

# 목 차

I . 오즈 엔터프라이즈 서버 <b>API</b> .....	<b>3</b>
Class Cache .....	5
Class ConnectionPool .....	9
Class DataBind .....	16
Class Log .....	19
Class Mail .....	22
Class Module .....	29
Class Monitor .....	40
Class Service .....	44
II . 오즈 스케줄러 서버 <b>API</b> .....	<b>47</b>
Class Program .....	49
Class Publisher .....	54
Class Scheduler .....	58
Class TaskHolidayInfo .....	91
Class TaskHolidayGroupInfo .....	94
III . 오즈 리파지토리 <b>API</b> .....	<b>97</b>
Interface Repository .....	98
Class RepositoryEX .....	142
오즈 리파지토리 구현 .....	167

<b>IV. User Defined Log .....</b>	<b>171</b>
UDL 인터페이스 .....	172
UDL 구현 .....	187
<b>V. User Defined Resource .....</b>	<b>189</b>
UDR 인터페이스 .....	190
UDR 구현 .....	193
<b>Appendix 1. 사용자 지정 파라미터 암호화 .....</b>	<b>199</b>

## I . 오즈 엔터프라이즈 서버 API

- Class Cache
- Class ConnectionPool
- Class DataBind
- Class Log
- Class Mail
- Class Module
- Class Monitor
- Class Service

오즈 엔터프라이즈 서버에 관련된 각종 정보 조회와 실시간 환경 설정 변경 기능을 사용자 애플리케이션에서 직접 제어할 수 있도록 오즈 엔터프라이즈 닷넷 서버 API를 제공합니다.

다음은 오즈 엔터프라이즈 서버 API로 제공하는 주요 클래스에 대한 설명입니다.

클래스 이름	설명
<b>Cache</b>	오즈 서버 캐시 매니저와 관련된 처리를 수행합니다. 캐시 매니저 환경 설정 및 캐시 항목 삭제 등을 처리합니다.
<b>Connection Pool</b>	오즈 서버의 데이터베이스 연결 Pool에 대한 인터페이스로 해당 데이터베이스의 ADO .NET/ODBC 정보의 등록, 삭제 등의 관리 기능을 수행합니다.
<b>DataBind</b>	데이터 바인딩 모듈 관련 기능을 수행합니다.
<b>Log</b>	오즈 서버 로그 매니저와 관련된 처리를 수행하며 서버 운용시 발생하는 각종 메시지, 정보, 에러 등을 기록하기 위한 환경 설정을 변경합니다.
<b>Mail</b>	메일 관련 정보를 설정하고 메일을 전송하는 기능을 수행합니다.
<b>Module</b>	오즈 데이터 모듈 관련 기능을 수행합니다.
<b>Monitor</b>	오즈 서버의 실시간 모니터링 관련 기능을 수행합니다.
<b>Service</b>	서버 정보, 상태, 처리, 메모리 수집 등의 처리 기능을 수행합니다.

오즈 엔터프라이즈 서버 API를 사용하기 위해서는 다음과 같은 라이브러리 파일을 클래스 패스에 추가해 주어야 합니다.

라이브러리 파일명	설명
<b>OZServer.NET.dll</b>	오즈 닷넷 서버와 접속하기 위한 파일입니다.

## Class Cache

### Constructor Summary

- Cache(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)
- Cache(string url, string id, string pw, bool autoLogin, bool useUSL)

### Method Summary

- OZAttributeList GetConfiguration()
- void SetConfiguration(OZAttributeList attrs)

### Constructor Detail

---

	<i>//TCP/IP 방식</i>
	public Cache(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)
<b>ASP .NET</b>	
	<i>//HTTP 방식</i>
	public Cache(string url, string id, string pw, bool autoLogin, bool useUSL)
	<hr/>
	<i>url</i> HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
	<hr/>
	<i>ip</i> TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
	<hr/>
	<i>port</i> TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003
<b>Argument</b>	
	<i>id</i> 사용자 아이디 ex) string id = "admin";
	<hr/>
	<i>pw</i> 사용자 패스워드 ex) string pw = "admin";
	<hr/>
	<i>autoLogin</i> 자동 로그인 여부 ex) bool autoLogin = true;

---

<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;
---------------	---------------------------------------

## Method Detail

### ■ GetCacheConfiguration

<b>Prototype</b>	public OZAttributeList GetCacheConfiguration() throws OZAPIException
------------------	--

<b>Definition</b>	오즈 서버의 캐시 설정을 가져옵니다. 조회 내용은 캐시 키 사용 여부, 캐시 저장 경로, 메모리 캐시 타임 아웃, 디스크 캐시 타임 아웃, 프리 메모리 퍼센트입니다.
-------------------	--

### ■ SetCacheConfiguration

<b>Prototype</b>	public void SetCacheConfiguration(OZAttributeList attrs) throws OZAPIException
------------------	--

<b>Definition</b>	캐시 설정을 변경합니다. 캐시 설정 변경은 해당 변경된 값을 캐시 키 사용 여부, 메모리 캐시 타임 아웃, 디스크 캐시 타임 아웃, 프리 메모리 퍼센트, 캐시 저장 경로명 등의 요소로 변경합니다
-------------------	--

<b>Argument</b>	<i>attrs</i> 캐시 설정 속성 값
-----------------	-------------------------

## 관련 Class

### ■ OZAPIException(oz.framework.cp.OZAPIException)

API를 사용할 때 생성되는 Exception입니다. 모든 API에서 OZAPIException을 발생시킵니다.

- 속성

- Message

<b>Prototype</b>	public string Message
------------------	-----------------------

<b>Definition</b>	에러 내용
-------------------	-------

### ■ OZAttributeList(oz.util.OZAttributeList.cs)

GetCacheConfiguration(), SetCacheConfiguration()의 설정 값을 반환하거나 설정하는데 사용되는 클래스입니다.

- 속성

▪ This

**Prototype** This[string key] {get;set;}

**Definition** key에 해당하는 속성 값

- Key

Property를 설정하거나 얻어오는 경우 사용되는 key는 다음 표와 같습니다.

Key	Value	설명
<b>Active</b>	"true" "false"	캐시키 사용 여부 ex) p["datamodule.active"] = "false";
<b>CACHE_FILE_PATH</b>	저장 경로	캐시 저장 경로(string) ex) p["CACHE_FILE_PATH"] = "%OZ_HOME%/cache";
<b>DM_CACHE_FILE_PATH</b>	저장 경로	Data Module 캐시 저장 경로(string) ex) p["DM_CACHE_FILE_PATH"] = "%OZ_HOME%/cache_dm/";
<b>memoryCacheValidTime</b>	시간	메모리 캐시 타임 아웃(단위:초) ex) p["datamodule.memoryCacheValidTime"] = "100";
<b>diskCacheValidTime</b>	시간	디스크 캐시 타임 아웃(단위:초) ex) p["datamodule.diskCacheValidTime"] = "100";
<b>FreeMemoryPercentage</b>	메모리	프리 메모리 퍼센트(%) ex) p["datamodule.freeMemoryPercentage"] = "20";

※ 참고사항 : 캐시 키 설정 방법은 "OZ Enterprise Server Administrator's Guide"의 "cachemngr.properties" 부분을 참조하시기 바랍니다.

**Sample : CacheSample.cs**

```
using System;

using oz.framework.api;

namespace sample{
/// <summary>
/// Cache
```

```
/// </summary>
public class CacheTest{
    public static void Main(){
        string url = "http://127.0.0.1/oztest/server.aspx";
        string id = "admin";
        string password = "admin";
        Cache c = new Cache(url, id, password, true, true);
        oz.util.OZAttributeList attrs = c.GetConfiguration();
        Console.WriteLine(attrs);
        c.SetConfiguration(attrs);
    }
}
```

## Class ConnectionPool

### Constructor Summary

- **ConnectionPool(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)**
- **ConnectionPool(string url, string id, string pw, bool autoLogin, bool useUSL)**

### Method Summary

- **void AddPool(ConnectionPoolInfo pool)**
- **void RemovePool(string alias)**
- **ConnectionPoolInfo[] GetPoolInfos()**
- **ConnectionPoolStatus[] GetPoolStatuses()**
- **ConnectionPoolInfo GetPoolInfo(string alias)**
- **void Save()**

### Constructor Detail

---

	<i>//TCP/IP 방식</i>	
		public ConnectionPool(string ip,int port, string id, string pw, bool autoLogin, bool useUSL)
<b>ASP .NET</b>	<i>//HTTP 방식</i>	
		public ConnectionPool(string url, string id, string pw, bool autoLogin, bool useUSL)
	<i>url</i>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
<b>Argument</b>	<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
	<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;

---

<i>id</i>	사용자 아이디 ex) string id = "admin";
<i>pw</i>	사용자 패스워드 ex) string pw = "admin";
<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ AddPool

<b>Prototype</b>	<code>public void AddPool(ConnectionPoolInfo pool) throws OZAPIException</code>
<b>Definition</b>	커넥션 풀 인포에 들어있는 정보에 맞춰서 커넥션 풀을 만들어 추가합니다. 커넥션 풀 인포에는 상태, 이름, 타입, 주소, 포트, SID, DB드라이버 종류, DB 서버이름, 사용자명, 암호, 최대접속, 초기 접속, 타임 아웃, URL, 프로퍼티, 사용자정의 드라이버가 있습니다.
<b>Argument</b>	<i>pool</i> 추가할 ConnectionPoolInfo의 내용

### ■ RemovePool

<b>Prototype</b>	<code>public void RemovePool(string alias) throws OZAPIException</code>
<b>Definition</b>	해당 앨리어스 이름의 ConnectionPool을 삭제합니다.
<b>Argument</b>	<i>alias</i> ConnectionPool 앨리어스 이름

### ■ GetPoolInfos

<b>Prototype</b>	<code>public ConnectionPoolInfo[] GetPoolInfos() throws OZAPIException</code>
<b>Definition</b>	모든 ConnectionPool에 대한 정보를 가져옵니다.

### ■ GetPoolStatuses

<b>Prototype</b>	<code>public ConnectionPoolStatus[] GetPoolStatuses() throws OZAPIException</code>
<b>Definition</b>	모든 ConnectionPool에 대한 상태 정보를 가져옵니다.

## ■ GetPoolInfo

<b>Prototype</b>	public ConnectionPoolInfo GetPoolInfo(string alias) throws OZAPIException
------------------	---

<b>Definition</b>	해당 ConnectionPool에 대한 ConnectionPoolInfo 리스트를 가져옵니다.
-------------------	--

<b>Argument</b>	<i>alias</i> ConnectionPool의 앨리어스 이름
-----------------	--------------------------------------

## ■ Save

<b>Prototype</b>	public void Save() throws OZAPIException throws OZAPIException
------------------	--

<b>Definition</b>	서버에 모든 ConnectionPool의 정보를 저장합니다.
-------------------	-----------------------------------

## 관련 Class

### ■ ConnectionPoolInfo(oz.framework.db.ConnectionPoolInfo)

데이터베이스 연결에 대한 각종 정보들을 가지고 있는 클래스입니다.

- Alias

<b>Prototype</b>	public string Alias{get;set;}
------------------	-------------------------------

<b>Definition</b>	해당 ConnectionPool 정보의 alias명
-------------------	------------------------------

- Vendor

<b>Prototype</b>	public string Vendor{get;set;}
------------------	--------------------------------

<b>Definition</b>	해당 ConnectionPool 정보의 Vendor명
-------------------	-------------------------------

- InitialConnections

<b>Prototype</b>	public int InitialConnections{get;set;}
------------------	---

<b>Definition</b>	해당 ConnectionPool 초기 커넥션 개수
-------------------	-----------------------------

- MaxConnections

<b>Prototype</b>	public int MaxConnections{get;set;}
------------------	-------------------------------------

<b>Definition</b>	해당 ConnectionPool 최대 커넥션 개수
-------------------	-----------------------------

- TimeOut

<b>Prototype</b>	public int TimeOut{get;set;}
------------------	------------------------------

<b>Definition</b>	해당 커넥션을 가져올 때 timeout 시간
-------------------	--------------------------

- InitSQLs
 

---

<b>Prototype</b>	<code>public string InitSQLs{get;set;}</code>
<b>Definition</b>	ConnectionPool에서 사용할 커넥션을 가져온 뒤 바로 실행하는 쿼리문

---
  
- CloseSQLs
 

---

<b>Prototype</b>	<code>public string CloseSQLs{get;set;}</code>
<b>Definition</b>	커넥션이 close또는 free되기 전 바로 실행하는 쿼리문

---
  
- SessionQuery
 

---

<b>Prototype</b>	<code>public string SessionQuery{get;set;}</code>
<b>Definition</b>	커넥션의 session 쿼리문

---
  
- DoConnectioncheck
 

---

<b>Prototype</b>	<code>public bool DoConnectionCheck{get;set;}</code>
<b>Definition</b>	커넥션 가져올 당시 커넥션이 유효한지 여부

---
  
- TestQuery
 

---

<b>Prototype</b>	<code>public string TestQuery{get;set;}</code>
<b>Definition</b>	커넥션이 유효한지 체크하기 위한 쿼리문

---
  
- Items
 

---

<b>Prototype</b>	<code>public IDictionary Items{get;}</code>
<b>Definition</b>	dbconfig.xml 파일의 Vendor마다 있는 Items의 key, Value

---
  
- Decode
 

---

<b>Prototype</b>	<code>public string Decode{get;set;}</code>
<b>Definition</b>	해당 ConnectionPool에 대한 디코딩셋

---
  
- Encode
 

---

<b>Prototype</b>	<code>public string Encode{get;set;}</code>
<b>Definition</b>	해당 ConnectionPool에 대한 인코딩셋

---
  
- FetchRow
 

---

<b>Prototype</b>	<code>public int FetchRow{get;set;}</code>
------------------	--

---

---

**Definition** JDBC 전송 ROW 측, 결과 셋 반환 시 한꺼번에 가져올 수 있는 행 수

---

■ **ConnectionPoolStatus(oz.framework.db.ConnectionPoolStatus)**

ConnectionPool의 상태 정보를 멤버 변수로 가지고 있는 클래스입니다.

- Status

---

**Prototype** public Status Status{get;set;}

---

현재 상태 Status  
public enum Status

**Definition** {  
OK = 1,  
DRIVER\_ERROR = -1,  
CONNECTION\_ERROR = -2  
}

---

- Statusstring

---

**Prototype** public string Statusstring{get;}

---

해당 상태 메시지

**Definition** STATUS 값이 1인 경우 "OK",  
STATUS 값이 -1인 경우 "Fail to load or register JDBC driver"  
그 외인 경우 "Fail to make initial connection"

---

- FreeConnectionCount

---

**Prototype** public int FreeConnectionCount{get;set;}

---

**Definition** 프리 상태의 커넥션 수

---

- CheckedOutConnectionCount

---

**Prototype** public int CheckedOutConnectionCount{get;set;}

---

**Definition** Checked out된 커넥션 수

---

- ErrorMessage

---

**Prototype** public string ErrorMessage{get;}

---

**Definition** ConnectionPool 생성 시 발생한 Error 메시지

---

- Error

---

**Prototype** public Exception Error{set;}

---

**Definition** ConnectionPool 생성 시 발생한 Error

---

**Sample : ConnectionPoolTest.cs**

```
using System;

using oz.framework.api;
using oz.framework.db;

namespace sample
{
    public class ConnectionPoolTest
    {
        public static void Main()
        {
            string url = "http://127.0.0.1/oz/server.aspx";
            string id = "admin";
            string password = "admin";

            ConnectionPool cp = new ConnectionPool(url, id,
password, true, true);

            string alias = "connection_pool_test";

            ConnectionPoolInfo poolInfo = new
ConnectionPoolInfo();
            poolInfo.Alias = alias;
            poolInfo.Vendor = "MSSQL";
            poolInfo.Items["serverAddress"] = "218.36.12.88";
            poolInfo.Items["portNo"] = "1433";
            poolInfo.Items["dbName"] = "QATEST";
            poolInfo.Items["user"] = "user1";
            poolInfo.Items["password"] = "user123";

            poolInfo.MaxConnections = 20;
            poolInfo.InitialConnections = 5;
            poolInfo.TimeOut = 5;
            //charater set
            //poolInfo.Encode = "ISO-8859-1";
            //poolInfo.Decode = "ks_c_5601-1987";

            cp.AddPool(poolInfo);

            ConnectionPoolInfo addedPoolInfo =
cp.GetPoolInfo(alias);
            Console.WriteLine(addedPoolInfo);

            cp.RemovePool(alias);

            ConnectionPoolInfo[] poolInfos = cp.GetPoolInfos();
            foreach(ConnectionPoolInfo cpi in poolInfos)
```

```
        {  
            Console.WriteLine(cpi);  
        }  
  
        ConnectionPoolStatus[] statuses =  
cp.GetPoolStatuses();  
        foreach(ConnectionPoolStatus status in statuses)  
        {  
            Console.WriteLine(status);  
        }  
    }  
}
```

## Class DataBind

### Constructor Summary

- **DataBind(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)**
- **DataBind(string url, string id, string pw, bool autoLogin, bool useUSL)**

### Method Summary

- **void SetConfiguration(OZAttributeList config)**
- **OZAttributeList GetConfiguration()**

### Constructor Detail

---

	<i>//TCP/IP 방식</i>
	public DataBind(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)
<b>ASP .NET</b>	<i>//HTTP 방식</i>
	public DataBind(string url, string id, string pw, bool autoLogin, bool useUSL)
	<hr/>
	<i>url</i> HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
	<hr/>
	<i>ip</i> TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
	<hr/>
<b>Argument</b>	<i>port</i> TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003
	<hr/>
	<i>id</i> 사용자 아이디 ex) string id = "admin";
	<hr/>
	<i>pw</i> 사용자 패스워드 ex) string pw = "admin";

---

<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ SetConfiguration

<b>Prototype</b>	public void SetConfiguration(OZAttributeList config) throws OZAPIException
<b>Definition</b>	DataBind 설정 즉, "databind.properties"에 설정되어 있는 값을 변경할 수 있습니다.
<b>Argument</b>	<i>config</i> DataBind 설정 속성

### ■ GetConfiguration

<b>Prototype</b>	public OZAttributeList GetConfiguration() throws OZAPIException
<b>Definition</b>	DataBind의 설정 값 즉, "databind.properties"에 설정되어 있는 값을 가져옵니다.

- Key

SetConfiguration()와 GetConfiguration()에서 사용되는 key는 다음 표와 같습니다.

Key	Value	설명
<b>ConcurrentFetchSize</b>	버퍼 크기	FetchType이 "Concurrent"일 때 클라이언트로 데이터를 전송할 때 Stream 버퍼의 크기를 설정합니다. 단위는 byte, 기본 값은 4096, 최소 값은 256입니다. ※ 주의사항 : 음수를 설정하거나 잘못된 형식으로 설정하면 기본 값으로 설정되고, 최소 값보다 작은 양수로 설정하면 최소 값으로 설정됩니다.

<b>ConcurrentFirstRow</b>	행 개수	<p>FetchType이 "Concurrent"일 때 클라이언트로 데이터를 전송할 때 맨 처음 전송되는 첫 데이터의 행 개수를 설정합니다.</p> <p>단위는 개수, 기본 값은 0입니다.</p> <p>※ 주의사항 : 0보다 작은 값을 설정한 경우에는 기본 값으로 설정됩니다.</p>
---------------------------	------	---

### Sample : DataBindTest.cs

```

using System;

using oz.util;
using oz.framework.api;

namespace sample
{
    /// <summary>
    /// DataBindTest
    /// </summary>
    public class DataBindTest
    {
        public static void Main()
        {
            string url = "http://127.0.0.1/oz/server.aspx";
            string id = "admin";
            string password = "admin";

            DataBind db = new DataBind(url, id, password, true, true);

            OZAttributeList attrs = db.GetConfiguration();
            foreach(stringDictionaryEntry attr in attrs)
            {
                Console.WriteLine(attr.Key + " : " + attr.Value);
            }

            db.SetConfiguration(attrs);
        }
    }
}

```

## Class Log

### Constructor Summary

- **Log(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)**
- **Log(string url, string id, string pw, bool autoLogin, bool useUSL)**

### Method Summary

- **string GetConfiguration()**
- **Stream DownloadLog()**
- **Stream DownloadLog(string fileName)**
- **void SetConfiguration(string config)**
- **string[] GetFileNames()**

### Constructor Detail

---

	<i>//TCP/IP 방식</i>
	public Log(string url, string id, string pw, bool autoLogin, bool useUSL)
<b>ASP .NET</b>	<i>//HTTP 방식</i>
	public Log(string url, string id, string pw, bool autoLogin, bool useUSL)
	<hr/>
	<i>url</i> HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
	<hr/>
	<i>ip</i> TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
<b>Argument</b>	<hr/>
	<i>port</i> TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003
	<hr/>
	<i>id</i> 사용자 아이디 ex) string id = "admin";

---

<i>pw</i>	사용자 패스워드 ex) string pw = "admin";
<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ GetConfiguration

**Prototype** public string GetConfiguration() throws OZAPIException

**Definition** 로그 설정을 서버로부터 가져옵니다.

### ■ DownloadLog

**Prototype** public Stream DownloadLog() throws OZAPIException

**Definition** 서버로부터 로그 파일을 다운로드합니다.

**Prototype** public Stream DownloadLog(string fileName) throws OZAPIException

**Definition** 서버로부터 다운로드한 로그 파일을 로컬에 저장합니다.

**Argument** *fileName* 저장할 파일명

### ■ SetConfiguration

**Prototype** public void SetConfiguration(string logs) throws OZAPIException

**Definition** 로그 환경 설정 값을 변경합니다.

**Argument** *logs* 변경할 로그 환경 설정 값, "key=value" 형식으로 설정  
ex) string logs="Priority=DEBUG"  
ex) string logs="CONSOLE.Layout=%r[%t]%p%c{1}%X-%m%n"

### ■ GetFileNames

**Prototype** public string[] GetFileNames() throws OZAPIException

**Definition** 생성된 로그 파일명을 가져옵니다.

## Sample : LogSample.cs

```
using System;
using System.IO;

using oz.framework.api;

namespace sample
{
    /// <summary>
    /// LogTest
    /// </summary>
    public class LogSample
    {
        public static void Main()
        {
            string url = "http://127.0.0.1/oz/server.aspx";
            string id = "admin";
            string password = "admin";

            Log log = new Log(url, id, password, true, true);

            string config = log.GetConfiguration();
            Console.WriteLine(config);

            string[] logs = log.GetFilesNames();
            foreach(string s in logs)
            {
                Console.WriteLine(s);
            }
            Stream logFile = log.DownloadLog();
        }
    }
}
```

## Class Mail

### Constructor Summary

- Mail(string ip, int port, string id, string pw, bool bAutoLogin, bool useUSL)
- Mail(string url, string id, string pw, bool bAutoLogin, bool useUSL)

### Method Summary

- void AddAlias(string aliasName, NameValueCollection configMap)
- NameValueCollection GetAliasConfig(string aliasName)
- string[] GetMailAliasNames()
- void ModifyAlias(string aliasName, string newAliasName, NameValueCollection configMap)
- void RemoveAlias(string aliasName)
- void Send(string aliasName, string from, string fromUserName, string to, string cc, string bcc, string subject, string context, bool isHTML, string localFileFullPath, string fileName)
- bool SendSync(string aliasName, string from, string fromUserName, string to, string cc, string bcc, string subject, string context, bool isHTML, string localFileFullPath, string fileName)

### Constructor Detail

---

	<code>//TCP/IP 방식</code>	
	<code>public Mail(string ip, int port, string id, string pw, bool bAutoLogin, bool useUSL)</code>	
<b>Prototype</b>	<code>//HTTP 방식</code>	
	<code>public Mail(string url, string id, string pw, bool bAutoLogin, bool useUSL)</code>	
<b>Argument</b>	<code>url</code>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";

---

<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;
<i>id</i>	사용자 아이디 ex) string id = "admin";
<i>pw</i>	사용자 패스워드 ex) string pw = "admin";
<i>bAutoLogin</i>	자동 로그인 여부 ex) bool bAutoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ AddAlias

<b>Prototype</b>	public void AddAlias(string aliasName, NameValueCollection configMap)				
<b>Definition</b>	오즈 서버에 설정한 앨리어스 이름으로 메일 설정 정보를 추가합니다.				
<b>Argument</b>	<table border="1"> <tr> <td><i>aliasName</i></td> <td>앨리어스 이름</td> </tr> <tr> <td><i>configMap</i></td> <td>메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> </table>	<i>aliasName</i>	앨리어스 이름	<i>configMap</i>	메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.
<i>aliasName</i>	앨리어스 이름				
<i>configMap</i>	메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.				

### ■ GetAliasConfig

<b>Prototype</b>	public NameValueCollection GetAliasConfig(string aliasName)
<b>Definition</b>	해당 앨리어스의 메일 설정 정보를 가져옵니다.
<b>Argument</b>	<i>aliasName</i> 앨리어스 이름

### ■ GetMailAliasNames

<b>Prototype</b>	public string[] GetMailAliasNames()
<b>Definition</b>	오즈 서버의 mail.properties에 설정된 메일 앨리어스를 모두 가져옵니다. ※ 참고사항 : 각 앨리어스의 active 여부와 관계없이 설정된 모든 앨리어스를 가져옵니다.

■ **ModifyAlias**

<b>Prototype</b>	<code>public void ModifyAlias(string aliasName, string newAliasName, NameValueCollection configMap)</code>						
<b>Definition</b>	해당 앨리어스의 앨리어스 이름 및 메일 설정 정보를 변경합니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>aliasName</i></td> <td>앨리어스 이름</td> </tr> <tr> <td><i>newAliasName</i></td> <td>변경할 앨리어스 이름</td> </tr> <tr> <td><i>configMap</i></td> <td>메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하십시오.</td> </tr> </table>	<i>aliasName</i>	앨리어스 이름	<i>newAliasName</i>	변경할 앨리어스 이름	<i>configMap</i>	메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하십시오.
<i>aliasName</i>	앨리어스 이름						
<i>newAliasName</i>	변경할 앨리어스 이름						
<i>configMap</i>	메일 설정 속성 값 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하십시오.						

■ **RemoveAlias**

<b>Prototype</b>	<code>public void RemoveAlias(string aliasName)</code>
<b>Definition</b>	해당 앨리어스의 메일 설정 정보를 삭제합니다.
<b>Argument</b>	<i>aliasName</i> 앨리어스 이름

■ **Send**

<b>Prototype</b>	<code>public void Send(string aliasName, string from, string fromUserName, string to, string cc, string bcc, string subject, string context, bool isHTML, string localFileFullPath, string fileName)</code>																		
<b>Definition</b>	해당 앨리어스에 설정된 메일 정보로 메일을 비동기 방식으로 전송합니다.																		
<b>Argument</b>	<table border="0"> <tr> <td><i>aliasName</i></td> <td>앨리어스 이름</td> </tr> <tr> <td><i>from</i></td> <td>발신자 메일 주소</td> </tr> <tr> <td><i>fromUserName</i></td> <td>발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨</td> </tr> <tr> <td><i>to</i></td> <td>수신자 메일 주소</td> </tr> <tr> <td><i>cc</i></td> <td>참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정</td> </tr> <tr> <td><i>bcc</i></td> <td>숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정</td> </tr> <tr> <td><i>subject</i></td> <td>메일 제목</td> </tr> <tr> <td><i>context</i></td> <td>메일 내용</td> </tr> <tr> <td><i>isHTML</i></td> <td>메시지 형식을 HTML로 설정할지 여부</td> </tr> </table>	<i>aliasName</i>	앨리어스 이름	<i>from</i>	발신자 메일 주소	<i>fromUserName</i>	발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨	<i>to</i>	수신자 메일 주소	<i>cc</i>	참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정	<i>bcc</i>	숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정	<i>subject</i>	메일 제목	<i>context</i>	메일 내용	<i>isHTML</i>	메시지 형식을 HTML로 설정할지 여부
<i>aliasName</i>	앨리어스 이름																		
<i>from</i>	발신자 메일 주소																		
<i>fromUserName</i>	발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨																		
<i>to</i>	수신자 메일 주소																		
<i>cc</i>	참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정																		
<i>bcc</i>	숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정																		
<i>subject</i>	메일 제목																		
<i>context</i>	메일 내용																		
<i>isHTML</i>	메시지 형식을 HTML로 설정할지 여부																		

<i>localFilePath</i>	파일을 첨부할 경우 첨부할 파일의 전체 경로 여러 값을 설정할 경우 " "를 구분자로 설정
<i>fileName</i>	파일을 첨부할 경우 첨부 파일명 ※ 주의사항 : 첨부 파일의 구분자인 " " 문자열은 파일명에 포함될 수 없습니다.

■ SendSync

<b>Prototype</b>	public bool SendSync(string aliasName, string from, string fromUserName, string to, string cc, string bcc, string subject, string context, bool isHTML, string localFilePath, string fileName)																						
<b>Definition</b>	해당 앨리어스에 설정된 메일 정보로 메일을 동기 방식으로 전송하고 메일 전송 성공 여부를 가져옵니다.																						
<b>Argument</b>	<table border="1"> <tr> <td><i>aliasName</i></td> <td>앨리어스 이름</td> </tr> <tr> <td><i>from</i></td> <td>발신자 메일 주소</td> </tr> <tr> <td><i>fromUserName</i></td> <td>발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨</td> </tr> <tr> <td><i>to</i></td> <td>수신자 메일 주소</td> </tr> <tr> <td><i>cc</i></td> <td>참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정</td> </tr> <tr> <td><i>bcc</i></td> <td>숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정</td> </tr> <tr> <td><i>subject</i></td> <td>메일 제목</td> </tr> <tr> <td><i>context</i></td> <td>메일 내용</td> </tr> <tr> <td><i>isHTML</i></td> <td>메시지 형식을 HTML로 설정할지 여부</td> </tr> <tr> <td><i>localFilePath</i></td> <td>파일을 첨부할 경우 첨부할 파일의 전체 경로 여러 값을 설정할 경우 " "를 구분자로 설정</td> </tr> <tr> <td><i>fileName</i></td> <td>파일을 첨부할 경우 첨부 파일명 ※ 주의사항 : 첨부 파일의 구분자인 " " 문자열은 파일명에 포함될 수 없습니다.</td> </tr> </table>	<i>aliasName</i>	앨리어스 이름	<i>from</i>	발신자 메일 주소	<i>fromUserName</i>	발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨	<i>to</i>	수신자 메일 주소	<i>cc</i>	참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정	<i>bcc</i>	숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정	<i>subject</i>	메일 제목	<i>context</i>	메일 내용	<i>isHTML</i>	메시지 형식을 HTML로 설정할지 여부	<i>localFilePath</i>	파일을 첨부할 경우 첨부할 파일의 전체 경로 여러 값을 설정할 경우 " "를 구분자로 설정	<i>fileName</i>	파일을 첨부할 경우 첨부 파일명 ※ 주의사항 : 첨부 파일의 구분자인 " " 문자열은 파일명에 포함될 수 없습니다.
<i>aliasName</i>	앨리어스 이름																						
<i>from</i>	발신자 메일 주소																						
<i>fromUserName</i>	발신자 이름 값을 null 또는 ""로 설정할 경우 발신자 메일 주소로 적용됨																						
<i>to</i>	수신자 메일 주소																						
<i>cc</i>	참조 메일 주소 값을 null 또는 ""로 설정할 경우 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정																						
<i>bcc</i>	숨은 참조 메일 주소 값을 null 또는 ""로 설정할 경우 숨은 참조 메일 주소로 적용되며, 여러 값을 설정할 경우 "," 또는 ";"를 구분자로 설정																						
<i>subject</i>	메일 제목																						
<i>context</i>	메일 내용																						
<i>isHTML</i>	메시지 형식을 HTML로 설정할지 여부																						
<i>localFilePath</i>	파일을 첨부할 경우 첨부할 파일의 전체 경로 여러 값을 설정할 경우 " "를 구분자로 설정																						
<i>fileName</i>	파일을 첨부할 경우 첨부 파일명 ※ 주의사항 : 첨부 파일의 구분자인 " " 문자열은 파일명에 포함될 수 없습니다.																						

## Option

- 메일 설정

Key	Value	설명
<b>active</b>	true/false	메일 전송 기능 사용 여부 ex) setProperty("active", "true")
<b>fromSend</b>	발신자 메일 주소	발신자 메일 주소 ex) setProperty("fromSend", "mail@forcs.com")
<b>toSend</b>	수신자 메일 주소	수신자 메일 주소 ex) setProperty("toSend", "mail@forcs.com")
<b>SMTPServer</b>	SMTP 서버 URL	SMTP 서버 URL ex) setProperty("SMTPServer", "mail.forcs.com")
<b>SMTPServerPort</b>	SMTP 서버 포트 번호	SMTP 서버 포트 번호 ex) setProperty("SMTPServerPort", "25")
<b>SMTPUserID</b>	SMTP 사용자 ID	SMTP 인증이 필요한 메일 서버를 이용할 때 인증할 사용자 ID ex) setProperty("SMTPUserID", "UserID")
<b>SMTPUserPassword</b>	SMTP 사용자 ID의 비밀번호	SMTP 인증이 필요한 메일 서버를 이용할 때 인증할 사용자 ID의 비밀번호 ex) setProperty("SMTPUserPassword", "Password")
<b>SMTPUserID_encrypted</b>	암호화된 SMTP 사용자 ID	SMTP 인증이 필요한 메일 서버를 이용할 때 인증할 암호화된 사용자 ID ex) setProperty("SMTPUserID_encrypted", "ScPdRcRgFgHdEbJaJbPcFc")
<b>SMTPUserPassword_encrypted</b>	암호화된 SMTP 사용자 ID의 비밀번호	SMTP 인증이 필요한 메일 서버를 이용할 때 인증할 암호화된 사용자 비밀번호 ex) setProperty("SMTPUserPassword_encrypted", "ScPdRcRgFgEbGaDbKbFbMaIbPa")
<b>EnableSSL</b>	true/false	SSL 사용 여부 ex) setProperty("EnableSSL", "true")
<b>SendRetryCount</b>	재전송 횟수	메일 전송 실패 시 재전송 횟수 ex) setProperty("SendRetryCount", "3")
<b>SendRetryPeriodTime</b>	재전송 주기 (단위 : 초)	메일 전송 실패 시 재전송 주기 ex) setProperty("SendRetryPeriodTime", "10")

<b>PrefixSubjectMessage</b>	메일 제목 접두어	메일 제목 앞에 사용할 접두어 ex) setProperty("PrefixSubjectMessage", "[Forcs]")
-----------------------------	-----------	---

### Sample : MailSample.cs

```

using System;
using System;
using System.Collections.Specialized;

using oz.framework.api;

namespace sample
{

    public class MailSample
    {
        public static void Main()
        {
            Mail mail = new
Mail("http://127.0.0.1/oz/server.aspx","admin","admin",true,false);

            NameValueCollection properties = new NameValueCollection();
            properties["active"] = "true";
            properties["fromSend"] = "gosys@forcs.com";
            properties["toSend"] = "oz@forcs.com";
            properties["SMTPServer"] = "smtp.gmail.com";
            properties["SMTPServerPort"] = "465";
            properties["SMTPUserID"] = "gosys@forcs.com";
            properties["SMTPUserPassword"] = "password";
            properties["EnableSSL"] = "true";

            mail.AddAlias("forcs", properties);

            mail.SendSync("forcs", "oz@forcs.com", "UserName", "oz@gmail.com",
null, null, "This is test","content", false, @"d:\parameter_test.ozd",
"test.ozd");

            NameValueCollection props = mail.GetAliasConfig("forcs");
            mail.ModifyAlias("forcs", "forcs2", props);

            mail.AddAlias("forcs", props);
            mail.AddAlias("forcs3", props);

            mail.RemoveAlias("forcs3");

            foreach(string name in mail.GetMailAliasNames())

```

```
        {
            NameValueCollection properties2 = mail.GetAliasConfig(name);
            foreach(string key in properties2.Keys)
            {
                Console.WriteLine(key + "=" + properties2[key]);
            }
            Console.WriteLine();
        }
    }
}
```

## Class Module

### Constructor Summary

- **Module(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)**
- **Module(string url, string id, string pw, bool autoLogin, bool useUSL)**

### Property Summary

- **bool MemoAllowed**
- **string Password**

### Method Summary

- **void AddODIPParameter(string odiName, string key, string value)**
- **void AddODIPParameter(string odiName, string item, string category, IDictionary parameters)**
- **IReportInfo AddReport(string itemName, string categoryName)**
- **IReportInfo AddReport(string itemName, string categoryName, string displayName)**
- **void AddParameter(string key, string value)**
- **OZParameter[] GetODIPParameter(string category, string odiName)**
- **Stream GetOZD()**
- **Stream GetOZD(string item, string category, string[] urls)**
- **Stream GetOZD(string itemName, string categoryName)**
- **OZODIItem[] GetOZQReportParameter(string category, string ozqItemName)**
- **OZReportParameter GetOZReportParameter(string category, string reportName)**
- **OZDataSetInfo[] GetOZReportDataSetInfo(string category, string reportName) throws OZCPEException**
- **void RedirectODIPath(string odiName, string path)**
- **void SaveOZD(string fileName)**

- void SaveOZD(string fileName, string item, string category, string[] urls)
- void SaveOZD(string filename, string itemName, string category)

## Constructor Detail

	<i>//TCP/IP 방식</i>	
	public Module(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)	
<b>ASP .NET</b>	<i>//HTTP 방식</i>	
	public Module(string url, string id, string pw, bool autoLogin, bool useUSL)	
	<i>url</i>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
	<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
	<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003
<b>Argument</b>	<i>id</i>	사용자 아이디 ex) string id = "admin";
	<i>pw</i>	사용자 패스워드 ex) string pw = "admin";
	<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
	<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Property Detail

### ■ MemoAllowed

**Prototype** public bool MemoAllowed{set;}

**Definition** 메모 허용 여부

### ■ Password

**Prototype** public string Password{set;}

**Definition** OZD 저장 시 password

## Method Detail

### ■ AddODIPParameter

<b>Prototype</b>	public void AddODIPParameter(string odiName, string key, string value) throws OZAPIException	
<b>Definition</b>	SDM 파일을 만들기 위해 ODI에 지정되어 있는 ODI 파라미터 값을 설정합니다. ODI 파라미터 값을 설정하지 않을 경우에는 ODI 기본 파라미터의 기본 값이 사용됩니다.	
<b>Argument</b>	<i>odiName</i>	ODI 이름
	<i>key</i>	ODI 파라미터 이름
	<i>value</i>	ODI 파라미터 값

### ■ AddODIPParameter

<b>Prototype</b>	public void AddODIPParameter(string odiName, string item, string category, IDictionary parameters) throws OZAPIException	
<b>Definition</b>	SDM 파일을 만들기 위해 ODI에 지정되어 있는 ODI 파라미터 값을 설정합니다. ODI 파라미터 값을 설정하지 않을 경우에는 ODI 기본 파라미터 기본 값이 사용됩니다. ODI 파라미터 값에 따라 각각 다른 SDM 파일이 필요할 경우 각각의 파라미터 값 별로 SDM 생성 단위를 만들고자 할 때 사용합니다.	
<b>Argument</b>	<i>odiName</i>	디자이너에서 지정한 ODI 이름
	<i>item</i>	ODI 파일명
	<i>category</i>	ODI 파일 경로
	<i>parameters</i>	파라미터 Key, Value 가 설정되어 있는 Dictionary

### ■ AddParameter

<b>Prototype</b>	public void AddParameter(string key, string value) throws OZAPIException	
<b>Definition</b>	SDM 파일을 만들기 위해 폼에 지정되어 있는 폼 파라미터 값을 설정합니다. 폼 파라미터 값을 설정하지 않을 경우에는 폼 파라미터 기본 값이 사용됩니다.	
<b>Argument</b>	<i>key</i>	폼 파라미터 이름
	<i>value</i>	폼 파라미터 값

### ■ AddReport

<b>Prototype</b>	public IReportInfo AddReport(string itemName, string categoryName)
------------------	--

<b>Definition</b>	다중 보고서를 하나의 OZD 파일로 만들기 위하여 리포트를 추가합니다.	
<b>Argument</b>	<i>itemName</i>	리포트를 추가할 아이템 이름
	<i>categoryName</i>	리포트를 추가할 아이템의 카테고리 이름

<b>Prototype</b>	public IReportInfo AddReport(string itemName, string categoryName, string displayName)	
<b>Definition</b>	다중 보고서를 하나의 OZD 파일로 만들기 위하여 리포트를 추가합니다.	
<b>Argument</b>	<i>itemName</i>	리포트를 추가할 아이템 이름
	<i>categoryName</i>	리포트를 추가할 아이템의 카테고리 이름
	<i>displayname</i>	뷰어의 보고서 트리에 표시할 보고서 이름

■ **GetODIParameter**

<b>Prototype</b>	public OZParameter[] GetODIParameter(string category, string odiName)	
<b>Definition</b>	ODI 파일에 추가된 ODI 파라미터를 가져옵니다.	
<b>Argument</b>	<i>category</i>	카테고리 이름
	<i>odiName</i>	ODI 이름

■ **GetOZD**

<b>Prototype</b>	public Stream GetOZD()	
<b>Definition</b>	다중 보고서를 갖는 OZD를 생성한 후 입력 스트림을 가져옵니다.	

<b>Prototype</b>	public Stream GetOZD(string item, string category, string[] urls) throws OZAPIException	
<b>Definition</b>	서버로부터 보고서 파일과 SDM 파일을 가져와 OZD 파일을 생성하여 가져옵니다. OZD 파일을 생성할 때 urls에 지정된 이미지 파일도 함께 저장합니다. ※ 주의사항 : API 함수를 이용하여 데이터를 바인딩할 경우에는 항상 DM_TYPE="Memory", FetchType="Batch"로 바인딩됩니다. 대용량 데이터를 바인딩할 경우에는 메모리를 많이 사용하거나 결과물을 얻는데 시간이 걸릴 수 있습니다.	
	<i>item</i>	아이템 이름 (보고서 파일인 OZR 파일 이름)
<b>Argument</b>	<i>category</i>	카테고리 이름
	<i>urls</i>	OZD에 함께 저장할 이미지 파일의 URL

<b>Prototype</b>	public Stream GetOZD(string itemName, string categoryName)	
<b>Definition</b>	다중 보고서를 갖는 OZD를 생성한 후 입력 스트림을 가져옵니다	
<b>Argument</b>	<i>itemName</i>	아이템 이름 (보고서 파일인 OZR 파일 이름)

---

<i>categoryName</i>	카테고리 이름
---------------------	---------

---

#### ■ GetOZQReportParameter

<b>Prototype</b>	public OZODIItem[] GetOZQReportParameter(string category, string ozqItemName)
------------------	---

<b>Definition</b>	OZQR 파일에 추가된 ODI 파라미터를 가져옵니다.
-------------------	-------------------------------

<b>Argument</b>	<i>category</i> 카테고리 이름
	<i>odiName</i> OZQR 이름

---

#### ■ GetOZReportParameter

<b>Prototype</b>	public OZReportParameter GetOZReportParameter(string category, string reportName)
------------------	---

<b>Definition</b>	리포트의 ODI 파라미터와 폼 파라미터를 가져옵니다.
-------------------	-------------------------------

※ 주의사항 : FX 데이터에 추가된 ODI 파라미터는 가져오지 않습니다.

<b>Argument</b>	<i>category</i> 카테고리 이름
	<i>reportName</i> 리포트 이름

---

#### ■ GetOZReportDataSetInfo

<b>Prototype</b>	public OZDataSetInfo[] GetOZReportDataSetInfo(string category, string reportName) throws OZCPEXception
------------------	--

<b>Definition</b>	리포트에 추가된 데이터 셋 정보를 가져옵니다. 데이터 트리에 추가된 순서대로 리턴됩니다.
-------------------	---

<b>Argument</b>	<i>category</i> 카테고리 이름
	<i>reportName</i> 리포트 이름

---

#### ■ RedirectODIPath

<b>Prototype</b>	public void RedirectODIPath(string odiName, string path)
------------------	--

<b>Definition</b>	OZD 생성시 동적으로 변경할 ODI파일을 등록합니다.
-------------------	--------------------------------

<b>Argument</b>	<i>odiName</i> ODI 이름
	<i>path</i> ODI 파일 경로

---

#### ■ SaveOZD

<b>Prototype</b>	public void SaveOZD(string fileName)
------------------	--------------------------------------

<b>Definition</b>	다중 보고서를 갖는 OZD를 생성한 후 특정 경로에 OZD 파일을 저장합니다.
-------------------	---

<b>Argument</b>	<i>filename</i> 저장할 카테고리 경로 및 파일 이름
-----------------	-------------------------------------

---

■ SaveOZD

<b>Prototype</b>	public void SaveOZD(string fileName, string item, string category, string[] urls) throws OZAPIException	
<b>Definition</b>	보고서 파일을 OZD 파일로 저장합니다. ※ 주의사항 : API 함수를 이용하여 데이터를 바인딩할 경우에는 항상 DM_TYPE="Momory", FetchType="Batch"으로 바인딩됩니다. 대용량 데이터를 바인딩할 경우에는 메모리를 많이 사용하거나 결과물을 얻는데 시간이 걸릴 수 있습니다.	
<b>Argument</b>	<i>fileName</i>	저장 경로를 포함한 OZD 파일명 설정한 저장 폴더가 없을 경우에는 자동으로 폴더 생성함
	<i>item</i>	보고서 파일(.ozr) 이름
	<i>category</i>	보고서 파일(.ozr)의 카테고리 이름
	<i>urls</i>	OZD에 함께 저장할 이미지 파일의 URL

■ SaveOZD

<b>Prototype</b>	public void SaveOZD(string filename,string itemName, string category)	
<b>Definition</b>	다중 보고서를 갖는 OZD를 생성한 후 특정 경로에 OZD 파일을 저장합니다.	
<b>Argument</b>	<i>filename</i>	저장할 카테고리 경로 및 파일 이름
	<i>itemName</i>	보고서 파일(.ozr) 이름
	<i>category</i>	보고서 파일(.ozr)의 카테고리 이름

관련 Class

■ OZReportParameter(oz.framework.odm.OZReportParameter.class)

리포트의 ODI 파라미터와 폼 파라미터 정보를 가지고 있는 클래스입니다.

- 메소드

▪ GetODIList

<b>Prototype</b>	public OZODIItem[] GetODIList()
<b>Definition</b>	ODI 정보 객체를 가져옵니다.
<b>Return</b>	<i>OZODIItem</i> ODI 정보 객체

▪ GetFormParameters

<b>Prototype</b>	public OZParameter[] GetFormParameters()
<b>Definition</b>	폼 파라미터 정보 객체 가져옵니다.

---

**Return**      *OZParameter*    Form Key, Value, Type, Desc 값이 저장된 객체

---

■ **OZDataSetInfo(oz.framework.odm.OZDataSetInfo.class)**

리포트에 추가된 데이터 셋 정보를 가지고 있는 클래스입니다.

- 메소드

▪ **GetDataSetServiceType**

---

**Prototype**    public string GetDataSetServiceType()

---

**Definition**    데이터 셋의 서비스 타입을 가져옵니다.  
ODI, DataService, FXData 중 하나의 값으로 리턴됩니다.

---

▪ **GetDataSetName**

---

**Prototype**    public string GetDataSetName()

---

**Definition**    데이터 셋의 이름을 가져옵니다.

---

▪ **GetDataSetFields**

---

**Prototype**    public OZAttributeList GetDataSetFields()

---

**Definition**    데이터 셋의 필드 정보를 가져옵니다.  
OZAttributeList는 fieldName과 fieldType으로 구성되어 있으며,  
fieldType은 JDBC Types과 동일합니다. 예를 들어, VARCHAR 타입인  
경우 12가 리턴됩니다.

---

▪ **GetMaxRowCount**

---

**Prototype**    public int GetMaxRowCount()

---

**Definition**    데이터 셋의 "최대 행 수" 속성 값을 가져옵니다.  
CSVDataSet, FxDataSet은 무조건 0으로 리턴됩니다.

---

▪ **GetUse**

---

**Prototype**    public bool GetUse()

---

**Definition**    데이터 셋의 "사용" 속성 값을 가져옵니다.  
CSVDataSet, FxDataSet은 무조건 true로 리턴됩니다.

---

▪ **GetExportable**

---

**Prototype**    public bool GetExportable()

---

**Definition**    데이터 셋의 "셋 저장 가능" 속성 값을 가져옵니다.

---

■ **OZODIItem(oz.framework.odm.OZODIItem.class)**

ODI 파라미터 정보를 가지고 있는 클래스입니다.

- 메소드

▪ GetCategory

---

**Prototype** public string GetCategory()

---

**Definition** 카테고리 이름을 가져옵니다.

---

▪ GetODIName

---

**Prototype** public string GetODIName()

---

**Definition** ODI 이름을 가져옵니다.

---

▪ GetODIFileName

---

**Prototype** public string GetODIFileName()

---

**Definition** ODI 파일 이름을 가져옵니다.

---

▪ GetODIParameters

---

**Prototype** public OZParameter[] GetODIParameters()

---

**Definition** 해당 ODI 파라미터를 가져옵니다.

---

**Return** *OZParameter* ODI Key, Value, Type, Desc 값이 저장된 객체

---

■ **OZParameter(oz.framework.odm.OZParameter.class)**

파라미터 정보를 가지고 있는 클래스입니다.

- 메소드

▪ GetName

---

**Prototype** public string GetName()

---

**Definition** 파라미터 이름을 가져옵니다.

---

▪ GetType

---

**Prototype** public int GetType()

---

파라미터 타입을 가져옵니다.

**Definition** 파라미터 타입은 JDBC Types과 동일합니다. 예를 들어, VARCHAR 타입인 경우 12가 리턴됩니다.

---

▪ GetValue

---

**Prototype** public string GetValue()

---

**Definition** 파라미터 값을 가져옵니다.

---

- GetDescription

---

**Prototype** public string GetDescription()

---

**Definition** 파라미터 설명을 가져옵니다.

---

## 관련 Interface

- IReportInfo(oz.framework.api.IReportInfo)

다중 보고서를 갖는 OZD 생성 시 보고서에 대한 부가 정보를 설정합니다.

- ItemName

---

**Prototype** string ItemName{get;}

---

**Definition** 아이템 이름

---

- CategoryName

---

**Prototype** string CategoryName{get;}

---

**Definition** 아이템의 카테고리 이름

---

- AddURL

---

**Prototype** void AddURL(params string[] urls)

---

**Definition** 보고서에 포함된 이미지 파일의 URL을 배열로 설정합니다.

---

**Argument** *urls* 이미지 파일 URL

---

- AddParameter

---

**Prototype** void AddParameter(string key, string value)

---

**Definition** 파라미터 값을 설정합니다.

---

**Argument** *key* 파라미터 이름

---

*value* 파라미터 값

---

- AddODIParameter

---

**Prototype** public void AddODIParameter(string odiName, string key, string value)

---

**Definition** ODI 파라미터 값을 설정합니다.

---

*odiName* ODI 이름

---

**Argument** *key* ODI 파라미터 이름

---

*value* ODI 파라미터 값

---

- AddProperties

<b>Prototype</b>	public void AddProperties(string key, string value)
<b>Definition</b>	보고서 저장 옵션을 설정합니다.
	<i>key</i> 옵션 이름
<b>Argument</b>	옵션 값
	<i>value</i> 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.

- AddFrameworkURL

<b>Prototype</b>	void AddFrameworkURL(string fxName, string url, string charset)
<b>Definition</b>	FX 데이터의 프레임워크 URL을 설정합니다.
	<i>fxName</i> FX 데이터 이름
	<i>url</i> 프레임워크 URL
<b>Argument</b>	프레임워크 URL의 파라미터를 처리할 문자 셋 ※ 참고사항
	<ul style="list-style-type: none"> <li>문자 셋을 null 또는 공백으로 설정한 경우 시스템의 기본 문자 셋으로 적용됩니다.</li> <li>문자 셋에 base64(대소문자 구분 안 함)를 포함하여 설정한 경우 예를 들어, base64/ksc5601 또는 base64/utf-8 등의 형태로 설정한 경우 ksc5601 또는 utf-8로 인코딩한 후 base64로 한 번 더 인코딩합니다.</li> </ul>
	<i>charset</i>

**Sample : MultiReportSample.cs**

```
using System;
using oz.framework.api;

namespace sample
{
    public class MultiReportSample
    {
        public static void Main()
        {
            Module maker = new Module("http://127.0.0.1/oz/server.aspx",
            "admin", "admin", true, false);

            string[] urls = new string[]{"file://D:\\pictures\\0.gif",
            "file://D:\\pictures\\1.gif", "file://D:\\pictures\\2.gif"};
        }
    }
}
```

```

        maker.AddODIPParameter("parameter_test.odi", "odiparam",
"testvalue");
        maker.AddParameter("formparam", "testvalue");
        IReportInfo reportInfo = maker.AddReport("parameter_test.ozr", "/"
        , "Report1");
        reportInfo.AddURL(urls);
        reportInfo.AddODIPParameter("parameter_test.odi", "odiparam1", "form
number 1");
        reportInfo.AddParameter("formparam1", "form parameter 1");

        reportInfo = maker.AddReport("parameter_test.ozr", "/", "Report2");
        reportInfo.AddURL(urls);
        reportInfo.AddODIPParameter("parameter_test.odi", "odiparam1", "form
number 2");
        reportInfo.AddParameter("formparam2", "form parameter 2");

        reportInfo = maker.AddReport("parameter_test.ozr", "/", "Report3");
        reportInfo.AddURL(urls);
        reportInfo.AddODIPParameter("parameter_test.odi", "odiparam1", "form
number 3");
        reportInfo.AddParameter("formparam2", "form parameter 3");

        maker.SaveOZD(@"d:\MultiReport.ozd");
    }
}
}

```

## UseOzdParameterSample.cs

```

using System;
using oz.framework.api;

namespace sample
{
    public class UseOzdParametersSample
    {
        public static void Main()
        {
            Module maker = new Module("http://127.0.0.1/oz/server.aspx",
"admin", "admin", true, false);

            maker.AddODIPParameter("parameter_test.odi", "odiparam1",
"testvalue");
            maker.AddParameter("formparam1", "testvalue");
            IReportInfo reportInfo = maker.AddReport("parameter_test.ozr",
"/");

```

```
        reportInfo.AddODIPParameter("parameter_test.odi", "odiparam1",  
"form number 1");  
        reportInfo.AddParameter("formparam1", "form parameter 1");  
        reportInfo.AddProperties("use_ozd_parameter", "false");  
  
        reportInfo = maker.AddReport("parameter_test.ozr", "/");  
        reportInfo.AddODIPParameter("parameter_test.odi", "odiparam1",  
"form number 2");  
        reportInfo.AddParameter("formparam1", "form parameter 2");  
        reportInfo.AddProperties("use_ozd_parameter", "false");  
  
        maker.AddReport("parameter_test.ozr", "/");  
  
        maker.SaveOZD(@"d:\sample.ozd");  
  
    }  
}
```

## Class Monitor

### Constructor Summary

- **Monitor(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)**
- **Monitor(string url, string id, string pw, bool autoLogin, bool useUSL)**

### Method Summary

- **OZServerInfo GetServerInfo()**
- **MemoryStatus GetMemoryInfo()**
- **Stream DownloadLog()**

### Constructor Detail

---

	<i>//TCP/IP 방식</i>	
	<code>public Monitor(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)</code>	
<b>ASP .NET</b>	<i>//HTTP 방식</i>	
	<code>public Monitor(string url, string id, string pw, bool autoLogin, bool useUSL)</code>	
	<i>url</i>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";
	<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
<b>Argument</b>	<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;
	<i>id</i>	사용자 아이디 ex) string id = "admin";
	<i>pw</i>	사용자 패스워드 ex) string pw = "admin";

---

<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ GetServerInfo

**Prototype** public OZServerInfo GetServerInfo() throws OZAPIException

**Definition** 서버의 버전 정보와 각 모듈별 버전 정보를 가져옵니다.

### ■ GetMemoryInfo

**Prototype** public MemoryStatus GetMemoryInfo() throws OZAPIException

**Definition** 현재의 메모리 사용량(전체 메모리, 사용 메모리, 비사용 메모리) 정보를 가져옵니다.

### ■ DownloadLog

**Prototype** public Stream DownloadLog() throws OZAPIException

**Definition** 서버에 저장된 모니터 로그 파일을 가져옵니다.

## 관련 Class

### ■ MemoryStatus(oz.server.monitor.MemoryStatus)

Server가 실행되고 있는 System의 메모리 상태를 나타냅니다.

- 멤버 변수
  - public long TotalMemory : 서버 VM의 토탈 메모리
  - public long FreeMemory : 서버 VM의 Free 메모리
  - public long UseMemory : 서버 VM의 사용 메모리
  - public long FreeMemoryPercentage : 서버 VM의 FreeMemory/TotalMemory

### ■ OZServerInfo(oz.server.monitor.OZServerInfo)

Server와 Server가 실행되고 있는 System의 버전 정보들을 나타냅니다.

- 멤버 변수
  - public string OSName : Server가 실행되고 있는 OS의 이름

- public string OSVersion : OS의 버전
- public string FrameworkVersion .NET Framework 버전
- public string FrameworkVendor : .NET Framework 공급자
- public string ServerVersion : 오즈 서버 버전

### Sample : MonitorSample.cs

```
using System;

using oz.util;
using oz.framework.api;
using oz.framework.monitor;

namespace sample{
    /// <summary>
    /// MonitorTest
    /// </summary>
    public class MonitorSample{
        public static void Main(){
            string url = "http://127.0.0.1/oz/server.aspx";
            string id = "admin";
            string password = "admin";

            Monitor m = new Monitor(url, id, password, true, true);

            OZServerInfo si = m.GetServerInfo();
            Console.WriteLine(si);

            MemoryStatus ms = m.GetMemoryInfo();
            Console.WriteLine(ms);

            System.IO.Stream monitorLog = m.DownloadLog();
        }
    }
}
```

## Class Service

### Constructor Summary

- `Service(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)`
- `Service(string url, string id, string pw, bool autoLogin, bool useUSL)`

### Method Summary

- `void GrabageCollect()`
- `void Stop()`
- `void Restart()`
- `bool Ping()`
- `int GetHandlerCount()`

### Constructor Detail

---

<b>ASP .NET</b>	<pre>//TCP/IP 방식 public service(string ip, int port, string id, string pw, bool autoLogin, bool useUSL)</pre>								
	<pre>//HTTP 방식 public Service(string url, string id, string pw, bool autoLogin, bool useUSL)</pre>								
<b>Argument</b>	<table border="0" style="width: 100%;"> <tr> <td style="padding-right: 10px;"><i>url</i></td> <td>HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";</td> </tr> <tr> <td style="padding-right: 10px;"><i>ip</i></td> <td>TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";</td> </tr> <tr> <td style="padding-right: 10px;"><i>port</i></td> <td>TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;</td> </tr> <tr> <td style="padding-right: 10px;"><i>id</i></td> <td>사용자 아이디 ex) string id = "admin";</td> </tr> </table>	<i>url</i>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";	<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";	<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;	<i>id</i>	사용자 아이디 ex) string id = "admin";
<i>url</i>	HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server.aspx";								
<i>ip</i>	TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";								
<i>port</i>	TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;								
<i>id</i>	사용자 아이디 ex) string id = "admin";								

---

<i>pw</i>	사용자 패스워드 ex) string pw = "admin";
<i>autoLogin</i>	자동 로그인 여부 ex) bool autoLogin = true;
<i>useUSL</i>	USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ GrabageCollect

**Prototype** public void GrabageCollect() throw OZCPEXception

**Definition** 사용되지 않는 메모리를 수거해 재사용 가능하게 해주는 garbage collection을 실행합니다.

### ■ Stop

**Prototype** public void Stop() throw OZCPEXception

**Definition** 동작 중인 작업을 완료시킨 후 서버를 정상 종료시킵니다.  
Daemon 타입 서버만 지원합니다.

### ■ Restart

**Prototype** public void Restart() throw OZCPEXception

**Definition** 동작 중인 서버를 정상 종료시키고 다시 실행합니다.  
Daemon 타입 서버만 지원합니다.

### ■ Ping

**Prototype** public bool Ping() throw OZCPEXception

**Definition** 서버의 구동 여부를 반환합니다.

### ■ GetHandlerCount

**Prototype** public int GetHandlerCount() throw OZCPEXception

**Definition** 서버의 동시 실행 개수를 가져옵니다.

## Sample : ServiceSample.cs

```
using System;
```

```
using System.IO;
using oz.framework.api;
using oz.framework.repositoryex.info;
using oz.util;

namespace server{
    public class ServicesSample{
        public ServicesSample(){
            //
            // TODO: 여기에 생성자 논리를 추가
            //
        }

        static Service service = null;
        public static string Main(string[] args){
            string result = "성공";
            string url = args[0];
            string ip = args[1];
            int port = Int32.Parse(args[2]);
            string id = args[3];
            string password = args[4];
            bool isDaemon = Boolean.Parse(args[5]);
            service = null;
            try{
                if(isDaemon){
                    service = new Service(ip, port, id, password, true,
false);
                }else{
                    service = new Service(url, id, password,true,false);
                }
                Console.WriteLine("ping=" + service.Ping());
                Console.WriteLine("GetHandlerCount=" +
service.GetHandlerCount());
                service.GarbageCollect();
                //service.Stop();
                //service.Restart();
                //Console.WriteLine("ping="+service.Ping());
            }catch(Exception e){
                Console.WriteLine(e.Message);
                result = e.Message;
            }
            return result;
        }
    }
}
```

## Ⅱ. 오즈 스케줄러 서버 API

- Class Program
- Class Publisher
- Class Scheduler
- Class TaskHolidayInfo
- Class TaskHolidayGroupInfo

오즈 스케줄러 서버에 관련된 각종 정보 조회와 실시간 환경 설정 변경 기능을 사용자 애플리케이션에서 직접 제어할 수 있도록 닷넷 API를 제공합니다.

다음은 오즈 스케줄러 서버 API로 제공하는 주요 클래스에 대한 설명입니다.

클래스 이름	설명
<b>Program</b>	오즈 스케줄러에서 외부 프로그램 등록 및 관리 기능을 수행합니다.
<b>Publisher</b>	오즈 스케줄러에 의해 생성되는 보고서 결과 파일을 관리합니다.
<b>Scheduler</b>	보고서 스케줄 작업을 생성하고 결과 파일 및 스케줄러 관리 기능을 수행합니다.
<b>TaskHolidayInfo</b>	태스크에 지정할 휴일을 정의합니다.
<b>TaskHolidayGroupInfo</b>	태스크에 지정할 휴일 그룹을 정의합니다.

오즈 스케줄러 서버 API를 사용하기 위해서는 다음과 같은 어셈블리 파일을 오즈 닷넷 서버가 설치된 경로의 bin 폴더에 위치시켜야 합니다.

라이브러리 파일명	설명
<b>OZServer.NET.dll</b>	오즈 서버 및 스케줄러 서버에 접속하기 위한 파일입니다.
<b>OZSchedulerAPI.dll</b>	오즈 스케줄러 서버에 접속하여 스케줄러 서버를 API로 제어하기 위한 파일입니다.

## Class Program

### Constructor Summary

- Program(string ip, int port)

### Method Summary

- void CreateFolder(ServerInfo s, string folder)
- Stream DownloadFile(ServerInfo s, string file)
- FileInfo[] GetExternalProgramInfos(ServerInfo s, string folder)
- void RemoveFiles(ServerInfo s, string folder, string[] files)
- void RemoveFolder(ServerInfo s, string folder, bool forciblyRemove)
- void UploadFile(ServerInfo s, string file, Stream inputStream)

### Constructor Detail

<b>Prototype</b>	public Program(string ip, int port)
<b>Argument</b>	<p><i>ip</i> 오즈 스케줄러 서버가 설치되어 있는 호스트 컴퓨터의 IP ex) string ip = "127.0.0.1";</p> <p><i>port</i> 스케줄러 포트 (기본 값 : 9521) ex) int port = 9521;</p>

### Method Detail

- CreateFolder

<b>Prototype</b>	public void CreateFolder(ServerInfo s, string folder)
<b>Definition</b>	스케줄러에서 외부 프로그램의 폴더를 생성합니다.
<b>Argument</b>	<p><i>s</i> 오즈 서버 정보</p> <p><i>folder</i> 생성할 폴더명</p>

■ **DownloadFile**

<b>Prototype</b>	public Stream DownloadFile(ServerInfo s, string file)				
<b>Definition</b>	등록된 외부 프로그램 파일을 다운로드합니다.				
<b>Argument</b>	<table border="0"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>file</i></td> <td>다운로드할 파일명</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>file</i>	다운로드할 파일명
<i>s</i>	오즈 서버 정보				
<i>file</i>	다운로드할 파일명				

■ **GetExternalProgramInfos**

<b>Prototype</b>	public FileInfo[] GetExternalProgramInfos(ServerInfo s, string folder)				
<b>Definition</b>	스케줄러에서 외부 프로그램 목록을 가져옵니다.				
<b>Argument</b>	<table border="0"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>folder</i></td> <td>폴더명</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>folder</i>	폴더명
<i>s</i>	오즈 서버 정보				
<i>folder</i>	폴더명				

■ **RemoveFiles**

<b>Prototype</b>	public void RemoveFiles(ServerInfo s, string folder, string[] files)						
<b>Definition</b>	등록된 외부 프로그램 파일을 삭제합니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>folder</i></td> <td>폴더명</td> </tr> <tr> <td><i>files</i></td> <td>삭제할 프로그램 파일명</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>folder</i>	폴더명	<i>files</i>	삭제할 프로그램 파일명
<i>s</i>	오즈 서버 정보						
<i>folder</i>	폴더명						
<i>files</i>	삭제할 프로그램 파일명						

■ **RemoveFolder**

<b>Prototype</b>	public void RemoveFolder(ServerInfo s, string folder, bool forciblyRemove)						
<b>Definition</b>	스케줄러에서 외부 프로그램의 폴더를 삭제합니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>folder</i></td> <td>삭제할 폴더명</td> </tr> <tr> <td><i>forciblyRemove</i></td> <td>                     하위 폴더가 존재할 경우 폴더 삭제 여부                      true : 폴더를 삭제함                      false : 폴더를 삭제하지 않음                 </td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>folder</i>	삭제할 폴더명	<i>forciblyRemove</i>	하위 폴더가 존재할 경우 폴더 삭제 여부 true : 폴더를 삭제함 false : 폴더를 삭제하지 않음
<i>s</i>	오즈 서버 정보						
<i>folder</i>	삭제할 폴더명						
<i>forciblyRemove</i>	하위 폴더가 존재할 경우 폴더 삭제 여부 true : 폴더를 삭제함 false : 폴더를 삭제하지 않음						

■ **UploadFile**

<b>Prototype</b>	public void UploadFile(ServerInfo s, string file, Stream inputStream)
<b>Definition</b>	스케줄러 서버에 외부 프로그램을 등록(업로드)합니다.

	<i>s</i>	오즈 서버 정보
<b>Argument</b>	<i>file</i>	등록할 파일명
	<i>inputStream</i>	등록할 파일의 입력 스트림

## 관련 Class

### ■ ServerInfo(oz.scheduler.ServerInfo)

오즈 서버에 대한 기본적인 정보를 가지고 있는 클래스입니다.

- 속성

#### ▪ IsDaemon

**Prototype** public bool IsDaemon{get;set;}

오즈 서버 타입

**Definition**

- true : TCP/IP 방식 서버
- false : HTTP 방식 서버

#### ▪ IP

**Prototype** public string IP{get;set;}

오즈 서버 IP

**Definition** Server가 TCP/IP 방식일 경우에만 설정합니다.

#### ▪ Port

**Prototype** public int Port{get;set;}

오즈 서버 Port 번호

**Definition** Server가 TCP/IP 방식일 경우에만 설정합니다.

#### ▪ URL

**Prototype** public string URL{get;set;}

오즈 서버 URL

**Definition** Server가 HTTP 방식일 경우에만 설정합니다.

#### ▪ ID

**Prototype** public string ID{get;set;}

**Definition** 사용자의 ID

#### ▪ Password

**Prototype** public string Password{get;set;}

---

**Definition**    사용자의 패스워드

---

- DirectoryName

---

**Prototype**    public string DirectoryName{get;set;}

---

**Definition**    경로

---

- UserType

---

**Prototype**    public UserType UserType{get;set;}

---

**Definition**    사용자 타입(Administrator,Normal,Everyone)

---

- **FileInfo(oz.scheduler.FileInfo)**

파일 또는 폴더에 대한 기본 정보 값(파일/폴더, 이름, 크기, 최종 수정 시각)들을 가지고 있습니다.

- 속성

- IsDirectory

---

**Prototype**    public bool IsDirectory{get;set;}

---

**Definition**    디렉토리(폴더)인지 파일인지 구분

---

- Name

---

**Prototype**    public string Name{get;set;}

---

**Definition**    파일명 또는 디렉토리명

---

- Size

---

**Prototype**    public long Size{get;set;}

---

**Definition**    파일 크기 또는 디렉토리 크기

---

- LastModified

---

**Prototype**    public DateTime LastModified{get;set;}

---

**Definition**    파일 또는 디렉토리의 최종 수정 시각, 1970년 1월 1일 00:00:00 GMT를 기준으로 일시초로 시간을 계산한 값

---

### Sample : ProgramSample.cs

```
using System;
```

```
namespace sample
{
    public class ProgramSample
    {
        public static void Main()
        {
            oz.framework.api.Program program = new
            oz.framework.api.Program("127.0.0.1", 9521);

            oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();
            serverInfo.IsDaemon = false;
            serverInfo.IP = "127.0.0.1";
            serverInfo.Port = 8003;
            serverInfo.URL = "http://127.0.0.1/oz/server.aspx";
            serverInfo.ID = "admin";
            serverInfo.Password = "admin";
            serverInfo.UserType = oz.scheduler.UserType.Administrator;

            program.CreateFolder(serverInfo, "/folder");

            System.IO.MemoryStream inputStream = new System.IO.MemoryStream(new
            byte[500]);

            program.UploadFile(serverInfo, "/folder/file1", inputStream);
            inputStream.Position = 0;
            program.UploadFile(serverInfo, "/folder/file2", inputStream);

            System.IO.Stream downloadedFile = program.DownloadFile(serverInfo,
            "/folder/file1");
            if(downloadedFile.Length != inputStream.Length)
                throw new InvalidOperationException("Invalid program status.
            Binary data has modified by outside. ");

            oz.scheduler.FileInfo[] extProgInfos =
            program.GetExternalProgramInfos(serverInfo, "folder");
            foreach(oz.scheduler.FileInfo extProgInfo in extProgInfos)
            {
                Console.WriteLine("IsDirectory : " + extProgInfo.IsDirectory);
                Console.WriteLine("LastModified : " + extProgInfo.LastModified);
                Console.WriteLine("Name : " + extProgInfo.Name);
                Console.WriteLine("Size : " + extProgInfo.Size);
            }

            program.RemoveFiles(serverInfo, "/folder", "file1", "file2");

            program.RemoveFolder(serverInfo, "/folder", true);
        }
    }
}
```

## Class Publisher

### Constructor Summary

- **Publisher(string ip, int port)**

### Method Summary

- **void CreateFolder(ServerInfo s, string folder)**
- **Stream DownloadFile(ServerInfo s, string file)**
- **FileInfo[] GetPublishedInfos(ServerInfo s, string folder)**
- **void RemoveFiles(ServerInfo s, string folder, string[] files)**
- **void RemoveFolder(ServerInfo s, string folder, bool forciblyRemove)**

### Constructor Detail

<b>Prototype</b>	public Publisher(string ip, int port)	
<b>Argument</b>	<i>ip</i>	오즈 스케줄러 서버가 설치되어 있는 호스트 컴퓨터의 IP ex) string ip = "127.0.0.1";
	<i>port</i>	스케줄러 포트 (기본 값 : 9521) ex) int port = 9521;

### Method Detail

- **CreateFolder**

<b>Prototype</b>	public void CreateFolder(ServerInfo s, string folder)	
<b>Definition</b>	스케줄러 익스포트 파일을 저장할 폴더를 생성합니다.	
<b>Argument</b>	<i>s</i>	오즈 서버 정보
	<i>folder</i>	생성할 폴더명

### ■ DownloadFile

<b>Prototype</b>	public Stream DownloadFile(ServerInfo s, string file)	
<b>Definition</b>	등록된 스케줄러 익스포트 파일을 다운로드합니다.	
<b>Argument</b>	<i>s</i>	오즈 서버 정보
	<i>file</i>	다운로드할 파일명

### ■ GetPublishedInfos

<b>Prototype</b>	public FileInfo[] GetPublishedInfos(ServerInfo s, string folder)	
<b>Definition</b>	스케줄러에서 익스포트 파일 목록을 가져옵니다.	
<b>Argument</b>	<i>s</i>	오즈 서버 정보
	<i>folder</i>	폴더명

### ■ RemoveFiles

<b>Prototype</b>	public void RemoveFiles(ServerInfo s, string folder, string[] files)	
<b>Definition</b>	등록된 스케줄러 익스포트 파일을 삭제합니다.	
<b>Argument</b>	<i>s</i>	오즈 서버 정보
	<i>folder</i>	삭제할 폴더명
	<i>files</i>	삭제할 파일명

### ■ RemoveFolder

<b>Prototype</b>	public void RemoveFolder(ServerInfo s, string folder, bool forciblyRemove)	
<b>Definition</b>	스케줄러 익스포트 파일이 저장되는 폴더를 삭제합니다.	
<b>Argument</b>	<i>s</i>	오즈 서버 정보
	<i>folder</i>	삭제할 폴더명
	<i>forciblyRemove</i>	하위 폴더가 존재할 경우 폴더 삭제 여부 true : 폴더를 삭제함 false : 폴더를 삭제하지 않음

## 관련 Class

### ■ ServerInfo(oz.scheduler.ServerInfo)

Program class의 "관련 class" 부분을 참조하시기 바랍니다.

**■ FileInfo(oz. scheduler.FileInfo)**

Program class의 "관련 class" 부분을 참조하시기 바랍니다.

**Sample : PublisherSample.cs**

```
using System;

namespace sample
{
    public class PublisherSample
    {
        public static void Main()
        {
            oz.framework.api.Publisher publisher = new
            oz.framework.api.Publisher("127.0.0.1", 9521);
            oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();
            serverInfo.IsDaemon = false;
            serverInfo.IP = "127.0.0.1";
            serverInfo.Port = 8003;
            serverInfo.URL = "http://127.0.0.1/oz/server.aspx";
            serverInfo.ID = "admin";
            serverInfo.Password = "admin";
            serverInfo.UserType = oz.scheduler.UserType.Administrator;

            publisher.CreateFolder(serverInfo, "/folder");

            oz.scheduler.FileInfo[] pubFileInfos =
            publisher.GetPublishedFileInfos(serverInfo, "/ozd");

            foreach(oz.scheduler.FileInfo pubFileInfo in pubFileInfos)
            {
                Console.WriteLine("IsDirectory : " + pubFileInfo.IsDirectory);
                Console.WriteLine("LastModified : " +
            pubFileInfo.LastModified);
                Console.WriteLine("Name : " + pubFileInfo.Name);
                Console.WriteLine("Size : " + pubFileInfo.Size);

                System.IO.Stream inputStream =
            publisher.DownloadFile(serverInfo, "/ozd/" + pubFileInfo.Name);
                Console.WriteLine(inputStream.Length);
            }

            try
            {
                publisher.RemoveFiles(serverInfo, "/folder", "file1", "file2");
            }
            catch(Exception)
```

```
    {  
        // there is no files so error is gonna be occurred.  
    }  
  
    publisher.RemoveFolder(serverInfo, "/folder", true);  
}   
}
```

## Class Scheduler

### Constructor Summary

- Scheduler(string ip, int port)

### Method Summary

- string CreateTask(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)
- ScheduledTask[] GetTaskInfos(ServerInfo s)
- TaskResult[] GetTaskResults(ServerInfo s, string from, string to, string taskId)
- int GetTaskWaitTime(ServerInfo s, string taskId)
- int GetTaskWaitCount()
- void RemoveTask(ServerInfo s, string taskId)
- string ModifyTask(ServerInfo s, string taskId, NameValueCollection configMap, NameValueCollection exportMap)
- void PauseTask(ServerInfo s, string taskId)
- void ResumeTask(ServerInfo s, string taskId)
- void Stop(ServerInfo s, bool forciblyStop)
- bool Export(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)
- bool MakePDF(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)
- bool Print(ServerInfo s, NameValueCollection configMap, NameValueCollection printMap)
- NameValueCollection GetConfiguration(ServerInfo s)
- void ModifyConfiguration(ServerInfo s, NameValueCollection configMap)
- bool Ping()
- string[] GetSchedulingInfos (string path)
- void ConvertSchedulingInfos(string oldPath, string newPath)
- bool AddTaskHoliday(TaskHolidayInfo value)

- `bool ModifyTaskHoliday(string old_key, TaskHolidayInfo new_value)`
- `bool DeleteTaskHoliday(string[] keys)`
- `bool AddTaskHolidayGroup(TaskHolidayGroupInfo value)`
- `bool ModifyTaskHolidayGroup(string old_key, TaskHolidayInfo new_value)`
- `bool DeleteTaskHolidayGroup(string key)`
- `TaskHolidayInfos GetTaskHolidayInfos()`
- `TaskHolidayGroupInfos GetTaskHolidayGroupInfos()`
- `void SaveTaskHoliday()`
- `DirectExportResult DirectExport(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)`
- `Hashtable[] DirectExportFiles(ServerInfo s, NameValueCollection properties, NameValueCollection exportProperties, string localTempPath)`
- `DirectExportResult DirectExportResult(ServerInfo s, NameValueCollection properties, NameValueCollection exportProperties)`
- `HashTable DirectExportByteArray(ServerInfo s, NameValueCollection configProperties, NameValueCollection exportProperties)`
- `DirectPrintResult DirectPrint(ServerInfo s, NameValueCollection configMap, NameValueCollection printProperties)`
- `void GetFile(ServerInfo s, string filename, string targetFile)`
- `void GetTaskProperties(ServerInfo s, string tasked, NameValueCollection configMap, NameValueCollection exportMap)`

## Constructor Detail

<b>Prototype</b>	<code>public scheduler(string ip, int port)</code>
<b>Argument</b>	<p><i>ip</i>                   오즈 스케줄러 서버가 설치되어 있는 호스트 컴퓨터의 IP ex) string ip = "127.0.0.1";</p> <p><i>port</i>                 스케줄러 포트 (기본 값 : 9521) ex) int port = 9521;</p>

## Method Detail

- `CreateTask`

<b>Prototype</b>	<code>public string CreateTask(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)</code>
<b>Definition</b>	<p>폼 파일을 기반으로 태스크를 생성한 후 생성된 태스크 ID를 가져옵니다.</p> <p>※ 참고사항 : CreateTask 함수로 여러 개의 태스크 생성 시 뷰어의 대기 큐와 상관없이 실행되는 뷰어의 개수만큼 Thread와 param 파일이 생성됩니다.</p>
<b>Argument</b>	<p><i>s</i>                    오즈 서버 정보</p> <p><i>configMap</i>            스케줄러 설정 옵션                           설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p> <p><i>exportMap</i>            익스포트 정보                           설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p>

■ **GetTaskInfos**

<b>Prototype</b>	<code>public ScheduledTask[] GetTaskInfos(ServerInfo s)</code>
<b>Definition</b>	현재 생성된 태스크의 결과물을 가져옵니다.
<b>Argument</b>	<i>s</i> 오즈 서버 정보

■ **GetTaskResults**

<b>Prototype</b>	<code>public TaskResult[] GetTaskResults(ServerInfo s, string from, string to, string taskId)</code>
<b>Definition</b>	태스크 결과를 가져옵니다.
<b>Argument</b>	<p><i>s</i>                    오즈 서버 정보</p> <p><i>from</i>                태스크 결과를 가져올 시작 시간</p> <p><i>to</i>                    태스크 결과를 가져올 종료 시간</p> <p><i>taskId</i>              가져올 태스크 아이디</p>

■ **GetTaskWaitTime**

<b>Prototype</b>	<code>public int GetTaskwaitTime(ServerInfo s, String taskId)</code>
<b>Definition</b>	태스크가 대기한 시간을 가져옵니다. (단위 : 초)
<b>Argument</b>	<p><i>s</i>                    오즈 서버 정보</p> <p><i>taskId</i>              가져올 태스크 아이디</p>

■ **GetTaskWaitCount**

<b>Prototype</b>	<code>public int GetTaskwaitCount()</code>
------------------	--

---

**Definition** 실행 대기 중인 태스크 개수를 가져옵니다.

---

■ **RemoveTask**

---

**Prototype** public void RemoveTask(ServerInfo s, string taskId)

---

**Definition** 태스크를 삭제합니다.

---

**Argument**

<i>s</i>	오즈 서버 정보
<i>taskId</i>	삭제할 태스크 아이디

---

■ **ModifyTask**

---

**Prototype** public string ModifyTask(ServerInfo s, string taskId, NameValueCollection configMap, NameValueCollection exportMap)

---

**Definition** 태스크 속성을 변경합니다.

---

**Argument**

<i>s</i>	오즈 서버 정보
<i>taskId</i>	속성을 변경할 태스크 아이디
<i>configMap</i>	스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하십시오. 기 바랍니다.
<i>exportMap</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하십시오. 기 바랍니다.

---

■ **PauseTask**

---

**Prototype** public void PauseTask(ServerInfo s, string taskId)

---

**Definition** 태스크를 일시 중지시킵니다.

---

**Argument**

<i>s</i>	오즈 서버 정보
<i>taskId</i>	중지할 태스크 아이디

---

■ **ResumeTask**

---

**Prototype** public void ResumeTask(ServerInfo s, string taskId)

---

**Definition** 중지된 태스크를 다시 실행시킵니다.

---

**Argument**

<i>s</i>	오즈 서버 정보
<i>taskId</i>	다시 실행시킬 태스크 아이디

---

■ **Stop**

---

**Prototype** public void Stop(ServerInfo s, bool forciblyStop)

---

<b>Definition</b>	스케줄러를 중지시킵니다.
<b>Argument</b>	<p><i>s</i> 오즈 서버 정보</p> <p><i>forciblystop</i> 스케줄러를 강제로 중지시킬지 여부</p>

■ **Export**

<b>Prototype</b>	public bool Export(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)
<b>Definition</b>	<p>뷰어 파라미터를 그대로 사용하여 익스포트한 후 익스포트 성공 여부를 반환합니다.</p> <p>※ 주의사항 : 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType=None"이 아닌 경우에만 사용할 수 있습니다.</p>
<b>Argument</b>	<p><i>s</i> 오즈 서버 정보</p> <p><i>configMap</i> 스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p> <p><i>exportMap</i> 익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p>

■ **MakePDF**

<b>Prototype</b>	public bool MakePDF(ServerInfo s, NameValueCollection configMap, NameValueCollection exportMap)
<b>Definition</b>	<p>스케줄러의 작업 목록에 등록하지 않고 PDF 익스포트만 수행한 후 익스포트 성공 여부를 반환합니다.</p> <p>※ 주의사항 : 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType=None"이 아닌 경우에만 사용할 수 있습니다.</p>
<b>Argument</b>	<p><i>s</i> 오즈 서버 정보</p> <p><i>configMap</i> 익스포트를 위한 폼 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. CreatTask()와 비슷하나 PDF 관련 정보만 입력하면 됩니다.</p> <p><i>exportMap</i> PDF 익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. CreatTask()와 비슷하나 PDF 관련 정보만 입력하면 됩니다.</p>

■ **Print**

<b>Prototype</b>	public bool Print(ServerInfo s, NameValueCollection configMap, NameValueCollection printMap)						
<b>Definition</b>	<p>뷰어 파라미터를 적용시켜 프린트하고 성공 여부를 반환합니다. (실제 프린트 작업의 성공 여부가 아니라 뷰어에 프린트 호출이 성공했는지 여부를 반환함)</p> <p>※ 주의사항 : "task_type=viewerTag"인 경우에만 사용될 수 있으며 이외의 값이 들어오면 무시합니다. 또한 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType=None"이 아닌 경우에만 사용할수 있습니다.</p>						
<b>Argument</b>	<table border="1"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>configMap</i></td> <td>스케줄러 설정 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> <tr> <td><i>printMap</i></td> <td>출력 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. 단, print.mode = silent로만 실행됩니다.</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>configMap</i>	스케줄러 설정 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.	<i>printMap</i>	출력 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. 단, print.mode = silent로만 실행됩니다.
<i>s</i>	오즈 서버 정보						
<i>configMap</i>	스케줄러 설정 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.						
<i>printMap</i>	출력 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. 단, print.mode = silent로만 실행됩니다.						

※ 참고사항

- print API 호출 시 태스크 작업 주기 옵션과 관계없이 즉시 한 번만 프린트 작업을 호출하고 태스크는 생성되지 않으며, print API 호출 시 메일 보내기 등의 파라미터를 설정하여도 메일 관련 작업은 일어나지 않고 오직 프린트만 합니다.
- print 함수를 API로 구현할 경우 뷰어 파라미터에서 설정한 파라미터 중 아래의 파라미터는 항상 해당 값으로 고정되어 동작합니다.

```
viewer.allowmultiframe=true
viewer.mode=print
viewer.printcommand=true
viewer.showerrorMessage=false
viewer.useprogressbar=false
print.ingnoreerror=false
print.mode=silent
export.confirmsave=false
export.format=""
information.debug=debug
```

■ GetConfiguration

<b>Prototype</b>	public NameValueCollection GetConfiguration(ServerInfo s)
<b>Definition</b>	스케줄러 설정 값을 가져옵니다.
<b>Argument</b>	<i>s</i> 오즈 서버 정보

■ ModifyConfiguration

<b>Prototype</b>	<code>public void ModifyConfiguration(ServerInfo s, NameValueCollection configMap)</code>
<b>Definition</b>	스케줄러의 설정 값을 변경합니다.
<b>Argument</b>	<p><i>s</i>                    오즈 서버 정보</p> <p><i>configMap</i>        스케줄러 설정 옵션                  설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p>

■ **Ping**

<b>Prototype</b>	<code>public bool Ping()</code>
<b>Definition</b>	서버의 구동 여부를 반환합니다.

■ **GetSchedulingInfos**

<b>Prototype</b>	<code>public string[] GetSchedulingInfos(string path)</code>
<b>Definition</b>	<p>설정된 경로에 있는 OZS 파일 목록을 가져옵니다. (path를 파일명으로 지정하면 해당 OZS 파일 정보만 가져오며, path를 폴더명으로 지정하면 해당 폴더에 있는 모든 OZS 파일 정보를 가져옵니다. 만일 해당 경로에 파일이 존재하지 않거나 존재하지 않는 폴더를 지정하면 빈 값을 리턴합니다.)</p>
<b>Argument</b>	<p><i>path</i>                    OZS 파일 목록을 가져올 경로</p> <p>※ 주의사항 : OZS 파일 목록을 가져올 경로는 /%SCH_HOME%/[path]/로 지정됩니다. 예를 들어 path를 sample로 설정할 경우에는 /%SCH_HOME%/sample/에 있는 OZS 파일 목록을 가져옵니다.</p>

■ **ConvertSchedulingInfos**

<b>Prototype</b>	<code>public void ConvertSchedulingInfos(string oldPath, string newPath)</code>
<b>Definition</b>	2.5 버전의 OZS 파일을 최신 버전으로 변경하여 저장합니다.
<b>Argument</b>	<p><i>oldPath</i>                변환할 OZS 파일명 또는 경로</p> <p>※ 참고사항 : 해당 경로는 /%SCH_HOME%/[oldPath]/입니다.</p> <p><i>newPath</i>                변환된 OZS 파일 저장 경로</p> <p>※ 참고사항 : 해당 경로는 /%SCH_HOME%/[newPath]/이며 newPath는 oldPath와 다르게 설정하여야 합니다.</p>

■ **AddTaskHoliday**

<b>Prototype</b>	<code>public bool AddTaskHoliday(TaskHolidayInfo value)</code>
------------------	--

---

**Definition**      태스크 휴일 정보를 추가합니다.

---

**Argument**      *value*            태스크 휴일 정보

---

#### ■ **ModifyTaskHoliday**

---

**Prototype**      `public bool ModifyTaskHoliday(string old_key, TaskHolidayInfo new_value)`

---

**Definition**      태스크 휴일 정보를 수정합니다.

---

**Argument**      *old\_key*            수정할 태스크 휴일 이름

---

**Argument**      *new\_value*        새로운 태스크 휴일 정보

---

#### ■ **DeleteTaskHoliday**

---

**Prototype**      `public bool DeleteTaskHoliday(string[] keys)`

---

**Definition**      태스크 휴일 정보를 삭제합니다.

---

**Argument**      *keys*                삭제할 태스크 휴일 이름 배열

---

#### ■ **AddTaskHolidayGroup**

---

**Prototype**      `public bool AddTaskHolidayGroup(TaskHolidayGroupInfo value)`

---

**Definition**      태스크 휴일 그룹 정보를 추가합니다.

---

**Argument**      *value*                태스크 휴일 그룹 정보

---

#### ■ **ModifyTaskHolidayGroup**

---

**Prototype**      `public bool ModifyTaskHolidayGroup(string old_key, TaskHolidayGroupInfo new_value)`

---

**Definition**      태스크 휴일 그룹 정보를 수정합니다.

---

**Argument**      *old\_key*            수정할 태스크 휴일 그룹 이름

---

**Argument**      *new\_value*        새로운 태스크 휴일 그룹 정보

---

#### ■ **DeleteTaskHolidayGroup**

---

**Prototype**      `public bool DeleteTaskHolidayGroup(string key)`

---

**Definition**      태스크 휴일 그룹 정보를 삭제합니다.

---

**Argument**      *key*                 삭제할 태스크 휴일 그룹 이름

---

#### ■ **GetTaskHolidayInfos**

---

**Prototype**      `public TaskHolidayInfos GetTaskHolidayInfos()`

---



<i>exportProperties</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.
<i>localTempPath</i>	익스포트한 결과를 저장할 로컬 임시 경로

■ **DirectExportResult**

<b>Prototype</b>	public DirectExportResult DirectExportResult(ServerInfo s, NameValueCollection properties, NameValueCollection exportProperties)						
<b>Definition</b>	뷰어 파라미터를 적용시켜 익스포트한 후 결과 정보를 반환합니다. ※ 주의사항 : 서버에 설정된 스케줄러의 환경 설정 중 task_type이 viewerTag 이고, ViewerType이 ActiveX 또는 Applet인 경우에만 사용할 수 있습니다.						
<b>Argument</b>	<table border="1"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>properties</i></td> <td>스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> <tr> <td><i>exportProperties</i></td> <td>익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>properties</i>	스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.	<i>exportProperties</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.
<i>s</i>	오즈 서버 정보						
<i>properties</i>	스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.						
<i>exportProperties</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.						

■ **DirectExportByteArray**

<b>Prototype</b>	public Hashtable DirectExportByteArray(ServerInfo s, NameValueCollection configProperties, NameValueCollection exportProperties)						
<b>Definition</b>	태스크를 실행한 후 실행 결과 정보를 반환합니다. 태스크 실행 결과를 파일로 저장하지 않고, 메모리로 익스포트하려면 ude.classname을 설정한 후 ViewerType=Applet, export_file=false로 설정하시기 바랍니다. ※ 주의사항 : 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType=None"이 아닌 경우에만 사용할 수 있습니다.						
<b>Argument</b>	<table border="1"> <tr> <td><i>s</i></td> <td>오즈 서버 정보</td> </tr> <tr> <td><i>configProperties</i></td> <td>스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> <tr> <td><i>exportProperties</i></td> <td>익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</td> </tr> </table>	<i>s</i>	오즈 서버 정보	<i>configProperties</i>	스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.	<i>exportProperties</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.
<i>s</i>	오즈 서버 정보						
<i>configProperties</i>	스케줄러 설정 옵션 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.						
<i>exportProperties</i>	익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.						

■ **DirectPrint**

<b>Prototype</b>	public DirectPrintResult DirectPrint(ServerInfo s, NameValueCollection configMap, NameValueCollection printMap)
------------------	---

	<p>뷰어 파라미터를 적용시켜 프린트하고 결과 정보를 반환합니다.</p> <p>※ 주의사항 : "task_type=viewerTag"인 경우에만 사용될 수 있으며 이외의 값이 들어오면 무시합니다. 또한 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType=None"이 아닌 경우에만 사용할 수 있습니다.</p>
<b>Definition</b>	
	<p><i>s</i> 오즈 서버 정보</p>
	<p>스케줄러 설정 정보</p>
<b>Argument</b>	<p><i>configMap</i> 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다.</p>
	<p>출력 정보</p>
	<p><i>printMap</i> 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. 단, print.mode = silent로만 실행됩니다.</p>

■ **GetFile**

<b>Prototype</b>	public void GetFile(ServerInfo s, string fileName, string targetFile)
<b>Definition</b>	파일을 가져옵니다.
	<p><i>s</i> 오즈 서버 정보</p>
<b>Argument</b>	<p><i>fileName</i> 가져올 파일명</p> <p><i>targetFile</i> 저장할 파일 명</p>

■ **GetTaskProperties**

<b>Prototype</b>	public void GetTaskProperties(ServerInfo s, string TaskID, NameValueCollection configMap, NameValueCollection exportMap)
<b>Definition</b>	태스크의 속성 즉 스케줄러에서 태스크를 생성할 때 사용되는 설정 옵션 (configMap)과 스케줄러에서 익스포트되는 파일의 설정 옵션(exportMap)을 가져옵니다.
	<p><i>s</i> 오즈 서버 정보</p>
	<p><i>taskID</i> 속성을 가져올 태스크 아이디</p>
<b>Argument</b>	<p><i>configMap</i> 익스포트를 위한 폼 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. CreatTask()와 비슷하나 PDF 관련 정보만 입력하면 됩니다.</p>
	<p><i>exportMap</i> PDF 익스포트 정보 설정할 수 있는 key와 설명은 아래 "Option" 부분을 참조하시기 바랍니다. CreatTask()와 비슷하나 PDF 관련 정보만 입력하면 됩니다.</p>

## 관련 Class

### ■ TaskResult(oz.scheduler.TaskResult)

태스크 실행 결과를 나타냅니다.

- 속성

#### ▪ TaskID

---

**Prototype** public string TaskID{get;set;}

---

**Definition** 태스크 ID

---

#### ▪ CompletedTime

---

**Prototype** public string CompletedTime{get;set;}

---

**Definition** 태스크 완료 시간

---

#### ▪ IsSuccessful

---

**Prototype** public bool IsSuccessful{get;set;}

---

**Definition** 태스크 성공 여부

---

#### ▪ FormFileName

---

**Prototype** public string FormFileName{get;set;}

---

**Definition** 보고서 이름

---

#### ▪ ParamInfo

---

**Prototype** public string ParamInfo{get;set;}

---

**Definition** 파라미터

---

#### ▪ SchedulingType

---

**Prototype** public string SchedulingType{get;set;}

---

**Definition** 스케줄 타입

---

#### ▪ ExportedFiles

---

**Prototype** public string ExportedFiles{get;set;}

---

**Definition** 익스포트된 파일 이름

---

#### ▪ ErrorMessage

---

**Prototype**    `public string ErrorMessage{get;set;}`

---

**Definition**    에러 발생 시 에러 메시지

---

#### ■ **DirectTaskResult(oz.scheduler.DirectTaskResult)**

태스크 실행 결과를 나타냅니다.

- 멤버 변수
  - `public string TaskID`: 태스크 ID
  - `public string CompletedTime` : 태스크 완료 시간
  - `public string ExecuteTime`: 태스크 수행 시간 (단위 : 초)
  - `public bool IsSuccessful`: 태스크 성공 여부
  - `public string FormName`: 보고서 이름
  - `public string ErrorMessage` : 에러 메시지

#### ■ **oz.scheduler.DirectExportResult extends DirectTaskResult**

익스포트 태스크 실행 결과를 나타냅니다.

- 멤버 변수
  - `public string ExportFileList`: 익스포트된 파일 이름
  - `public int PageCount` : 보고서의 익스포트된 페이지 수

##### 주의사항

- ① 한 보고서를 여러 개의 포맷으로 동시에 익스포트할 경우 보고서의 익스포트된 페이지 수 합계를 나타냅니다.  
예를 들어, 페이지 수가 4 페이지인 보고서를 xls, doc 포맷으로 익스포트 시 `PageCount` 값으로 8이 리턴됩니다.
- ② 보고서 익스포트 시 "`viewer.largebundle=true`"일 경우 `PageCount` 값으로 1이 리턴됩니다.

#### ■ **oz.scheduler.DirectPrintResult extends DirectTaskResult**

프린트 태스크 실행 결과를 나타냅니다.

- 멤버 변수
  - `public int PageCount`: 페이지 수
  - `public int PageCopy`: 인쇄 매수
  - `public string PageRange`: 인쇄 범위
  - `public string PrinterName` : 프린터 이름
  - `public string PrinterDriverName` : 프린터 드라이버 이름

## Option

※ 주의사항

- 서버에 설정된 스케줄러의 환경 설정 중 "ViewerType"을 "None"으로 설정한 경우에는 익스포트 시에 뷰어를 사용하지 않고, 서버 API를 이용하여 익스포트하므로, OZD 파일로만 익스포트할 수 있습니다.
- 뷰어 파라미터 중 아래 표와 같이 몇몇 파라미터는 사용자가 설정한 값은 무시되고, 항상 고정된 값으로만 설정되어 동작됩니다.

뷰어 파라미터	값
<b>viewer.mode</b>	export
<b>viewer.useprogressbar</b>	false
<b>viewer.allowmultiframe</b>	true
<b>export.mode</b>	silent
<b>export.confirmsave</b>	false
<b>information.debug</b>	debug
<b>viewer.showerrormessage</b>	false

■ 기본 설정

- 리포트 이름 설정

Key	Value	설명
<b>report_name</b>	리포트명	태스크 생성할 리포트명
<b>category_name</b>	카테고리명	태스크를 생성할 리포트의 카테고리명

- 서버 데이터 모듈

Key	Value	설명
<b>dm_server_check</b>	"check" "null"	서버 데이터 모듈 선택
<b>dm_server_name</b>	SDM 파일명	서버 데이터 모듈 파일명 (Repository의 odi가 있는 category에 저장된 SDM 파일명)
<b>odi_name</b>	ODI명	서버 데이터 모듈을 만들 ODI명
<b>odi_category_name</b>	카테고리명	서버 데이터 모듈을 만들 ODI가 저장된 카테고리명 (루트인 경우 "/" 로 입력)

- 뷰어 파라미터 설정

Key	Value	설명
-----	-------	----

<b>task_type</b>	"viewerTag"	뷰어 파라미터 그대로 사용 여부 설정 이 옵션은 뷰어에서 사용하는 파라미터를 그대로 사용할 수 있도록 설정해 주는 옵션으로, 주로 멀티 폼을 사용하는 보고서를 익스포트할 때 사용됩니다. 사용 방법은 "SchedulerViewerTagSample.cs"를 참조하시기 바랍니다.
------------------	-------------	--

- 태스크 타임 아웃 설정

Key	Value	설명
<b>taskwait_timeout</b>	타임아웃 (단위 : 초)	태스크 실행 대기 타임아웃 (기본값 : 0) ※ 주의사항 <ul style="list-style-type: none"> <li>주기적으로 실행되는 태스크는 동작 안 함</li> <li>scheduler_server.properties 파일의 ViewerConcurrentCount 옵션 값을 n으로 설정한 경우 동작 안 함</li> </ul>

- 외부 프로그램 설정

Key	Value	설명
<b>external_program_check</b>	"check" "null"	파라미터 생성과 파일 경로의 동적 생성을 위한 외부 프로그램 체크 여부
<b>external_program_command</b>	외부 프로그램명	외부 프로그램명 ("SCH_HOME/External"에서의 상대 경로)

■ 전자 메일 발송

Key	Value	설명
<b>mail_check</b>	"check" "null"	전자 메일 발송 여부
<b>mail_notify_error_check</b>	"check" "null"	수신자에게 에러 통보 여부 선택
<b>mail_recipient_to</b>	수신자 메일 주소	전자 메일 수신자
<b>mail_recipient_cc</b>	참조자 메일 주소	참조할 사람 메일
<b>mail_recipient_bcc</b>	숨은 참조자 메일 주소	숨은 참조할 사람 메일

<b>mail_subject</b>	메일 제목	전자 메일 제목
<b>mail_text_message</b>	메일 내용	전자 메일 내용
<b>mail_html_comment</b>	"check" "null"	전자 메일 내용에 HTML 사용 여부
<b>html_mail_content</b>	"check" "null"	전자 메일 내용에 생성한 HTML 파일로 대체
<b>mail_attach_list</b>	첨부 파일 리스트	전자 메일에 첨부할 스케줄러의 생성된 파일 리스트 (다수 선택이 가능하며 '/'로 구분)
<b>mail_attach_zipcheck</b>	"check" "null"	전자 메일에 첨부할 파일을 하나의 파일로 압축하여 전송
<b>mail_attach_zipfilename</b>	파일명	전자 메일에 첨부할 파일을 하나의 파일로 압축하여 전송 시 파일 이름
<b>mail_attach_zippassword</b>	파일 암호	전자 메일에 첨부할 파일을 하나의 파일로 압축하여 전송할 때 파일 암호

■ 스케줄러 파일 변환

Key	Value	설명
<b>file_export_list</b>	포맷 리스트	변환 파일 리스트 (다수 선택이 가능하며 '/' 로 구분)

■ 결과 파일 저장 여부

Key	Value	설명
<b>export_file</b>	true/false	DirectExportByteArray 함수 실행 시 태스크 실행 결과를 파일로 저장할지 여부 ActiveX 뷰어에서는 항상 true로 동작함 ex) setProperty("export_file", "false")

■ 리포트 파라미터

Key	Value	설명
<b>parameter_count</b>	파라미터 수	입력 파라미터 수

<b>parameter_name_1</b> ... <b>parameter_name_n</b>	파라미터명	입력 파라미터명 (n : 입력 파라미터 수)
<b>parameter_value_1</b> ... <b>parameter_value_n</b>	파라미터 값	입력 파라미터 값 (n : 입력 파라미터 수)

※ 참고사항

- 파라미터 수 : 파라미터의 수는 품과 각각의 ODI에서 사용하는 파라미터 수의 합을 입력 파라미터 수로 설정하고, 그 개수만큼 파라미터 이름과 해당 값을 설정해 주어야 합니다.
- 파라미터명
  - ① 리포트 파라미터명은 "[FORM]." 으로 시작합니다. 예를 들어 empNo라는 파라미터는 "[FORM].empNo"로 입력하여야 합니다.
  - ② ODI 파라미터명은 ODI명으로 시작합니다. 예를 들어 ODI명이 "testodi"이고 파라미터명이 "id"일 경우에는 "testodi.id"로 입력하여야 합니다.

■ ODI 파라미터

Key	Value	설명
<b>odi_parameter_count</b>	파라미터 수	입력 파라미터 수
<b>odi_parameter_name_1</b> ... <b>odi_parameter_name_n</b>	파라미터명	입력 파라미터명 (n : 입력 파라미터 수)
<b>odi_parameter_value_1</b> ... <b>odi_parameter_value_n</b>	파라미터 값	입력 파라미터 값 (n : 입력 파라미터 수)

※ 참고사항 : ODI의 파라미터 설정은 서버 데이터 모듈을 저장할 때만 사용되며, ODI는 서버 데이터 모듈을 저장할 때 지정한 ODI를 사용합니다.

■ 파라미터 값 적용 여부

Key	Value	설명
-----	-------	----

<b>use_ozd_parameter</b>	true/false	OZD 바인딩 시 OZD에 저장된 파라미터 값을 적용할지 여부
--------------------------	------------	------------------------------------

■ 스케줄 시간

- 실행 타입

Key	Value	설명
<b>launch_type</b>	"once" "immediately" "periodically"	실행 타입 once : 한번 실행 immediately : 즉시 실행 periodically : 주기적 실행

▪ launch\_type = once인 경우

Key	Value	설명
<b>execution_year</b>	년도	실행 시간 - 년
<b>execution_month</b>	월	실행 시간 - 월
<b>execution_day</b>	일	실행 시간 - 일
<b>execution_hour</b>	시간	실행 시간 - 시
<b>execution_min</b>	분	실행 시간 - 분

▪ launch\_type = periodically인 경우

Key	Value	설명
<b>start_year</b>	년도	시작 시간 - 년
<b>start_month</b>	월	시작 시간 - 월
<b>start_day</b>	일	시작 시간 - 일
<b>end_year</b>	년도	종료 시간 - 년
<b>end_month</b>	월	종료 시간 - 월
<b>end_day</b>	일	종료 시간 - 일
<b>periodically_execution_day_type</b>	"daily" "weekly" "monthly"	주기 타입

▪ periodically\_execution\_day\_type = daily인 경우

Key	Value	설명
<b>daily_type</b>	weekday	평일
<b>daily_every_days</b>	간격 일수	설정된 일 간격(며칠)

- periodically\_execution\_day\_type = weekly인 경우

Key	Value	설명
<b>weekly_every_weeks</b>	매주 간격	매주 몇번째 주
<b>weekly_monday_check</b>	"check" "null"	매주 월요일
<b>weekly_tuesday_check</b>	"check" "null"	매주 화요일
<b>weekly_wednesday_check</b>	"check" "null"	매주 수요일
<b>weekly_thursday_check</b>	"check" "null"	매주 목요일
<b>weekly_friday_check</b>	"check" "null"	매주 금요일
<b>weekly_saturday_check</b>	"check" "null"	매주 토요일
<b>weekly_sunday_check</b>	"check" "null"	매주 일요일

- periodically\_execution\_day\_type = monthly인 경우

Key	Value	설명
<b>monthly_every_months</b>	월 간격	설정된 달 간격(몇 달)
<b>monthly_type</b>	"specific_day" , "day_of_week", "user_defined"	특정 일이나 주
<b>monthly_days</b>	"1" ~ "31" "LAST"	1 ~ 31 : 매달 특정 일 LAST : 매달 마지막 일
<b>monthly_which_week</b>	"T1" "T2" "T3" "T4" "T5" "TL"	매달 몇 번째 주 T1 : 매달 첫 번째 주 T2 : 매달 두 번째 주 T3 : 매달 세 번째 주 T4 : 매달 네 번째 주 T5 : 매달 다섯 번째 주 TL : 매달 마지막 주

<b>monthly_which_week_day</b>	"sunday" "monday" "tuesday" "wednesday" "thursday" "friday" "saturday"	매달 특정 요일
<b>monthly_user_defined_days</b>	설정할 일 리스트	설정할 일을 오름차순으로 콤마(,)로 연결하여 설정

- 실행 횟수

Key	Value	설명
<b>periodically_execution_time_type</b>	"once" "repeat" "user_defined"	실행 횟수(한번, 여러 번, 사용자 정의)

- periodically\_execution\_time\_type = once인 경우

Key	Value	설명
<b>once_hour</b>	시	실행시킬 시간 - 시
<b>once_min</b>	분	실행시킬 시간 - 분

- periodically\_execution\_time\_type = repeat인 경우

Key	Value	설명
<b>repeat_every_hours</b>	시	주기적으로 실행 간격 - 시
<b>repeat_every_minutes</b>	분	주기적으로 실행 간격 - 분
<b>repeat_start_hour</b>	시	주기적 실행 시작 시간 - 시
<b>repeat_start_minute</b>	분	주기적 실행 시작 시간 - 분
<b>repeat_end_hour</b>	시	주기적 실행 종료 시간 - 시
<b>repeat_end_minute</b>	분	주기적 실행 종료 시간 - 분

※ 참고사항 : 위 6가지 예제는 2시 5분에서 9시 35분 사이에 1시간 4분 간격으로  
태스크 실행하라는 명령을 나타냅니다.

- periodically\_execution\_time\_type = user\_defined인 경우

Key	Value	설명
-----	-------	----

<b>user_defined_time</b>	설정할 시간 리스트	설정할 시간(시:분)을 오름차순으로 콤마(,)로 연결하여 설정
--------------------------	---------------	---------------------------------------

■ 파일 저장

"파일형식.mailattach" 파라미터는 오즈 엔터 프라이즈 매니저에서만 사용하는 파라미터로  
메일 전송 시 첨부 파일로 해당 포맷의 파일을 첨부할지 여부를 설정합니다. true로 설정할  
경우에는 파일을 첨부하며, false로 설정할 경우에는 파일을 첨부하지 않습니다.

ex) setProperty("hdm.mailattach", "true")

그 밖의 파일 저장 시 설정할 수 있는 key에 대한 자세한 사항은 오즈 리포트 뷰어  
매뉴얼의 "오즈 뷰어 호출 옵션"의 각 파일 형식 저장 관련 파라미터 부분을 참조하시기  
바랍니다.

※ 참고사항 : 익스포트 시 Value 값을 "디렉토리이름/파일이름.파일형식"으로 설정할  
경우에는 "%스케줄러디렉토리%\Repository/디렉토리이름" 안의 파일이  
익스포트됩니다. 예를 들어, "FORCS"라는 디렉토리 안의 파일을 CSV  
파일로 익스포트하고자 할 경우 setProperty("csv.filename",  
"FORCS/test.csv")으로 설정하고,  
"%스케줄러디렉토리%\Repository/FORCS" 안에 "test.csv" 파일로  
저장됩니다.

■ modifyConfiguration의 ConfigMap

Key	Value	설명
<b>SchedulerPort</b>	포트 번호	스케줄러 포트 번호 (기본 값 : "9521")
<b>SchedulingInfoFile Path</b>	저장 경로	스케줄링 정보 파일 저장 경로명
<b>SMTPServer</b>	서버 주소	SMTP 서버 주소
<b>SMTPServerProt</b>	서버 포트 번호	SMTP 서버 포트 번호
<b>MailFrom</b>	메일 주소	전자 메일 발송 주소
<b>TempRepositoryF ilePath</b>	저장 경로	임시 파일 저장 경로
<b>RepositoryFileRo otPath</b>	저장 경로	익스포트 파일 루트 저장 경로 설정
<b>ExternalProgram FilePath</b>	저장 경로	외부 프로그램 저장 경로 설정

<b>ErrorNotifyToSender</b>	"true" "false"	관리자에게 에러 통보 여부 설정 <ul style="list-style-type: none"> <li>• true : 통보함</li> <li>• false : 통보하지 않음</li> </ul>
----------------------------	-------------------	--

### Sample : SchedulerSample.cs

```

using System;
using System.Collections.Specialized;

using oz.scheduler.holiday;
using oz.framework.api;

namespace sample
{
    public class SchedulerSample
    {
        public static void Main()
        {
            oz.framework.api.Scheduler scheduler = new
            oz.framework.api.Scheduler("127.0.0.1", 9521);
            if(scheduler.Ping())
                Console.WriteLine("Scheduler is alive. ");
            else
                Console.WriteLine("Scheduler is not being run. ");

            oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();
            serverInfo.IsDaemon = false;
            serverInfo.IP = "127.0.0.1";
            serverInfo.Port = 8003;
            serverInfo.URL = "http://127.0.0.1/oz/server.aspx";
            serverInfo.ID = "admin";
            serverInfo.Password = "admin";
            serverInfo.UserType = oz.scheduler.UserType.Administrator;

            NameValueCollection properties = new NameValueCollection();

            properties["report_name"] = "parameter_test.ozr";
            properties["category_name"] = "/";
            properties["cfg.type"] = "new";

            properties["odi_name"] = "parameter_test.odi";
            properties["odi_category_name"] = "/";
            properties["dm_server_check"] = "check";
            properties["dm_server_name"] = "test1.sdm";

            properties["odi_name"] = "parameter_test.odi";

```

```
properties["odi_category_name"] = "/";
properties["external_program_check"] = "check";
properties["external_program_command"] = "notepad.bat";

properties["parameter_count"] = "2";
properties["parameter_name_1"] = "[FORM].formparam1";
properties["parameter_value_1"] = "form param 1";

properties["parameter_name_2"] = "[FORM].formparam2";
properties["parameter_value_2"] = "form param 1";

properties["odi_parameter_count"] = "2";
properties["odi_parameter_name_1"] = "odiparam1";
properties["odi_parameter_value_1"] = "odi param 1";

properties["odi_parameter_name_2"] = "odiparam2";
properties["odi_parameter_value_2"] = "odi param 1";

properties["launch_type"] = "periodically";
properties["start_year"] = DateTime.Now.Year.ToString();
properties["start_month"] = DateTime.Now.Month.ToString();
properties["start_day"] = DateTime.Now.Day.ToString();

properties["execution_year"] = DateTime.Now.Year.ToString();
properties["execution_month"] = DateTime.Now.Month.ToString();
properties["execution_day"] = DateTime.Now.Day.ToString();
properties["execution_hour"] = DateTime.Now.Hour.ToString();
properties["execution_min"] = DateTime.Now.Minute.ToString();
properties["periodically_execution_day_type"] = "daily";

properties["daily_type"] = "weekday";
properties["periodically_execution_time_type"] = "once";

properties["once_hour"] = DateTime.Now.Hour.ToString();
properties["once_min"] = DateTime.Now.Minute.ToString();

properties["mail_check"] = "check";
properties["html_mail_content"] = "check";
properties["mail_notify_error_check"] = "null";
properties["mail_recipient_to"] = "gosys@forcs.com";
//properties["mail_recipient_cc"] = "abc@forcs.com";
//properties["mail_recipient_bcc"] = "lan@forcs.com";
properties["mail_subject"] = "mail subject";
properties["mail_text_message"] = "mail content";
properties["mail_attach_list"] =
"csv/excel/html/ozd/pdf/text/tiff/word/ppt/jpg/svg";

properties["file_export_list"] =
"csv/xls/html/ozd/pdf/txt/tif/doc/ppt/jpg/svg";
```

```
NameValueCollection exportProperties = new NameValueCollection();

exportProperties["csv.filename "] = "schedulerSample.csv";
exportProperties["csv.pagetitle "] = "page";
exportProperties["csv.pageline "] = "7";
exportProperties["csv.pagestyle"] = "none";
exportProperties["csv.separator "] = "Tab";
exportProperties["csv.removeorange "] = "1,3";
exportProperties["csv.exceptfirstpage "] = "true";
exportProperties["csv.savetointeger "] = "true";

exportProperties["excel.filename "] = "schedulerSample.xls";
exportProperties["excel.numberformat "] = "#,##0.00";
exportProperties["excel.savefont"] = "Arial";
exportProperties["excel.matchmode"] = "columnpersheet";
exportProperties["excel.matchesubmode"] = "rowfirst";
exportProperties["excel.removeorange"] = "1,3";
exportProperties["excel.removeoption "] = " firstpageonly";
exportProperties["excel.removeblank "] = " true";

exportProperties["html.filename "] = "schedulerSample.html";
exportProperties["html.imagepath "] = "file://c:/image";
exportProperties["html.vertical "] = "1";
exportProperties["html.horizontal "] = "1";
exportProperties["html.savebypage "] = "true";
exportProperties["html.offsetx "] = "1";
exportProperties["html.offsety "] = "1";

exportProperties["ozd.filename "] = "schedulerSample.ozd";
exportProperties["ozd.memoallowed "] = "true";
exportProperties["ozd.password"] = "admin";

exportProperties["pdf.filename "] = "schedulerSample.pdf";
//exportProperties["pdf.saverange "] = "1.3";
exportProperties["pdf.title "] = "Report";
exportProperties["pdf.subject "] = "OZ";
exportProperties["pdf.creator "] = "hong";
exportProperties["pdf.author "] = "hong";
exportProperties["pdf.keyword "] = "oz";
exportProperties["pdf.userpassword"] = "user";
exportProperties["pdf.masterpassword "] = "admin";
exportProperties["pdf.printprotected "] = "true";

exportProperties["text.filename "] = "schedulerSample.txt";
exportProperties["text.pagetitle "] = "page";
exportProperties["text.pageline "] = "7";
exportProperties["text.pagestyle"] = "none";
exportProperties["text.separator "] = "Tab";
```

```
exportProperties["text.removepage "] = "1,3";
exportProperties["text.exceptfirstpage "] = "true";
exportProperties["text.savetointeger "] = "true";

exportProperties["tiff.filename "] = "schedulerSample.tif";
exportProperties["tiff.encode "] = "G3";

exportProperties["word.filename "] = "schedulerSample.doc";

exportProperties["ppt.filename "] = "schedulerSample.ppt ";

exportProperties["jpg.filename "] = "schedulerSample.jpg ";

exportProperties["svg.filename "] = "schedulerSample.svg";

string taskID = scheduler.CreateTask(serverInfo, properties,
exportProperties);
Console.WriteLine("taskID : " + taskID);
scheduler.MakePDF(serverInfo, properties, exportProperties);
scheduler.MakePDFByPooling(serverInfo, properties,
exportProperties);

NameValueCollection schedulerProperties = new
NameValueCollection();

schedulerProperties["SchedulerPort"] = "9521";

schedulerProperties["ErrorNotifyToSender"] = "false";
schedulerProperties["ExternalProgramFilePath"] =
"%SCH_HOME%/External";
schedulerProperties["MailFrom"] = "gosys@forcs.com";
schedulerProperties["RepositoryFileRootPath"] =
"%SCH_HOME%/Repository";
schedulerProperties["SMTPServer"] = "mail.forcs.com";
schedulerProperties["SMTPServerPort"] = "465";
schedulerProperties["SMTPUserID"] = "gosys@forcs.com";
schedulerProperties["SMTPUserPassword"] = "password";
schedulerProperties["SchedulingInfoFilePath"] =
"%SCH_HOME%/ScheduledTask";
schedulerProperties["TempRepositoryFilePath"] =
"%SCH_HOME%/TempRepository";

schedulerProperties["ViewerType"] = "ActiveX";
schedulerProperties["ViewerConcurrentCount"] = "10";
schedulerProperties["ViewerTimeout"] = "0";
schedulerProperties["MailSendRetryCount"] = "0";
schedulerProperties["MailSendRetryPeriodTime"] = "30";
schedulerProperties["TaskHoliday"] = "true";
schedulerProperties["PrefixSubjectMessage"] = "";
```

```

schedulerProperties["EnableSSL"] = "false";

scheduler.ModifyConfiguration(serverInfo, schedulerProperties);

foreach(oz.scheduler.ScheduledTask taskInfo in
scheduler.GetTaskInfos(serverInfo))
{
    Console.WriteLine("CategoryName : " + taskInfo.CategoryName);
    Console.WriteLine("LastExecutionTime : " +
taskInfo.LastExecutionTime);
    Console.WriteLine("NextExecutionTime : " +
taskInfo.NextExecutionTime);
    Console.WriteLine("ReportName : " + taskInfo.ReportName);
    Console.WriteLine("Status : " + taskInfo.Status);
    Console.WriteLine("TaskID : " + taskInfo.TaskID);
    Console.WriteLine("Type : " + taskInfo.Type);
}

foreach(oz.scheduler.TaskResult taskResult in
scheduler.GetTaskResults(serverInfo, string.Empty, string.Empty,
string.Empty))
{
    Console.WriteLine("CompletedTime : " +
taskResult.CompletedTime);
    Console.WriteLine("ErrorMessage : " + taskResult.ErrorMessage);
    Console.WriteLine("ExportedFiles : " +
taskResult.ExportedFiles);
    Console.WriteLine("FormFileName : " + taskResult.FormFileName);
    Console.WriteLine("IsSuccessful : " + taskResult.IsSuccessful);
    Console.WriteLine("ParamInfo : " + taskResult.ParamInfo);
    Console.WriteLine("SchedulingType : " +
taskResult.SchedulingType);
    Console.WriteLine("TaskID : " + taskResult.TaskID);
}

foreach(oz.scheduler.ScheduledTask taskInfo in
scheduler.GetTaskInfos(serverInfo))
{
    if(taskInfo.TaskID==taskID)
    {
        if(taskInfo.Status!=oz.scheduler.TaskStatus.Running)
        {
            scheduler.PauseTask(serverInfo, taskID);
            scheduler.ResumeTask(serverInfo, taskID);
            scheduler.RemoveTask(serverInfo, taskID);
        }
    }
}
}

```

```
        scheduler.Stop(serverInfo, true);  
  
    }  
}
```

### Sample : SchedulerViewerTagSample.cs

```
using System;  
using System.Collections.Specialized;  
  
using oz.scheduler.holiday;  
using oz.framework.api;  
  
namespace sample  
{  
    /// <summary>  
    /// SchedulerViewerTag  
    /// </summary>  
    public class SchedulerViewerTagSample  
    {  
        public static void Main()  
        {  
            oz.framework.api.Scheduler scheduler = new  
oz.framework.api.Scheduler("127.0.0.1", 9521);  
            if(scheduler.Ping())  
                Console.WriteLine("Scheduler is alive. ");  
            else  
                Console.WriteLine("Scheduler is not being run. ");  
  
            oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();  
            serverInfo.IsDaemon = false;  
            serverInfo.IP = "127.0.0.1";  
            serverInfo.Port = 8003;  
            serverInfo.URL = "http://127.0.0.1/oz/server.aspx";  
            serverInfo.ID = "admin";  
            serverInfo.Password = "admin";  
            serverInfo.UserType = oz.scheduler.UserType.Administrator;  
  
            NameValueCollection properties = new NameValueCollection();  
  
            properties["task_type"] = "viewerTag";  
  
            properties["cfg.type"] = "new";  
        }  
    }  
}
```

```

properties["launch_type"] = "immediately";

NameValueCollection exportProperties = new NameValueCollection();

if(serverInfo.IsDaemon)
{
    exportProperties["connection.server"] = serverInfo.IP;
    exportProperties["connection.port"] =
serverInfo.Port.ToString();
}
else
    exportProperties["connection.servlet"] = serverInfo.URL;

// 뷰어 태그
exportProperties["connection.reportName"] = "/parameter_test.ozr";
exportProperties["applet.mode"] = "export";
exportProperties["applet.useprogressbar"] = "false";
exportProperties["applet.allowmultiframe"] = "true";
exportProperties["connection.pcount"] = "2";
exportProperties["connection.args1"] = "formparam1=form param 1";
exportProperties["connection.args2"] = "formparam2=form param 2";
exportProperties["export.mode"] = "silent";
exportProperties["export.saveonefile"] = "true";
exportProperties["connection.fetchtype"] = "BATCH";
exportProperties["export.confirmsave"] = "false";
exportProperties["information.debug"] = "debug";
exportProperties["applet.showerrormessage"] = "false";
exportProperties["odi.parameter_test.pcount"] = "2";
exportProperties["odi.parameter_test.args1"] = "odiparam1=odi param
1";
exportProperties["odi.parameter_test.args2"] = "odiparam2=odi param
2";

exportProperties["odi.odinames"] = "parameter_test";
exportProperties["export.format"] =
"csv/xls/html/ozd/pdf/txt/tif/doc/ppt/jpg/svg/mht";
exportProperties["csv.filename"] = "1.csv";
exportProperties["excel.filename"] = "1.xls";
exportProperties["html.filename"] = "1.html";
exportProperties["ozd.filename"] = "1.ozd";
exportProperties["pdf.filename"] = "1.pdf";
exportProperties["text.filename"] = "1.txt";
exportProperties["tiff.filename"] = "1.tif";
exportProperties["word.filename"] = "1.doc";
exportProperties["ppt.filename"] = "1.ppt";
exportProperties["jpg.filename"] = "1.jpg";
exportProperties["svg.filename"] = "1.svg";
exportProperties["mht.filename"] = "1.mht";
exportProperties["hdm.filename"] = "1.hdm";

exportProperties["viewer.childcount"] = "1";

```

```

        if(serverInfo.IsDaemon)
        {
            exportProperties["child1.connection.server"] = serverInfo.IP;
            exportProperties["child1.connection.port"] =
serverInfo.Port.ToString();
        }
        else
            exportProperties["child1.connection.servlet"] = serverInfo.URL;

        exportProperties["child1.connection.reportName"] =
"/parameter_test.ozr";
        exportProperties["child1.applet.mode"] = "export";
        exportProperties["child1.applet.useprogressbar"] = "false";
        exportProperties["child1.applet.allowmultiframe"] = "true";
        exportProperties["child1.export.mode"] = "silent";
        exportProperties["child1.connection.fetchtype"] = "BATCH";
        exportProperties["child1.export.confirmsave"] = "false";
        exportProperties["child1.information.debug"] = "debug";
        exportProperties["child1.applet.showerrorMessage"] = "false";
        exportProperties["child1.connection.pcount"] = "2";
        exportProperties["child1.connection.args1"] = "formparam1=form
param 1";
        exportProperties["child1.connection.args2"] = "formparam2=form
param 2";
        exportProperties["child1.odi.odinames"] = "parameter_test";
        exportProperties["child1.odi.parameter_test.pcount"] = "2";
        exportProperties["child1.odi.parameter_test.args1"] =
"odiparam1=odi param 1";
        exportProperties["child1.odi.parameter_test.args2"] =
"odiparam2=odi param 2";
        exportProperties["child1.export.format"] =
"csv/xls/html/ozd/pdf/txt/tif/doc/ppt/jpg/svg/mht";
        exportProperties["child1.csv.filename"] = "child_1.csv";
        exportProperties["child1.excel.filename"] = "child_1.xls";
        exportProperties["child1.html.filename"] = "child_1.html";
        exportProperties["child1.ozd.filename"] = "child_1.ozd";
        exportProperties["child1.pdf.filename"] = "child_1.pdf";
        exportProperties["child1.text.filename"] = "child_1.txt";
        exportProperties["child1.tiff.filename"] = "child_1.tif";
        exportProperties["child1.word.filename"] = "child_1.doc";
        exportProperties["child1.ppt.filename"] = "child_1.ppt";
        exportProperties["child1.jpg.filename"] = "child_1.jpg";
        exportProperties["child1.svg.filename"] = "child_1.svg";
        exportProperties["child1.mht.filename"] = "child_1.mht";
        exportProperties["child1.hdm.filename"] = "child_1.hdm";

        string taskID = scheduler.CreateTask(serverInfo, properties,
exportProperties);
    }
}

```

```
}

```

### Sample : DirectExportResultSample.cs

```
using System;
using System.Collections.Specialized;
using oz.framework.api;
using oz.scheduler;

namespace sample
{
    /// <summary>
    /// DirectExportResultSample
    /// </summary>
    public class DirectExportResultSample
    {
        public DirectExportResultSample()
        {

        }

        public static void Main()
        {

            oz.framework.api.scheduler scheduler = new
            oz.framework.api.scheduler("127.0.0.1", 9521);

            oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();

            serverInfo.IsDaemon = false;
            serverInfo.IP = "127.0.0.1";
            serverInfo.Port = 8003;
            serverInfo.ID = "admin";
            serverInfo.URL = "http://127.0.0.1/oz/server.aspx";
            serverInfo.Password = "admin";
            serverInfo.UserType = oz.scheduler.UserType.Administrator;

            NameValueCollection properties = new NameValueCollection();

            properties["task_type"] = "viewerTag";
            properties["cfg.type"] = "new";
            properties["launch_type"] = "immediately";

            NameValueCollection exportProperties = new NameValueCollection();
            // viewer tag

            if(serverInfo.IsDaemon)
            {

```

```
        exportProperties["connection.server"] = serverInfo.IP;
        exportProperties["connection.port"] =
serverInfo.Port.ToString();
    }
    else
        exportProperties["connection.servlet"] = serverInfo.URL;
        exportProperties["connection.reportName"] = "/parameter_test.ozr";
        exportProperties["viewer.mode"] = "export";
        exportProperties["export.mode"] = "silent";
        exportProperties["export.confirmsave"] = "false";
        exportProperties["export.format"] = "xls";
        exportProperties["xls.filename"] = "sample.xls";

        DirectExportResult result = scheduler.DirectExport(serverInfo,
properties, exportProperties);

        Console.WriteLine("TaskID : " + result.TaskID);
        Console.WriteLine("CompletedTime : " + result.CompletedTime);
        Console.WriteLine("ExecuteTime ; " + result.ExecuteTime);
        Console.WriteLine("Success : " + result.ISSuccessful);
        Console.WriteLine("FormName : " + result.FormName);
        Console.WriteLine("ExportFileList " + result.ExportFileList);
        Console.WriteLine("PageCount " + result.PageCount);

    }
}
}
```

### Sample : DirectPrintResultSample.cs

```
using System;
using System.Collections.Specialized;
using oz.framework.api;
using oz.scheduler;

namespace sample
{
    /// <summary>
    /// DirectPrintResultSample
    /// </summary>
    public class DirectPrintResultSample
    {
        public DirectPrintResultSample()
        {
        }

        public static void Main()
        {
        }
    }
}
```

```

{

    oz.framework.api.scheduler scheduler = new
oz.framework.api.scheduler("127.0.0.1", 9521);

    oz.scheduler.ServerInfo serverInfo = new oz.scheduler.ServerInfo();

    serverInfo.IsDaemon = false;
    serverInfo.IP = "127.0.0.1";
    serverInfo.Port = 8003;
    serverInfo.URL = "http://127.0.0.1/oz/server.aspx";
    serverInfo.ID = "admin";
    serverInfo.Password = "admin";
    serverInfo.UserType = oz.scheduler.UserType.Administrator;

    NameValueCollection properties = new NameValueCollection();

    properties["task_type"] = "viewerTag";
    properties["cfg.type"] = "new";
    properties["launch_type"] = "immediately";

    NameValueCollection printProperties = new NameValueCollection();
    // Viewer tag
    if(serverInfo.IsDaemon)
    {
        printProperties["connection.server"] = serverInfo.IP;
        printProperties["connection.port"] = serverInfo.Port.ToString();
    }
    else
        printProperties["connection.servlet"] = serverInfo.URL;

    printProperties["connection.reportName"] = "/parameter_test.ozr";
    printProperties["applet.mode"] = "export";
    printProperties["connection.pcount"] = "2";
    printProperties["connection.args1"] = "formparam1=formparam 1";
    printProperties["connection.args2"] = "formparam2=formparam 1";
    printProperties["export.mode"] = "silent";
    printProperties["export.saveonefile"] = "true";
    printProperties["connection.fetchtype"] = "BATCH";
    printProperties["export.confirmsave"] = "false";
    printProperties["information.debug"] = "debug";
    printProperties["applet.showerrorMessage"] = "false";
    printProperties["odi.parameter_test.pcount"] = "2";
    printProperties["odi.parameter_test.args1"] = "odiparam1=odiparam
1";
    printProperties["odi.parameter_test.args2"] = "odiparam2=odiparam
2";

    printProperties["odi.odinames"] = "parameter_test";

    printProperties["viewer.mode"] = "print";

```

```
printProperties["print.mode"] = "silent";
printProperties["viewer.printcommand"] = "true";

DirectPrintResult result = scheduler.DirectPrint(serverInfo,
properties, printProperties);

Console.WriteLine("TaskID 아이디 : " + result.TaskID);
Console.WriteLine("CompletedTime : " + result.CompletedTime);
Console.WriteLine("ExecuteTime ; " + result.ExecuteTime);
Console.WriteLine("FormName : " + result.FormName);

Console.WriteLine("PageCount : " + result.PageCount);
Console.WriteLine("PageCopy : " + result.PageCopy);
Console.WriteLine("PageRange : " + result.PageRange);
Console.WriteLine("PrinterName : " + result.PrinterName);
Console.WriteLine("PrinterDriverName : " +
result.PrinterDriverName);

    }
}
}
```

## Class TaskHolidayInfo

### Constructor Summary

- TaskHolidayInfo(string name)

### Property Summary

- string Name{get;set;}
- string BaseDay{get;set;}
- HolidayType Type{get;set;}
- int Start{get;set;}
- int End{get;set;}

### Constructor Detail

<b>Prototype</b>	public TaskHolidayInfo(string name)	
<b>Argument</b>	<i>name</i>	이름 ※ 주의사항 : 이름에 ";" 문자가 포함될 수 없습니다.

### Property Detail

- Name

**Prototype** public string Name{get;set;}

**Definition** 태스크 휴일 이름

- BaseDay

**Prototype** public string BaseDay{get;set;}

**Definition** 태스크 휴일로 지정할 기준일

		기준일의 패턴
		※ 참고사항 : 기준일 포맷
<b>Argument</b>	<i>pattern</i>	<ul style="list-style-type: none"> <li>• yyyy-MM-dd : 항상 휴일로 설정</li> <li>• yyyy-MM-01 : 매월 1일을 휴일로 설정</li> <li>• yyyy-01-01 : 매년 1월 1일을 휴일로 설정</li> <li>• 2007-01-01 : 2007년 1월 1일을 휴일로 설정</li> </ul>

■ **Type**

<b>Prototype</b>	public string Type{get;set;}
<b>Definition</b>	태스크 휴일 등록 시 지정한 타입(양력, 음력) solar : 양력 lunar : 음력

■ **Start**

<b>Prototype</b>	public int Start{get;set;}
<b>Definition</b>	"baseday"를 기준으로 며칠 전부터 휴일로 지정할지에 대한 시작 값

■ **End**

<b>Prototype</b>	public int End{get;set;}
<b>Definition</b>	"baseday"를 기준으로 며칠 후까지 휴일로 지정할지에 대한 값

**Sample : SchedulerTaskHoliday.cs**

```
using System;

namespace sample
{
    public class SchedulerTaskHoliday
    {
        public static void Main()
        {
            oz.framework.api.Scheduler scheduler = new
            oz.framework.api.Scheduler("127.0.0.1", 9521);

            oz.scheduler.holiday.TaskHolidayInfos holidayInfos =
            scheduler.GetTaskHolidayInfos();
            foreach(oz.scheduler.holiday.TaskHolidayInfo holidayInfo in
            holidayInfos.Values)
            {
                Console.WriteLine("Name : " + holidayInfo.Name);
                Console.WriteLine("Type : " + holidayInfo.Type);
            }
        }
    }
}
```

```
        Console.WriteLine("BaseDay : " + holidayInfo.BaseDay);
        Console.WriteLine("Start : " + holidayInfo.Start);
        Console.WriteLine("End : " + holidayInfo.End);
        Console.WriteLine();
    }

    oz.scheduler.holiday.TaskHolidayInfo anniversary = new
oz.scheduler.holiday.TaskHolidayInfo("a foundation day");
    anniversary.BaseDay = "yyyy-07-25";
    anniversary.Type = oz.scheduler.holiday.HolidayType.Solar;
    anniversary.Start = 0;
    anniversary.End = 0;
    scheduler.AddTaskHoliday(anniversary);

    scheduler.SaveTaskHoliday();
}
}
```

## Class TaskHolidayGroupInfo

### Constructor Summary

- TaskHolidayGroupInfo(string name)

### Property Summary

- string Name{get;}
- string[] Reference{get;}

### Method Summary

- void AddReference(params string[] names)
- void RemoveReferences(string[] names)
- void ClearReferences()

### Constructor Detail

---

Prototype	public TaskHolidayGroupInfo(string name)
-----------	--

---

Argument	<i>name</i>	이름 ※ 주의사항 : 이름에 ";" 문자가 포함될 수 없습니다.
----------	-------------	--

---

### Property Detail

- Name

---

Prototype	public string Name{get;}
-----------	--------------------------

---

Definition	태스크 휴일 그룹 이름
------------	--------------

---

## ■ References

---

**Prototype** public string[] References{get;}

---

**Definition** 태스크 휴일 그룹에서 참조할 Reference 이름의 배열

---

## Method Detail

### ■ AddReference

---

**Prototype** public void AddReference(string[] names)

---

**Definition** 태스크 휴일 그룹이 참조하는 태스크 휴일 이름을 등록합니다.

---

**Argument** *names*      태스크 휴일 이름  
(동일한 이름이 있을 경우 해당 값 무시함)

---

### ■ RemoveReferences

---

**Prototype** public void RemoveReferences(string[] names)

---

**Definition** 태스크 휴일 그룹에서 참조하는 태스크 휴일 이름을 삭제합니다.

---

**Argument** *names*      태스크 휴일 그룹에서 삭제할 태스크 휴일 이름의 배열

---

### ■ ClearReferences

---

**Prototype** public void clearReferences()

---

**Definition** 태스크 휴일 그룹에서 참조하는 태스크 휴일 이름을 모두 삭제합니다.

---

## Sample : SchedulerTaskHolidayGroup.cs

```
using System;

namespace sample
{
    /// <summary>
    /// TaskHolidayGroup
    /// </summary>
    public class SchedulerTaskHolidayGroup
    {
        public static void Main()
        {
            oz.framework.api.Scheduler scheduler = new
            oz.framework.api.Scheduler("127.0.0.1", 9521);
        }
    }
}
```

```
        oz.scheduler.holiday.TaskHolidayGroupInfo groupInfo = new
oz.scheduler.holiday.TaskHolidayGroupInfo("group3");
        groupInfo.AddReference("New Year's Day", "Lunar New Year's Day",
"Independence Movement Day");
        scheduler.AddTaskHolidayGroup(groupInfo);
        groupInfo.AddReference("Labor Day");
        scheduler.ModifyTaskHolidayGroup(groupInfo.Name, groupInfo);

        oz.scheduler.holiday.TaskHolidayGroupInfos holidayGroupInfos =
scheduler.GetTaskHolidayGroupInfos();
        foreach(oz.scheduler.holiday.TaskHolidayGroupInfo holidayGroupInfo
in holidayGroupInfos.values)
        {
            Console.WriteLine("Name : " + holidayGroupInfo.Name);
            foreach(string reference in holidayGroupInfo.References)
            {
                Console.Write(reference + "\t");
            }
            Console.WriteLine();
        }

        scheduler.DeleteTaskHolidayGroup(groupInfo.Name);

        scheduler.SaveTaskHoliday();
    }
}
```

### Ⅲ. 오즈 리파지토리 API

- Interface Repository
- Class RepositoryEx
- 오즈 리파지토리 구현

## Interface Repository

### Interface Summary

#### [NameSpace : oz.framework.repositoryex]

- 기본 리파지토리 인터페이스
  - **public interface IRepository**
  
- 아이템 관련 리파지토리 인터페이스
  - **public interface IRepositoryCategory**
  - **public interface IRepositoryItem**
  - **public interface IRepositoryHistory**
  - **public interface IRepositoryArchive**
  
- 사용자 관련 리파지토리 인터페이스
  - **public interface IRepositoryGroup**
  - **public interface IRepositoryUser**
  - **public interface IRepositoryMultiLoginUser**
  - **public interface OZRepositoryExternalLogin**
  
- Exception 관련 클래스
  - **public class OZRepositoryException extends Exception**

#### [패키지명 : oz.framework.repositoryex.auth]

- 권한 관련 인터페이스
  - **public interface IRepositoryAuthorityToGroupCategory**
  - **public interface IRepositoryGroupAuthorityToItem**
  - **public interface IRepositoryUserAuthorityToCategory**
  - **public interface IRepositoryUserAuthorityToItem**

#### [패키지명 : oz.framework.repositoryex.info]

- 리파지토리 정보(bean) 관련 인터페이스
  - **public interface ICategoryInfo**

- **public interface IItemInfo**
- **public interface IHistoryInfo**
- **public interface IGroupInfo**
- **public interface IUserInfo**
- **public interface ILoginInfo**
- **public interface ISearchResult**

## Interface Method Summary

### [IRepository]

- **IStringDictionary Property{set;}**
- **int Version{get;}**
- **void Open ()**
- **void Close()**
- **object Login(string userName, string password, ILoginInfo loginInfo, HttpContext ctx)**
- **void UpdateSessionState(object session, SessionState state)**
- **bool Logout(object session, string username, ILoginInfo loginInfo, HttpContext ctx)**
- **IRepositoryCategory Category{get;}**
- **IRepositoryItem Item{get;}**
- **IRepositoryHistory History{get;}**
- **IRepositoryGroup Group{get;}**
- **IRepositoryUser User{get;}**
- **IRepositoryMultiLoginUser MultiLoginUser{get;}**
- **IRepositoryGroupAuthorityToCategory GroupAuthorityToCategory{get;}**
- **IRepositoryGroupAuthorityToItem GroupAuthorityToItem{get;}**
- **IRepositoryUserAuthorityToCategory UserAuthorityToCategory{get;}**
- **IRepositoryUserAuthorityToItem UserAuthorityToItem{get;}**

### [IRepositoryCategory]

- **IRepository Repository{get;set;}**
- **CategoryAccess AccessType{get;}**
- **string[] CreateItems(object session, string[] itemNames, string[]**

descriptions, string[] categoryIDs, Stream[] items, string comment, OZErrorCode[] errCodes, string[] errMsgs)

- string[] CreateCategories(object session, string[] categoryNames, string[] parentCategoryIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- string ModifyCategoryName(object session, string categoryID, string newCategoryName, string comment)
- bool[] DeleteCategories(object session, string[] categoryIDs, bool[] toBeDestroyed, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- bool[] UndeleteCategories(object session, string[] categoryIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- int GetItemCount(object session, string categoryID)
- IItemInfo[] GetItemInfos(object session, string categoryID)
- string GetCategoryID(object session, string itemID)
- ICategoryInfo GetCategoryInfo(object session, string categoryID)
- IItemInfo[] GetDeletedItemInfos(object session, string categoryID)
- ICategoryInfo[] GetCategoryInfos(object session, string categoryID)
- bool TransferItems(object session, string[] itemIDs, string categoryID)
- bool TransferCategory(object session, string categoryID, string targetCategoryID)

### **[IRepositoryItem]**

- IRepository Repository{get;set;}
- ItemAccess AccessType{get;}
- string[] CreateItems(object session, string[] itemNames, string[] descriptions, Stream[] items, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- string ModifyItemName(object session, string itemID, string newItemName, string comment)
- bool[] DeleteItems(object session, string[] itemIDs, bool[] toBeDestroyed, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- bool[] UndeleteItems(object session, string[] itemIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)
- bool ModifyItemDescription(object session, string itemID, string description)
- IItemInfo GetItemInfo(object session, string itemID)

- **bool HasItem(object session, string itemID)**
- **Stream[] GetItems(object session, string[] itemIDs, OZErrorCode[] errCodes, string[] errMsgs)**
- **Stream[] GetItems(object session, string[] itemIDs, long[] modifiedTimes, ItemOptions[] options, OZErrorCode[] errCodes, string[] errMsgs)**
- **Stream[] CheckOut(object session, string[] itemIDs, string[] localCheckOutFolders, long[] localFileTimes, ItemOptions[] options, OZErrorCode[] errCodes, string[] errMsgs)**
- **bool[] CheckIn(object session, string[] itemIDs, Stream[] items, string comment, bool[] keepCheckOut, OZErrorCode[] errCodes, string[] errMsgs)**
- **Stream[] UndoCheckOut(object session, string[] itemIDs, ItemOptions[] options, OZErrorCode[] errCodes, string[] errMsgs)**
- **bool[] IsCheckOutUser(object session, string[] itemIDs)**

#### **[IRepositoryHistory]**

- **IRepository Repository{get;set;}**
- **HistoryAccess AccessType{get;}**
- **bool RollBack(object session, string itemID, int version, string comment)**
- **Stream GetItem(object session, string itemID, int version)**
- **IHistoryInfo[] GetHistoryInfos(object session, string itemID)**
- **IHistoryInfo[] GetDeletedItemHistoryInfos(object session, string itemID)**
- **bool RemoveHistory(object session, string itemID, int version)**

#### **[IRepositoryGroup]**

- **IRepository Repository{get;set;}**
- **GroupAccess AccessType{get;}**
- **string CreateGroup(object session, string groupName, string parentGroupID, string description)**
- **string ModifyGroupName(object session, string groupID, string groupName)**
- **bool ModifyGroupDescription(object session, string groupID, string description)**
- **bool DeleteGroup(object session, string groupID)**
- **string CreateUser(object session, string userName, string password, string groupID, string description)**

- **IUserInfo[] GetUserInfos(object session, string groupID)**
- **IGroupInfo GetGroupInfo(object session, string groupID)**
- **IGroupInfo[] GetSubGroupInfos(object session, string groupID)**
- **IGroupInfo GetParentGroupInfo(object session, string groupID)**
- **string GetGroupID(object session, string userID)**
- **bool AddGroupAdministrator(object session, string userID, string groupID)**
- **bool RemoveGroupAdministrator(object session, string userID, string groupID)**
- **bool IsGroupAdministrator(object session, string userID, string groupID)**
- **IUserInfo[] GetAdministrators(object session, string groupID)**
- **bool TransferUser(object session, string userID, string newGroupID)**
- **bool TransferGroup(object session, string groupID, string targetGroupID)**

#### **[IRepositoryUser]**

- **IRepository Repository{get;set;}**
- **UserAccess AccessType{get;}**
- **string CreateUser(object session, string userName, string password, string description)**
- **string ModifyName(object session, string userID, string newUserName)**
- **bool ModifyPassword(object session, string userID, string oldPassword, string newPassword)**
- **bool ModifyDescription(object session, string userID, string description)**
- **bool DeleteUser(object session, string userID)**
- **IUserInfo GetUserInfo(object session, string userID)**
- **bool CheckPassword(object session, string userID, string password)**
- **IUserInfo[] GetUserInfos(object session)**
- **bool IsAdministrator(object session, string userID)**
- **bool AddAdmin(object sID, string uID, string client\_ip)**
- **bool RemoveAdmin(object sID, string uID, string client\_ip)**
- **bool ModifyAllowip(object sID, String uID, String allowip, string client\_ip)**
- **bool IsExternalLogin()**

#### **[IRepositoryMultiLoginUser]**

- **IRepository Repository{get;set;}**
- **MultiLoginAccess AccessType{get;}**

- void DisableLogin(object session, string userID)
- void EnableLogin(object session, string userID)
- bool IsLoginEnabled(object session, string userID)

#### [OZRepositoryExternalLogin]

- string CreateUser(oz.framework.log.OZLog log, string id, string pwd)
- string Login(oz.framework.log.OZLog log, string id, string pwd)

#### [IRepositoryArchive]

- ArchiveAccess Acesstype{get;}
- Stream Distribute(object session, string categoryID, bool isRecursive)
- Stream Distribute(object session, string[] itemIDs)
- Bool Upload(object session, string categoryID, Stream @in)
- Stream DistributeGroupUser(object session, string groupID, bool isRecursive);

#### [IRepositoryGroupAuthorityToCategory]

- IRepository Repository{get;set;}
- AuthorityAccess AccessType{get;}
- bool Modify(object session, string categoryID, string groupID, Authority permission)
- Authority GetAuthority(object session, string groupID, string categoryID)
- ICategoryInfo[] GetCategoryInfos(object session, string groupID, string categoryID, Authority permission)
- IGroupInfo[] GetGroupInfos(object session, string categoryID, Authority permission)
- IItemInfo[] GetItemInfos(object session, string groupID, string categoryID, Authority permission)

#### [IRepositoryGroupAuthorityToItem]

- IRepository Repository{get;set;}
- AuthorityAccess AccessType{get;}
- bool Modify(object session, string groupID, string itemID, Authority permission)

- **Authority GetAuthority(object session, string groupID, string itemID)**
- **IGroupInfo[] GetGroupInfos(object session, string itemID, Authority permission)**
- **IItemInfo[] GetItemInfos(object session, string groupID, Authority permission)**

#### **[IRepositoryUserAuthorityToCategory]**

- **IRepository Repository{get;set;}**
- **AuthorityAccess AccessType{get;}**
- **bool Modify(object session, string userID, string categoryID, Authority permission)**
- **Authority GetAuthority(object session, string userID, string categoryID)**
- **ICategoryInfo[] GetCategoryInfos(object session, string userID, string categoryID, Authority permission)**
- **IUserInfo[] GetUserInfos(object session, string categoryID, Authority permission)**
- **IItemInfo[] GetItemInfos(object session, string userID, string categoryID, Authority permission)**

#### **[IRepositoryUserAuthorityToItem]**

- **IRepository Repository{get;set;}**
- **AuthorityAccess AccessType{get;}**
- **bool Modify(object session, string userID, string itemID, Authority permission)**
- **Authority GetAuthority(object session, string userID, string itemID)**
- **IUserInfo[] GetUserInfos(object session, string itemID, Authority permission)**
- **IItemInfo[] GetItemInfos(object session, string userID, Authority permission)**

### **Interface Method Detail**

#### **[IRepository Interface Method Summary]**

- **Property**

<b>Prototype</b>	<code>oz.util.IStringDictionary Property{set;}</code>
<b>Definition</b>	리포지토리 설정을 변경합니다. "repository.properties"에 설정되어 있는 값을 변경할 수 있으며, 설정할 수 있는 Key와 값은 아래 표를 참조하시기 바랍니다.
<b>Argument</b>	<i>IStringDictionary</i> 리포지토리 설정 속성
<b>■ Version</b>	
<b>Prototype</b>	<code>int Version{get;}</code>
<b>Definition</b>	리포지토리 버전을 정의한 후 정의한 버전을 가져옵니다.
<b>■ Open</b>	
<b>Prototype</b>	<code>void Open()</code>
<b>Definition</b>	서버 구동 시 리포지토리를 엽니다.
<b>■ Close</b>	
<b>Prototype</b>	<code>void Close()</code>
<b>Definition</b>	서버 종료 시 리포지토리를 종료합니다.
<b>■ Login</b>	
<b>Prototype</b>	<code>object Login(string username, string password, ILoginInfo loginInfo, HttpContext ctx)</code>
<b>Definition</b>	클라이언트를 호출할 때 해당 메소드를 호출하여 로그인한 세션 정보를 가져옵니다.
<b>Argument</b>	<i>UserName</i> 접속한 사용자 이름
	<i>password</i> 접속한 사용자의 패스워드
	<i>loginInfo</i> 로그인 정보 ※ 참고사항 : 자세한 로그인 정보는 ILoginInfo 인터페이스를 참조하시기 바랍니다.
<i>ctx</i>	HTTP 접속 시 현재 컨텍스트 ※ 참고사항 : 윈도우즈 서비스 형태의 서버인 경우 null 값을 설정합니다.
<b>■ UpdateSessionState</b>	
<b>Prototype</b>	<code>void UpdateSessionState(object session, SessionState state)</code>
<b>Definition</b>	클라이언트를 호출할 때 해당 메소드를 호출하여 로그인한 세션 정보를 가져옵니다.
<b>Argument</b>	<i>session</i> 세션 ID

---

	서버 상태
<i>state</i>	<ul style="list-style-type: none"> <li>• None</li> <li>• BareServer</li> <li>• FromServer</li> </ul>

---

■ Logout

---

<b>Prototype</b>	<code>bool Logout(object session, string userName, ILoginInfo loginInfo, HttpContext ctx)</code>
<b>Definition</b>	지정한 사용자 ID에 해당하는 사용자를 로그 아웃 즉, 접속 해제시키며 로그아웃 성공 여부를 가져옵니다.
	<i>session</i> 세션 ID
	<i>userName</i> 사용자 이름
<b>Argument</b>	<i>loginInfo</i> 로그인 정보 (자세한 정보는 해당 인터페이스 참조)
	ctx HTTP 접속 시 현재 컨텍스트 ※ 참고사항 : 윈도우즈 서비스 형태의 서버인 경우 null 값을 설정합니다.

---

■ Category

---

<b>Prototype</b>	<code>IRepositoryCategory Category{get;}</code>
<b>Definition</b>	카테고리 인터페이스를 구현한 경우 구현된 객체를 가져옵니다.

---

■ Item

---

<b>Prototype</b>	<code>IRepositoryItem Item{get;}</code>
<b>Definition</b>	아이템 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

■ ItemHistory

---

<b>Prototype</b>	<code>IRepositoryHistory ItemHistory{get;}</code>
<b>Definition</b>	아이템 히스토리 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

■ Group

---

<b>Prototype</b>	<code>IRepositoryGroup Group{get;}</code>
<b>Definition</b>	그룹 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

■ User

---

<b>Prototype</b>	<code>IRepositoryUser User{get;}</code>
<b>Definition</b>	유저 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

### ■ MultiLoginUser

---

**Prototype** IRepositoryMultiLoginUser MultiLoginUser{get;}

**Definition** 멀티 로그인 유저 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

### ■ GroupAuthorityToCategory

---

**Prototype** IRepositoryGroupAuthorityToCategory  
GroupAuthorityToCategory{get;}

**Definition** 카테고리에 대한 그룹의 권한 인터페이스를 구현한 경우 해당 구현된 클래스를 가져옵니다.

---

### ■ GroupAuthorityToItem

---

**Prototype** IRepositoryGroupAuthorityToItem GroupAuthorityToItem{get;}

**Definition** 아이템에 대한 그룹의 권한 인터페이스를 구현한 경우 해당 구현된 객체를 가져옵니다.

---

### ■ UserAuthorityToCategory

---

**Prototype** IRepositoryUserAuthorityToCategory  
UserAuthorityToCategory{get;}

**Definition** 카테고리에 대한 사용자의 권한 인터페이스를 구현한 경우 해당 구현된 클래스를 가져옵니다.

---

### ■ UserAuthorityToItem

---

**Prototype** IRepositoryUserAuthorityToItem UserAuthorityToItem

**Definition** 아이템에 대한 사용자의 권한 인터페이스를 구현한 경우 해당 구현된 클래스를 가져옵니다.

---

## [IRepositoryCategory Interface Method Summary]

### ■ AccessType

---

**Prototype** CategoryAccess AccessType{get;}

---

카테고리 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

None = 0x00000000  
 All = 0x7FFFFFFF  
 CreateCategories = 0x00000002  
 ModifyCategoryName = 0x00000004  
 DeleteCategories = 0x00000008  
 UndeleteCategories = 0x00000010  
 HasItem = 0x00000020  
**Definition** GetItemCount = 0x00000040  
 GetItemInfos = 0x00000080  
 GetCategoryID = 0x00000100  
 GetCategoryInfo = 0x00000200  
 GetDeletedItemInfos = 0x00000400  
 SearchItem = 0x00000800,  
 GetCategoryInfos = 0x00001000  
 TransferItems = 0x00002000  
 TransferCategory = 0x00004000

■ **Repository**

**Prototype** IRepository Repository{get;set;}

**Definition** IRepository 인터페이스를 구현한 객체를 가져옵니다.

■ **CreateItems**

**Prototype** string[] CreateItems(object session, string[] itemNames, string[] descriptions, string[] categoryIDs, Stream[] items, string comment, OZErrorCode[] errCodes, string[] errMsgs)

**Definition** 새로운 아이템을 생성하고 생성된 아이템의 ID를 반환합니다.

<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemNames</i>	새로 생성할 아이템의 ID
<b>Argument</b>	<i>descriptions</i>	새로 생성할 아이템의 설명 내용
	<i>categoryIDs</i>	새로 생성할 아이템이 카테고리 ID
	<i>items</i>	새로 생성할 아이템의 입력 스트림
	<i>comment</i>	주석문
	<i>errCodes</i>	에러 코드
	<i>errMsgs</i>	에러 메시지

■ **CreateCategories**

**Prototype** string[] CreateCategories(object session, string[] categoryNames, string[] parentCategoryIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)

<b>Definition</b>	카테고리를 새로 생성하고 생성된 카테고리의 ID를 반환합니다.	
	<i>session</i>	세션 ID
	<i>categoryNames</i>	생성할 카테고리의 이름
<b>Argument</b>	<i>parentCategoryIDs</i>	생성할 카테고리의 상위 카테고리 ID
	<i>comment</i>	주석문
	<i>errCodes</i>	에러 코드
	<i>errMsgs</i>	에러 메시지

#### ■ ModifyCategoryName

<b>Prototype</b>	string ModifyCategoryName(object session, string categoryID, string newCategoryName, string comment)	
<b>Definition</b>	지정한 카테고리 ID에 해당하는 카테고리의 이름을 변경합니다.	
	<i>session</i>	세션 ID
<b>Argument</b>	<i>categoryID</i>	이름을 변경할 카테고리의 ID
	<i>newCategoryName</i>	변경할 카테고리 이름
	<i>comment</i>	주석문

#### ■ DeleteCategories

<b>Prototype</b>	bool[] DeleteCategories(object session, string[] categoryIDs, bool[] toBeDestroyed, string comment, OZErrorCode[] errCodes, string[] errMsgs)	
<b>Definition</b>	지정한 카테고리 ID에 해당하는 카테고리를 삭제합니다.	
	<i>session</i>	세션 ID
	<i>categoryIDs</i>	삭제할 카테고리의 ID
<b>Argument</b>	<i>toBeDestroyed</i>	카테고리를 영구 삭제할 지 여부
	<i>comment</i>	주석문
	<i>errorCodes</i>	에러 코드
	<i>errorMsgs</i>	에러 메시지

#### ■ UndeleteCategories

<b>Prototype</b>	bool[] UndeleteCategories(object session, string[] categoryIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)	
<b>Definition</b>	삭제된 카테고리를 복원하고 복원 성공 여부를 가져옵니다. ※ 주의사항 : 서버 버전이 2007년 09월 01일 이전 버전일 경우 해당 메소드가 지원되지 않습니다. ※ 참고사항 : 카테고리 삭제 시 "isDestroys=false"로 설정하였을 경우 삭제된	

카테고리의 복원이 가능합니다.

	<i>session</i>	세션 ID
	<i>CategoryIDs</i>	복원할 카테고리의 ID
<b>Argument</b>	<i>comment</i>	주석문
	<i>errorCode</i>	에러 코드
	<i>errorMsg</i>	에러 메시지

#### ■ GetItemCount

<b>Prototype</b>	int GetItemCount(object session, string categoryID)	
<b>Definition</b>	지정한 카테고리에 속해 있는 모든 아이템의 개수를 가져옵니다.	
	<i>session</i>	세션 ID
<b>Argument</b>	<i>categoryID</i>	아이템의 개수를 얻을 카테고리의 ID

#### ■ GetItemInfos

<b>Prototype</b>	IItemInfo[] GetItemInfos(object session, string categoryID)	
<b>Definition</b>	지정한 카테고리에 속해 있는 모든 아이템 정보를 가져옵니다.	
	<i>session</i>	세션 ID
<b>Argument</b>	<i>categoryID</i>	아이템의 정보를 얻을 카테고리의 ID

#### ■ GetCategoryInfos

<b>Prototype</b>	ICategoryInfo[] GetCategoryInfos(object session, string categoryID)	
<b>Definition</b>	지정한 카테고리의 정보를 가져옵니다.	
	<i>session</i>	세션 ID
<b>Argument</b>	<i>categoryID</i>	정보를 가져올 카테고리 ID

#### ■ GetCategoryID

<b>Prototype</b>	string GetCategoryID(object session, string itemID)	
<b>Definition</b>	지정한 아이템이 존재하는 카테고리의 ID를 반환합니다.	
	<i>session</i>	세션 ID
<b>Argument</b>	<i>itemID</i>	카테고리의 ID를 얻을 아이템 ID

### ■ GetCategoryInfo

<b>Prototype</b>	ICategoryInfo GetCategoryInfo(object session, string categoryID)	
<b>Definition</b>	지정한 카테고리의 정보를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>categoryID</i>	정보를 가져올 카테고리 ID

### ■ GetDeletedItemInfos

<b>Prototype</b>	IItemInfo[] GetDeletedItemInfos(object session, string categoryID)	
<b>Definition</b>	지정한 카테고리의 삭제된 아이템 정보를 가져옵니다. ※ 참고사항 : "toBeDestroyed=false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>categoryID</i>	삭제된 아이템 정보를 가져올 카테고리 ID

### ■ TransferItems

<b>Prototype</b>	bool TransferItems(object session, string[] itemIDs, string targetCategoryID)	
<b>Definition</b>	지정한 아이템의 카테고리를 이동하고 카테고리 이동 성공 여부를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemIDs</i>	카테고리를 이동할 아이템 ID
	<i>targetCategoryID</i>	이동할 카테고리 ID

### ■ TransferCategory

<b>Prototype</b>	bool TransferCategory(object session, string categoryID, string targetCategoryID)	
<b>Definition</b>	지정한 카테고리를 다른 카테고리로 이동하고 카테고리 이동 성공 여부를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>categoryID</i>	카테고리를 이동할 카테고리 ID
	<i>targetCategoryID</i>	이동할 카테고리 ID

## [IOZRepositoryItem]

### ■ AccessType

<b>Prototype</b>	CategoryAccess AccessType{get;}
------------------	---------------------------------

아이템 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

None = 0x00000000  
 All = 0x7FFFFFFF  
 CreateItems = 0x00000001  
 ModifyItemName = 0x00000002  
 DeleteItems = 0x00000004  
 UndeleteItems = 0x00000008  
**Definition** ModifyItemDescription = 0x00000010  
 GetItemInfo = 0x00000040  
 HasItem = 0x00000020  
 GetItemsUnconditionally = 0x00000080,  
 GetItems = 0x00000100  
 CheckOut = 0x00000200,  
 CheckIn = 0x00000400  
 UndoCheckOut = 0x00000800  
 IsCheckOutUser = 0x00001000

■ **Repository**

**Prototype** IRepository Repository{get;set;}

**Definition** IRepostiroy 인터페이스를 구현한 객체를 가져옵니다.

■ **CreateItems**

**Prototype** string[] CreateItems(object session, string[] itemNames, string[] descriptions, Stream[] items, string comment, OZErrorCode[] errCodes, string[] errMsgs)

**Definition** 새로운 아이템을 생성하고 생성된 아이템의 ID를 반환합니다.

<i>session</i>	세션 ID
<i>itemNames</i>	새로 생성할 아이템의 ID
<i>descriptions</i>	새로 생성할 아이템의 설명 내용
<b>Argument</b> <i>items</i>	새로 생성할 아이템의 입력 스트림
<i>comment</i>	주석문
<i>errCodes</i>	에러 코드
<i>errMsgs</i>	에러 메시지

■ **ModifyItemName**

**Prototype** string ModifyItemName(object session, string itemID, string newItemName, string comment)

**Definition** 지정한 아이템 ID에 해당하는 아이템 이름을 변경하고 변경된 아이템 ID를 반환합니다.

<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemID</i>	이름을 변경할 아이템 ID
	<i>newItemName</i>	변경할 아이템 이름
	<i>comment</i>	주석문

#### ■ DeleteItems

<b>Prototype</b>	<code>bool[] DeleteItems(object session, string[] itemIDs, bool[] toBeDestroyed, string comment, OZErrorCode[] errCodes, string[] errMsgs)</code>	
<b>Definition</b>	지정한 아이템을 리파지토리에서 삭제하고 아이템 삭제 성공 여부를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemIDs</i>	삭제할 아이템의 ID
	<i>toBeDestroyed</i>	아이템을 영구 삭제할 지 여부
	<i>comment</i>	주석문
	<i>errCodes</i>	에러 코드
	<i>errMsgs</i>	에러 메시지

#### ■ UnDeleteItems

<b>Prototype</b>	<code>bool[] UndeleteItems(object session, string[] itemIDs, string comment, OZErrorCode[] errCodes, string[] errMsgs)</code>	
<b>Definition</b>	삭제된 아이템을 복원하고 복원 성공 여부를 가져옵니다. ※ 참고사항 : 아이템 삭제 시 "toBeDestroyed =false"로 설정하였을 경우 삭제된 아이템의 복원이 가능합니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemIDs</i>	복원할 아이템 ID
	<i>comment</i>	주석문
	<i>errCodes</i>	에러 코드
	<i>errMsgs</i>	에러 메시지

#### ■ ModifyItemDescription

<b>Prototype</b>	<code>bool ModifyItemDescription(object session, string itemID, string description)</code>	
<b>Definition</b>	지정한 아이템 ID에 해당하는 아이템 설명을 변경하고 변경 성공 여부를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemID</i>	변경할 아이템 ID

---

*description*    변경할 내용

---

#### ■ GetItemInfo

---

**Prototype**    IItemInfo GetItemInfo(object session, string itemID)

---

**Definition**    지정된 아이템 ID에 해당하는 아이템 정보를 가져옵니다.

---

**Argument**    *session*            세션 ID

---

*itemID*            아이템 정보를 가져올 아이템 ID

---

#### ■ HasItem

---

**Prototype**    bool HasItem(object session, string itemID)

---

**Definition**    지정된 아이템의 존재 여부를 가져옵니다.

---

**Argument**    *session*            세션 ID

---

*itemID*            아이템 ID

---

#### ■ GetItems

---

Stream[] GetItems(object session, string[] itemIDs, OZErrorCode[] errCodes, string[] errMsgs)

---

**Prototype**    Stream[] GetItems(object session, string[] itemIDs, long[] modifiedTimes, ItemOptions[] options, OZErrorCode[] errCodes, string[] errMsgs)

---

**Definition**    지정된 아이템을 가져온 후 아이템 스트림을 가져옵니다.

---

**Argument**    *session*            세션 ID

---

*itemIDs*            가져올 아이템 ID

---

**Argument**    *modifiedTimes*    클라이언트 시간

---

*options*            에러 메시지 처리에 대한 옵션

---

*errCodes*            에러 코드

---

*errMsgs*            에러 메시지

---

#### ■ CheckOut

---

**Prototype**    Stream[] CheckOut(object session, string[] itemIDs, string[] localCheckOutFolders, long[] localFileTimes, OZErrorCode[] errCodes, string[] errMsgs)

---

**Definition**    지정된 아이템을 지정된 사용자 ID로 체크 아웃할 폴더에 체크 아웃을 하고 로컬 파일의 시간이 아이템 시간 보다 작을 경우 아이템 스트림을 가져옵니다.

---

**Argument**    *session*            세션 ID

---

*itemIDs*            체크 아웃할 아이템의 ID

---

<i>localCheckoutFolders</i>	체크 아웃할 로컬 폴더 이름
<i>localFileTimes</i>	체크 아웃할 아이템의 로컬 파일 시간
<i>errCodes</i>	에러 코드
<i>errMsgs</i>	에러 메시지

#### ■ CheckIn

<b>Prototype</b>	<code>bool[] CheckIn(object session, string[] itemIDs, Stream[] items, string comment, bool[] keepCheckout, OZErrorCode[] errCodes, string[] errMsgs)</code>														
<b>Definition</b>	지정한 아이템을 지정한 사용자 ID로 체크인하고 체크인 성공 여부를 가져옵니다.														
<b>Argument</b>	<table border="1"> <tr> <td><i>session</i></td> <td>세션 ID</td> </tr> <tr> <td><i>itemIDs</i></td> <td>체크인할 아이템 ID</td> </tr> <tr> <td><i>items</i></td> <td>체크인할 아이템 스트림</td> </tr> <tr> <td><i>comment</i></td> <td>주석문</td> </tr> <tr> <td><i>keepCheckout</i></td> <td>체크 아웃 상태를 유지할지 여부</td> </tr> <tr> <td><i>errCodes</i></td> <td>에러 코드</td> </tr> <tr> <td><i>errMsgs</i></td> <td>에러 메시지</td> </tr> </table>	<i>session</i>	세션 ID	<i>itemIDs</i>	체크인할 아이템 ID	<i>items</i>	체크인할 아이템 스트림	<i>comment</i>	주석문	<i>keepCheckout</i>	체크 아웃 상태를 유지할지 여부	<i>errCodes</i>	에러 코드	<i>errMsgs</i>	에러 메시지
<i>session</i>	세션 ID														
<i>itemIDs</i>	체크인할 아이템 ID														
<i>items</i>	체크인할 아이템 스트림														
<i>comment</i>	주석문														
<i>keepCheckout</i>	체크 아웃 상태를 유지할지 여부														
<i>errCodes</i>	에러 코드														
<i>errMsgs</i>	에러 메시지														

#### ■ UndoCheckout

<b>Prototype</b>	<code>Stream[] UndoCheckout(object session, string[] itemIDs, bool[] replace, OZErrorCode[] errCodes, string[] errMsgs)</code>										
<b>Definition</b>	지정한 아이템을 체크 아웃 취소합니다.										
<b>Argument</b>	<table border="1"> <tr> <td><i>session</i></td> <td>세션 ID</td> </tr> <tr> <td><i>itemIDs</i></td> <td>체크 아웃 취소할 아이템의 ID</td> </tr> <tr> <td><i>replaces</i></td> <td>로컬 파일 변경 여부</td> </tr> <tr> <td><i>errCodes</i></td> <td>에러 코드</td> </tr> <tr> <td><i>errMsgs</i></td> <td>에러 메시지</td> </tr> </table>	<i>session</i>	세션 ID	<i>itemIDs</i>	체크 아웃 취소할 아이템의 ID	<i>replaces</i>	로컬 파일 변경 여부	<i>errCodes</i>	에러 코드	<i>errMsgs</i>	에러 메시지
<i>session</i>	세션 ID										
<i>itemIDs</i>	체크 아웃 취소할 아이템의 ID										
<i>replaces</i>	로컬 파일 변경 여부										
<i>errCodes</i>	에러 코드										
<i>errMsgs</i>	에러 메시지										

#### ■ IsCheckoutUser

<b>Prototype</b>	<code>bool[] IsCheckoutUser(object session, string[] itemIDs)</code>				
<b>Definition</b>	지정한 사용자가 지정한 아이템을 체크 아웃했는지 여부를 확인합니다				
<b>Argument</b>	<table border="1"> <tr> <td><i>session</i></td> <td>세션 ID</td> </tr> <tr> <td><i>itemIDs</i></td> <td>아이템 ID</td> </tr> </table>	<i>session</i>	세션 ID	<i>itemIDs</i>	아이템 ID
<i>session</i>	세션 ID				
<i>itemIDs</i>	아이템 ID				

**[IRepositoryHistory]**■ **AccessType**


---

**Prototype** HistoryAccess AccessType{get;}
 

---

아이템 히스토리 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

None = 0x00000000

All = 0x7FFFFFFF

**Definition** RollBack = 0x00000001
 

---

GetItem = 0x00000002

GetHistoryInfos = 0x00000004

GetDeletedItemHistoryInfos = 0x00000008

RemoveHistory = 0x00000010

---

■ **Repository**


---

**Prototype** IRepository Repository{get;set;}
 

---

**Definition** IRepository 인터페이스를 구현한 객체를 가져옵니다.
 

---

■ **RollBack**


---

**Prototype** bool RollBack(object session, string itemID, int version, string comment)
 

---

**Definition** 지정한 아이템을 지정한 버전으로 복원시킵니다.
 

---

*session* 세션 ID

*itemID* 복원시킬 아이템의 ID

**Argument** *version* 복원시킬 버전
 

---

*comment* 주석문

---

■ **GetItem**


---

**Prototype** Stream GetItem(object session, string itemID, int version)
 

---

**Definition** 지정한 아이템 ID에 해당하는 아이템 중 지정한 버전의 아이템을 가져옵니다.
 

---

*session* 세션 ID

**Argument** *itemID* 가져올 아이템의 ID
 

---

*version* 가져올 아이템의 버전

---

■ **GetHistoryInfos**


---

**Prototype** IHistoryInfo[] GetHistoryInfos(object session, string itemID)
 

---

**Definition** 지정한 아이템의 히스토리 정보를 가져옵니다.
 

---

<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemID</i>	히스토리 정보를 가져올 아이템의 ID

■ **GetDeleteHistoryItemInfo**

<b>Prototype</b>	IHistoryInfo[] GetDeletedItemHistoryInfos(object session, string itemID)	
<b>Definition</b>	삭제된 히스토리 리스트를 가져옵니다. ※ 참고사항 : "toBeDestroyed =false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemIDs</i>	히스토리 정보를 가져올 아이템의 ID

■ **RemoveHistory**

<b>Prototype</b>	bool RemoveHistory(object session, string itemID, int version)	
<b>Definition</b>	지정한 아이템에 대해 특정 버전의 히스토리를 삭제합니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>itemID</i>	히스토리를 삭제할 아이템의 ID
	<i>version</i>	히스토리를 삭제할 버전

**[IRepositoryGroup]**

■ **AccessType**

<b>Prototype</b>	GroupAccess AccessType{get;}
<b>Definition</b>	<p>그룹 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.</p> <p>None = 0x00000000                  All = 0x7FFFFFFF                  CreateGroup = 0x00000001                  ModifyGroupName = 0x00000002                  ModifyGroupDescription = 0x00000004                  DeleteGroup = 0x00000008,                  CreateUser = 0x00000010                  TransferUser = 0x00000020                  GetUserInfos = 0x00000040                  GetGroupInfo = 0x00000080                  GetSubGroupInfos = 0x00000100                  GetParentGroupInfo = 0x00000200                  GetGroupID = 0x00000400                  AddGroupAdministrator = 0x00000800                  RemoveGroupAdministrator = 0x00001000</p>

IsGroupAdministrator = 0x00002000

GetAdministrators = 0x00004000

TransferGroup = 0x00008000

### ■ Repository

**Prototype** IRepository Repository{get;set;}

**Definition** IRepostiroy 인터페이스를 구현한 객체를 가져옵니다.

### ■ CreateGroup

**Prototype** string CreateGroup(object session, string groupName, string parentGroupID, string description)

**Definition** 새로운 그룹을 생성하고 생성된 그룹의 ID를 반환합니다.

<i>session</i>	세션 ID
<i>groupName</i>	새로 생성할 그룹의 이름
<i>parentGroupID</i>	새로 생성할 그룹의 상위 그룹 ID
<i>description</i>	그룹에 대한 설명 내용

### ■ ModifyGroupName

**Prototype** string ModifyGroupName(object session, string groupID, string groupName)

**Definition** 지정한 그룹 ID에 해당하는 그룹의 이름을 변경하고 변경된 그룹 ID를 반환합니다.

<i>session</i>	세션 ID
<i>groupID</i>	그룹의 이름을 변경할 그룹 ID
<i>groupName</i>	변경할 그룹의 이름

### ■ ModifyGroupDescription

**Prototype** bool ModifyGroupDescription(object session, string groupID, string description)

**Definition** 지정한 그룹 ID에 해당하는 그룹의 설명을 변경하고 변경 성공 여부를 반환합니다.

<i>session</i>	세션 ID
<i>groupID</i>	그룹의 설명을 변경할 그룹 ID
<i>description</i>	변경할 그룹의 설명

### ■ DeleteGroup

<b>Prototype</b>	bool DeleteGroup(object session, string groupID)	
<b>Definition</b>	지정한 그룹 ID에 해당하는 그룹을 삭제하고 삭제 성공 여부를 반환합니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	삭제할 그룹 ID

### ■ CreateUser

<b>Prototype</b>	string CreateUser(object session, string userName, string password, string groupID, string description)	
<b>Definition</b>	지정한 그룹에 새로운 사용자 ID를 생성하고 생성된 사용자 ID를 반환합니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>userName</i>	사용자 이름
	<i>password</i>	패스워드
	<i>groupID</i>	그룹ID
	<i>description</i>	사용자에 대한 설명 내용

### ■ GetUserInfos

<b>Prototype</b>	IUserInfo[] GetUserInfos(object session, string groupID)	
<b>Definition</b>	지정한 그룹 ID에 등록되어 있는 모든 사용자의 정보를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	사용자의 정보를 가져올 그룹 ID

### ■ GetGroupInfo

<b>Prototype</b>	IGroupInfo GetGroupInfo(object session, string groupID)	
<b>Definition</b>	지정한 그룹 ID에 해당하는 그룹의 정보를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	그룹 ID

### ■ GetSubGroupInfos

<b>Prototype</b>	IGroupInfo[] GetSubGroupInfos(object session, string groupID)	
<b>Definition</b>	지정한 그룹의 하위 그룹 정보를 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	하위 그룹의 정보를 가져올 그룹 ID

### ■ GetParentGroupInfo

**Prototype** IGroupInfo GetParentGroupInfo(object session, string groupID)

**Definition** 지정한 그룹의 상위 그룹의 정보를 가져옵니다.

**Argument**

<i>session</i>	세션 ID
<i>groupID</i>	상위 그룹의 정보를 가져올 그룹 ID

### ■ GetGroupID

**Prototype** string GetGroupID(object session, string userID)

**Definition** 지정한 사용자가 속해 있는 그룹의 정보를 가져옵니다.

**Argument**

<i>session</i>	세션 ID
<i>userID</i>	그룹 정보를 가져올 사용자 ID

### ■ AddGroupAdministrator

**Prototype** bool AddGroupAdministrator(object session, string userID, string groupID)

**Definition** 지정한 그룹의 그룹 관리자를 추가하고 추가 성공 여부를 반환합니다.

**Argument**

<i>session</i>	세션 ID
<i>userID</i>	그룹 관리자로 추가할 사용자 ID
<i>groupID</i>	그룹 관리자를 추가할 그룹 ID

### ■ RemoveGroupAdministrator

**Prototype** bool RemoveGroupAdministrator(object session, string userID, string groupID)

**Definition** 지정한 그룹의 그룹 관리자를 해제하고 해제 성공 여부를 반환합니다.

**Argument**

<i>session</i>	세션 ID
<i>userID</i>	그룹 관리자 권한을 해제할 사용자 ID
<i>groupID</i>	그룹 관리자를 해제할 그룹 ID

### ■ IsGroupAdministrator

**Prototype** bool IsGroupAdministrator(object session, string userID, string groupID)

**Definition** 지정한 사용자 ID에 해당하는 사용자가 해당 그룹의 관리자인지 여부를 확인합니다.

**Argument**

<i>session</i>	세션 ID
<i>userID</i>	관리자인지 체크할 사용자 ID

---

<i>groupID</i>	그룹 ID
----------------	-------

---

#### ■ GetAdministrators

---

**Prototype** `IUserInfo[] GetAdministrators(object session, string groupID)`

---

**Definition** 지정한 그룹의 그룹 관리자 정보를 가져옵니다.

---

**Argument** *session* 세션 ID

---

*groupID* 그룹 관리자 정보를 가져올 그룹 ID

---

#### ■ TransferUser

---

**Prototype** `bool TransferUser(object session, string userID, string newGroupID)`

---

**Definition** 지정한 사용자의 그룹을 이동하고 그룹 이동 성공 여부를 가져옵니다.

---

*session* 세션 ID

---

**Argument** *userID* 그룹을 이동할 사용자 ID

---

*newGroupID* 이동할 그룹 ID

---

#### ■ TransferGroup

---

**Prototype** `bool TransferGroup(object session, string groupID, string targetGroupID)`

---

**Definition** 지정한 그룹을 다른 그룹으로 이동하고 그룹 이동 성공 여부를 가져옵니다.

---

*session* 세션 ID

---

**Argument** *groupID* 그룹을 이동할 그룹 ID

---

*targetGroupID* 이동할 그룹 ID

---

### [IRepositoryUser]

#### ■ AccessType

---

**Prototype** `UserAccess AccessType{get;}`

---

---

사용자 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

None = 0x00000000

All = 0x7FFFFFFF

CreateUser = 0x00000001

ModifyName = 0x00000002

**Definition** ModifyPassword = 0x00000004

ModifyDescription = 0x00000008

DeleteUser = 0x00000010

GetUserInfo = 0x00000020

CheckPassword = 0x00000040

GetUserInfos = 0x00000080

IsAdministrator = 0x00000100

---

## ■ Repository

**Prototype** IRepository Repository{get;set;}

**Definition** IRepostiroy 인터페이스를 구현한 객체를 가져옵니다.

---

## ■ CreateUser

**Prototype** string CreateUser(object session, string userName, string password, string description)

**Definition** 새로운 사용자를 생성하고 생성된 사용자의 ID를 반환합니다.

<i>session</i>	세션 ID
<i>userName</i>	사용자 이름
<i>password</i>	패스워드
<i>description</i>	사용자에 대한 설명 내용

---

## ■ ModifyName

**Prototype** string ModifyName(object session, string userID, string newUserName)

**Definition** 지정한 사용자 ID에 해당하는 사용자의 이름을 변경하고 사용자 이름이 변경된 사용자 ID를 반환합니다.

<i>session</i>	세션 ID
<i>userID</i>	사용자 ID
<i>newUserName</i>	변경할 사용자 이름

---

## ■ ModifyPassword

**Prototype** bool ModifyPassword(object session, string userID, string oldPassword, string newPassword)

---

**Definition**    지정한 사용자의 패스워드를 변경하고 변경 성공 여부를 반환합니다.

*session*            세션 ID

**Argument**    *userID*            패스워드를 변경할 사용자 ID

*oldPassword*      변경 전 패스워드

*newPpassword*    변경할 패스워드

#### ■ **ModifyUserDescription**

**Prototype**    bool ModifyDescription(object session, string userID, string description)

**Definition**    지정한 사용자 ID에 해당하는 사용자의 사용자 설명을 변경하고 변경 성공 여부를 반환합니다.

*session*            세션 ID

**Argument**    *userID*            사용자 설명 내용을 변경할 사용자 ID

*description*      변경할 사용자 설명

#### ■ **DeleteUser**

**Prototype**    bool DeleteUser(object session, string userID)

**Definition**    지정한 사용자 ID에 해당하는 사용자의 모든 정보를 삭제하고 삭제 성공 여부를 반환합니다.

*session*            세션 ID

**Argument**    *userID*            삭제할 사용자의 ID

#### ■ **GetUserInfo**

**Prototype**    IUserInfo GetUserInfo(object session, string userID)

**Definition**    지정한 사용자 ID에 해당하는 사용자의 사용자 정보를 가져옵니다.

*session*            세션 ID

**Argument**    *userID*            사용자 정보를 가져올 사용자 ID

#### ■ **CheckPassword**

**Prototype**    bool CheckPassword(object session, string userID, string password)

**Definition**    사용자의 패스워드가 맞는지 여부를 체크합니다.

*session*            세션 ID

**Argument**    *userID*            패스워드를 체크할 사용자 ID

*password*          체크할 패스워드

### ■ GetUserInfos

**Prototype** IUserInfo[] GetUserInfos(object session)

**Definition** 모든 사용자의 사용자 정보를 가져옵니다.

**Argument** *session* 세션 ID

### ■ IsAdministrator

**Prototype** bool IsAdministrator(object session, string userID)

**Definition** 지정한 사용자 ID에 해당하는 사용자가 리파지토리의 관리자인지 여부를 확인합니다.

**Argument** *session* 세션 ID

*userID* 관리자인지 체크할 사용자 ID

### ■ AddAdmin

**Prototype** bool AddAdmin(object sID, string uID, string client\_ip)

**Definition** 관리자 권한을 추가하고 추가 성공 여부를 반환합니다.

**Argument** *sID* 로그인한 사용자 ID

*uID* 관리자 권한을 추가할 사용자 ID

*client\_ip* 클라이언트 IP

### ■ RemoveAdmin

**Prototype** bool RemoveAdmin(object sID, string uID, string client\_ip)

**Definition** 관리자 권한을 삭제하고 삭제 성공 여부를 반환합니다.

**Argument** *sID* 로그인한 사용자 ID

*uID* 관리자 권한을 삭제할 사용자 ID

*client\_ip* 클라이언트 IP

### ■ ModifyAllowip

**Prototype** bool ModifyAllowip(object sID, string uID, string allowip, string client\_ip)

**Definition** 사용자의 허용 IP를 변경하고 변경 성공 여부를 반환합니다.

**Argument** *sID* 로그인한 사용자 ID

*uID* 허용 IP를 변경할 사용자 ID

*allowip* 허용 IP

*client\_ip* 클라이언트 IP

### ■ IsExternalLogin

**Prototype** bool IsExternalLogin

**Definition** 외부 API와 연계한 로그인 기능의 사용 여부를 가져옵니다.

## [IRepositoryMultiLoginUser]

### ■ AccessType

**Prototype** MultiLoginAccess AccessType{get;}

멀티로그인 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

int ACCESS\_MULTILOGINUSER\_NOT = 0x00000000

**Definition** int ACCESS\_DISABLE\_USER\_LOGIN = 0x00000001

int ACCESS\_ENABLE\_USER\_LOGIN = 0x00000002

int ACCESS\_IS\_LOGIN\_ENABLED = 0x00000004

### ■ Repository

**Prototype** IRepository Repository{get;set;}

**Definition** IRepository 인터페이스를 구현한 객체를 가져옵니다.

### ■ DisableLogin

**Prototype** void DisableLogin(object session, string userID)

**Definition** 지정한 세션 ID의 사용자를 로그인하지 못하도록 설정합니다.

**Argument** *session* 세션 ID

*userID* 로그인하지 못하도록 할 사용자 ID

### ■ EnableUserLogin

**Prototype** void EnableLogin(object session, string userID)

**Definition** 지정한 세션 ID의 사용자를 로그인 가능하도록 설정합니다.

**Argument** *session* 세션 ID

*userID* 로그인 가능하도록 할 사용자 ID

### ■ IsLoginEnabled

**Prototype** bool IsLoginEnabled(object session, string userID)

**Definition** 지정한 세션 ID의 사용자가 로그인이 가능한지 여부를 가져옵니다.

**Argument** *session* 세션 ID

*userID* 로그인 가능 여부 체크할 사용자 ID

**[OZRepositoryExternalLogin]**■ **CreateUser**

<b>Prototype</b>	string CreateUser(oz.framework.log.OZLog log, string id, string pwd)	
<b>Definition</b>	외부 API와 연계하여 로그인할 사용자 ID를 생성합니다.	
	<i>log</i>	서버 로그
<b>Argument</b>	<i>id</i>	생성할 사용자 ID
	<i>pwd</i>	생성할 사용자 비밀번호

■ **Login**

<b>Prototype</b>	string Login(oz.framework.log.OZLog log, string id, string pwd)	
<b>Definition</b>	외부 API와 연계된 로그인 ID로 로그인하고 에러 발생 시 에러 메시지를 리턴합니다.	
	<i>log</i>	서버 로그
<b>Argument</b>	<i>id</i>	로그인할 사용자 ID
	<i>pwd</i>	로그인할 사용자 비밀번호

**[IRepositoryGroupAuthorityToCategory]**■ **AccessType**

<b>Prototype</b>	AuthorityAccess AccessType{get;}
	카테고리에 대한 그룹 권한 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다. None = 0x00000000 All = 0x7FFFFFFF
<b>Definition</b>	ModifyGroupAuthToCategory = 0x00000001 GetGroupAuthToCategory = 0x00000002 GetGroupInfosOfCategory = 0x00000004 GetItemInfosInCategoryOfGroup = 0x00000008 GetCategoryInfosOfGroup = 0x00010000

■ **Repository**

<b>Prototype</b>	IRepository Repository{get;set;}
<b>Definition</b>	IRepository 인터페이스를 구현한 객체를 가져옵니다.

## ■ Modify

---

<b>Prototype</b>	bool Modify(object session, string categoryID, string groupID, Authority permission)
<b>Definition</b>	지정한 카테고리 와 그룹의 권한 정보를 변경하고 변경 성공 여부를 가져옵니다.
	<i>session</i> 세션 ID
	<i>categoryID</i> 권한 정보를 변경할 카테고리 ID
	<i>groupID</i> 권한 정보를 변경할 그룹 ID
<b>Argument</b>	권한 정보
	<ul style="list-style-type: none"> <li>• 1 : VIEW</li> <li>• 2 : READ</li> <li>• 4 : WRITE</li> </ul>
	<i>permission</i>

※ 참고사항 : 권한 정보는 OR 조건으로 처리해야 합니다.

## ■ GetAuthority

---

<b>Prototype</b>	Authority GetAuthority(object session, string groupID, string categoryID)
<b>Definition</b>	지정한 카테고리에 대한 지정한 그룹의 그룹 권한을 가져옵니다.
	<i>session</i> 세션 ID
<b>Argument</b>	<i>groupID</i> 그룹 ID
	<i>categoryID</i> 권한 정보를 가져올 카테고리 ID

## ■ GetGroupInfos

---

<b>Prototype</b>	IGroupInfo[] GetGroupInfos(object session, string categoryID, Authority permission)
<b>Definition</b>	지정한 카테고리 ID에 대해 permission 이상의 권한을 가지고 있는 모든 사용자 정보를 가져옵니다.
	<i>session</i> 세션 ID
<b>Argument</b>	<i>categoryID</i> 카테고리 ID
	<i>permission</i> 권한 정보

## ■ GetItemInfos

---

<b>Prototype</b>	IItemInfo[] GetItemInfos(object session, string groupID, string categoryID, Authority permission)
<b>Definition</b>	지정한 카테고리의 아이템 중에 지정한 그룹의 권한이 permission 이상인 모든 아이템의 목록을 가져옵니다.

<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	그룹 ID
	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

■ **GetCategoryInfos**

<b>Prototype</b>	ICategoryInfo[] GetCategoryInfos(object session, string groupID, string categoryID, Authority permission)	
<b>Definition</b>	지정한 카테고리의 하위 카테고리 중에 지정한 그룹의 권한이 permission 이상인 모든 카테고리의 목록을 가져옵니다.	
<b>Argument</b>	<i>session</i>	세션 ID
	<i>groupID</i>	그룹 ID
	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

**[IRepositoryGroupAuthorityToItem]**

■ **AccessType**

<b>Prototype</b>	AuthorityAccess AccessType{get;}	
<b>Definition</b>	아이템에 대한 그룹 권한 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.	
	None = 0x00000000	
	All = 0x7FFFFFFF	
	ModifyGroupAuthToItem = 0x00000010	
	GetGroupAuthToItem = 0x00000020	
	GetGroupInfosOfItem = 0x00000040	
GetItemInfosOfGroup = 0x00000080		

■ **Repository**

<b>Prototype</b>	IRepository Repository{get;set;}	
<b>Definition</b>	IRepository 인터페이스를 구현한 객체를 가져옵니다.	

■ **Modify**

<b>Prototype</b>	bool Modify(object session, string groupID, string itemID, Authority permission)	
<b>Definition</b>	지정한 아이템 ID에 대한 지정한 그룹 ID의 권한 정보를 변경하고 변경 성공 여부를 반환합니다.	
<b>Argument</b>	<i>session</i>	세션 ID

<i>groupID</i>	그룹 ID
<i>itemID</i>	아이템 ID
<i>permission</i>	권한 정보

#### ■ GetAuthority

<b>Prototype</b>	Authority GetAuthority(object session, string groupID, string itemID)
<b>Definition</b>	지정한 아이템 ID에 대한 지정한 그룹 ID의 권한 정보를 가져옵니다.
	<i>session</i> 세션 ID
<b>Argument</b>	<i>groupID</i> 그룹 ID
	<i>itemID</i> 아이템 ID

#### ■ GetGroupInfos

<b>Prototype</b>	IGroupInfo[] GetGroupInfos(object session, string itemID, Authority permission)
<b>Definition</b>	지정한 아이템 ID에 대해 permission 이상의 권한이 부여된 모든 그룹 정보를 가져옵니다.
	<i>session</i> 세션 ID
<b>Argument</b>	<i>itemID</i> 아이템 ID
	<i>permission</i> 권한 정보

#### ■ GetItemInfos

<b>Prototype</b>	IItemInfo[] GetItemInfos(object session, string groupID, Authority permission)
<b>Definition</b>	지정한 그룹 ID에 해당하는 그룹에게 permission 이상의 권한이 부여된 모든 아이템의 정보를 가져옵니다.
	<i>session</i> 세션 ID
<b>Argument</b>	<i>groupID</i> 그룹 ID
	<i>permission</i> 권한 정보

### [IRepositoryUserAuthorityToCategory]

#### ■ AccessType

<b>Prototype</b>	AuthorityAccess AccessType{get;}
------------------	----------------------------------

카테고리에 대한 사용자 권한 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.

None = 0x00000000

All = 0x7FFFFFFF

**Definition** ModifyUserAuthToCategory = 0x00000100  
 GetUserAuthToCategory = 0x00000200,  
 GetUserInfosOfCategory = 0x00000400  
 GetItemInfosInCategoryOfUser = 0x00000800  
 GetCategoryInfosOfUser = 0x00020000

■ **Repository**

**Prototype** IRepository Repository{get;set;}

**Definition** OZRepository 인터페이스를 구현한 클래스를 가져옵니다.

■ **Modify**

**Prototype** bool Modify(object session, string userID, string categoryID, Authority permission)

**Definition** 지정한 카테고리 ID에 대한 지정한 사용자 ID의 권한 정보를 변경하고 변경 성공 여부를 반환합니다.

<b>Argument</b>	<i>session</i>	세션 ID
	<i>userID</i>	사용자 ID
	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

■ **GetAuthority**

**Prototype** Authority GetAuthority(object session, string userID, string categoryID)

**Definition** 지정한 카테고리 ID에 대한 지정한 사용자 ID의 권한 정보를 가져옵니다.

<b>Argument</b>	<i>session</i>	세션 ID
	<i>userID</i>	사용자 ID
	<i>categoryID</i>	카테고리 ID

■ **GetUserInfos**

**Prototype** IUserInfo[] GetUserInfos(object session, string categoryID, Authority permission)

**Definition** 지정한 카테고리 ID에 대해 permission 이상의 권한이 부여된 모든 사용자 정보를 가져옵니다.

<b>Argument</b>	<i>session</i>	세션 ID
-----------------	----------------	-------

<i>categoryID</i>	카테고리 ID
<i>permission</i>	권한 정보

■ **GetItemInfos**

<b>Prototype</b>	IItemInfo[] GetItemInfos(object session, string userID, string categoryID, Authority permission)
<b>Definition</b>	지정한 카테고리의 아이템 중에 지정한 사용자의 권한이 permission 이상인 모든 아이템의 목록을 가져옵니다.
	<i>session</i> 세션 ID
	<i>userID</i> 사용자 ID
<b>Argument</b>	<i>categoryID</i> 카테고리 ID
	<i>permission</i> 권한 정보

■ **GetCategoryInfos**

<b>Prototype</b>	ICategoryInfo[] GetCategoryInfos(object session, string userID, string categoryID, Authority permission)
<b>Definition</b>	지정한 카테고리의 하위 카테고리 중에 지정한 사용자의 권한이 permission 이상인 모든 카테고리의 목록을 가져옵니다.
	<i>session</i> 세션 ID
	<i>userID</i> 사용자 ID
<b>Argument</b>	<i>categoryID</i> 카테고리 ID
	<i>permission</i> 권한 정보

**[IRepositoryUserAuthorityItem]**

■ **AccessType**

<b>Prototype</b>	AuthorityAccess AccessType{get;}
	아이템에 대한 사용자 권한 인터페이스에 접근 가능한 메소드를 숫자 값으로 가져옵니다.
	None = 0x00000000
<b>Definition</b>	All = 0x7FFFFFFF
	ModifyUserAuthToItem = 0x00001000
	GetUserAuthToItem = 0x00002000
	GetUserInfosOfItem = 0x00004000
	GetItemInfosOfUser = 0x00008000

■ **Repository**

<b>Prototype</b>	IRepository Repository{get;set;}
------------------	----------------------------------

---

**Definition** IRepostiroy 인터페이스를 구현한 객체를 가져옵니다.

---

#### ■ **Modify**

---

**Prototype** bool Modify(object session, string userID, string itemID, Authority permission)

**Definition** 지정한 아이템에 대한 사용자의 권한 정보를 변경하고 변경 성공 여부를 가져옵니다.

<i>session</i>	세션 ID
<i>userID</i>	사용자 ID
<i>iItemID</i>	아이템 ID
<i>permission</i>	권한 정보

---

#### ■ **GetAuthority**

---

**Prototype** Authority GetAuthority(object session, string userID, string itemID)

**Definition** 지정한 아이템에 대한 사용자의 권한 정보를 가져옵니다.

<i>session</i>	세션 ID
<i>userID</i>	사용자 ID
<i>itemID</i>	권한 정보

---

#### ■ **GetUserInfos**

---

**Prototype** IUserInfo[] GetUserInfos(object session, string itemID, Authority permission)

**Definition** 지정한 아이템에 대한 사용자의 권한이 permission 이상인 모든 아이템 정보를 가져옵니다.

<i>session</i>	세션 ID
<i>itemID</i>	아이템 ID
<i>permission</i>	권한 정보

---

#### ■ **GetItemInfos**

---

**Prototype** IItemInfo[] GetItemInfos(object session, string userID, Authority permission)

**Definition** 지정한 사용자의 권한이 permission 이상인 모든 아이템 정보를 가져옵니다.

<i>session</i>	세션 ID
<i>userID</i>	아이템 ID
<i>permission</i>	권한 정보

---

**[ICategoryInfo]**■ **ID**


---

**Prototype**    `string ID{get;}`


---

**Definition**    현재 카테고리의 ID를 가져옵니다.

---

■ **Name**


---

**Prototype**    `string Name{get;}`


---

**Definition**    현재 카테고리의 이름을 가져옵니다.

---

■ **ParentID**


---

**Prototype**    `string ParentID{get;}`


---

**Definition**    현재 카테고리의 상위 카테고리 ID를 가져옵니다.

---

■ **ParentName**


---

**Prototype**    `string ParentName{get;}`


---

**Definition**    현재 카테고리의 상위 카테고리 이름을 가져옵니다.

---

■ **Comment**


---

**Prototype**    `string Comment{get;}`


---

**Definition**    현재 카테고리의 주석문을 가져옵니다.

---

■ **Description**


---

**Prototype**    `string Description{get;}`


---

**Definition**    현재 카테고리의 설명 내용을 가져옵니다.

---

■ **UpdateTime**


---

**Prototype**    `long UpdateTime{get;}`


---

**Definition**    현재 카테고리를 갱신한 시간 정보를 가져옵니다.

---

■ **CreatedUserID**


---

**Prototype**    `string CreatedUserID{get;}`


---

**Definition**    현재 카테고리를 생성한 사용자 ID를 가져옵니다.

---

### ■ CreatedUserName

---

**Prototype** string CreatedUserName{get;}

**Definition** 현재 카테고리를 생성한 사용자 이름을 가져옵니다.

---

### ■ Permission

---

**Prototype** Authority Permission{get;}

**Definition** 현재 카테고리의 권한 정보를 가져옵니다.

---

## [IItemInfo]

### ■ IsCheckedOut

---

**Prototype** bool IsCheckedOut{get;}

**Definition** 해당 아이템의 체크 아웃 여부를 가져옵니다

---

### ■ IsDeletable

---

**Prototype** bool IsDeletable{get;}

해당 아이템의 삭제 여부를 가져옵니다.

**Definition** ※ 참고사항 : "toBeDestroyed =false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.

---

### ■ ID

---

**Prototype** string ID{get;}

**Definition** 해당 아이템의 아이템 ID를 가져옵니다.

---

### ■ Name

---

**Prototype** string Name{get;}

**Definition** 해당 아이템의 아이템 이름을 가져옵니다.

---

### ■ CategoryID

---

**Prototype** string CategoryID{get;}

**Definition** 해당 아이템이 속한 카테고리의 카테고리 ID를 가져옵니다.

---

### ■ CategoryName

---

**Prototype** string CategoryName{get;}

**Definition** 해당 아이템이 속한 카테고리의 카테고리 이름을 가져옵니다.

---

### ■ UpdateTime

---

**Prototype**    `long UpdateTime{get;}`

---

**Definition**    해당 아이템을 갱신한 시간 정보를 가져옵니다.

---

### ■ Description

---

**Prototype**    `string Description{get;}`

---

**Definition**    해당 아이템의 설명 내용을 가져옵니다.

---

### ■ Comment

---

**Prototype**    `string Comment{get;}`

---

**Definition**    해당 아이템의 주석문을 가져옵니다.

---

### ■ CreatedUserID

---

**Prototype**    `string CreatedUserID{get;}`

---

**Definition**    해당 아이템을 생성한 사용자 ID를 가져옵니다.

---

### ■ CreatedUserName

---

**Prototype**    `string CreatedUserName{get;}`

---

**Definition**    해당 아이템을 생성한 사용자 이름을 가져옵니다.

---

### ■ CheckOutUser

---

**Prototype**    `string CheckOutUser{get;}`

---

**Definition**    해당 아이템을 체크 아웃한 사용자 이름을 가져옵니다.

---

### ■ CheckOutLocalPath

---

**Prototype**    `string CheckOutLocalPath{get;}`

---

**Definition**    체크 아웃한 아이템의 폴더 이름을 가져옵니다.

---

### ■ Permission

---

**Prototype**    `Authority Permission{get;}`

---

**Definition**    해당 아이템의 권한 정보를 가져옵니다.

---

## [IHistoryInfo]

### ■ ItemID

---

**Prototype**    `string ItemID{get;}`

---

**Definition**    아이템 ID를 가져옵니다.

---

■ **ItemName**

---

**Prototype**    `string ItemName{get;}`

---

**Definition**    아이템 이름을 가져옵니다.

---

■ **Version**

---

**Prototype**    `int Version{get;}`

---

**Definition**    아이템의 버전 개수를 가져옵니다.

---

■ **CheckInTime**

---

**Prototype**    `long CheckInTime{get;}`

---

**Definition**    아이템의 버전 별 체크인 시간을 가져옵니다.

---

■ **CheckInUser**

---

**Prototype**    `string CheckInUser{get;}`

---

**Definition**    아이템의 버전 별 체크인 사용자 ID를 가져옵니다.

---

■ **CheckInComment**

---

**Prototype**    `string CheckInComment{get;}`

---

**Definition**    아이템의 버전 별 체크인 주석문을 가져옵니다.

---

## [IGroupInfo]

■ **Administrators**

---

**Prototype**    `IDictionary Administrators{get;}`

해당 그룹의 관리자 정보를 가져옵니다.

---

**Definition**    ※ 참고사항 : 관리자 정보는 "key=사용자ID, value=사용자 이름"으로 구성된 HashMap으로 반환됩니다.

---

■ **ID**

---

**Prototype**    `string ID{get;}`

---

**Definition**    해당 그룹의 그룹 ID를 가져옵니다.

---

---

**■ Name**


---

**Prototype**    `string Name{get;}`

---

**Definition**    해당 그룹의 그룹 이름을 가져옵니다.

---

**■ ParentID**


---

**Prototype**    `string ParentID{get;}`

---

**Definition**    해당 그룹의 상위 그룹 ID를 가져옵니다.

---

**■ ParentName**


---

**Prototype**    `string ParentName{get;}`

---

**Definition**    해당 그룹의 상위 그룹 이름을 가져옵니다.

---

**■ Description**


---

**Prototype**    `string Description{get;}`

---

**Definition**    해당 그룹의 설명 내용을 가져옵니다.

---

**■ UpdateTime**


---

**Prototype**    `long UpdateTime{get;}`

---

**Definition**    해당 그룹을 갱신한 시간 정보를 가져옵니다.

---

**■ Permission**


---

**Prototype**    `Authority Permission{get;}`

---

**Definition**    해당 그룹의 권한 정보를 가져옵니다.

---

**[IUserInfo]**
**■ ID**


---

**Prototype**    `string ID{get;}`

---

**Definition**    해당 사용자의 사용자 ID를 가져옵니다.

---

**■ Name**


---

**Prototype**    `string Name{get;}`

---

**Definition**    해당 사용자의 사용자 이름을 가져옵니다.

---

**■ GroupID**

---

**Prototype** string GroupID{get;}

---

**Definition** 해당 사용자가 속한 그룹의 그룹 ID를 가져옵니다.

---

**■ GroupName**

---

**Prototype** string GroupName{get;}

---

**Definition** 해당 사용자가 속한 그룹의 그룹 이름을 가져옵니다.

---

**■ Description**

---

**Prototype** string Description{get;}

---

**Definition** 해당 사용자의 설명 내용을 가져옵니다.

---

**■ UpdateTime**

---

**Prototype** long UpdateTime{get;}

---

**Definition** 해당 사용자를 갱신한 시간 정보를 가져옵니다.

---

**■ IsLoggedIn**

---

**Prototype** bool IsLoggedIn{get;}

---

**Definition** 해당 사용자의 로그인 여부를 가져옵니다.

---

**■ SessionID**

---

**Prototype** string SessionID{get;}

---

**Definition** 해당 사용자가 로그인 시 생성된 세션 ID를 가져옵니다.

---

**■ IsLoginEnabled**

---

**Prototype** bool IsLoginEnabled{get;}

---

**Definition** 해당 사용자의 로그인 가능 여부를 가져옵니다.

---

**■ LoginIP**

---

**Prototype** string LoginIP{get;}

---

**Definition** 해당 사용자가 로그인한 PC의 IP 정보를 가져옵니다.

---

**■ LastLoginTime**

---

**Prototype** long LastLoginTime{get;}

---

**Definition** 해당 사용자가 마지막으로 로그인한 시간 정보를 가져옵니다.

---

■ **Permission**

---

**Prototype** Authority Permission{get;}

---

**Definition** 해당 사용자의 권한 정보를 가져옵니다.

---

**[ILoginInfo]**

■ **ClientType**

---

**Prototype** ClientType ClientType{get;}

---

**Definition** 클라이언트에서 호출한 클라이언트의 종류를 가져옵니다.

---

■ **MACAddress**

---

**Prototype** string MACAddress{get;}

---

**Definition** 클라이언트로부터 MAC Address 정보를 가져옵니다.

---

■ **IPAddress**

---

**Prototype** string IPAddress{get;}

---

**Definition** 클라이언트로부터 IP Address 정보를 가져옵니다.

---

■ **HDDSerialNo**

---

**Prototype** string HDDSerialNO{get;}

---

**Definition** 클라이언트로부터 HDD Serial 번호를 가져옵니다.

---

■ **ProtocolVersion**

---

**Prototype** int ProtocolVersion{get;}

---

**Definition** 클라이언트로부터 프로토콜 버전을 가져옵니다.

---

■ **UserID**

---

**Prototype** string UserID{get;}

---

**Definition** 클라이언트가 로그인한 사용자 ID를 가져옵니다.

---

■ **UserName**

---

**Prototype** string UserName{get;}

---

**Definition** 클라이언트가 로그인한 사용자 이름을 가져옵니다.

---

■ **SessionID**

---

**Prototype**    string SessionID{get;}

---

**Definition**   클라이언트가 로그인 시 생성된 세션 ID를 가져//옵니다.

---

■ **ClientIP**

---

**Prototype**    string ClientIP{get;}

---

**Definition**   클라이언트가 요청한 클라이언트 IP를 가져옵니다.

---

## Class OZRepositoryException

[패키지명 : oz.framework.repositoryex]

■ **public class OZRepositoryException extends Exception**

■ **Method Summary**

- **public OZRepositoryException(string \_msg)**
- **public OZRepositoryException(int code,string \_msg)**
- **public OZErrorCode ErrorCode{get;}**
- **public string Message{get;}**

■ **Method Detail**

- OZRepositoryException

---

	public OZRepositoryException(string _msg)				
<b>Prototype</b>	public OZRepositoryException(int code,string _msg)				
<b>Definition</b>	OZRepositoryException 생성자				
<b>Argument</b>	<table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;"><i>code</i></td> <td>에러 코드</td> </tr> <tr> <td><i>_msg</i></td> <td>에러 메시지</td> </tr> </table>	<i>code</i>	에러 코드	<i>_msg</i>	에러 메시지
<i>code</i>	에러 코드				
<i>_msg</i>	에러 메시지				

---

- ErrorCode

---

	public OZErrorCode ErrorCode{get;}
<b>Prototype</b>	
<b>Definition</b>	에러 코드

---

- Message

---

**Prototype**    `public string Message{get;}`

---

**Definition**    에러 메시지

---

## Class RepositoryEx

### Constructor Summary

- **RepositoryEx(string ip, int port, string id, string pw, bool useUSL)** throws **OZCPEException**
- **RepositoryEx(string iurl, string id, string pw, bool useUSL)** throws **OZCPEException**

### Method Summary

- **OZErrorCode GetLastErrorCode(int index)**
- **string GetLastErrorMessage(int index)**
- **void SetConfiguration(NameValueCollection conf)**
- **NameValueCollection GetConfiguration()**
- **void Reload()**
- **string[] CreateItems(string[] itemNames, string[] descriptions, string[] categoryIDs, bool[] isCompressed, Stream[] items, string comment)**
- **string[] CreateItem(string[] itemNames, string[] descriptions, string[] categoryIDs, bool[] isCompressed, Stream[] items, string comment, long[] createdTimes)**
- **string ModifyItemName(string itemID, string newItemName, string comment)**
- **bool ModifyItemDatetime(string itemID, long datetime)**
- **bool[] DeleteItems(string[] itemIDs, bool[] toBeDestroyed, string comment)**
- **bool[] UndeleteItems(string[] itemIDs, string comment)**
- **bool ModifyHistoryItemComment(string itemID, int itemVersion, string newComment)**
- **bool ModifyHistoryItemDatetime(string itemID, int itemVersion, long datetime)**
- **bool ModifyItemDescription(string itemID, string description)**
- **IItemInfo GetItemInfo(string itemID)**

- **IItemInfo[] GetItemInfoListInCategory(string categoryID, bool isRecursive)**
- **bool HasItem(string itemID)**
- **Stream[] GetItems(string[] itemIDs, bool[] isCompressed, bool[] isObjStreams)**
- **Stream[] GetItems(string[] itemIDs, long[] modifiedTimes, bool[] isCompressed, bool[] isObjStreams)**
- **Stream[] CheckOut(string[] itemIDs, string[] checkOutFolders, long[] localFileTimes, bool[] isCompressed, string[] checkOutCmts)**
- **bool[] CheckIn(string[] itemIDs, bool[] isCompressed, Stream[] items, string comment, bool[] keepCheckOut)**
- **bool[] CheckIn(string[] itemIDs, bool[] isCompressed, Stream[] items, string comment, bool[] keepCheckOut, long[] datetimes)**
- **Stream[] UndoCheckOut(string[] itemIDs, bool[] replaceItems, bool[] isCompressed)**
- **bool[] IsCheckOutUser(string[] itemIDs)**
- **bool RollBack(string itemID, int version, string comment)**
- **Stream GetItem(string itemID, int version, bool isCompressed)**
- **IHistoryInfo[] GetHistoryInfos(string itemID)**
- **IHistoryInfo[] GetHistoryItemListByDatetime(string itemID)**
- **Stream GetHistoryItemByDatetime(string itemID, long datetime, bool isCompressed, bool isObjStream)**
- **IHistoryInfo[] GetDeletedItemHistoryInfos(string itemID)**
- **bool RemoveHistory(string itemID, int version)**
- **bool TransferItems(string[] itemIDs, string targetCategoryID)**
- **bool TransferCategory(string categoryID, string targetCategoryID)**
- **string[] CreateCategories(string[] categoryNames, string[] parentCategoryIDs, string comment)**
- **bool ModifyCategoryDesc(string categoryID, string description)**
- **string ModifyCategoryName(string categoryID, string newCategoryName, string comment)**
- **bool[] DeleteCategories(string[] categoryIDs, bool[] toBeDestroyed, string comment)**
- **bool[] UndeleteCategories(string[] categoryIDs, string comment)**
- **int GetItemCount(string categoryID)**
- **IItemInfo[] GetItemInfos(string categoryID)**

- **ICategoryInfo[] GetCategoryInfos(string categoryID)**
- **string GetCategoryID(string itemID)**
- **ICategoryInfo GetCategoryInfo(string categoryID)**
- **IItemInfo[] GetDeletedItemInfos(string categoryID)**
- **ISearchResult[] SearchItem(string categoryID, string checkOutUser, string startDate, string endDate, SearchOptions options, string fileFilter, string criteria)**
- **string CreateUser(string userName, string password, string description)**
- **string ModifyUserName(string userID, string userName)**
- **bool ModifyUserPassword(string userID, string oldPwd, string newPwd)**
- **bool ModifyUserDescription(string userID, string description)**
- **bool DeleteUser(string userID)**
- **IUserInfo GetUserInfo(string userID)**
- **bool CheckPassword(string userID, string password)**
- **IUserInfo[] GetUserInfos()**
- **void DisableLogin(string userID)**
- **void EnableLogin(string userID)**
- **bool IsLoginEnabled(string userID)**
- **string CreateGroup(string groupName, string parentGroupID, string description)**
- **string ModifyGroupName(string groupID, string groupName)**
- **bool ModifyGroupDescription(string groupID, string description)**
- **bool DeleteGroup(string groupID)**
- **string CreateUser(string userName, string password, string groupID, string description)**
- **bool TransferUser(string userID, string newGroupID)**
- **bool TransferGroup(string groupID, string targetGroupID)**
- **IUserInfo[] GetUserInfos(string groupID)**
- **IUserInfo[] GetUserInfoListInGroup(string groupID, bool isRecursive)**
- **IGroupInfo GetGroupInfo(string groupID)**
- **IGroupInfo[] GetGroupInfoListInGroup(string groupID, bool isRecursive)**
- **IGroupInfo[] GetSubGroupInfos(string groupID)**
- **IGroupInfo GetParentGroupInfo(string groupID)**
- **string GetGroupID(string userID)**
- **bool AddGroupAdministrator(string userID, string groupID)**
- **bool RemoveGroupAdministrator(string userID, string groupID)**

- **bool IsGroupAdministrator(string userID, string groupID)**
- **IUserInfo[] GetGroupAdministrators(string groupID)**
- **bool IsAdministrator(string userID)**
- **bool AddAdministrator(string userID)**
- **bool RemoveAdministrator(string userID)**
- **Stream Distribute(string categoryID, bool isRecursive)**
- **Stream DistributeRepository(string[] itemIDs)**
- **Stream DistributeRepositoryByDatetime(string[] itemIDs)**
- **bool Upload(string categoryID, Stream in)**
- **bool ModifyUserAuthorityToItem(string userID, string itemID, Authority permission)**
- **bool ModifyUserAuthorityToCategory(string userID, string categoryID, Authority permission)**
- **bool ModifyGroupAuthorityToItem(string groupID, string itemID, Authority permission)**
- **bool ModifyGroupAuthorityToCategory(string groupID, string categoryID, Authority permission)**
- **Authority GetUserAuthorityToItem(string userID, string itemID)**
- **Authority GetUserAuthorityToCategory(string userID, string categoryID)**
- **Authority GetGroupAuthorityToItem(string groupID, string itemID)**
- **Authority GetGroupAuthorityToCategory(string groupID, string categoryID)**
- **IGroupInfo[] GetGroupInfosOfItem(string itemID, Authority permission)**
- **IGroupInfo[] GetGroupInfosOfCategory(string categoryID, Authority permission)**
- **IUserInfo[] GetUserInfosOfItem(string itemID, Authority permission)**
- **IUserInfo[] GetUserInfosOfCategory(string categoryID, Authority permission)**
- **ICategoryInfo[] GetCategoryInfosOfUser(string userID, string categoryID, Authority permission)**
- **ICategoryInfo[] GetCategoryInfosOfGroup(string groupID, string categoryID, Authority permission)**
- **IItemInfo[] GetItemInfosOfUser(string userID, string categoryID, Authority permission)**
- **IItemInfo[] GetItemInfosOfGroup(string groupID, string categoryID, Authority permission)**
- **int SetUserDefinedResourceSync(string alias, string[] itemIDs, string**

**comment, string[] errorMsg)**

- **OZAttributeList EncryptFile(string[] itemIDs)**
- **OZAttributeList EncryptFile(string categoryID, bool isRecursive)**
- **OZAttributeList DecryptFile(string[] itemIDs)**
- **OZAttributeList DecryptFile(string categoryID, bool isRecursive)**

## Constructor Detail

### ■ RepositoryEx

<b>Prototype</b>	<i>//TCP/IP 방식</i> public RepositoryEx(string ip, int port, string id, string pw, bool useUSL) throws OZCPEXception
	<i>//HTTP 방식</i> public RepositoryEx(string iurl, string id, string pw, bool useUSL) throws OZCPEXception
<b>Argument</b>	<i>ip</i> HTTP 방식 오즈 서버의 URL ex) string url = "http://127.0.0.1/oz/server";
	<i>port</i> TCP/IP 방식 오즈 서버의 IP ex) string ip = "127.0.0.1";
	<i>id</i> TCP/IP 방식 오즈 서버의 포트 번호 ex) int port = 8003;
	<i>pw</i> 사용자 아이디 ex) string id = "admin";
	<i>useUSL</i> USL 사용 여부 ex) bool useUSL = false;

## Method Detail

### ■ GetLastErrorCode

<b>Prototype</b>	public OZErrorCode GetLastErrorCode(int index)
<b>Definition</b>	마지막에 호출한 메소드의 에러 코드를 가져옵니다.
<b>Argument</b>	<i>index</i> 메소드 호출 시 요청 인덱스

### ■ GetLastErrorMessage

<b>Prototype</b>	public OZErrorCode GetLastErrorMessage(int index)
<b>Definition</b>	마지막에 호출한 메소드의 에러 메시지를 가져옵니다.

**Argument**    *index*    메소드 호출 시 요청 인덱스

■ **SetConfiguration**

**Prototype**    public void SetConfiguration(NameValueCollection conf)

**Definition**    리파지토리 설정을 변경합니다.

**Argument**    *conf*    변경할 설정

■ **GetConfiguration**

**Prototype**    public NameValueCollection GetConfiguration()

**Definition**    리파지토리 설정을 가져옵니다.

■ **Reload**

**Prototype**    public void Reload()

**Definition**    리파지토리를 재시작합니다.

■ **CreateItems**

**Prototype**    public string[] CreateItems(string[] itemNames, string[] descriptions, string[] categoryIDs, bool[] isCompressed, Stream[] items, string comment)

**Definition**    아이템을 생성하고 생성된 아이템 ID를 반환합니다.

<i>itemNames</i>	새로 생성할 아이템의 이름
<i>descriptions</i>	새로 생성할 아이템의 설명 내용
<i>categoryIDs</i>	새로 생성할 아이템이 카테고리 이름
<b>Argument</b> <i>isCompressed</i>	압축 여부
<i>items</i>	새로 생성할 아이템의 입력 스트림
<i>comment</i>	주석문

■ **CreateItem**

**Prototype**    public string[] CreateItem(string[] itemNames, string[] descriptions, string[] categoryIDs, bool[] isCompressed, Stream[] items, string comment, long[] createdTimes)

**Definition**    시간을 지정하여 아이템을 생성하고 생성된 아이템 ID를 반환합니다.(NONE 타입은 지원 안 함)

<i>itemNames</i>	새로 생성할 아이템의 이름
<b>Argument</b> <i>descriptions</i>	새로 생성할 아이템의 설명 내용
<i>categoryIDs</i>	새로 생성할 아이템이 카테고리 이름

<i>isCompressed</i>	압축 여부
<i>items</i>	새로 생성할 아이템의 입력 스트림
<i>comment</i>	주석문
<i>createdTimes</i>	설정할 아이템 생성시간 설정할 시간이 0, 음수이면 현재시간으로 설정

#### ■ ModifyItemName

<b>Prototype</b>	<code>public string ModifyItemName(string itemID, string newItemName, string comment)</code>
<b>Definition</b>	아이템 이름을 변경하고 변경된 아이템 ID를 반환합니다.
	<i>itemID</i> 이름을 변경할 아이템 ID
<b>Argument</b>	<i>newItemName</i> 변경할 아이템 이름
	<i>comment</i> 주석문

#### ■ ModifyItemDatetime

<b>Prototype</b>	<code>public bool ModifyItemDatetime(string itemID, long datetime)</code>
<b>Definition</b>	아이템 생성 시간을 변경합니다.(NONE 타입은 지원 안 함)
	<i>itemID</i> 시간을 변경할 아이템의 ID
<b>Argument</b>	<i>datetime</i> 변경할 시간 0 또는 음수로 설정할 경우 현재 시간으로 적용됨

#### ■ DeleteItems

<b>Prototype</b>	<code>public bool[] DeleteItems(string[] itemIDs, bool[] toBeDestroyed, string comment)</code>
<b>Definition</b>	지정한 아이템을 리파지토리에서 삭제하고 아이템 삭제 성공 여부를 가져옵니다.
	<i>itemIDs</i> 삭제할 아이템 ID
<b>Argument</b>	<i>toBeDestroyed</i> 아이템을 영구 삭제할 지 여부
	<i>comment</i> 주석문

#### ■ UnDeleteItems

<b>Prototype</b>	<code>public bool[] UndeleteItems(string[] itemIDs, string comment)</code>
<b>Definition</b>	삭제된 아이템을 복원하고 복원 성공 여부를 가져옵니다. ※ 참고사항 : 아이템 삭제 시 "toBeDestroyed=false"로 설정하였을 경우 삭제된 아이템의 복원이 가능합니다.
<b>Argument</b>	<i>itemIDs</i> 복원할 아이템 ID

---

*comment*           주석문

---

■ **ModifyHistoryItemComment**

---

**Prototype**   public bool ModifyHistoryItemComment(string itemID, int itemVersion, string newComment) throws OZCPEXception

---

**Definition**   아이템 히스토리 중 특정 버전의 주석을 수정합니다.

---

*itemID*           아이템 ID(Full Path로 설정)

---

**Argument**   *itemVersion*   주석을 수정할 버전

---

*newComment*   수정할 주석

---

■ **ModifyHistoryItemDatetime**

---

**Prototype**   public bool ModifyHistoryItemDatetime(string itemID, int itemVersion, long datetime)

---

**Definition**   지정한 버전의 아이템 시간을 변경합니다.(NONE 타입은 지원 안 함)

---

*itemID*           시간을 변경할 아이템의 ID

---

**Argument**   *itemVersion*   시간을 변경할 버전

---

*datetime*        변경할 시간  
0 또는 음수로 설정할 경우 현재 시간으로 적용됨

---

■ **ModifyItemDescription**

---

**Prototype**   public bool ModifyItemDescription(string itemID, string description)

---

**Definition**   지정한 아이템 ID에 해당하는 아이템의 설명을 변경하고 변경 성공 여부를 반환합니다.

---

*itemID*           설명을 변경할 아이템의 ID

---

**Argument**   *description*   아이템 설명 내용

---

■ **GetItemInfo**

---

**Prototype**   public IItemInfo GetItemInfo(string itemID)

---

**Definition**   지정한 아이템 ID에 해당하는 아이템의 정보를 가져옵니다.

---

**Argument**   *itemID*           정보를 가져올 아이템 ID

---

■ **GetItemInfoListInCategory**

---

**Prototype**   public IItemInfo[] GetItemInfoListInCategory(string categoryID, bool isRecursive)

---

**Definition**   아이템 정보를 가져옵니다.(NONE 타입은 지원 안 함)

---

**Argument**   *categoryID*   카테고리 ID(Full Path로 설정)

---

---

*isRecursive* 하위 카테고리 포함 여부

---

■ **HasItem**

---

**Prototype** public bool HasItem(string itemID)

**Definition** 지정한 아이템의 존재 여부를 확인합니다.

**Argument** *itemID*           아이템 ID

---

■ **GetItems**

---

**Prototype** public Stream[] GetItems(string[] itemIDs, bool[] isCompressed, bool[] isObjStreams)

**Definition** 지정한 아이템ID에 해당하는 아이템을 가져옵니다.

*itemIDs*                   가져올 아이템 ID

**Argument** *isCompressed*       압축 여부

*isObjStreams*           ODI 객체 스트림 필요 여부

---

■ **GetItems**

---

**Prototype** public Stream[] GetItems(string[] itemIDs, long[] modifiedTimes, bool[] isCompressed, bool[] isObjStreams)

**Definition** 지정한 아이템ID에 해당하는 아이템을 가져옵니다.

*itemIDs*                   가져올 아이템 ID

**Argument** *modifiedTimes*       압축 여부

*isCompressed*           아이템을 최종 변경한 시간

*isObjStreams*           ODI 객체 스트림 필요 여부

---

■ **CheckOut**

---

**Prototype** public Stream[] CheckOut(string[] itemIDs, string[] checkoutFolders, long[] localFileTimes, bool[] isCompressed, string[] checkoutCmts)

**Definition** 지정한 아이템을 체크 아웃할 폴더에 체크 아웃합니다.

*itemIDs*                   체크 아웃할 아이템의 ID

*checkoutFolders*       체크 아웃할 폴더 이름

**Argument** *localFileTimes*       체크 아웃할 아이템의 로컬 파일 시간

*isCompressed*           압축 여부

*checkoutCmts*           주석

---

## ■ CheckIn

	<code>public bool[] CheckIn(string[] itemIDs, bool[] isCompressed, Stream[] items, string comment, bool[] keepCheckout)</code>
<b>Prototype</b>	<code>public bool[] CheckIn(string[] itemIDs, bool[] isCompressed, Stream[] items, string comment, bool[] keepCheckout, long[] datetimes)</code>
<b>Definition</b>	시간을 지정하여 아이템을 체크인합니다.(NONE 타입은 지원 안 함)
<b>Argument</b>	<i>itemIDs</i> 체크인할 아이템의 ID
	<i>isCompressed</i> 압축 여부
	<i>items</i> 아이템의 입력 스트림
	<i>comment</i> 주석문
	<i>keepCheckout</i> 체크 아웃 상태를 유지할지 여부
	<i>datetimes</i> 설정할 시간 0 또는 음수로 설정할 경우 현재 시간으로 적용됨

## ■ UndoCheckout

<b>Prototype</b>	<code>public Stream[] UndoCheckout(string[] itemIDs, bool[] replaceItems, bool[] isCompressed)</code>
<b>Definition</b>	지정한 아이템을 체크 아웃 취소합니다.
<b>Argument</b>	<i>itemIDs</i> 체크 아웃 취소할 아이템의 ID
	<i>replaceItems</i> 로컬 작업 폴더의 아이템을 가져올지 여부
	<i>isCompressed</i> 압축 여부

## ■ IsCheckoutUser

<b>Prototype</b>	<code>public bool[] isCheckoutUser(string[] ItemIDs) throws OZCPEXception</code>
<b>Definition</b>	현재 사용자가 지정한 아이템을 체크 아웃했는지 여부를 확인합니다.
<b>Argument</b>	<i>itemIDs</i> 아이템 ID

## ■ RollBack

<b>Prototype</b>	<code>public bool RollBack(string itemID, int version, string comment)</code>
<b>Definition</b>	지정한 아이템을 지정한 버전으로 복원시킵니다.
<b>Argument</b>	<i>itemID</i> 복원시킬 아이템의 ID
	<i>version</i> 복원시킬 버전
	<i>comment</i> 주석문

■ **GetItem**

<b>Prototype</b>	public Stream GetItem(string itemID, int version, bool isCompressed)						
<b>Definition</b>	지정한 아이템 ID에 해당하는 아이템 중 지정한 버전의 아이템을 가져옵니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>itemID</i></td> <td>가져올 아이템의 ID</td> </tr> <tr> <td><i>version</i></td> <td>가져올 아이템의 버전</td> </tr> <tr> <td><i>isCompressed</i></td> <td>압축 여부</td> </tr> </table>	<i>itemID</i>	가져올 아이템의 ID	<i>version</i>	가져올 아이템의 버전	<i>isCompressed</i>	압축 여부
<i>itemID</i>	가져올 아이템의 ID						
<i>version</i>	가져올 아이템의 버전						
<i>isCompressed</i>	압축 여부						

■ **GetHistoryInfos**

<b>Prototype</b>	public IHistoryInfo[] GetHistoryInfos(string itemID)
<b>Definition</b>	지정한 아이템의 히스토리 정보를 가져옵니다.
<b>Argument</b>	<i>itemID</i> 히스토리 정보를 가져올 아이템의 ID

■ **GetHistoryItemListByDatetime**

<b>Prototype</b>	public IHistoryInfo[] GetHistoryItemListByDatetime(string itemID)
<b>Definition</b>	지정한 아이템의 히스토리 정보를 날짜순으로 가져옵니다.(NONE 타입은 지원 안 함)
<b>Argument</b>	<i>itemID</i> 히스토리 정보를 가져올 아이템의 ID

■ **GetHistoryItemByDatetime**

<b>Prototype</b>	public Stream GetHistoryItemByDatetime(string itemID, long datetime, bool isCompressed, bool isObjStream)								
<b>Definition</b>	지정한 시간 기준으로 최근 아이템을 가져옵니다.(NONE 타입은 지원 안 함)								
<b>Argument</b>	<table border="0"> <tr> <td><i>itemID</i></td> <td>가져올 아이템 ID</td> </tr> <tr> <td><i>datetime</i></td> <td>가져올 시간</td> </tr> <tr> <td><i>isCompressed</i></td> <td>압축 여부</td> </tr> <tr> <td><i>isObjStream</i></td> <td>가져올 아이템이 ODI인 경우 obj stream으로 가져올지 여부</td> </tr> </table>	<i>itemID</i>	가져올 아이템 ID	<i>datetime</i>	가져올 시간	<i>isCompressed</i>	압축 여부	<i>isObjStream</i>	가져올 아이템이 ODI인 경우 obj stream으로 가져올지 여부
<i>itemID</i>	가져올 아이템 ID								
<i>datetime</i>	가져올 시간								
<i>isCompressed</i>	압축 여부								
<i>isObjStream</i>	가져올 아이템이 ODI인 경우 obj stream으로 가져올지 여부								

■ **GetDeletedItemHistoryInfos**

<b>Prototype</b>	public IHistoryInfo[] GetDeletedItemHistoryInfos(string itemID)				
<b>Definition</b>	해당 삭제된 아이템의 정보를 가져옵니다.				
<b>Argument</b>	<table border="0"> <tr> <td>※ 참고사항 :</td> <td>"toBeDestroyed=false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.</td> </tr> <tr> <td><i>itemID</i></td> <td>정보를 가져올 아이템의 ID</td> </tr> </table>	※ 참고사항 :	"toBeDestroyed=false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.	<i>itemID</i>	정보를 가져올 아이템의 ID
※ 참고사항 :	"toBeDestroyed=false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.				
<i>itemID</i>	정보를 가져올 아이템의 ID				

### ■ RemoveHistory

<b>Prototype</b>	public bool RemoveHistory(string itemID, int version)	
<b>Definition</b>	지정한 아이템에 대해 특정 버전의 히스토리를 삭제합니다.	
<b>Argument</b>	<i>itemID</i>	히스토리를 삭제할 아이템의 ID
	<i>version</i>	히스토리를 삭제할 버전

### ■ CreateCategories

<b>Prototype</b>	public string[] CreateCategories(string[] categoryNames, string[] parentCategoryIDs, string comment)	
<b>Definition</b>	카테고리를 새로 생성하고 생성된 카테고리의 ID를 반환합니다.	
<b>Argument</b>	<i>categoryName</i>	생성할 카테고리의 이름
	<i>parentCategoryIDs</i>	생성할 카테고리의 상위 카테고리 ID
	<i>comment</i>	주석문

### ■ ModifyCategoryDesc

<b>Prototype</b>	public bool ModifyCategoryDesc(string categoryID, string description)	
<b>Definition</b>	카테고리 설명을 변경합니다.(NONE 타입은 지원 안 함)	
<b>Argument</b>	<i>categoryID</i>	카테고리 ID(Full Path로 설정)
	<i>description</i>	카테고리 설명

### ■ ModifyCategoryName

<b>Prototype</b>	public string ModifyCategoryName(string categoryID, string newCategoryName, string comment)	
<b>Definition</b>	지정한 카테고리 ID에 해당하는 카테고리의 이름을 변경합니다.	
<b>Argument</b>	<i>categoryID</i>	이름을 변경할 카테고리의 ID
	<i>newCategoryName</i>	변경할 카테고리 이름
	<i>comment</i>	주석문

### ■ DeleteCategories

<b>Prototype</b>	public bool[] DeleteCategories(string[] categoryIDs, bool[] toBeDestroyed, string comment)	
<b>Definition</b>	지정한 카테고리 ID에 해당하는 카테고리를 삭제합니다.	
<b>Argument</b>	<i>categoryIDs</i>	삭제할 카테고리의 ID
	<i>toBeDestroyed</i>	카테고리를 영구 삭제할 지 여부
	<i>comment</i>	주석문

### ■ UnDeleteCategories

<b>Prototype</b>	<code>public bool[] UndeleteCategories(string[] categoryIDs, string comment)</code>
<b>Definition</b>	삭제된 카테고리를 복원하고 복원 성공 여부를 가져옵니다. ※ 참고사항 : 카테고리 삭제 시 "toBeDestroyed=false"로 설정하였을 경우 삭제된 카테고리의 복원이 가능합니다.
<b>Argument</b>	<i>categoryIDs</i> 삭제된 카테고리의 ID <i>comment</i> 주석문

### ■ GetItemCount

<b>Prototype</b>	<code>public int GetItemCount(string categoryID)</code>
<b>Definition</b>	지정한 카테고리에 속해 있는 모든 아이템의 개수를 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 아이템의 개수를 얻을 카테고리의 ID

### ■ GetItemInfos

<b>Prototype</b>	<code>public IItemInfo[] GetItemInfos(string categoryID)</code>
<b>Definition</b>	지정한 카테고리에 속해 있는 모든 아이템 정보를 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 아이템 정보를 얻을 카테고리 ID

### ■ GetCategoryInfos

<b>Prototype</b>	<code>public ICategoryInfo[] GetCategoryInfos(string categoryID)</code>
<b>Definition</b>	지정한 카테고리에 속해 있는 모든 하위 카테고리 정보를 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 아이템의 개수를 얻을 카테고리의 ID

### ■ GetCategoryID

<b>Prototype</b>	<code>public string GetCategoryID(string itemID)</code>
<b>Definition</b>	지정한 아이템이 존재하는 카테고리의 ID를 반환합니다.
<b>Argument</b>	<i>itemID</i> 카테고리의 ID를 얻을 아이템 ID

### ■ GetCategoryInfo

<b>Prototype</b>	<code>public ICategoryInfo GetCategoryInfo(string categoryID)</code>
<b>Definition</b>	지정한 카테고리의 정보를 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 정보를 가져올 카테고리 ID

### ■ GetDeletedItemInfos

<b>Prototype</b>	<code>public IItemInfo[] GetDeletedItemInfos(string categoryID)</code>
------------------	--

---

	지정한 카테고리의 삭제된 아이템 정보를 가져옵니다.
<b>Definition</b>	※ 참고사항 : "toBeDestroyed=false"로 설정하여 삭제한 아이템의 정보만 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 삭제된 아이템 정보를 가져올 카테고리 ID

---

#### ■ TransferCategory

---

<b>Prototype</b>	public bool TransferCategory(string categoryID, string targetCategoryID)
<b>Definition</b>	지정한 카테고리를 다른 카테고리로 이동하고 카테고리 이동 성공 여부를 가져옵니다.
<b>Argument</b>	<i>categoryID</i> 카테고리를 이동할 카테고리 ID <i>targetCategoryID</i> 이동할 카테고리 ID

---

#### ■ TransferItems

---

<b>Prototype</b>	public bool TransferItems(string[] itemIDs, string targetCategoryID)
<b>Definition</b>	지정한 아이템의 카테고리를 이동하고 카테고리 이동 성공 여부를 가져옵니다
<b>Argument</b>	<i>itemIDs</i> 카테고리를 이동할 아이템 ID <i>targetCategoryID</i> 이동할 카테고리 ID

---

#### ■ SearchItem

---

<b>Prototype</b>	public ISearchResult[] SearchItem(string categoryID, string checkOutUser, string startDate, string endDate, SearchOptions options, string fileFilter, string criteria)
<b>Definition</b>	지정한 카테고리의 아이템 중 지정한 옵션 및 조회 조건에 해당하는 아이템을 가져옵니다
<b>Argument</b>	<i>categoryID</i> 카테고리를 이동할 아이템 ID <i>checkOutUser</i> 체크아웃 계정 <i>startDate</i> 시작일 <i>endDate</i> 종료일 <i>options</i> 조회 옵션 <i>fileFilter</i> 파일 확장자 <i>criteria</i> 조회 조건

---

#### ■ CreateUser

---

<b>Prototype</b>	public string CreateUser(string userName, string password, string description)
------------------	--

---

---

**Definition** 새로운 사용자를 생성하고 생성된 사용자의 ID를 반환합니다.

*userName* 사용자 이름

**Argument** *password* 패스워드

*description* 사용자에 대한 설명 내용

---

#### ■ ModifyUserName

---

**Prototype** `public string ModifyUserName(string userID, string userName)`

**Definition** 지정한 사용자 ID에 해당하는 사용자의 이름을 변경하고 변경 성공 여부를 반환합니다.

*userID* 사용자 ID

**Argument** *userName* 변경할 사용자 이름

---

#### ■ ModifyUserPassword

---

**Prototype** `public bool ModifyUserPassword(string userID, string oldPwd, string newPwd)`

**Definition** 지정한 사용자 ID에 해당하는 사용자의 패스워드를 변경하고 변경 성공 여부를 반환합니다.

*userID* 사용자 ID

**Argument** *oldPwd* 변경 전 패스워드

*newPwd* 변경 후 패스워드

---

#### ■ ModifyUserDescription

---

**Prototype** `public bool ModifyUserDescription(string userID, string description)`

**Definition** 지정한 사용자 ID에 해당하는 사용자의 설명 내용을 변경하고 변경 성공 여부를 반환합니다.

*userID* 사용자 ID

**Argument** *description* 변경할 설명 내용

---

#### ■ DeleteUser

---

**Prototype** `public bool DeleteUser(string userID)`

**Definition** 지정한 사용자 ID에 해당하는 사용자의 모든 정보를 삭제하고 삭제 성공 여부를 반환합니다.

**Argument** *userID* 삭제할 사용자의 ID

---

#### ■ GetUserInfo

---

**Prototype** `public IUserInfo GetUserInfo(string userID)`

---

---

**Definition**    지정한 사용자 ID에 해당하는 사용자의 사용자 정보를 가져옵니다.

---

**Argument**    *userID*            사용자 정보를 가져올 사용자 ID

---

#### ■ CheckPassword

---

**Prototype**    public bool CheckPassword(string userID, string password)

---

**Definition**    사용자의 패스워드가 맞는지 여부를 체크합니다.

---

**Argument**    *userID*            패스워드를 체크할 사용자 ID

---

**Argument**    *password*        체크할 패스워드

---

#### ■ GetUserInfos

---

**Prototype**    public IUserInfo[] GetUserInfos()

---

**Definition**    모든 사용자의 정보를 가져옵니다.

---

#### ■ DisableLogin

---

**Prototype**    public void DisableLogin(string userID)

---

**Definition**    지정한 사용자 ID의 사용자를 로그인하지 못하도록 설정합니다.

---

**Argument**    *userID*            로그인하지 못하도록 할 사용자 ID

---

#### ■ EnableLogin

---

**Prototype**    public void EnableLogin(string userID)

---

**Definition**    지정한 사용자 ID의 사용자를 로그인 가능하도록 설정합니다

---

**Argument**    *userID*            로그인 가능하도록 할 사용자 ID

---

#### ■ IsLoginEnabled

---

**Prototype**    public bool IsLoginEnabled(string userID)

---

**Definition**    지정한 사용자 ID의 사용자가 로그인이 가능한지 여부를 가져옵니다.

---

**Argument**    *userID*            로그인 가능 여부를 체크할 사용자 ID

---

#### ■ CreateGroup

---

**Prototype**    public string CreateGroup(string groupName, string  
parentGroupID, string description)

---

**Definition**    새로운 그룹을 생성하고 생성된 그룹의 ID를 반환합니다.

---

**Argument**    *groupName*        새로 생성할 그룹의 이름

---

**Argument**    *parentGroupID*    새로 생성할 그룹의 상위 그룹 ID

---

**Argument**    *description*        그룹에 대한 설명 내용

---

### ■ ModifyGroupName

<b>Prototype</b>	public string ModifyGroupName(string groupID, string groupName)				
<b>Definition</b>	지정한 사용자 ID에 해당하는 사용자의 이름을 변경하고 변경 성공 여부를 반환합니다.				
<b>Argument</b>	<table border="0"> <tr> <td><i>groupID</i></td> <td>그룹의 이름을 변경할 그룹 ID</td> </tr> <tr> <td><i>groupName</i></td> <td>변경할 그룹의 이름</td> </tr> </table>	<i>groupID</i>	그룹의 이름을 변경할 그룹 ID	<i>groupName</i>	변경할 그룹의 이름
<i>groupID</i>	그룹의 이름을 변경할 그룹 ID				
<i>groupName</i>	변경할 그룹의 이름				

### ■ ModifyGroupDescription

<b>Prototype</b>	public bool ModifyGroupDescription(string groupID, string description)				
<b>Definition</b>	지정한 사용자 ID에 해당하는 사용자의 설명을 변경하고 변경 성공 여부를 반환합니다.				
<b>Argument</b>	<table border="0"> <tr> <td><i>groupID</i></td> <td>그룹의 설명을 변경할 그룹 ID</td> </tr> <tr> <td><i>description</i></td> <td>변경할 그룹의 설명</td> </tr> </table>	<i>groupID</i>	그룹의 설명을 변경할 그룹 ID	<i>description</i>	변경할 그룹의 설명
<i>groupID</i>	그룹의 설명을 변경할 그룹 ID				
<i>description</i>	변경할 그룹의 설명				

### ■ DeleteGroup

<b>Prototype</b>	public bool DeleteGroup(string groupID)
<b>Definition</b>	지정한 그룹 ID에 해당하는 그룹을 삭제하고 삭제 성공 여부를 반환합니다.
<b>Argument</b>	<i>groupID</i> 삭제할 그룹 ID

### ■ CreateUserInGroup

<b>Prototype</b>	public string CreateUserInGroup(string uName, string pwd, string gID, string desc) throws OZCPEException								
<b>Definition</b>	지정한 그룹에 새로운 사용자를 생성하고 생성된 사용자의 ID를 반환합니다.								
<b>Argument</b>	<table border="0"> <tr> <td><i>uName</i></td> <td>새로 생성할 사용자의 이름</td> </tr> <tr> <td><i>Pwd</i></td> <td>패스워드</td> </tr> <tr> <td><i>gID</i></td> <td>사용자를 생성할 그룹 ID</td> </tr> <tr> <td><i>desc</i></td> <td>사용자에 대한 설명 내용</td> </tr> </table>	<i>uName</i>	새로 생성할 사용자의 이름	<i>Pwd</i>	패스워드	<i>gID</i>	사용자를 생성할 그룹 ID	<i>desc</i>	사용자에 대한 설명 내용
<i>uName</i>	새로 생성할 사용자의 이름								
<i>Pwd</i>	패스워드								
<i>gID</i>	사용자를 생성할 그룹 ID								
<i>desc</i>	사용자에 대한 설명 내용								

### ■ TransferUser

<b>Prototype</b>	public bool TransferUser(string userID, string newGroupID)				
<b>Definition</b>	지정한 사용자의 그룹을 이동하고 그룹 이동 성공 여부를 가져옵니다				
<b>Argument</b>	<table border="0"> <tr> <td><i>userID</i></td> <td>그룹을 이동할 사용자 ID</td> </tr> <tr> <td><i>newGroupID</i></td> <td>이동할 그룹 ID</td> </tr> </table>	<i>userID</i>	그룹을 이동할 사용자 ID	<i>newGroupID</i>	이동할 그룹 ID
<i>userID</i>	그룹을 이동할 사용자 ID				
<i>newGroupID</i>	이동할 그룹 ID				

### ■ TransferGroup

<b>Prototype</b>	public bool TransferGroup(string groupID, string targetGroupID)				
<b>Definition</b>	지정한 그룹을 다른 그룹으로 이동하고 그룹 이동 성공 여부를 가져옵니다.				
<b>Argument</b>	<table border="0"> <tr> <td><i>groupID</i></td> <td>그룹을 이동할 그룹 ID</td> </tr> <tr> <td><i>targetGroupID</i></td> <td>이동할 그룹 ID</td> </tr> </table>	<i>groupID</i>	그룹을 이동할 그룹 ID	<i>targetGroupID</i>	이동할 그룹 ID
<i>groupID</i>	그룹을 이동할 그룹 ID				
<i>targetGroupID</i>	이동할 그룹 ID				

### ■ GetUserInfo

<b>Prototype</b>	public IUserInfo[] GetUserInfos(string groupID)
<b>Definition</b>	지정한 그룹 ID에 등록되어 있는 모든 사용자의 정보를 가져옵니다.
<b>Argument</b>	<i>groupID</i> 사용자의 정보를 가져올 그룹 ID

### ■ GetUserInfoListInGroup

<b>Prototype</b>	public IUserInfo[] GetUserInfoListInGroup(string groupID, bool isRecursive)				
<b>Definition</b>	그룹에 추가된 사용자 정보를 가져옵니다.(NONE 타입은 지원 안 함)				
<b>Argument</b>	<table border="0"> <tr> <td><i>groupID</i></td> <td>사용자의 정보를 가져올 그룹 ID</td> </tr> <tr> <td><i>isRecursive</i></td> <td>하위 그룹 사용자 포함 여부</td> </tr> </table>	<i>groupID</i>	사용자의 정보를 가져올 그룹 ID	<i>isRecursive</i>	하위 그룹 사용자 포함 여부
<i>groupID</i>	사용자의 정보를 가져올 그룹 ID				
<i>isRecursive</i>	하위 그룹 사용자 포함 여부				

### ■ GetGroupInfo

<b>Prototype</b>	public IGroupInfo GetGroupInfo(string groupID)
<b>Definition</b>	지정한 그룹 ID에 해당하는 그룹의 정보를 가져옵니다.
<b>Argument</b>	<i>groupID</i> 그룹 정보를 가져올 그룹 ID

### ■ GetGroupInfoListInGroup

<b>Prototype</b>	public IGroupInfo[] GetGroupInfoListInGroup(string groupID, bool isRecursive)				
<b>Definition</b>	그룹에 추가된 그룹 정보를 가져옵니다.(NONE 타입은 지원 안 함)				
<b>Argument</b>	<table border="0"> <tr> <td><i>groupID</i></td> <td>그룹 ID(Full Path로 설정)</td> </tr> <tr> <td><i>isRecursive</i></td> <td>하위 그룹 포함 여부</td> </tr> </table>	<i>groupID</i>	그룹 ID(Full Path로 설정)	<i>isRecursive</i>	하위 그룹 포함 여부
<i>groupID</i>	그룹 ID(Full Path로 설정)				
<i>isRecursive</i>	하위 그룹 포함 여부				

### ■ GetSubGroupInfos

<b>Prototype</b>	public IGroupInfo[] GetSubGroupInfos(string groupID)
<b>Definition</b>	지정한 그룹의 하위 그룹 정보를 가져옵니다.
<b>Argument</b>	<i>groupID</i> 하위 그룹의 정보를 가져올 그룹 ID

■ **GetParentGroupInfo**

<b>Prototype</b>	public IGroupInfo GetParentGroupInfo(string groupID)
<b>Definition</b>	지정한 그룹의 상위 그룹의 정보를 가져옵니다.
<b>Argument</b>	<i>groupID</i> 상위 그룹의 정보를 가져올 그룹 ID

■ **GetGroupID**

<b>Prototype</b>	public string GetGroupID(string userID)
<b>Definition</b>	지정한 사용자가 속해 있는 그룹의 정보를 가져옵니다.
<b>Argument</b>	<i>userID</i> 그룹 정보를 가져올 사용자 ID

■ **AddGroupAdministrator**

<b>Prototype</b>	public bool AddGroupAdministrator(string userID, string groupID)
<b>Definition</b>	지정한 그룹의 그룹 관리자를 추가하고 추가 성공 여부를 반환합니다.
<b>Argument</b>	<i>userID</i> 그룹 관리자로 추가할 사용자 ID <i>groupID</i> 그룹 관리자를 추가할 그룹 ID

■ **RemoveGroupAdministrator**

<b>Prototype</b>	public bool RemoveGroupAdministrator(string userID, string groupID)
<b>Definition</b>	지정한 그룹의 그룹 관리자를 해제하고 해제 성공 여부를 반환합니다.
<b>Argument</b>	<i>userID</i> 그룹 관리자 권한을 해제할 사용자 ID <i>groupID</i> 그룹 관리자를 해제할 그룹 ID

■ **IsGroupAdministrator**

<b>Prototype</b>	public bool IsGroupAdministrator(string userID, string groupID)
<b>Definition</b>	지정한 사용자 ID에 해당하는 사용자가 해당 그룹의 관리자인지 여부를 확인합니다.
<b>Argument</b>	<i>userID</i> 관리자인지 체크할 사용자 ID <i>groupID</i> 그룹 ID

■ **GetGroupAdministrators**

<b>Prototype</b>	public IUserInfo[] GetGroupAdministrators(string groupID)
<b>Definition</b>	지정한 그룹의 그룹 관리자 정보를 가져옵니다.

---

**Argument**    *groupID*    그룹 관리자 정보를 가져올 그룹 ID

---

#### ■ IsAdministrator

---

**Prototype**    public bool IsAdministrator(string userID)

---

**Definition**    지정한 사용자 ID가 관리자인지 여부를 확인합니다.

---

**Argument**    *userID*    관리자인지 체크할 사용자 ID

---

#### ■ AddAdministrator

---

**Prototype**    public bool AddAdministrator(string userID)

---

**Definition**    지정한 사용자 ID를 관리자로 설정합니다.(NONE 타입은 지원 안 함)

---

**Argument**    *userID*    관리자로 설정할 사용자 ID

---

#### ■ RemoveAdministrator

---

**Prototype**    public bool RemoveAdministrator(string userID)

---

**Definition**    지정한 사용자 ID에게 설정된 관리자 권한을 삭제합니다.(NONE 타입은 지원 안 함)

---

**Argument**    *userID*    관리자 권한을 삭제할 사용자 ID

---

#### ■ Distribute

---

**Prototype**    public Stream Distribute(string categoryID, bool isRecursive)

---

**Definition**    지정한 카테고리의 아이템을 배포 파일 형식으로 저장합니다.

---

**Argument**    *categoryID*    카테고리 ID

---

**Argument**    *isRecursive*    하위 카테고리 포함 여부

---

#### ■ DistributeRepository

---

**Prototype**    public Stream DistributeRepository(string[] itemIDs)

---

**Definition**    아이템을 배포 파일 형식으로 저장합니다.

---

**Argument**    *itemIDs*    아이템 ID(Full Path로 설정)

---

#### ■ DistributeRepositoryByDatetime

---

**Prototype**    public Stream DistributeRepositoryByDatetime(string[] itemIDs)

---

**Definition**    지정한 아이템마다 가장 마지막에 체크인된 아이템을 가져와서 배포 파일 형식으로 저장합니다.(NONE 타입은 지원 안 함)

---

**Argument**    *itemIDs*    아이템 ID(Full Path로 설정)

---

### ■ Upload

<b>Prototype</b>	public bool Upload(string categoryID, Stream in)	
<b>Definition</b>	압축된 아이템을 지정한 카테고리에 압축 해제한 후 압축 해제 성공 여부를 반환합니다.	
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>in</i>	압축한 스트림

### ■ ModifyUserAuthorityToItem

<b>Prototype</b>	public bool ModifyUserAuthorityToItem(string userID, string itemID, Authority permission)	
<b>Definition</b>	지정한 아이템에 대한 사용자의 권한 정보를 변경하고 변경 성공 여부를 가져옵니다.	
<b>Argument</b>	<i>userID</i>	사용자 ID
	<i>itemID</i>	아이템 ID
	<i>permission</i>	권한 정보

### ■ ModifyUserAuthorityToCategory

<b>Prototype</b>	public bool ModifyUserAuthorityToCategory(string userID, string categoryID, Authority permission)	
<b>Definition</b>	지정한 카테고리 ID에 대한 지정한 사용자 ID의 권한 정보를 변경하고 변경 성공 여부를 반환합니다.	
<b>Argument</b>	<i>userID</i>	사용자 ID
	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

### ■ ModifyGroupAuthorityToItem

<b>Prototype</b>	public bool ModifyGroupAuthorityToItem(string groupID, string itemID, Authority permission)	
<b>Definition</b>	지정한 아이템에 대한 지정한 그룹 ID의 권한 정보를 변경하고 변경 성공 여부를 반환합니다.	
<b>Argument</b>	<i>groupID</i>	그룹 ID
	<i>itemID</i>	아이템 ID
	<i>permission</i>	권한 정보

### ■ ModifyGroupAuthorityToCategory

<b>Prototype</b>	public bool ModifyGroupAuthorityToCategory(string groupID, string categoryID, Authority permission)	
------------------	---	--

<b>Definition</b>	지정한 카테고리 ID에 대한 지정한 그룹 ID의 권한 정보를 변경하고 변경 성공 여부를 반환합니다.	
	<i>groupID</i>	그룹 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

#### ■ GetUserAuthorityToItem

<b>Prototype</b>	public Authority GetUserAuthorityToItem(string userID, string itemID)	
<b>Definition</b>	지정한 아이템에 대한 지정한 사용자 ID의 권한 정보를 가져옵니다.	
	<i>userID</i>	사용자 ID
<b>Argument</b>	<i>itemID</i>	아이템 ID

#### ■ GetUserAuthorityToCategory

<b>Prototype</b>	public Authority GetUserAuthorityToCategory(string userID, string categoryID)	
<b>Definition</b>	지정한 카테고리에 대한 지정한 사용자 ID의 권한 정보를 가져옵니다.	
	<i>userID</i>	사용자 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID

#### ■ GetGroupAuthorityToItem

<b>Prototype</b>	public Authority GetGroupAuthorityToItem(string groupID, string itemID)	
<b>Definition</b>	지정한 아이템에 대한 지정한 그룹 ID의 권한 정보를 가져옵니다.	
	<i>groupID</i>	그룹 ID
<b>Argument</b>	<i>itemID</i>	아이템 ID

#### ■ GetGroupAuthorityToCategory

<b>Prototype</b>	public Authority GetGroupAuthorityToCategory(string groupID, string categoryID)	
<b>Definition</b>	지정한 카테고리 ID에 대한 지정한 그룹 ID의 권한 정보를 가져옵니다.	
	<i>groupID</i>	그룹 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID

### ■ GetGroupInfosOfItem

---

<b>Prototype</b>	public IGroupInfo[] GetGroupInfosOfItem(string itemID, Authority permission)				
<b>Definition</b>	지정한 아이템 ID에 대해 permission 이상의 권한이 부여된 모든 그룹 정보를 가져옵니다.				
<b>Argument</b>	<table> <tr> <td><i>itemID</i></td> <td>아이템 ID</td> </tr> <tr> <td><i>permission</i></td> <td>권한 정보</td> </tr> </table>	<i>itemID</i>	아이템 ID	<i>permission</i>	권한 정보
<i>itemID</i>	아이템 ID				
<i>permission</i>	권한 정보				

---

### ■ GetGroupInfosOfCategory

---

<b>Prototype</b>	public IGroupInfo[] GetGroupInfosOfCategory(string categoryID, Authority permission)				
<b>Definition</b>	지정한 카테고리 ID에 대해 permission 이상의 권한이 부여된 모든 그룹 정보를 가져옵니다.				
<b>Argument</b>	<table> <tr> <td><i>categoryID</i></td> <td>카테고리 ID</td> </tr> <tr> <td><i>permission</i></td> <td>권한 정보</td> </tr> </table>	<i>categoryID</i>	카테고리 ID	<i>permission</i>	권한 정보
<i>categoryID</i>	카테고리 ID				
<i>permission</i>	권한 정보				

---

### ■ GetUserInfosOfItem

---

<b>Prototype</b>	public IUserInfo[] GetUserInfosOfItem(string itemID, Authority permission)				
<b>Definition</b>	지정한 아이템 ID에 대해 permission 이상의 권한이 부여된 모든 사용자 정보를 가져옵니다.				
<b>Argument</b>	<table> <tr> <td><i>itemID</i></td> <td>아이템 ID</td> </tr> <tr> <td><i>permission</i></td> <td>권한 정보</td> </tr> </table>	<i>itemID</i>	아이템 ID	<i>permission</i>	권한 정보
<i>itemID</i>	아이템 ID				
<i>permission</i>	권한 정보				

---

### ■ GetUserInfosOfCategory

---

<b>Prototype</b>	public IUserInfo[] GetUserInfosOfCategory(string categoryID, Authority permission)				
<b>Definition</b>	지정한 카테고리 ID에 대해 permission 이상의 권한이 부여된 모든 사용자 정보를 가져옵니다.				
<b>Argument</b>	<table> <tr> <td><i>categoryID</i></td> <td>카테고리 ID</td> </tr> <tr> <td><i>permission</i></td> <td>권한 정보</td> </tr> </table>	<i>categoryID</i>	카테고리 ID	<i>permission</i>	권한 정보
<i>categoryID</i>	카테고리 ID				
<i>permission</i>	권한 정보				

---

### ■ GetCategoryInfosOfUser

---

<b>Prototype</b>	public ICategoryInfo[] GetCategoryInfosOfUser(string userID, string categoryID, Authority permission)
<b>Definition</b>	지정한 카테고리의 하위 카테고리 중에 지정한 사용자의 권한이 permission 이상인 모든 카테고리의 목록을 가져옵니다.

---

	<i>userID</i>	사용자 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

#### ■ GetCategoryInfosOfGroup

<b>Prototype</b>	public ICategoryInfo[] GetCategoryInfosOfGroup(string groupID, string categoryID, Authority permission)	
<b>Definition</b>	지정한 카테고리의 하위 카테고리 중에 지정한 그룹 ID의 권한이 permission 이상인 모든 카테고리의 목록을 가져옵니다.	
	<i>groupID</i>	그룹 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

#### ■ GetItemInfosOfUser

<b>Prototype</b>	public IItemInfo[] GetItemInfosOfUser(string userID, string categoryID, Authority permission)	
<b>Definition</b>	지정한 카테고리의 아이템 중에 지정한 사용자의 권한이 permission 이상인 모든 아이템 목록을 가져옵니다.	
	<i>userID</i>	사용자 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

#### ■ GetItemInfosOfGroup

<b>Prototype</b>	public IItemInfo[] GetItemInfosOfGroup(string groupID, string categoryID, Authority permission)	
<b>Definition</b>	지정한 카테고리의 아이템 중에 지정한 그룹 ID의 권한이 permission 이상인 모든 아이템 목록을 가져옵니다.	
	<i>groupID</i>	그룹 ID
<b>Argument</b>	<i>categoryID</i>	카테고리 ID
	<i>permission</i>	권한 정보

#### ■ SetUserDefinedResourceSync

<b>Prototype</b>	public int SetUserDefinedResourceSync(string alias, string[] itemIDs, string comment, string[] errorMsg)	
<b>Definition</b>	보고서를 사용자가 정의한 리소스로 동기화합니다.	
<b>Argument</b>	<i>alias</i>	ozudr.properties에서 설정한 앨리어스 이름

	<i>itemIDs</i>	보고서 전체 경로
	<i>comment</i>	주석
	<i>errorMsg</i>	에러 메시지
<b>Return</b>	<i>responseCode</i>	응답 코드(1 : 전체 성공, 2 : 부분 성공, 3 : 실패)

■ **EncryptFile**

	<code>public OZAttributeList EncryptFile(string[] itemIDs)</code>	
<b>Prototype</b>	<code>public OZAttributeList EncryptFile(string categoryID, bool isRecursive)</code>	
<b>Definition</b>	OZR 파일을 암호화하여 저장하고, 암호화를 실패한 파일 리스트를 리턴합니다. 이미 암호화된 아이템은 다시 암호화하지 않습니다.	
<b>Argument</b>	<i>itemIDs</i>	암호화할 아이템 ID(Full Path로 설정)
	<i>categoryID</i>	암호화할 아이템을 가진 카테고리 ID(Full Path로 설정)
	<i>isRecursive</i>	하위 카테고리 포함 여부

■ **DecryptFile**

	<code>public OZAttributeList DecryptFile(string[] itemIDs)</code>	
<b>Prototype</b>	<code>public OZAttributeList DecryptFile(string categoryID, bool isRecursive)</code>	
<b>Definition</b>	OZR 파일을 복호화하여 저장하고, 복호화를 실패한 파일 리스트를 리턴합니다. 이미 복호화된 아이템은 다시 복호화하지 않습니다.	
<b>Argument</b>	<i>itemIDs</i>	복호화할 아이템 ID(Full Path로 설정)
	<i>categoryID</i>	복호화할 아이템을 가진 카테고리 ID(Full Path로 설정)
	<i>isRecursive</i>	하위 카테고리 포함 여부

## 오즈 리파지토리 구현

### 리파지토리 매니저 호출 방식

리파지토리 매니저에서 API(oz.framework.cp.client.OZFrameworkAPI)를 사용하는 방법은 기존과 동일하며 리파지토리 매니저에서 API를 호출할 경우 아래와 같은 순서로 API가 처리됩니다.

- 리파지토리 매니저에서 서버에 요청하기 전에 리파지토리 매니저가 새로운 CP를 지원하는 버전인지 확인합니다.
- 리파지토리 매니저가 새로운 CP를 지원하는 버전일 경우 현재 호출한 매개 변수를 새 버전의 CP에 맞게 변환하여 새로 추가된 API를 호출합니다.
- 리파지토리 매니저가 새로운 CP를 지원하지 않는 버전일 경우 이전과 동일한 방식으로 API를 호출합니다.

※ 참고사항 : 새로 추가된 API는 서버의 RepositoryEx API와 형태가 동일하며 클라이언트에서 직접 사용하지 않습니다.

### 개발 시 주의사항

- Adapter 추가 개발 시 IOZRepository 인터페이스를 반드시 구현하여야 합니다.
- Adapter 추가 개발 시 Category를 개발할 경우 반드시 Item을 개발하여야 하며 Group을 개발하는 경우에는 반드시 User를 개발하여야 합니다.
- 현재 리파지토리에서 지원하지 않는 기능은 다음과 같습니다.
  - 카테고리, 아이템 등을 삭제한 후 다시 복원하는 기능은 지원하지 않습니다.
  - 주석문 기능은 지원하지 않습니다.

### Sample : RepositoryExSample.cs

```
using System;
using System.IO;
using System.Reflection;
```

```

using System.Collections.Specialized;
using RepositoryEx = oz.framework.api.RepositoryEx;
using oz.framework.repositoryex.info;

namespace sample
{
    /// <summary>
    /// 전체 경로를 ID로 사용하는 Repository에 접근하기 위한 API 샘플
    /// </summary>
    public class RepositoryExTest
    {
        public static void Main()
        {
            const string URL = "http://127.0.0.1/oz/server.aspx"; // 서버가 구동중인
            URL
            const string ID = "admin"; // default
            const string PASSWORD = "admin"; // default

            using(RepositoryEx repository = new RepositoryEx(URL, ID, PASSWORD,
            false))
            {
                // 테스트를 위해 임시로 리파지토리 설정을 변경합니다
                NameValueCollection prevConf = repository.GetConfiguration();

                NameValueCollection tempConf = new NameValueCollection();
                tempConf["REPOSITORY_TYPE"] = "BUILTIN";
                tempConf["REPOSITORY_FILE_PATH"] = Path.GetTempPath();
                tempConf["REPOSITORY_ITEM_NUMBER_PER_DIRECTORY"] = "100";
                tempConf["REPOSITORY_HISTORY_ITEM_VALID_DAYS"] = "20";
                repository.SetConfiguration(tempConf);
                repository.Reload();

                string itemID = null;

                string categoryID = repository.CreateCategories(new
                string[]{"test"}, new string[]{"/"}, "comment")[0];

                // 아이템 생성, 체크아웃/인, 변경, 삭제
                using(Stream @in =
                Assembly.GetExecutingAssembly().GetManifestResourceStream("sample.parameter_test.odi"))
                {
                    // 아이템 생성
                    itemID = repository.CreateItems(new
                string[]{"parameter_test.odi"}, new string[]{"Repository API test"}, new
                string[]{" /test"},
                    new bool[1], new Stream[]{@in}, "comment")[0];

                    if(null == itemID || 0 == itemID.Length)
                    {

```

```

        throw new Exception("Failed to create item; " +
repository.GetLastErrorMessage(0));
    }
}

// 아이템에 대한 독점점 수정권한을 요청합니다
Stream checkedOutItem = repository.CheckOut(new string[]{itemID},
new string[]{"d:"}, new long[1], new bool[1])[0];

// 아이템의 수정 사항을 반영합니다
repository.CheckIn(new string[]{itemID}, new bool[1], new
Stream[]{checkedOutItem}, null, new bool[1]);

// 아이템의 이름 수정
itemID = repository.ModifyItemName(itemID, "modified_item.odi",
"comment");

string tempCategoryID = repository.CreateCategories(new
string[]{"newCategory"}, new string[]{categoryID}, null)[0];

Console.WriteLine(repository.GetItemInfo(itemID).ToString());

repository.TransferItems(new string[]{itemID}, tempCategoryID);

foreach(IItemInfo itemInfo in
repository.GetItemInfos(tempCategoryID))
{
    Console.WriteLine(itemInfo.ToString());
    itemID = itemInfo.ID;
}

// 아이템 제거
repository.DeleteItems(new string[]{itemID}, null, null);

// 카테고리 제거
repository.DeleteCategories(new string[]{categoryID}, new bool[1],
null);

// 사용자 생성, 변경, 삭제
string userID = repository.CreateUser("userName", "password", "/",
"description");

userID = repository.ModifyUserName("userName", "brandNewName");

repository.ModifyUserDescription(userID, "new description");

repository.DeleteUser(userID);

// 그룹 생성, 변경, 삭제
string groupID = repository.CreateGroup("newGroupName", "/",

```

```
"description");  
  
        userID = repository.CreateUser("newUserName", "password", groupID,  
"description");  
  
        groupID = repository.ModifyGroupName(groupID, "newGroupName2");  
  
        repository.DeleteGroup(groupID);  
  
        repository.SetConfiguration(prevConf);  
        repository.Reload();  
    }  
}  
}
```

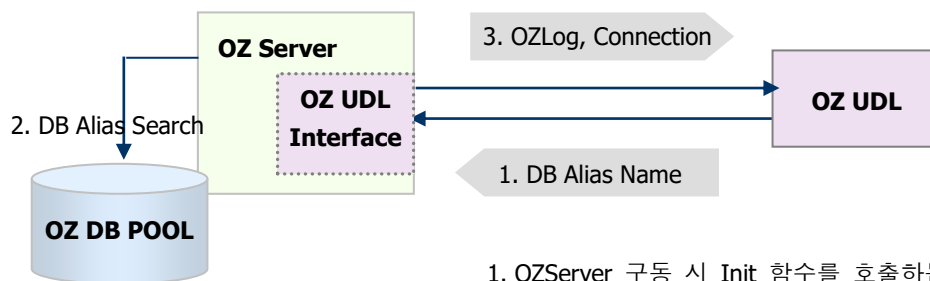
## IV. User Defined Log

- UDL 인터페이스
- UDL 구현

## UDL 인터페이스

UDL(User Defined Log)은 오즈 서버를 통하여 데이터를 조회할 경우 데이터 베이스나 보고서 관련 정보를 임의의 파일이나 DB에 사용자가 정의한 타입의 형식으로 기록할 수 있도록 하기 위한 인터페이스입니다.

### UDL 인터페이스 호출 구조



1. OZServer 구동 시 Init 함수를 호출하는 경우
2. 서비스 호출하여 Trace하는 경우

### oz.udl.OZUserDefinedLogger 클래스

OZ UDL을 사용하기 위하여 `oz.udl.OZUserDefinedLogger`를 먼저 구현하여야 합니다. `oz.udl.OZUserDefinedLogger` 구현 시 `Init`, `Trace`, `DBAlias`를 반드시 구현하여야 하며, 해당 메소드는 인터페이스를 정의하는 방법 외에 애트리뷰트를 이용하여 지정할 수 있습니다.

#### ■ Init

**Prototype** `public void Init(OZConnection con, OZLog log)`

**Definition** 오즈 서버 구동 시 사용자 정의 로그를 남길 대상을 정의합니다.

**Argument** `con` DB에 사용자 정의 로그를 기록할 경우 사용할 DB 커넥션 정보 및 기록할 로그 형식

*log*      파일에 사용자 정의 로그를 기록할 경우 사용할 파일 정보 및 기록할 로그 형식

■ **Trace**

**Prototype**    public void Trace(IUserDefinedLogTarget target, OZConnection con, OZLog log)

**Definition**    로그에 기록할 타겟 내용을 정의합니다.

*target*      사용자 정의 로그 타겟  
 설정할 수 있는 사용자 정의 로그 타겟은 아래 IUserDefinedLogTarget 인터페이스 부분을 참조하시기 바랍니다.

**Argument**    *con*      DB에 사용자 정의 로그를 기록할 경우 사용할 DB 커넥션 정보 및 기록할 로그 형식

*log*      파일에 사용자 정의 로그를 기록할 경우 사용할 파일 정보 및 기록할 로그 형식

■ **DBAlias**

**Prototype**    public string DbAlias{get;}

**Definition**    DB에 사용자 정의 로그를 기록할 경우 오즈 서버 커넥션 풀의 앨리어스 명 중 로그를 기록할 앨리어스 명

※ 참고사항 : 애트리뷰트를 이용하여 구현하는 방법

애트리뷰트를 이용하여 구현할 경우에는 대괄호를 이용하여 애트리뷰트명을 지정(ex. [OSUDLInitialize])하고 그 다음 줄에 함수를 선언하여 구현합니다.

만일 다음과 같이 구현된 메소드를

```
public class UserDefinedLogger : IUserDefinedLogger{
    public void Init(OZConnection con, OZLog log){}
    public void Trace(IUserDefinedLogTarget target, OZConnection con,
    OZLog log){}

    public string DbAlias{ get; }
}
```

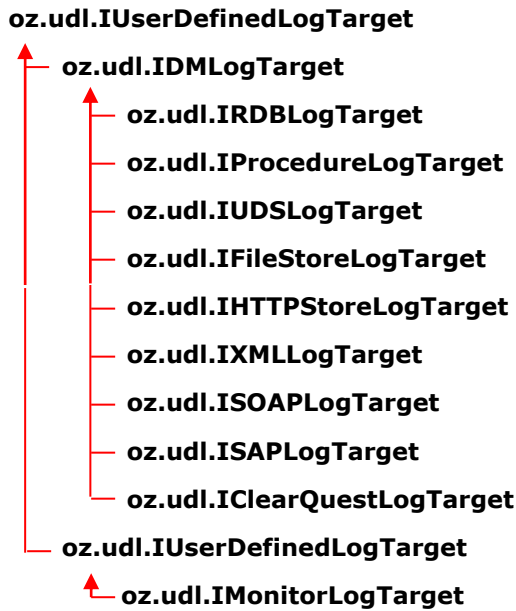
아래처럼 애트리뷰트를 이용하여 구현할 수 있습니다.

```
public class UserDefinedLogger{
    [OSUDLInitialize]
    public void Init(OZConnection con, OZLog log){}

    [OSUDLTrace]
    public void Trace(IUserDefinedLogTarget target, OZConnection con,
    OZLog log){}
```

```
[OZUDLDbAlias]
public string DbAlias{ get; }
}
```

## IUserDefinedLogTarget 인터페이스의 클래스 구조



### ■ IUserDefinedLogTarget 클래스

OZ UDL에서 제공되는 정보를 요청하여 사용하기 위한 인터페이스입니다.

- IPAddress

<b>Prototype</b>	public string IPAddress(){get;}
<b>Definition</b>	클라이언트가 요청한 오즈 서버 IP 주소를 가져옵니다
<b>Return</b>	오즈 서버 IP 주소

- HttpRequest

<b>Prototype</b>	public HttpRequest HttpRequest(){get;}
<b>Definition</b>	클라이언트에서 요청한 HttpRequest를 가져옵니다.
<b>Return</b>	HttpRequest

- HttpContext

<b>Prototype</b>	public HttpContext HttpContext(){get;}
------------------	--

---

**Definition** 클라이언트에서 HttpContext를 가져옵니다.

**Return** HttpContext

---

- UserID

---

**Prototype** public string UserID(){get;}

**Definition** 클라이언트에서 쿼리문을 실행한 오즈 서버 사용자 ID를 가져옵니다

**Return** 사용자 ID

---

#### ■ IDMLogTarget 클래스

OZ UDL에서 데이터 모듈 관련 정보를 사용하기 위한 인터페이스입니다.

- ODIName

---

**Prototype** public string ODIName(){get;}

**Definition** 쿼리문을 실행한 ODI 이름을 가져옵니다.

**Return** ODI 이름

---

- DataSetName

---

**Prototype** public string DataSetName(){get;}

**Definition** 쿼리문을 실행한 데이터 셋 이름을 가져옵니다.

**Return** 데이터 셋 이름

---

- DataStoreName

---

**Prototype** public string DataStoreName(){get;}

**Definition** 쿼리문을 실행한 데이터 스토어 이름을 가져옵니다.

**Return** 데이터 스토어 이름

---

- Parameters

---

**Prototype** public system.Connection.IDictionary Parameters{get;}

**Definition** 오즈 사용자 지정 파라미터를 Dictionary 형태로 가져옵니다.

**Return** 오즈 사용자 지정 파라미터의 key와 Value

---

#### ■ IRDBLogTarget 클래스

OZ UDL에서 RDB 스토어 관련 정보를 사용하기 위한 인터페이스로 쿼리문이 실행된 후 로그가 기록됩니다.

- DBAlias

---

**Prototype** public string Alias{get;}

---

<b>Definition</b>	실행한 ODI에서 사용한 커백션 풀의 DB 앨리어스 명을 가져옵니다. DB 앨리어스 명
<b>Return</b>	※ 참고사항 : 데이터 스토어에서 앨리어스를 사용하지 않는 경우 커백션 풀의 키 값을 가져옵니다.
<hr/>	
- Query	
<b>Prototype</b>	public string Query{get;}
<b>Definition</b>	클라이언트에서 요청한 쿼리문을 가져옵니다.
<b>Return</b>	쿼리문
<hr/>	
- PreparedQueryValue	
<b>Prototype</b>	public IList PreparedQueryValue{get;}
<b>Definition</b>	컴파일된 질의문을 사용할 경우 사용된 값을 가져옵니다.
<b>Return</b>	컴파일된 질의문에 사용된 값 (바이너리 데이터 포함)
<hr/>	
- QueryState	
<b>Prototype</b>	public string QueryState{get;}
<b>Definition</b>	클라이언트에서 요청한 쿼리문이 정상적으로 실행되었는지 여부를 가져옵니다.
<b>Return</b>	<i>true</i> 쿼리문이 정상적으로 실행됨 <i>false</i> 쿼리문이 정상적으로 실행되지 않음
<hr/>	
- QueryExecuteTime	
<b>Prototype</b>	public long QueryExecuteTime{get;}
<b>Definition</b>	클라이언트에서 요청한 쿼리문이 수행되는 시간을 가져옵니다.
<b>Return</b>	쿼리 수행 시간 (단위 : mesc)
<hr/>	
- IsPrepared	
<b>Prototype</b>	public bool IsPrepared{get;}
<b>Definition</b>	컴파일된 질의문의 사용 여부를 가져옵니다.
<b>Return</b>	<i>true</i> 컴파일된 질의문을 사용함 <i>false</i> 컴파일된 질의문을 사용하지 않음
<hr/>	
- QueryType	
<b>Prototype</b>	public QueryType QueryType{get;}
<b>Definition</b>	클라이언트에서 요청하여 실행된 쿼리문의 타입을 가져옵니다.

<b>Return</b>	<i>select</i>	Select 문이 사용됨
	<i>insert</i>	Insert 문이 사용됨
	<i>update</i>	Update 문이 사용됨
	<i>delete</i>	Delete 문이 사용됨

■ **IProcedureLogTarget** 클래스 함수

OZ UDL에서 프로시저 데이터 관련 정보를 사용하기 위한 인터페이스로 프로시저가 실행된 후 로그가 기록됩니다.

- Alias

<b>Prototype</b>	public string Alias{get;}
<b>Definition</b>	실행한 ODI에서 사용한 커넥션 풀의 DB 앨리어스 명을 가져옵니다. DB 앨리어스 명
<b>Return</b>	※ 참고사항 : 데이터 스토어에서 앨리어스를 사용하지 않는 경우 커넥션 풀의 키 값을 가져옵니다.

- ProcedureName

<b>Prototype</b>	public string ProcedureName{get;}
<b>Definition</b>	클라이언트에서 요청한 프로시저 이름을 가져옵니다.
<b>Return</b>	프로시저 이름

- ParameterList

<b>Prototype</b>	public IList ParameterList{get;}
<b>Definition</b>	클라이언트에서 요청한 프로시저 파라미터를 IList 형태로 가져옵니다.
<b>Return</b>	프로시저 파라미터

- ProcedureExecuteTime

<b>Prototype</b>	public long ProcedureExecuteTime{get;}
<b>Definition</b>	클라이언트에서 요청한 프로시저가 수행되는 시간을 가져옵니다.
<b>Return</b>	프로시저 수행 시간 (단위 : msec)

- ProcedureState

<b>Prototype</b>	public string ProcedureState{get;}	
<b>Definition</b>	클라이언트에서 요청한 프로시저가 정상적으로 실행되었는지 여부를 가져옵니다.	
<b>Return</b>	<i>true</i>	프로시저가 정상적으로 실행됨
	<i>false:ErrMsg</i>	프로시저가 정상적으로 실행되지 않았을 경우 "false:"

---

"에러메시지" 형태로 리턴됨

---

#### ■ IUDSLLogTarget 클래스 함수

OZ UDL에서 사용자 데이터 스토어 정보를 사용하기 위한 인터페이스로 사용자 데이터 스토어 실행문이 수행된 후 로그가 기록됩니다.

##### - ExecuteCommand

---

**Prototype** public string ExecuteCommand{get;}

---

**Definition** 클라이언트에서 요청한 실행문을 가져옵니다.

---

**Return** 실행문

---

##### - QueryType

---

**Prototype** public QueryType QueryType{get;}

---

**Definition** 클라이언트에서 요청하여 실행된 쿼리문의 타입을 가져옵니다.

---

*select* Select 문이 사용됨

---

*insert* Insert 문이 사용됨

---

**Return** *update* Update 문이 사용됨

---

*delete* Delete 문이 사용됨

---

##### - CommandExecuteTime

---

**Prototype** public long CommandExecuteTime{get;}

---

**Definition** 클라이언트에서 요청한 실행문이 수행되는 시간을 가져옵니다.

---

**Return** 실행문 수행 시간 (단위 : mesc)

---

##### - CommandState

---

**Prototype** public string CommandState{get;}

---

**Definition** 클라이언트에서 요청한 실행문이 정상적으로 실행되었는지 여부를 가져옵니다.

---

*true* 실행문이 정상적으로 실행됨

---

**Return** *false:ErrMsg* 실행문이 정상적으로 실행되지 않았을 경우 "false:에러메시지" 형태로 리턴됨

---

#### ■ IFileStoreLogTarget 클래스 함수

OZ UDL에서 파일 스토어 정보를 사용하기 위한 인터페이스로 파일 스토어의 파일을 읽어온 뒤 로그가 기록됩니다.

##### - FilePath

---

**Prototype** public string FilePath{get;}

---

<b>Definition</b>	클라이언트에서 요청한 파일 경로를 가져옵니다.	
<b>Return</b>	파일 경로	
- FileAccessTime		
<b>Prototype</b>	public long FileAccessTime{get;}	
<b>Definition</b>	클라이언트에서 요청한 파일을 가져오는 시간을 가져옵니다.	
<b>Return</b>	파일 처리 시간 (단위 : mesc)	
- FileAccessState		
<b>Prototype</b>	public string FileAccessState{get;}	
<b>Definition</b>	클라이언트에서 요청한 파일이 정상적으로 실행되었는지 여부를 가져옵니다.	
	<i>true</i>	파일이 정상적으로 실행됨
<b>Return</b>	<i>false:ErrMsg</i>	파일이 정상적으로 실행되지 않았을 경우 "false: <i>에러 메시지</i> " 형태로 리턴됨

#### ■ IHttpStoreLogTarget 클래스 함수

OZ UDL에서 HTTP 스토어 정보를 사용하기 위한 인터페이스로 HTTP 스토어의 파일을 읽어온 뒤 로그가 기록됩니다.

- URL		
<b>Prototype</b>	public string URL{get;}	
<b>Definition</b>	클라이언트에서 요청한 URL을 가져옵니다.	
<b>Return</b>	URL	
- HTTPAccessTime		
<b>Prototype</b>	public long HTTPAccessTime{get;}	
<b>Definition</b>	클라이언트에서 요청한 HTTP 경로의 파일을 가져오는 시간을 가져옵니다.	
<b>Return</b>	파일 처리 시간 (단위 : mesc)	
- HTTPAccessState		
<b>Prototype</b>	public string HTTPAccessState{get;}	
<b>Definition</b>	클라이언트에서 요청한 HTTP 경로의 파일이 정상적으로 실행되었는지 여부를 가져옵니다.	
	<i>true</i>	파일이 정상적으로 실행됨
<b>Return</b>	<i>false:ErrMsg</i>	파일이 정상적으로 실행되지 않았을 경우 "false: <i>에러 메시지</i> " 형태로 리턴됨

■ **IXMLLogTarget** 클래스 함수

OZ UDL에서 XML 스토어 정보를 사용하기 위한 인터페이스로 XML 파일을 읽어온 뒤 로그가 기록됩니다.

- URL

<b>Prototype</b>	public string URL{get;}
<b>Definition</b>	클라이언트에서 요청한 파일 경로 또는 URL을 가져옵니다.
<b>Return</b>	파일 경로 또는 URL

- URLAccessTime

<b>Prototype</b>	public long URLAccessTime{get;}
<b>Definition</b>	클라이언트에서 요청한 URL의 파일을 가져오는 시간을 가져옵니다.
<b>Return</b>	파일 처리 시간 (단위 : mesc)

- URLAccessState

<b>Prototype</b>	public string URLAccessState{get;}				
<b>Definition</b>	클라이언트에서 요청한 URL의 파일이 정상적으로 실행되었는지 여부를 가져옵니다.				
<b>Return</b>	<table border="0"> <tr> <td><i>true</i></td> <td>파일이 정상적으로 실행됨</td> </tr> <tr> <td><i>false:ErrMsg</i></td> <td>파일이 정상적으로 실행되지 않았을 경우 "false: <i>에러 메시지</i>" 형태로 리턴됨</td> </tr> </table>	<i>true</i>	파일이 정상적으로 실행됨	<i>false:ErrMsg</i>	파일이 정상적으로 실행되지 않았을 경우 "false: <i>에러 메시지</i> " 형태로 리턴됨
<i>true</i>	파일이 정상적으로 실행됨				
<i>false:ErrMsg</i>	파일이 정상적으로 실행되지 않았을 경우 "false: <i>에러 메시지</i> " 형태로 리턴됨				

■ **ISOAPLogTarget** 클래스 함수

OZ UDL에서 SOAP 스토어 정보를 사용하기 위한 인터페이스로 Service Response 후에 로그가 기록됩니다.

- ServiceName

<b>Prototype</b>	public string ServiceName{get;}
<b>Definition</b>	클라이언트에서 요청한 서비스 이름을 가져옵니다.
<b>Return</b>	서비스 이름

- Port

<b>Prototype</b>	public string Port{get;}
<b>Definition</b>	클라이언트에서 요청한 서비스의 포트를 가져옵니다.
<b>Return</b>	서비스 포트

- Operation

<b>Prototype</b>	public string Operation{get;}
------------------	-------------------------------

<b>Definition</b>	클라이언트에서 요청한 서비스의 오퍼레이션을 가져옵니다.
<b>Return</b>	서비스 오퍼레이션
<hr/>	
- EndPoint	
<b>Prototype</b>	<code>public string EndPoint{get;}</code>
<b>Definition</b>	클라이언트에서 요청한 서비스를 호출한 EndPoint를 가져옵니다.
<b>Return</b>	서비스 EndPoint
<hr/>	
- RequestXML	
<b>Prototype</b>	<code>public string RequestXML{get;}</code>
<b>Definition</b>	클라이언트에서 요청한 XML 내용을 가져옵니다.
<b>Return</b>	XML 내용
<hr/>	
- ServiceExecuteTime	
<b>Prototype</b>	<code>public long ServiceExecuteTime{get;}</code>
<b>Definition</b>	클라이언트에서 요청한 서비스가 실행되는 시간을 가져옵니다.
<b>Return</b>	서비스 수행 시간 (단위 : mesc)
<hr/>	
- ServiceState	
<b>Prototype</b>	<code>public string ServiceState{get;}</code>
<b>Definition</b>	클라이언트에서 요청한 SOAP의 서비스가 정상적으로 실행되었는지 여부를 가져옵니다.
<b>Return</b>	<i>true</i> 서비스가 정상적으로 실행됨 <i>false:ErrMsg</i> 서비스가 정상적으로 실행되지 않았을 경우 "false:에러메시지" 형태로 리턴됨

■ ISAPLogTarget 클래스 함수

OZ UDL에서 SAP 스토어 정보를 사용하기 위한 인터페이스로 함수가 실행된 후에 로그가 기록됩니다.

- FunctionName	
<b>Prototype</b>	<code>public string FunctionName{get;}</code>
<b>Definition</b>	클라이언트에서 요청한 함수 이름을 가져옵니다.
<b>Return</b>	함수 이름
<hr/>	
- FunctionType	
<b>Prototype</b>	<code>public string FunctionType{get;}</code>

<b>Definition</b>	클라이언트에서 요청한 함수 타입을 가져옵니다.	
<b>Return</b>	함수 타입	
<hr/>		
-	InputParameters	
<b>Prototype</b>	public IDictionary InputParameters{get;}	
<b>Definition</b>	클라이언트에서 요청한 입력 파라미터를 IDictionary 형태로 가져옵니다.	
<b>Return</b>	입력 파라미터의 key와 value	
<hr/>		
-	ResultSetTypes	
<b>Prototype</b>	public string ResultSetTypes{get;}	
<b>Definition</b>	클라이언트에서 요청한 ResultSet의 타입을 가져옵니다.	
	<i>Structure</i>	Structure 타입
<b>Return</b>	<i>Table</i>	Table 타입
	<i>SimpleFields</i>	SimpleFields 타입
<hr/>		
-	FunctionExecuteTime	
<b>Prototype</b>	public long FunctionExecuteTime{get;}	
<b>Definition</b>	클라이언트에서 요청한 함수가 실행되는 시간을 가져옵니다.	
<b>Return</b>	함수 실행 시간 (단위 : mesc)	
<hr/>		
-	FunctionState	
<b>Prototype</b>	public string FunctionState{get;}	
<b>Definition</b>	클라이언트에서 요청한 함수가 정상적으로 실행되었는지 여부를 가져옵니다.	
	<i>true</i>	함수가 정상적으로 실행됨
<b>Return</b>	<i>false:ErrMsg</i>	함수가 정상적으로 실행되지 않았을 경우 "false:에러 메시지" 형태로 리턴됨
<hr/>		

■ **IClearQuestLogTarget** 클래스 함수

OZ UDL에서 Clear Quest 스토어 정보를 사용하기 위한 인터페이스로 Clear Quest 쿼리문이 실행된 후에 로그가 기록됩니다.

-	ExecuteType	
<b>Prototype</b>	public string ExecuteType{get;}	
<b>Definition</b>	클라이언트에서 요청한 Clear Quest 실행 타입을 가져옵니다.	
	<i>ExecuteSQL</i>	SQL 타입
<b>Return</b>	<i>ExecuteQuery</i>	쿼리 타입
<hr/>		

---

*ExecuteDynamicQuery*    다이내믹 쿼리 타입

---

- QuerySubType

---

**Prototype**    public string QuerySubType{get;}**Definition**    클라이언트에서 요청한 QuerySub 타입을 가져옵니다.**Return**  
*Query*        Query 타입  
*Chart*        Chart 타입

---

- Query

---

**Prototype**    public string Query{get;}**Definition**    클라이언트에서 요청한 쿼리문을 가져옵니다.**Return**        쿼리문

---

- QueryExecuteTime

---

**Prototype**    public long QueryExecuteTime{get;}**Definition**    클라이언트에서 요청한 쿼리문이 실행되는 시간을 가져옵니다.**Return**        쿼리문 실행 시간 (단위 : msec)

---

- QueryState

---

**Prototype**    public string QueryState{get;}**Definition**    클라이언트에서 요청한 쿼리문이 정상적으로 실행되었는지 여부를 가져옵니다.**Return**  
*true*            쿼리문이 정상적으로 실행됨  
*false:ErrMsg*    쿼리문이 정상적으로 실행되지 않았을 경우 "false:에러메시지" 형태로 리턴됨

---

## ■ IMonitorLogTarget 클래스

OZ UDL에서 모니터 로그 관련 정보를 사용하기 위한 인터페이스입니다.

- ThreadID

---

**Prototype**    public int ThreadID(){get;}**Definition**    서비스 스레드 아이디를 가져옵니다.**Return**        스레드 아이디

---

- Mark

---

**Prototype**    public string Mark(){get;}

<b>Definition</b>	서비스 시작 및 종료 표시를 가져옵니다.
<b>Return</b>	<i>start</i> 서비스 시작 <i>end</i> 서비스 종료

- ThreadName

<b>Prototype</b>	<code>public string ThreadName(){get;}</code>
<b>Definition</b>	서비스 스레드 이름을 가져옵니다.
<b>Return</b>	스레드 이름

- ServiceTime

<b>Prototype</b>	<code>public long ServiceTime(){get;}</code>
<b>Definition</b>	서비스 시작 시간과 종료 시간을 가져옵니다. (단위 : msec) ※ 참고사항 : 1970년 1월 1일부터 현재 시간을 Millisecond로 표시합니다.
<b>Return</b>	서비스 시작 시간 및 종료 시간

- FreeMemory

<b>Prototype</b>	<code>public long FreeMemory(){get;}</code>
<b>Definition</b>	사용 가능한 메모리 크기를 가져옵니다. (단위 : byte)
<b>Return</b>	메모리 크기

- TotalMemory

<b>Prototype</b>	<code>public long TotalMemory(){get;}</code>
<b>Definition</b>	전체 JVM 메모리 크기를 가져옵니다. (단위 : byte)
<b>Return</b>	메모리 크기

- ServiceCode

<b>Prototype</b>	<code>public int ServiceCode(){get;}</code>
<b>Definition</b>	서비스 구분 코드를 가져옵니다. ※ 참고사항 - MARK가 "end"일 경우에만 값이 리턴되며 "start"일 경우 -1을 리턴합니다. - 자세한 서비스 구분 코드는 오즈 엔터 프라이즈 서버 관리자 가이드의 "IV. 오즈 엔터프라이즈 서버 설정" 부분을 참조하시기 바랍니다.
<b>Return</b>	서비스 구분 코드

## - ServiceStatus

<b>Prototype</b>	public int ServiceStatus(){get;}
	서비스 상태 코드를 가져옵니다. ※ 참고사항
<b>Definition</b>	- MARK가 "end"일 경우에만 값이 리턴되며 "start"일 경우 -1을 리턴합니다. - 성공일 경우 "9001", 실패일 경우 "9002"를 리턴합니다.
<b>Return</b>	9001 서비스 성공 9002 서비스 실패

## - ServiceParameter

<b>Prototype</b>	public string ServiceParameter(){get;}
	클라이언트에서 서비스 요청 시 전송되는 파라미터 값을 가져옵니다.
<b>Definition</b>	※ 참고사항 : MARK가 "end"일 경우에만 값이 리턴되며 "start"일 경우 -1을 리턴합니다.
<b>Return</b>	파라미터 값

## - DBSessionID

<b>Prototype</b>	public string DBSessionID(){get;}
	DBMS 세션 ID를 가져옵니다.
<b>Definition</b>	※ 참고사항 : MARK가 "end"일 경우에만 값이 리턴되며 "start"일 경우 -1을 리턴합니다.
<b>Return</b>	DBMS 세션 ID

## - ExecuteTime

<b>Prototype</b>	public string ExecuteTime(){get;}
	서비스 처리 시 소요된 시간을 가져옵니다. (단위 : mesc)
<b>Definition</b>	※ 참고사항 : MARK가 "end"일 경우에만 값이 리턴되며 "start"일 경우 -1을 리턴합니다.
<b>Return</b>	서비스 처리 시간

## - ErrorCode

<b>Prototype</b>	public string ErrorCode(){get;}
	에러 발생 시 에러 코드를 가져옵니다.
<b>Definition</b>	※ 참고사항 : 에러 코드 내용은 conf/server_error_msg_언어명_국가명.xml 파일을 참고하시기 바랍니다.
<b>Return</b>	에러 메시지 코드

## - ErrorMessage

---

**Prototype** public string errorMsg(){get;}

**Definition** 에러 발생 시 에러 메시지를 가져옵니다.

**Return** 에러 메시지

---

- ErrorStackTrace

---

**Prototype** public string ErrorStackTrace(){get;}

**Definition** 에러 발생 시 Stack Trace를 가져옵니다.

**Return** 에러 Stack Trace

---

- DBConns

---

**Prototype** public string DBConns(){get;}

**Definition** Alias별 DB사용갯수

**Return** Alias별 DB사용갯수를 string화 하여 보여줌

---

## UDL 구현

서버 로그와 모니터 로그에 UDL을 적용하여 사용자 정의 로그를 파일이나 데이터 베이스에 기록하기 위한 방법에 대해 설명합니다.

### 오즈 서버 설정

#### ■ 관련 파일

- UDL 관련 라이브러리 파일 : OZ\_HOME/bin/ozudl.dll
- Lof4j 관련 라이브러리 파일 : OZ\_HOME/bin/log4net.dll
- UDL 관련 설정 파일 : OZ\_HOME/conf/ozudl.properties

#### ■ 파일 설정 값 변경

- udlmngr.properties 파일을 열어 UDL 사용 여부를 설정하고 사용할 경우 구현한 클래스의 경로를 설정합니다.

```
#-----
# configuration of OZ User Defined log
#-----

OZ_USER_DEFINED_LOG.Active=true
OZ_USER_DEFINED_LOG.Class=oz.udl.UDL클래스명

OZ_UDL_MONITOR.Active= true
OZ_UDL_MONITOR.Class=oz.udl.UDL클래스명
```

- ozudl.properties에 아래와 같이 설정하여 UDL을 사용함으로 설정합니다.
    - 서버 로그를 사용자 정의로그로 기록할 경우
      - OZ\_USER\_DEFINED\_LOG.Active=true
    - 모니터 로그를 사용자 정의로그로 기록할 경우
      - OZ\_UDL\_MONITOR.Active= true
  - 서버로그와 모니터 로그의 UDL 클래스를 설정합니다.
    - OZ\_USER\_DEFINED\_LOG.Class=oz.udl.UDL클래스명
    - OZ\_UDL\_MONITOR.Class=oz.udl.UDL클래스명
- ex)

OZ\_USER\_DEFINED\_LOG.Class=oz.udl.file.OZUserDefinedLogger

OZ\_UDL\_MONITOR.Class=oz.udl.db.OZMonitorLogForDB

- OZ\_HOME/Web.config 파일을 열어 사용자 정의 로그를 기록할 파일의 경로를 설정합니다.

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
<httpRuntime maxRequestLength="100000" />
  </system.web>
  <udl>
    <file>
      <add key="path" value="C:/Program Files/Forcs/
Server.NET/logs/UDL.Log" />
      <add key="layout" value="%d{yyyy-MM-dd HH:mm:ss.fff}
[%-5p] %C{1}.%M - %m%n" />
      <add key="daily" value="false" />
      <add key="size" value="100KB" />
      <add key="maxBackup" value="5" />
      <!--add key="daily" value="true" />
      <add key="pattern" value="yyyy-MM-dd" /-->
    </file>
    <db>
      <add key="alias" value="qa_ms" />
    </db>
  </udl>
</configuration>
```

## V . User Defined Resource

- UDR 인터페이스
- UDR 구현

## UDR 인터페이스

UDR(User Defined Resource)은 보고서의 리소스 필드를 생성할 때 리소스 필드 값을 리포트 디자이너에서 일일이 설정하지 않고 데이터베이스에 입력된 리소스 정보로 한꺼번에 설정할 수 있도록 하는 인터페이스입니다.

### 관련 Interface

#### ■ IOZUserDefinedResource(oz.udr.IOZUserDefinedResource)

<b>Prototype</b>	<code>public OZUDRResult SyncResource(OZConnection ozconn, OZLog cat, OZUDRInfo info)</code>
<b>Definition</b>	리소스를 동기화합니다.
<b>Argument</b>	<i>ozconn</i> DB의 Connection
	<i>cat</i> 서버 로그에 남을 카테고리
	<i>props</i> 사용자가 정의한 Property
	<i>info</i> 보고서에 있는 리소스 정보 객체
<b>Return</b>	OZUDRResult DB 정보 동기화한 결과 객체

### 관련 Class - oz.udr.OZUDRInfo

#### ■ Constructor

- OZUDRInfo

<b>Prototype</b>	<code>public OZUDRInfo(string category, string reportname, string[] language, Resource[] resource)</code>
	<i>category</i> 보고서 카테고리
<b>Argument</b>	<i>reportname</i> 보고서 이름
	<i>language</i> 언어 리스트 ex) ko/kr, en/us
	<i>resource</i> 리소스 정보 객체

#### ■ Property

- Category

<b>Prototype</b>	<code>public string Category{get;set;}</code>
------------------	---

<b>Definition</b>	카테고리 이름
- Reportname	
<b>Prototype</b>	public string Reportname{get;set;}
<b>Definition</b>	리포트 이름
- Language	
<b>Prototype</b>	public string[] Language{get;set;}
<b>Definition</b>	언어 리스트
- Resource	
<b>Prototype</b>	public Resource[] Resource{get;set;}
<b>Definition</b>	리소스

## 관련 Class - oz.udr.Resource

### ■ Constructor

- Resource							
<b>Prototype</b>	public Resource(string name, string desc, string[] value)						
<b>Argument</b>	<table border="1"> <tr> <td><i>name</i></td> <td>리소스 필드 이름</td> </tr> <tr> <td><i>desc</i></td> <td>리소스 설명</td> </tr> <tr> <td><i>value</i></td> <td>언어별 리소스 값(OZUDRInfo language index와 같은 index 값이어야 함)</td> </tr> </table>	<i>name</i>	리소스 필드 이름	<i>desc</i>	리소스 설명	<i>value</i>	언어별 리소스 값(OZUDRInfo language index와 같은 index 값이어야 함)
<i>name</i>	리소스 필드 이름						
<i>desc</i>	리소스 설명						
<i>value</i>	언어별 리소스 값(OZUDRInfo language index와 같은 index 값이어야 함)						

### ■ Property

- Name	
<b>Prototype</b>	public string Name{get;set;}
<b>Definition</b>	리소스 필드 이름
- Desc	
<b>Prototype</b>	public string Desc{get;set;}
<b>Definition</b>	리소스 설명

- Value

---

<b>Prototype</b>	public string[] value{get;set;}
------------------	---------------------------------

---

<b>Definition</b>	언어별 리소스 값
-------------------	-----------

---

- DescHash

---

<b>Prototype</b>	public Hashtable DescHash{get;}
------------------	---------------------------------

---

<b>Definition</b>	리소스 설명을 파싱할 Hash에 key, value 형태로 가지고 있는 객체
-------------------	--

---

## 관련 Class - oz.udr.OZUDRResult

### ■ Constructor

- OZUDRResult

---

<b>Prototype</b>	public OZUDRResult(bool isUpdate, OZUDRInfo info)
------------------	---

---

<i>isUpdate</i>	업데이트 여부
-----------------	---------

---

<b>Argument</b>	<i>info</i> 사용자가 정의한 리소스 정보를 다시 저장한 객체
-----------------	--

---

### ■ Property

- Update

---

<b>Prototype</b>	public bool Update{get;}
------------------	--------------------------

---

<b>Definition</b>	업데이트 여부
-------------------	---------

---

- UDRInfo

---

<b>Prototype</b>	public OZUDRInfo UDRInfo{get;}
------------------	--------------------------------

---

<b>Definition</b>	사용자가 정의한 리소스 정보를 다시 저장한 객체
-------------------	----------------------------

---

## UDR 구현

### UDR 인터페이스 예제

예를 들어, 리소스 정보를 저장한 테이블의 데이터 구조가 아래와 같고,

데이터 필드 이름	데이터 필드 타입	설명
ProgramID	varchar(40)	OZR 파일 전체 경로
ObjectID	varchar(50)	리소스 필드 이름
DD_KO	varchar(400)	한국어 리소스 필드 값
DD_EN	varchar(400)	영어 리소스 필드 값
DD_JP	varchar(400)	일본어 리소스 필드 값
DD_CN	varchar(400)	중국어 리소스 필드 값

필드 이름과 한국어, 영어, 일본어, 중국어에 대한 리소스 필드의 필드 이름과 필드 값을 동기화할 경우 다음과 같이 구현합니다.

```
using System;
using System.Collections;
using System.Data;
using System.Data.SqlClient;

using oz.framework.db;
using oz.framework.log;
using oz.uds;
using oz.util;

namespace oz.uds
{
    public class OZUserDefinedResourceImpl : IOZUserDefinedResource
    {
        OZLog log = null;
        public OZUserDefinedResourceImpl()
        {
        }
        private int languageLength = 0;
        private bool isupdate = false;
        private string languages = string.Empty;
        public OZUDRResult syncResource(OZConnection ozconn, OZLog log, OZUDRInfo
info)
        {
            this.log = log;
        }
    }
}
```

```

        log.Info("OZUserDefinedResourceImpl sync begin...");
        if (ozconn == null) throw new Exception("Connection is null.");
        string programID = getProgramID(info);
        LanguageLength = info.Language.Length;
        Languages =
        oz.util.OZTokenizer.join(oz.util.OZTokenizer.Delim,info.Language);
        int size = info.Resource.Length;
        for(int i=0; i<size; i++)
        {
            Resource resource = info.Resource[i];
            string objectID = resource.Name;
            if (objectID != null && !"".Equals(objectID) )
                condition(ozconn.Connection, programID, objectID, info, i);
        }
        OZUDRRResult result = new OZUDRRResult(isupdate, info);
        log.Info("OZUserDefinedResourceImpl sync end...");
        return result;
    }

    private string getProgramID(OZUDRInfo info)
    {
        string reportname = info.ReportName;
        int idx = reportname.ToLower().LastIndexOf(".ozr");
        if (idx >= 0)
        {
            reportname = reportname.Substring(0, idx);
        }
        string categoryName = info.Category;
        return info.Category+reportname;
    }

    public void condition(IDbConnection conn, string programID, string
    objectID, OZUDRInfo info, int index)
    {
        SqlDataReader dr = null;
        SqlCommand comm = null;
        bool isInsert = false;
        try
        {
            string query = "select * from DD where ProgramID = @ProgramID and
            ObjectID = @ObjectID ";
            //DD_REPORT
            comm = (SqlCommand)conn.CreateCommand();
            //comm.Connection = (SqlConnection)conn;
            comm.CommandType = CommandType.Text;
            comm.CommandText = query;
            comm.Prepare();
            AddParameter(comm,"@ProgramID",programID,DbType.String);
            AddParameter(comm,"@ObjectID",objectID,DbType.String);
            dr = comm.ExecuteReader();

```

```

ArrayList vec = null;
ArrayList vecValue = null;;
string[] targetLanguage = null;
string[] targetValue = null;
string dd_name = null;
if (dr.Read())
{
    vec = new ArrayList();
    vecValue = new ArrayList();
    //ko/kr
    dd_name = (string)dr["DD_KO"];
    if (dd_name != null)
    {
        vec.Add("ko/kr");
        vecValue.Add(dd_name);
    }
    //en/us
    dd_name = (string)dr["DD_EN"];
    if (dd_name != null)
    {
        vec.Add("en/us");
        vecValue.Add(dd_name);
    }
    //zh/cn
    dd_name = (string)dr["DD_CN"];
    if (dd_name != null)
    {
        vec.Add("zh/cn");
        vecValue.Add(dd_name);
    }
    if (dr != null) dr.Close();
    if (comm != null) comm.Dispose();
    targetLanguage = new string[vec.Count];
    targetValue = new string[vec.Count];
    vec.CopyTo(targetLanguage);
    vecValue.CopyTo(targetValue);
    string compare1 =
OZTokenizer.Join(OZTokenizer.Delim, targetLanguage);
    if(!isupdate)
    {
        if(!compare1.Equals(languges))
        {
            isUpdate = true;
        }
    }
    compare1 = OZTokenizer.Join(OZTokenizer.Delim, targetValue);
    string compare2 =
OZTokenizer.Join(OZTokenizer.Delim, info.Resource[index].Value);
    if(!isupdate)
    {

```

```
        if(!compare1.Equals(compare2))
        {
            isUpdate = true;
        }
    }
    info.Language = targetLanguage;
    info.Resource[index].Value = targetValue;
    isInsert = false;
}
else
{
    isInsert = true;
}
}
catch(Exception e)
{
    log.Error(e.Message,e);
    throw e;
}
finally
{
    try
    {
        if (dr != null) dr.Close();
        if (comm != null) comm.Dispose();
    }
    catch(Exception)
    {
    }
    if (isInsert)
    {
        insert(conn, programID, objectID, info, index);
    }
}
//return isInsert;
}

private void AddParameter(SqlCommand comm,string paramName,object
paramValue,DbType type)
{
    SqlParameter param = comm.CreateParameter();
    param.DbType = type;
    param.Direction = ParameterDirection.Input;
    param.ParameterName = paramName;
    //param1.Size = 4000;
    param.Value = paramValue;
    comm.Parameters.Add(param);
}

public void insert(IDbConnection conn, string programID, string objectID,
```

```

OZUDRInfo info, int index)
{
    SqlCommand comm = null;
    try
    {
        comm = (SqlCommand)conn.CreateCommand();
        string query = "insert into DD
(ProgramID,ObjectID,DD_KO,DD_EN,DD_CN,DD_JP)" +
            " values(@ProgramID,@ObjectID,@DD_KO,@DD_EN,@DD_CN,@DD_JP)";
        //comm.Connection = (SqlConnection)conn;
        comm.CommandType = CommandType.Text;
        comm.CommandText = query;
        comm.Prepare();
        comm.Parameters.Clear();
        AddParameter(comm,"@ProgramID",programID,DbType.String);
        AddParameter(comm,"@ObjectID",objectID,DbType.String);
        AddParameter(comm,"@Type","S",DbType.String);
        AddParameter(comm,"@INSERT_DT",DateTime.Now,DbType.DateTime);
        bool[] flag = new bool[4];
        string dd_ko = string.Empty;
        for(int i=0; i<info.Language.Length; i++)
        {
            if (info.Language[i].ToLower().Equals("ko/kr"))
            {
                dd_ko = info.Resource[index].Value[i];
                AddParameter(comm,"@DD_KO",dd_ko,DbType.String);
                flag[0] = true;
            }
            else if (info.Language[i].ToLower().Equals("en/us"))
            {
                AddParameter(comm,"@DD_EN",info.Resource[index].Value[i],DbType.String);
                flag[1] = true;
            }
            else if (info.Language[i].ToLower().Equals("zh/cn"))
            {
                AddParameter(comm,"@DD_CN",info.Resource[index].Value[i],DbType.String);
                flag[2] = true;
            }
            else if (info.Language[i].ToLower().Equals("ja/jp"))
            {
                AddParameter(comm,"@DD_JP",info.Resource[index].Value[i],DbType.String);
                flag[3] = true;
            }
        }
        if(!flag[1])
        {
            AddParameter(comm,"@DD_EN",dd_ko,DbType.String);
        }
        if(!flag[2])
        {

```

```
        AddParameter(comm,"@DD_CN",dd_ko,DbType.String);
    }
    if(!flag[3])
    {
        AddParameter(comm,"@DD_JP",dd_ko,DbType.String);
    }
    comm.ExecuteNonQuery();
    log.Info("[DD_REPORT] table succeeded insert.");
}
catch (Exception e)
{
    log.Error("UDR Error : " + e.Message, e);
    throw e;
}
finally
{
    comm.Dispose();
}
}
}
```

## Appendix 1. 사용자 지정 파라미터 암호화

---



본 장에서는 **ODI** 파일에 추가된 사용자 지정 파라미터를 암호화하기 위한 서버 설정 방법과 암호화 기능을 실행하는 클래스를 만들기 위한 함수에 대해 설명합니다.

## 오즈 서버 설정

### ■ 암호화 모듈 설정

- 오즈 서버가 구동될 때 암호화 모듈이 함께 실행되도록 설정합니다.
- `spmgr.properties` 파일을 열어 사용자 지정 파라미터 암호화 여부를 설정하고, 암호화 모듈의 클래스명과 암호화 모듈이 실행될 때 전달할 파라미터 값이 입력된 파일 경로를 설정합니다.

```
odiparam_usersecurity.Active=true
odiparam_usersecurity.Class=oz.framework.policy.클래스명
odiparam_usersecurity.InitParam=파일경로
```

## 사용자 지정 파라미터 암호화 관련 인터페이스

### ■ Init

<b>Prototype</b>	<code>public void Init(OZLog cat, OZConnection ozconn, string initParam)</code>						
<b>Definition</b>	사용자 지정 파라미터 암호화 모듈을 초기화합니다. 오즈 서버 구동 시 실행됩니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>cat</i></td> <td>오즈 서버에 로그를 기록할 경우 로그 파일 정보 및 기록할 로그 형식</td> </tr> <tr> <td><i>ozconn</i></td> <td>DB에 로그를 기록할 경우 사용할 DB 커넥션 정보 및 기록할 로그 형식</td> </tr> <tr> <td><i>initParam</i></td> <td>클래스 구동 시 설정할 파라미터</td> </tr> </table>	<i>cat</i>	오즈 서버에 로그를 기록할 경우 로그 파일 정보 및 기록할 로그 형식	<i>ozconn</i>	DB에 로그를 기록할 경우 사용할 DB 커넥션 정보 및 기록할 로그 형식	<i>initParam</i>	클래스 구동 시 설정할 파라미터
<i>cat</i>	오즈 서버에 로그를 기록할 경우 로그 파일 정보 및 기록할 로그 형식						
<i>ozconn</i>	DB에 로그를 기록할 경우 사용할 DB 커넥션 정보 및 기록할 로그 형식						
<i>initParam</i>	클래스 구동 시 설정할 파라미터						

### ■ ConvertParam

<b>Prototype</b>	<code>public string ConvertParam(OZLog cat, HttpContext context, OZConnection ozconn, string paramName, string paramValue)</code>						
<b>Definition</b>	사용자 지정 파라미터를 암호화한 후 암호화된 값을 리턴합니다.						
<b>Argument</b>	<table border="0"> <tr> <td><i>cat</i></td> <td>오즈 서버에 기록할 로그</td> </tr> <tr> <td><i>context</i></td> <td>HttpContext 정보</td> </tr> <tr> <td><i>ozconn</i></td> <td>DB 커넥션 정보</td> </tr> </table>	<i>cat</i>	오즈 서버에 기록할 로그	<i>context</i>	HttpContext 정보	<i>ozconn</i>	DB 커넥션 정보
<i>cat</i>	오즈 서버에 기록할 로그						
<i>context</i>	HttpContext 정보						
<i>ozconn</i>	DB 커넥션 정보						

---

*paramName* 사용자 지정 파라미터 이름

---

*paramValue* 사용자 지정 파라미터 값

---

#### ■ Release

---

**Prototype** public void Release()

---

**Definition** 오즈 서버 종료 시 호출됩니다.

---

#### ■ GetOZDBAlias

---

**Prototype** public string GetOZDBAlias()

---

**Definition** DB에 로그를 기록할 경우 오즈 서버에 정의된 커넥션 풀의 앨리어스 명을 리턴합니다.

---

### Sample : IUserSecurityParameterImpl.cs

```
using System;
using System.Web;
using System.IO;
using System.Text;
using System.Security.Cryptography;
using oz.framework.db;
using oz.framework.log;
using oz.framework.policy;

namespace oz.security
{
    public class IUserSecurityParameterImpl : IUserSecurityParameter
    {
        public IUserSecurityParameterImpl()
        {
        }

        public void Init(oz.framework.log.OZLog cat, OZConnection ozcon, string
initParam)
        {
            cat.Debug("ozcon = " + ozcon);
            cat.Debug("Init parameter = " + initParam);
        }

        public string ConvertParam(oz.framework.log.OZLog cat, HttpRequest
request, OZConnection con, string paramName, string paramValue)
        {

```

```
        cat.Debug("con = " + con);
        cat.Debug(paramName + "=" + paramValue);
        return Decrypt(paramValue, "_dhqxlak2010_");
    }

    public string getOZDBAlias()
    {
        return "";
    }

    public void Release()
    {
    }

    private string Decrypt(string textToDecrypt, string key)
    {
        //"_dhqxlak2010_"
        RijndaelManaged rijndaelCipher = new RijndaelManaged();
        rijndaelCipher.Mode = CipherMode.CBC;
        rijndaelCipher.Padding = PaddingMode.PKCS7;
        rijndaelCipher.KeySize = 128;
        rijndaelCipher.BlockSize = 128;
        byte[] encryptedData = Convert.FromBase64String(textToDecrypt);
        byte[] pwdBytes = Encoding.UTF8.GetBytes(key);
        byte[] keyBytes = new byte[16];
        int len = pwdBytes.Length;
        if (len > keyBytes.Length)
        {
            len = keyBytes.Length;
        }
        Array.Copy(pwdBytes, keyBytes, len);
        rijndaelCipher.Key = keyBytes;
        rijndaelCipher.IV = keyBytes;
        byte[] plainText =
rijndaelCipher.CreateDecryptor().TransformFinalBlock(encryptedData,
encryptedData.Length);
        return Encoding.UTF8.GetString(plainText);
    }

    private string Encrypt(string textToEncrypt, string key)
    {
        //"_dhqxlak2010_"
        RijndaelManaged rijndaelCipher = new RijndaelManaged();
        rijndaelCipher.Mode = CipherMode.CBC;
        rijndaelCipher.Padding = PaddingMode.PKCS7;
        rijndaelCipher.KeySize = 128;
```

```
rijndaelCipher.BlockSize = 128;
byte[] pwdBytes = Encoding.UTF8.GetBytes(key);
byte[] keyBytes = new byte[16];
int len = pwdBytes.Length;
if (len > keyBytes.Length)
{
    len = keyBytes.Length;
}
Array.Copy(pwdBytes, keyBytes, len);
rijndaelCipher.Key = keyBytes;
rijndaelCipher.IV = keyBytes;
ICryptoTransform transform = rijndaelCipher.CreateEncryptor();
byte[] plainText = Encoding.UTF8.GetBytes(textToEncrypt);
return
Convert.ToBase64String(transform.TransformFinalBlock(plainText, 0,
plainText.Length));
}
}
```