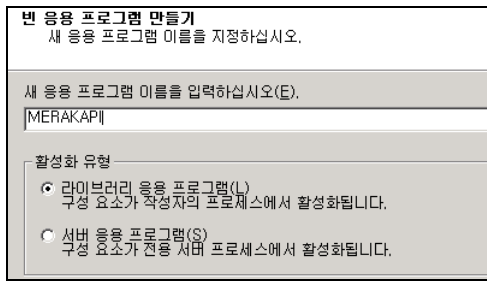


COM+ 응용 프로그램 설치 마법사가 실행됩니다. 다음 버튼을 눌러 진행합니다.

새 응용 프로그램 설치 또는 만들기 단계에서 빈 응용 프로그램 만들기를 선택합니다.

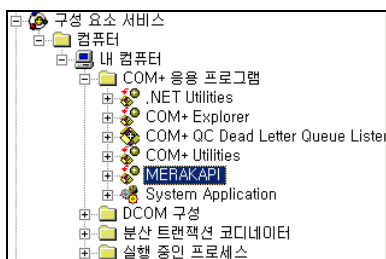


빈 응용 프로그램 만들기 단계에서 응용 프로그램의 이름을 입력하고, 라이브러리 응용 프로그램을 선택합니다. 그리고 다음 버튼을 클릭합니다.

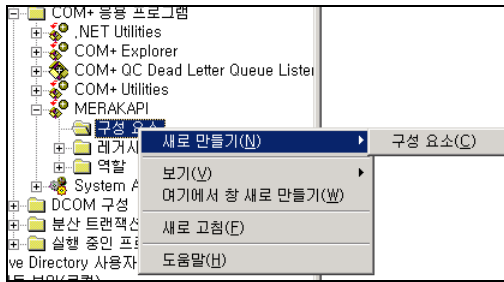


마침 버튼을 클릭하여 마법사를 종료합니다.

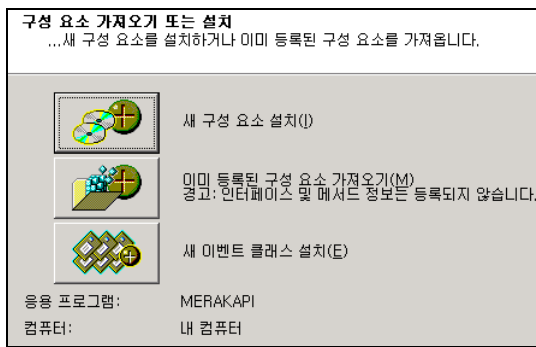
이제 아래와 같이 COM+ 응용 프로그램의 하위 폴더에 메라크에 대한 COM+가 등록되었습니다.



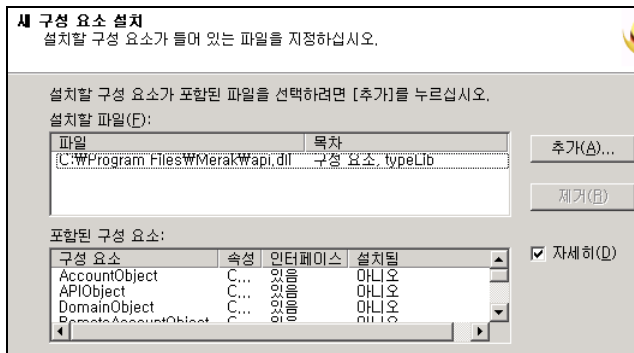
이제 COM+에 실제 DLL 파일을 등록하는 단계입니다. 위의 그림에서 MERAKAPI를 확장합니다. 아래 그림과 같이 구성 요소를 마우스 오른쪽 버튼으로 클릭하고 새로 만들기 -> 구성 요소를 선택합니다.



COM+ 구성 요소 설치 마법사가 실행됩니다. 다음 버튼을 클릭합니다.
 구성 요소 가져오기 또는 설치 단계에서 새 구성 요소 설치를 클릭합니다.

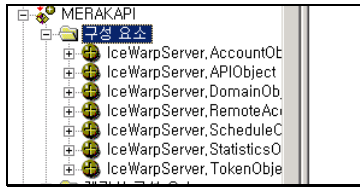


메라크 메일서버가 설치된 폴더에 있는 **api.dll** 파일을 찾아 선택합니다. 아래와 같이 새 구성 요소 설치 단계에서 API의 객체들을 찾아 볼 수 있습니다. 다음 버튼을 클릭합니다.



마침 버튼을 클릭합니다.

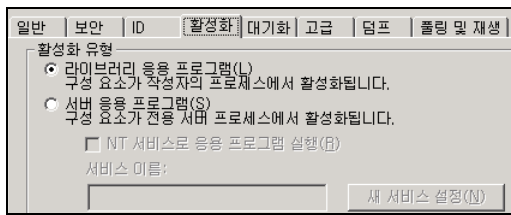
아래와 같이 등록된 객체를 볼 수 있습니다.



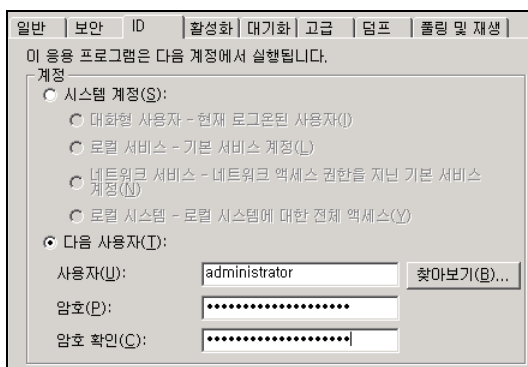
5.3 AD 및 문제 해결

지금까지 설정한 사항으로도 권한 문제가 해결되지 않는 경우가 있습니다. 또한 AD 환경 하에서 이러한 문제점이 해결되지 않는 경우가 다반사입니다. 이러한 경우에는 API가 동작하는 권한(Credential)을 관리자로 강제로 지정하여 변경하여 문제를 해결합니다.

등록한 MERAKAPI 구성 요소를 마우스 오른쪽 버튼으로 클릭하고 속성을 선택합니다. 활성화 탭을 클릭하고 서버 응용 프로그램 버튼을 선택합니다.



그리고 ID 탭을 클릭합니다. 다음 사용자를 선택하고 관리자 권한을 가지는 사용자 ID와 비밀번호를 입력합니다. 그리고 적용 -> 확인 버튼을 클릭합니다.



제 6 장 규칙(B&W 리스트 필터) 관리

6.1 개요

메라크 메일 서버에서는 다양한 필터를 제공합니다. 이러한 필터를 이용하여 관리자는 송수신하는 메일을 특정한 조건에 따라 처리할 수 있으며, 사용자들은 수신하는 메일을 조건에 맞게 처리할 수 있습니다.

필터에 대한 자세한 사항은 **관리자 설명서의 제 6장. 필터** 편을 참고하십시오.

참고: 메라크 메일 서버 9 버전에서는 규칙(rule)으로 부르며, 그 아래 버전에서는 B&W 리스트 필터(Black&White List Filter)라고 칭합니다. 여기에서는 규칙으로 통칭하여 언급합니다.

규칙은 수신하는 메일에 대해서만 적용할 수 있습니다. 그리고, 사용자 수준, 도메인 수준, 시스템 수준으로 모두 3가지로 나뉘서 설정할 수 있습니다. 규칙은 우선순위가 있으며 이는 사용자 > 도메인 > 시스템 순입니다.

도메인 규칙은 기본적으로 사용할 수 없도록 설정되어 있습니다. 만약 사용하고자 하는 경우에는 **메라크 관리도구 -> 도메인과 계정 노드 -> 전역 설정 노드 -> 도메인** 탭에서 제한한 항목을 모두 켜 주고 오른쪽 아래 **적용** 버튼을 클릭합니다.

도메인 규칙과 시스템 규칙은 당연히 메일 서버 관리자만이 설정할 수 있습니다. 다만, 도메인 관리자를 추가로 지정하여 도메인 규칙을 별도로 관리하도록 구성할 수 있지만 가급적 사용하지 않는 것이 좋습니다.

본 단원에서는 그룹웨어에서 사용자가 지정하는 블랙리스트, 화이트리스트를 메라크 메일 서버의 사용자 수준의 규칙에 어떠한 방식으로 통합하여 적용할 수 있는지에 대한 기술적인 자료를 제공합니다.

알림: 적용할 수 있는 버전은 메라크 7.24버전부터 9.4(현재 최신)까지이며, 각 버전 별로는 개발하는 과정에서 약간씩 차이점이 존재합니다.

6.2 규칙에 관련된 API

메라크 메일 서버는 API를 통해 프로그래밍할 수 있도록 설계되어 있습니다. 이 부분에 대한 자세한 사항은 제 1장. 개요 부분을 참고하십시오.

먼저 메라크 버전 별로 객체의 차이점은 아래와 같습니다.

- 메라크 9.0 버전 이상: **IceWarpServer**
- 메라크 8.9 버전 이하: **MerakCOM**

그리고, 규칙을 사용하기 위해 필요한 메소드(함수)는 아래와 같습니다. 메소드는 메라크 전 버전에 걸쳐 동일한 이름과 값 형식을 가집니다.

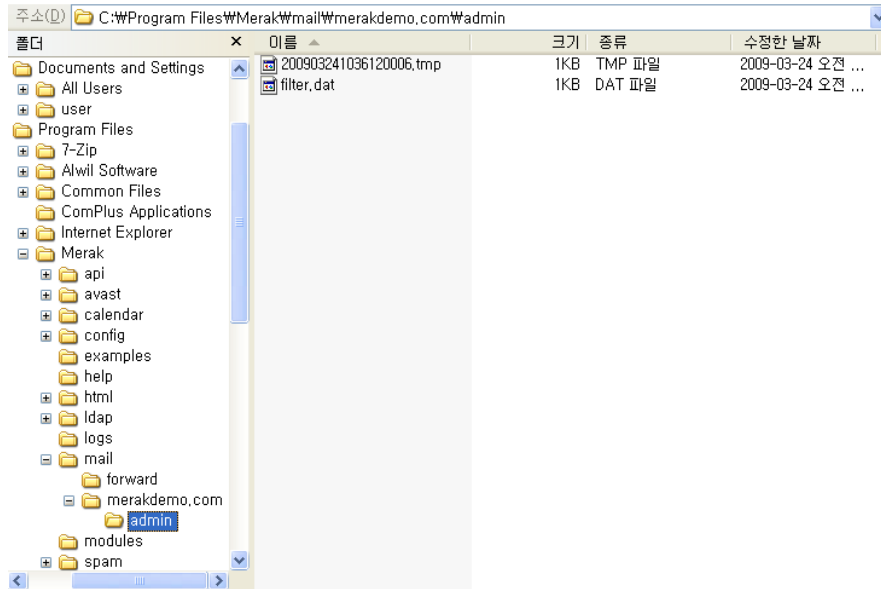
- **U_BlackWhiteFilter** = Bool // 사용자가 규칙을 사용할지 여부 설정
- **C_MailPath** = String // 메일 데이터 저장 폴더 위치

6.3 규칙 파일의 저장 위치

규칙은 파일로 저장됩니다. 저장 위치는 앞서 언급한 메일 데이터 저장 위치 아래 도메인 이름 아래 사용자 ID 폴더 아래 **filter.dat** 파일입니다.

Ex) C:\Program Files\Wmerak\Wmail\Wmerakdemo.com\Wadmin\Wfilter.dat

(주: 진하게 표시한 부분이 C_MailPath 값입니다.)



6.4 규칙 파일 형식

Filter.dat 파일은 이진(binary)이 아닌 텍스트 형식입니다. 따라서, FSO(File System Object)등을 통해 손쉽게 처리할 수 있습니다.

아래 표는 메라크 메일서버 7 버전에서 사용되는 규칙 파일의 형식입니다.

```
~Subject: 제목-거부
1:~Subject: 제목-허용
2:~Subject: 제목-삭제
~From: 주소-거부
1:~From: 주소-허용
2:~From: 주소-삭제
```

형식은 처리 방식:~헤더 조건: 문자열 순서입니다.

처리 방식은 아래와 같습니다.

- 없음 - 거부(reject)

- 1 - 허용(accept)
- 2 - 삭제(delete)

메라크 메일서버 8 버전에서는 다음과 같습니다.

```
0:~Subject: 제목-거부
1:~Subject: 제목-허용
2:~Subject: 제목-삭제
0:~From: 주소-거부
1:~From: 주소-허용
2:~From: 주소-삭제
```

메라크 메일서버 9 버전에서는 다음과 같습니다. 형식은 **처리 방식:H~헤더 조건: 문자열** 순서입니다.

```
0:H~Subject: 제목-거부
1:H~Subject: 제목-허용
2:H~Subject: 제목-삭제
0:H~From: 주소-거부
1:H~From: 주소-허용
2:H~From: 주소-삭제
```

3가지 버전에서 모두 형식이 다릅니다. 하지만, 메라크 메일서버 8 버전의 형식을 사용해도 7 버전과 9 버전 모두에서 사용이 가능합니다. 따라서, 8 버전의 형식을 기준으로 개발합니다.

참고로, 발송자(From:)에서 사용할 수 있는 정책은 사용자 메일 주소 차단/허용 또는 도메인 이름 차단/허용입니다.

예제) 0:~From: a@b.com a@b.com 메일 주소에서 오는 메일 차단
0:~From: @b.com b.com 도메인에서 오는 메일 모두 차단

참고: 규칙에서는 문자열 값을 세미콜론(;)으로 구분하여 여러 개를 동시에 입력할

수 있습니다.

Merak

Mail Server

Developer Guide

[메라크 메일서버 개발자용 설명서]

SoftMail[®]

(주)소프트메일

연락처: (주)소프트메일

홈페이지: <http://www.softmail.co.kr>

전화: 02-598-9220

팩스: 02-3486-9331

주소: 서울 서초구 서초동 1330-6 롯데골드로즈 1905