JEUS WebT 안내서

Tmax Soft

Document info

Document name: “JEUS WebT 안내서”

Document date: 2005-02-01

Manual release version: 1

Software Version: JEUS 5

차 례

그림 목차

매뉴얼에 대해서

매뉴얼의 대상

이 문서는 Java 를 기반으로 한 Client Application program 개발자들에게 WebT Library 를 통해서 Tmax Server 에서 제공하는 서비스들을 용이하게 이용할 수 있도록 도움을 주고자 한다.

매뉴얼의 전제 조건

이 문서를 보기 전에 기본적인 Tmax s 의 Client/Server 프로그램에 대한 이해가 요구된다. 또한 Jeus Configuration file 에 대한 기본적인 지식과 Web Application 에 대한 기반 지식이 습득되어 있어야 한다. Jeus 의 WebContainer manual 과 Tmax Application program manual 을 참조하기 바란다.

매뉴얼의 구성

이 매뉴얼은 8 장의 chapter 와 2 장의 appendix 로 구성되어 있다.

1. 소개: WebT 소개.
2. **WebT 모듈의 개요**: WebT 의 기본 개념 및 구조 소개
3. **WebT DataSource 설정**: Jeus WebContainer 에서 Tmax server 로의 연결에 대한 connection pool 설명.
4. **WebT 환경파일 설정**
5. **WebT API**.
6. **WebT Application 예제**: WebT library 를 이용한 Web Application samples

7. The WebT-Server System
8. 결론

A Web Container XML Configuration Reference(WEBMain.xml): a complete reference to the Web container's WEBMain.xml configuration file.

B WebT API Reference

관련 매뉴얼

Related documents are:

- The J2EE 1.3 Specification.
- The Servlet 2.3 Specification.
- The JSP 1.2 Specification.
- The JEUS Web Guide.
- The WebT1.8 Application Guide
- The Tmax Application Guide
- The Tmax RQ Application Guide

용어 설명

Convention	Meaning
Some text	Regular text
Some text	Emphasized text
CTRL+C	Keys to press at the same time
`public class myClass { }`	Java Code

`<SystemConfig>`	XML fragment
Note: / Warning:	Note or warning
Open the **Configuration** menu	GUI components (buttons etc.)
`jeusadmin nodename`	Console commands and syntax
`[]`	Optional command argument
`|`	Exclusive OR between commands
…	Indicates that a command parameter may be repeated
?, +, *	Zero or one, one or more, zero or more (in XML reference documentation)
⋮	Skipped sections (in XML and code)
Figure 1.	Figure or table caption

연락처

Korea
Tmax Soft Co., Ltd
18F Glass Tower, 946-1, Daechi-Dong, Kangnam-Gu, Seoul 135-708
South Korea
Tel: 82-2-6288-2114
Fax: 82-2-6288-2115
Email: info@tmax.co.kr
Web (Korean): http://www.tmax.co.kr

USA
Tmax Soft, Inc.
560 Sylvan Ave, Englewood Cliffs NJ 07632
USA
Tel: 1-201-567-8266
FAX: 1-201-567-7339
Email: info@tmaxsoft.com
Web (English): http://www.tmaxsoft.com

Japan
Tmax Soft Japan Co., Ltd.
6-7 Sanbancho, Chiyoda-ku, Tokyo 102-0075
Japan
Tel: 81-3-5210-9270
FAX: 81-3-5210-9277
Email: info@tmaxsoft.co.jp
Web (Japanese): http://www.tmaxsoft.co.jp

China
Beijing Silver Tower, RM 1507, 2# North Rd Dong San Huan,
Chaoyang District, Beijing, China, 100027
Tel: 86-10-6410-6148
Fax: 86-10-6410-6144
E-mail : info@tmaxchina.com.cn
Web (Chinese): http://www.tmaxchina.com.cn

1 소개

WebT 는 “Web Transaction”의 약자로 C/S 환경의 Middleware 제품인 Tmax 와 Web 환경의 WAS(Web Application Server)제품인 JEUS 간의 Transaction 서비스를 지원하는 제품이다.

WWW 를 이용한 인터넷 서비스는 그 자체가 인터넷을 대변할 정도로 가장 대중적인 인터페이스이다. 단순한 정적인 정보만 제공하던 WWW 서비스가 최근에는 CGL, JSP, Servlet, Applet, PHP 등 동적 정보를 제공할 수 있는 기술이 등장하면서 동적 데이터 서비스의 비중이 점점 커지고 있다.

또한, 근래에 들어서 뱅킹 서비스, 전자 상거래 등 Transaction 업무를 가장 대중적인 서비스 방법인 Transaction 을 통하여 제공하려는 시도가 이루어 지고 있다. 이러한 Transaction 서비스나 동적 데이터 서비스를 서버에 가중되는 부하가 상대적으로 높으며 웹 서버 자체에는 Transaction 단위의 일 처리 개념이 없기 때문에 그 보완책으로 TP – Monitor 와 같은 Middleware 를 Web Server 와 연동하는 방식이 많이 시도되고 있다.

TP –Monitor 와 연동하여 Web Server 를 운영할 경우 Transaction 서비스를 Web 을 통해 제공할 수 있을 뿐만 아니라, TP – Monitor 의 또 다른 주기능인 부하조절(Load balancing) 기능을 통하여 적은 자원으로 효율적인 서버 시스템을 구축할 수 있게 된다.

WWW 를 통한 서비스의 종류가 다양해지고 복잡해짐에 따라, Web Server 본연의 기능보다는 서비스 처리에 걸리는 부하량이 상대적으로 커지고 있다. 따라서, 효율적인 서버를 구축하기 위해서는 Web Server 와 서비스처리 응용프로그램은 서로 다른 Machine 으로 분리하는 것이 필수적이다. 이와 같은 효율적인 시스템 구축에 있어서 WebT 와 Tmax 를 이용하여 다수의 서버 Machine 을 연동하는 것을 손쉽게 구현할 수 있다.

2 WebT 모듈의 개요

2.1 소개

이 장에서는 WebT module 에서 다음의 중심 주제에 대해서 간략하게 살펴보도록 한다.

- WebT 기능과 역할
- WebT connection pool 설정
- WebT 환경파일 설정
- WebT Java API 프로그래밍
- WebT-Server 시스템

2.2 WebT 의 기능과 역할

WebT 와 Tmax Server 간의 서비스 흐름에 대한 구성은 다음과 같다.

아래의 그림은 클라이언트 프로그램으로부터 요청을 받으면 WebT 모듈을 통해서 Tmax Server 의 서버 프로그램이 수행되어 서비스가 이루어지는 흐름을 간략하게 설명하고 있다.

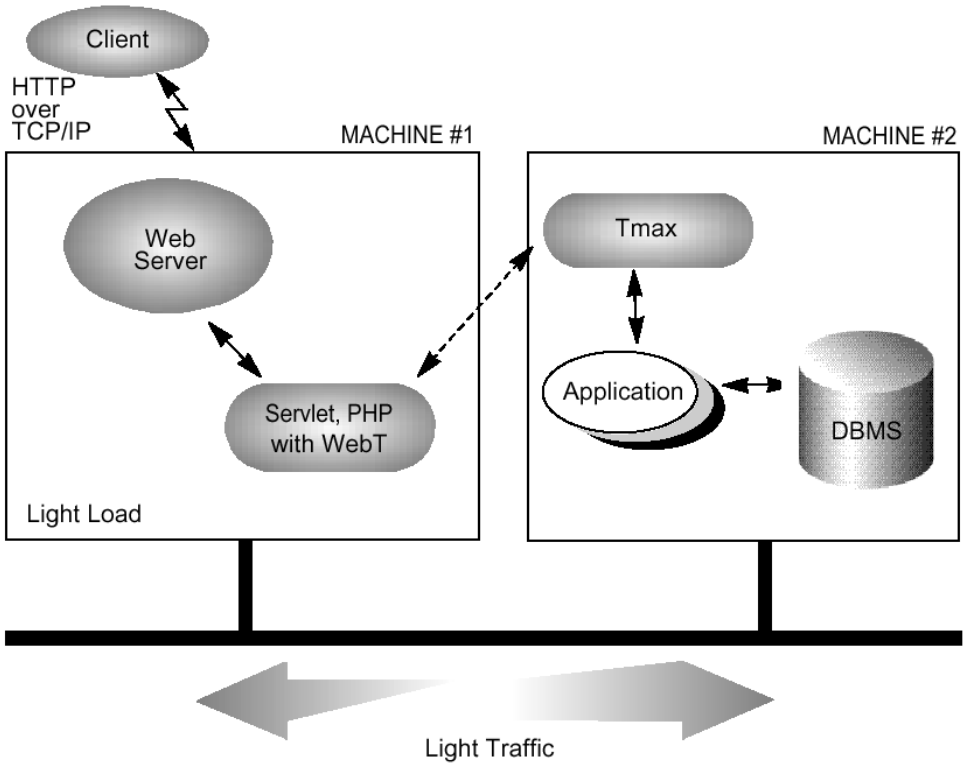

그림 1 WebT Overview

2.3 WebT Connection Pool 설정

JEUSMain.xml 에서 WebT *connection pool* 을 resource 로 등록하여 설정할 수 있다. 이 pool 설정은 EJB 혹은 Servlet, JSP Application Service 에서 Tmax server 의 TP service 를 이용할 수 있도록 해준다. 좀 더 자세한 사항은 이 매뉴얼의 3 장에서 설명하도록 하겠다.

2.4 WebT 환경파일 설정

JEUSMain.xml 에 WebT 관련 configuration 을 설정하는 방법 이외에 “webt.properties” 라고 불리는 properties file을 정의함으로써 JEUSMain.xml 환경 파일의 설정 요소인 WebT logging 과 monitoring 환경 정보, Tmax server와의 연결을 위한 기본 값 들을 설정할 수 있다.

WebT 환경 설정에 대해서 이후 4 장 에서 좀더 자세히 설명하겠다.

2.5 WebT Java API 프로그래밍

WebT 는 당사의 Tmax(TP -Monitor)와 Web Server 혹은 WAS 간을 연동하는 기술로써 Library 형태로 제공된다. 웹 환경을 지원하는 WAS 와 연동하는 경우에 기존의 WAS 가 Java 기반의 Software 라면 이를 C 기반의 TP-Monitor 계열의 Middleware 와의 연동을 위해서 별도의 Interface 를 제공해 주어야 한다. WebT 는 Java 언어를 통해 구현하였기 때문에 Applet, Servlet 과 같은 보편적인 방법을 사용하여 쉽게 Web Server 와 Tmax(TP-Monitor)를 연동한 Transaction 및 동적 데이터 서비스를 사용자에게 제공해 줄 수 있다

WebT Java API 는 5 장에서 설명하였고 6 장에서 관련 예제들을 살펴볼 수 있다. Appendix B 에는 WebT 관련 API reference 를 정의해 놓았다.

WebT 를 사용해서 application을 개발하기 위해서는 앞에 언급한 관련 장들를 반드시 살펴보도록 한다.

2.6 WebT-Server 시스템

WebT-Server system 은 JEUS (Web application server)에서 Tmax 로 연결되는 gateway 중의 하나이다. WebT 는 Java client 에서 Tmax service 를 호출할 수 있도록 하였다. WebT-Server 는 반면에 Tmax 에서 JEUS EJB service 의 결과를 받아올 수 있도록 한다.

WebT-Server system 에 대한 자세한 설명과 예제는 7 장에서 살펴보도록 한다..

2.7 결론

WebT 와 관련하여 간략하게 환경 설정, Web container-hosted WebT connection–pool, WebT application development 과 WebT-Server system 에 대해서 살펴보았다. 이에 언급된 주제에 대해서 지금부터 자세히 설명하도록 한다.

3 WebT DataSource 설정

3.1 소개

Jeus 와 Tmax 를 연동하는 service 에서는 JEUSMain.xml 에 WebT DataSource resource 를 등록해서 Tmax 와 연결을 하는 것이 효율적이다.

이 장에서는 JEUSMain.xml 에서의 WebT DataSource 설정에 대해서 설명한다.

3.2 WebT Connection Pool 개요

3.2.1 소개

대개 DB 나 legacy EIS 와 같이 JNDI 에 등록함으로써 동적으로 JEUS 에 추가할 수 있는 외부 데이터 저장소나 서비스들을 resource 라고 한다. resource 는 비즈니스 데이터와 백엔드 서비스로 완벽한 JEUS WAS 솔루션을 형성한다.

현재 JEUS 에서는 다섯 종류의 resource 들을 지원하고 이들은 모두 JEUS Manager 에서 최상위 레벨에 설정한다. [그림 2]

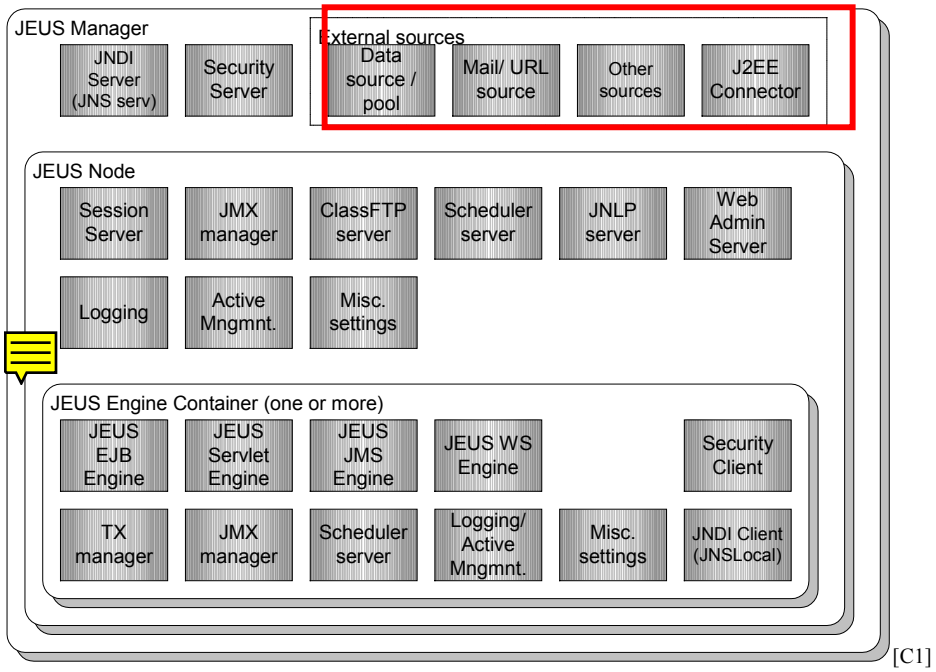

그림 2 JEUS architecture 에서의 resource

이 장에서 Other sources 의 한 종류로 EJB 혹은 Servlet, JSP Application 이 Tmax 의 TP monitor server 와의 서비스를 이루고자 할 때 이용되는 WebT DataSource 를 등록하는 resource 에 관하여 자세히 알아보도록 한다.

이후 설명에서 “resource”와 “source”라는 단어를 같은 의미로 사용한다.

3.2.2 WebT DataSource

WebT 는 TP Monitor 인 Tmax 와 웹 어플리케이션을 연결하는 gateway 제품이다. 자바 언어를 이용하여 개발된 제품으로써 JEUS 웹 컨테이너와 같은 자바 기반의 어플리케이션에서 Tmax 서비스를 호출할 수 있는 기능을 제공한다. JEUS 에서는 EJB 나 서블릿, JSP 에서 손쉽게 Tmax 서비스에 접근할 수 있도록 WebT DataSource 설정 기능을 가지고 있다. 여기서는 JEUSMain.xml 의 tmax external-source 에 대해서 알아보기로 한다.

tmax external-source 에 WebT DataSource 를 등록하여 Tmax 서버로의 네트워크 연결인 WebtConnection 객체를 관리하는 pool 기능을 수행한다.

WebtConnectionPool 클래스를 이용하면 Tmax 서비스 요청 시마다 매 번 WebtConnection 객체를 새로 생성하지 않고 이전에 사용한 WebtConnection 객체를 재 사용할 수 있으므로 Tmax 서버로의 네트워크 연결을 설정/종료하는데 소비되는 자원 및 시간을 절약할 수 있다.

WebtConnectionPool 은 한 개 이상의 WebtConnectionGroup 으로 구성된다. 각 WebtConnectionGroup 은 한 개의 Tmax Server 와 연결된다. 사용자 프로그램에서는 WebtConnectionGroup 의 GroupName 을 이용하여 원하는 Tmax Server 에 대한 연결을 얻을 수 있다.

해당 WebT 모듈을 JEUS 와 연동하였을 경우 JEUS Container 에서 Webt Connection Pool 에 대한 관리를 하여 webt.properties 환경파일을 이용하거나 JEUS 환경파일을 이용하여 pool 을 생성하고 User 가 getConnection 후 반환되지 않은 connection 에 대해서 자동적으로 pool 에 connection 을 반환하도록 한다.

3.2.3 결론

이 장에서는 JEUS 에서의 WebtConnectionPool 을 tmax external-source 에 등록하는 구성들을 살펴보았다. 이제 우리는 이에 대한 JEUSMain.xml 의 환경 설정 방법에 대해서 살펴보도록 한다.

3.3 WebT DataSource 설정

3.3.1 소개

이 장에서는 JEUSMain.xml 에 WebT DataSource 에 대한 설정방법을 살펴보도록 한다.

3.3.2 Tmax External Source 설정

JEUSMain.xml 파일에 다음의 값들을 설정할 수 있다.

- **export-name:** JNDI 에 등록되어 서비스되는 이름을 정한다.

- **webt-datasource-type:** webt datasource 를 구성하는데 필요한 webt connection pool 에 대한 속성을 정의한다. XA Transaction 을 지원하는 서비스인 경우에는 WebtXADataSource, 그 이외의 서비스인 경우에는 WebtConnectionPoolDataSource 로 정의한다.
- **webt-logging file name(default:webt.log):** 생성될 WebT log file 경로와 이름을 정의한다. OS 가 Windows 인 경우에는 파일 구분자를 '\'가 아닌 '\\'로 해 주어야 한다.
- **webt-logging level (default:none):** WebT log level 다음의 none|info|debug 값들 중 하나를 설정한다. log.level 이 info 이상이면 monitoring thread 을 설정하였을 경우 webt connection pool 의 상태를 살펴볼 수 있다.
- **webt-logging valid-day:** webt log file 에 대한 valid-days 값을 설정한다. 이 항목에 0 이상의 값이 설정되어 있으면 monitoring check 주기 마다 현재 시간값과 비교하여 valid-day 별로 로그 파일이 생기도록 한다.
- **webt-logging buffer-size(default:0):** log file 에 대한 buffering size 를 설정한다.
- **host-name:** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고자 하는 Tmax server 의 host address 를 설정한다.
- **port** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고자 하는 Tmax server 가 listen 하는 port 값을 설정한다.
- **backup-host-name:** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고 있는 Main Tmax server 에 장애가 발생하였을 경우 Backup Tmax server 와의 connection 을 시도하기 위해 Backup Tmax server 의 host address 를 설정한다.
- **backup-port:**각각의 WebtConnectionPoolGroup 이 connection 을 맺고 있는 Main Tmax server 에 장애가 발생하였을 경우 Backup Tmax server 와의 connection 을 시도하기 위해 Backup Tmax server 의 port 를 설정한다.
- **fdl-file:** tmax fdl file 의 위치를 설정한다.
- **default-charset(default: system default charset):** Application 에 적용될 default character set 을 설정한다. 기본적으로 WebtSystem.setDefaultCharset(java.lang.String charset)수행과 동일하다.

- **enable-pipe(default:false):** WebT 와 Tmax server 가 한 machine 에 존재할 경우 pipe 통신 여부를 설정하여 performance 를 꾀할 수 있다.
- **enable-extended-header(default:false):** Tmax Header가 확장된 3.11.x 이후 버전과의 서비스가 성공적으로 이루어 지도록 설정하는 option 값이다.
- **input-size(default:4096):** Tmax 로부터 데이터 수신 시 한번에 읽어 들일 buffer size 단위를 설정한다.
- **output size(default:4096):** Tmax 로부터 데이터 송신 시 한번에 송신할 buffer size 단위를 설정한다.
- **max-idle-timeout**: connection pool 에서 이 항목에 설정된 maxIdleTime 동안 사용되지 않은 idle 상태의 connection 이 있을 경우, 해당 connection 객체를 connection pool 에서 remove 한다. 최소한의 iniCapacity 값은 유지하도록 한다.
- **service-timeout:** Tmax Server 로 service를 요청한 후 이 항목에 설정된 시간 내에 서비스가 이루어지지 않으면 해당 서비스 요청을 중지한다.
- **transaction-timeout:** Tmax Server 로 Transaction service 를 요청한 경우 이 항목에 설정된 시간 내에 서비스가 이루어지지 않으면 요청된 서비스를 중지한다.
- **transaction-block-timeout:** Tmax Server 로 Transaction commit service 를 요청한 경우 이 항목에 설정된 시간 내에 commit 서비스의 정상적인 종료가 이루어지지 않으면 요청된 서비스를 중지한다.
- **security user-name:** Tmax server 로의 연결 시 Tmax server 에 등록된 user 에 대한 인증을 위해 user name 을 설정한다.
- **security user-password:** Tmax server 로의 연결 시 Tmax server 에 등록된 user 에 대한 인증을 위해 user password 을 설정한다.
- **security domain-name:** Tmax server 로의 연결 시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain name 을 설정한다.
- **securiy domain-password:** Tmax server 로의 연결 시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain password 을 설정한다.
- **monitoring enable-check-alive(default:false):** ConnectionPool service 들의 connection 여부를 주기적으로 check 할지 여부를 설정한다.

- **monitoring enable-check-idle(default:false):** ConnectionPool service 들의 connection 의 idle 상태 여부를 주기적으로 check 할지 여부를 설정한다.
- **monitoring check-pool-interval(default:60sec):** ConnectionPool service 들의 connection 여부 check 주기값을 설정한다.
- **pooling min(default:2):** 각 WebtConnectionPoolGroup 에 대한 초기 connection 수이자 최소한 유지해야 할 connection 수를 설정한다.
- **pooling max(default:30):** 각 WebtConnectionPoolGroup 에 최대 connection 수를 설정한다.
- **pooling step(default:4):** 각 WebtConnectionPoolGroup 의 connection thread 증가치를 설정한다.
- **pooling period(default:3600000):** pooling 되는 객체를 정리하는 시간을 지정한다.
- **wait-free-connectionType enable-wait(default:false):** 이 태그는 pool 안에 이용 가능한 connection 이 없거나 pool 안에 connection 들이 이미 최대값이 되었을 때 connection 을 요청 처리하는 방법을 결정한다. 만약 true 라면 시스템은 이용 가능한 connection 을 얻기 위해 대기한다. 만약 false 라면, 시스템은 사용자 요청이 올 때 새로운 connection 을 만들고 사용이 끝난 이후에 pool 에 반납하지 않는다.
- **wait-free-connectionType wait-time (default:10000):** 이 태그는 <enable-wait>가 true 일 때만 유효한 항목으로. 사용자가 connection 을 위해 대기하는 시간을 설정한다. 만약 사용자가 이 시간 동안 대기하여도 connection 을 이용할 수 없을 경우라면 시스템은 사용자에게 exception 을 던져준다.
- **tmax-property :** 이 항목은 일반적인 속성 이외에 사용자가 Tmax 와의 통신 시 추가로 정의하고자 하는 속성 값을 정의할 때 사용한다. WebtEventConnection 에 대한 pooling service 를 이용하고자 한다면 다음의 property name 에 EventSvcType 와 eventHandler를 설정해야 한다. EventSvcType에는 broadcast, tpacall,sendtocli,all 중에 하나를 선택하고, eventHandler 에는WebtEventConnection 에 적용될 WebtEventHandler 를 implement 한 handler object 객체의 클래스를 정의한다.

다음은 JEUSMain.xml 에 external-source 를 설정한 예제이다.

```
<resource>
 <external-source>
     <tmax>
        <export-name>tmax1</export-name>
        <webt-datasource-type>WebtConnectionPoolDataSource</webt-
datasource-type> <!-- WebtXADataSource -->
        <host-name>61.77.153.50</host-name>
        <port>8855</port>
        <backup-host-name>61.77.153.51</backup-host-name>
        <backup-port>8888</backup-port>
        <fdl-file>c:/tmax/fdl/tmax.fdl</fdl-file>
        <default-charset>euc-kr</default-charset>
        <enable-pipe>false</enable-pipe>
        <enable-extended-header>false</enable-extended-header>
        <inbuf-size>4096</inbuf-size>
        <outbuf-size>4096</outbuf-size>
        <max-idle-timeout>60000</max-idle-timeout>
        <service-timeout>120000</service-timeout>
        <transaction-timeout>100000</transaction-timeout>
        <transaction-block-timeout>30000</transaction-block-
timeout>
        <security>
               <user-name>choco</user-name>
               <user-password>1234</user-password>
               <domain-name>tmax</domain-name>
               <domain-password>4321</domain-name>
        </security>
        <tmax-connection-pool>
               <pooling>
                      <min>10</min>
                      <max>20</max>
                      <step>2</step>
```

```
                <period>2</period>
            </pooling>
            <wait-free-connection>
                <enable-wait>true</enable-wait>
                <wait-time>60000</wait-time>
            </wait-free-connection>
        </tmax-connection-pool>

        <webt-logging>
            <level>debug</level>
        </webt-logging>

        <monitoring>
            <enable-check-alive>true</enable-check-alive>
            <enable-check-idle>true</enable-check-idle>
            <check-pool-interval>3000</check-pool-interval>
        </monitoring>
        <tmax-property>
            <name>eventSvcType</name>
            <value>all</value>
        </tmax-property>
        <tmax-property>
            <name>eventHandler</name>
            <value>GenericServlet</value>
        </tmax-property>
    </tmax>
        </external-source>
    </resource>
```

3.3.3 결론

이번 장에서 Webt connection pool 과 WebT connection group 에 대한 설정 방법을 살펴보았다.

다음장에서는 사용자 프로그램에서 Webt connection pool 을 어떻게 이용하는지 살펴보도록 한다.

3.4 WebT Connection Pool 프로그래밍

위에서 설정한 것 처럼 JEUSMain.xml 에 등록되어 container 에 생성된 WebtConnectionPool 을 이용하여 Tmax server 의 service 를 어떻게 호출하는 지를 클라이언트 프로그램을 통해서 살펴보도록 한다.

```
// WebtConnection Object connected in TmaxServer1 is obtained from
// pool
Context ctx = new InitialContext();
WebtDataSource wds = (WebtDataSource)ctx.lookup("tmax1");
WebtConnection conn = wds.getConnection();
if (conn == null)
    return;
WebtServiceRemote service = new WebtRemoteService(conn, "MY_SVC");
WebtBuffer sndbuf = service.createStringBuffer();
sndbuf.setString("test data");

try {
    WebtBuffer rcvbuf = service.tpcall(sndbuf);
    System.out.println("service call success. Result = " +
Rcvbuf.getString());
} catch (WebtException e) {
    System.out.println("service call fail. Error is = " +
        service.getTPErrorMessage());
    e.printStackTrace();
} finally {
    // Used WebtConnection is returned to pool
    WebtConnectionPool.putConnection(conn);
}
```

3.5 결론

이번 장에서 JEUSMain.xml 에 WebT DataSource 에 대한 설정 방법과 생성된 connection pool 을 사용자의 클라이언트 프로그램에서 어떻게 이용하는지 살펴보았다. 다음 장에서는 WebT Library 가 Web Application program 에 어떻게 적용되는지 간단한 예제를 통해서 살펴보도록 한다.

4 WebT 환경파일 설정

4.1 소개

webt.properties 는 WebT 관련 환경 파일이다. WebT 에서 제공하는 library 를 참조하여 구성된 클라이언트 프로그램이 구동 될 때 기초 환경 설정 값을 초기화하여 적용시킨다. 또한 Application program 에서 WebtSystem API 를 사용하여 초기화 할 일을 webt.properties 파일에 설정할 수 있다.

4.2 webt.properties 설정

webt.properties 파일에 다음의 값들을 설정할 수 있다.

- **Log level (default:none)** WebT log level 다음의 none|info|debug 값들 중 하나를 설정한다. log.level 이 info 이상이면 monitoring thread 을 설정하였을 경우 webt connection pool 의 상태를 살펴볼 수 있다.
- **Log file path(default:stdout)** WebT log file 이 생성될 위치를 설정한다. OS 가 Windows 인 경우에는 파일 구분자를 '\'가 아닌 '\\'로 해 주어야 한다.
- **Log file name(default:webt.log)** WebT log file name 을 정한다.
- **Log buffering size(default:1024)** log file 에 대한 buffering size 를 설정한다.
- **FDL file path** fdl file 의 위치를 설정한다.
- **Pipe(default:false)** WebT 와 Tmax server 가 한 machine 에 존재할 경우 pipe 통신 여부를 설정하여 performance 를 꾀할 수 있다.
- **Input buffer size(default:4096)** Tmax 으로부터 데이터 수신 시 한번에 읽어 들일 buffer size 단위를 설정한다.
- **Output buffer size(default:4096)** Tmax 으로부터 한번에 데이터 송신할 buffer size 단위를 설정한다.

- **Default charset(default: system default charset)** Application 에 적용될 default character set 을 설정한다. 기본적으로 WebtSystem.setDefaultCharset(java.lang.String charset)수행과 동일하다.
- **ExtendedHeader(default:false)** Tmax Header 확장된 3.11.x 이후 버전과의 서비스가 성공적으로 이루어 지도록 설정하는 option 값이다.
- **Monitoring checkAlive(default:false)** ConnectionPool service 들의 connection 여부를 주기적으로 check 할지 여부를 설정한다.
- **Monitoring checkIdle(default:false)** ConnectionPool service 들의 connection 의 idle 상태 여부를 주기적으로 check 할지 여부를 설정한다.
- **Monitoring interval(default:60sec)** ConnectionPool service 들의 connection 여부 check 주기값을 설정한다.
- **Monitoring Valid-days** webt log file 에 대한 valid-days 값을 설정한다. 이 값이 0 이상의 값이 설정되어 있으면 monitoring check 주기 마다 현재 시간값과 비교하여 valid-day 별로 로그 파일이 생기도록 한다.
- **Enable to webt.properties modify** run-time 시 webt.properties 의 log.level 값이 수정되었을 경우 적용될 수 있는지의 여부를 설정한다. 이 값이 true 이면 monitoring check 주기 마다 webt.properties 파일의 수정 여부를 비교하여 log.level 값을 적용한다.
- **Enable to ConnectionPool(default:false)** WebtConnectionPool 사용 여부를 설정한다.
- **ConnectionPool group list** WebtConnectionPool 의 WebtConnectionPoolGroup name list 설정한다.
- **ConnectionPool type** 각 WebtConnectionPoolGroup 의 Type 을 설정한다. 설정값은 shared|non-shared|non-shared2 중의 한가지 값을 선택한다. Default 값은 shared 값이고 non-shared 와 non-shared2 은 Jeus Engine 과 함께 구성하여 서비스를 이용할 때만 설정 의미가 있다.
- **Tmax server host address** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고자 하는 Tmax server 의 host address 를 설정한다.
- **Tmax server host port** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고자 하는 Tmax server 가 listen 하는 port 값을 설정한다.

- **Backup Tmax server host address**
  각각의 WebtConnectionPoolGroup 이 connection 을 맺고 있는 Main Tmax server 에 장애가 발생하였을 경우 Backup Tmax server 와의 connection 을 시도하기 위해 Backup Tmax server 의 host address 를 설정한다.
- **Backup Tmax server host port** 각각의 WebtConnectionPoolGroup 이 connection 을 맺고 있는 Main Tmax server 에 장애가 발생하였을 경우 Backup Tmax server 와의 connection 을 시도하기 위해 Backup Tmax server 의 port 를 설정한다.
- **Tmax user name** Tmax server 로의 연결 시 Tmax server 에 등록된 user 에 대한 인증을 위해 user name 을 설정한다.
- **Tmax user password** Tmax server 로의 연결 시 Tmax server 에 등록된 user 에 대한 인증을 위해 user password 을 설정한다.
- **Tmax domain name** Tmax server 로의 연결 시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain name 을 설정한다.
- **Tmax domain password** Tmax server 로의 연결 시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain password 을 설정한다.
- **ConnectionPool iniCapacity** 각 WebtConnectionPoolGroup 에 대한 초기 connection 수이자 최소한에 유지해야 할 connection 수를 설정한다.
- **ConnectionPool maxCapacity** 각 WebtConnectionPoolGroup 에 최대 connection 수를 설정한다
- **ConnectionPool incrementRate** 각 WebtConnectionPoolGroup 의 connection thread 증가치를 설정한다.
- **ConnectionPool maxIdleTime** connection pool 에서 여기서 설정된 maxIdleTime 동안 사용되지 않은 idle 상태의 connection 이 있을 경우 해당 connection 객체를 connection pool 에서 remove 한다. 최소한의 iniCapacity 값은 유지하도록 한다.
- **ConnectionPool tpTimeout** Tmax Server 로 service 요청하여 여기서 설정된 시간 내에 서비스가 이루어지지 않으면 해당 서비스 요청을 중지한다.
- **ConnectionPool txTimeout** Tmax Server 로 Transaction service 를 요청한 경우 여기서 설정된 시간 내에 서비스가 이루어지지 않으면 요청된 서비스를 중지한다.

- **ConnectionPool connectTimeout** Tmax Server 로 connection 연결 시 해당 설정값 이내에 connection 이 연결이 되지 않으면 timeout 으로 해당 connection 을 close 하여 다른 작업이 이루어 질 수 있도록 한다.
- **Enable to WebtConnectionPool about WebtEventConnection** WebtEventConnection 에 대한 pooling service 를 이용할 지의 여부를 설정 한다.
- **Event Service Type** WebtConnectionPool 에 대한 connection 이 WebtEventConnection 일 경우 해당 event service type 일 설정한다.다음의 broadcast, tpacall,sendtocli,all 값들 중에 선택을 한다. Tppost 서비스의 경우는 이 기능이 제공되지 않는다.
- **Event Handler Object** WebtConnectionPool 에 실제 WebtEventConnection 에 적용될 WebtEventHandler 를 implement 한 handler object 객체의 클래스를 정의한다.

다음은 webt.properties 예제이다.

Note: 라인의 시작이'#' 시작되었으면 주석 문장을 의미한다.

```
#
# logging related parameters
############################

# set log level valid values are none, info, debug. default is
none
log.level=none

# set directory in which the log file places. if not set, log is
# printed to standard out
log.dir=/tmp

# set the name of the log file. default is webt.log
log.file=webt.log
```

```
# set log buffering size. default is 0
#log.bufsize=1024

#
# FDL related parameters
########################

# set the fdl file.
#fdl.file=/east/hope/prod/jeus/system/tmax.fdl

#
# pipe related paramters
########################

# enable/disable pipe. default is disable(false)
#pipe=false

#
# input/output buffer size parameter

# set input buffer size. default is 4096. minimum is 1024
#inbuf.size=4096

# set output buffer size. default is 4096. minimum is 1024
#outbuf.size=4096

# set application wide default character set. default is system
default
defaultCharset=euc-kr

# Tmax 3.11.x extended Header size
#extendedHeader=true

#
# monioring related parameter
```

```
###################################

# enable/disable alive check. default is disable(false)
# monitoring.pool.checkAlive=true

# enable/disable idle check. default is disable(false)
# monitoring.pool.checkidle=true

# set monitoring interval. default is 60sec
# monitoring.pool.interval=5

# set monitoring log file valid-days.default is -1
# monitoring.log.validDays=1

# enable/disable webt.properties modify. Default is disable(false)
# enableModify=false

# connection pool realted paramter
###################################

# enable/disable connection pool. default is disable(false)
#enableConnectionPool=true

# WebtConnectionGroup name list
#connectionPool.groups=tmax1

# set connection group type valid values are shared, non-shared,
non-shared2. default is shared
#connectionPool.tmax1.type=shared

# set Tmax Server Address.
#connectionPool.tmax1.hostAddr=61.77.153.1

# set Tmax Server Port.
#connectionPool.tmax1.hostPort=8888
```

```
# set Backup Tmax Server Address.
#connectionPool.tmax1.hostBackupAddr=61.77.153.1

# set Backup Tmax Server Port.
#connectionPool.tmax1.hostBackupPort=8889

# set default user name for security.
#connectionPool.tmax1.userName=tmax

# set default user passward for security.
#connectionPool.tmax1.userPasswd=1234

# set default domain name for security.
#connectionPool.tmax1.domainName=choco

# set default domain password for security.
#connectionPool.tmax1.domainPasswd=1234

# set initial pool size. default is 10
#connectionPool.tmax1.initCapacity=1

# set max pool size. default is 20
#connectionPool.tmax1.maxCapacity=10

# set increment step size. default is 5
#connectionPool.tmax1.incrementRate=2

# set connection idle time. default is 60sec
#connectionPool.tmax1.maxIdleTime=60

# set tptimeout.
#connectionPool.tmax1.tpTimeout=40

# set txtimeout.
```

```
#connectionPool.tmax1.txTimeout=10

# set connection timeout.
#connectionPool.tmax1.connectTimeout=10

# enable to event service.
#connectionPool.tmax1.enableEvent=true

# set event service Type.
#connectionPool.tmax1.eventSvcType=10

# set event handler object.
#connectionPool.tmax1.eventHandler=tcpservet.event.GenericEvent
```

4.3 webt.properties 적용

1. classpath 경로에 해당 webt.properties 파일이 존재하면 설정 값들이 적용될 수 있다.(참고:ClassLoader.getResourceAsStream("webt.properties")). 다시 말하면 Webt Service 를 하고자 하는 resourse 가 구동 시 현 current directory 에서 클래스 로딩시 webt.properties 파일이 존재하면 해당 파일에 설정된 Webt 환경 변수값들이 적용이 된다. 혹은 Jeus System 에서 기본적으로 classpath 잡혀 있는 $JEUS_HOME/lib/application 에 위치하였을 경우에는 Jeus 전 engine 에 공통으로 webt.properties 값을 적용 시킬 수 있다.
2. JeusMain.xml 에 CommandOption element 에 –Dwebt.properties=webt.properties 의 위치 경로를 설정되어 있으면 해당 properties 파일을 찾을 수 있다. Properties filename 변경도 가능하다.
3. 1,2 항목 이외에는 webt.properties 파일에 설정될 속성들이 default 값으로 적용된다.
4. Applet Service 인 경우 Applet 구동 시 webt.properties 파일을 로딩하여 WebT 관련 환경 변수 값을 초기화 하려면 작성한 webt.properties 파일을 applet tag 에 정의된 codebase 가 가리키는 디렉토리에 위치해 두면 된다.
5. Applet 구동 시 webt.properties 에 정의된 fdl 에 파일을 함께 다운로드 하려면 fdl 파일을 codebase 바로 밑에 혹은 그 하위 디렉토리에 위치해 놓아야 한다. 이때 fdl.file 경로가 '/'로 시작되면 WebT 는 이를 파일 시스템으로 간주하기 때문에 webt.properties 파일의 fdl.file 경로는 '/'시작되어서는 안되고 상대 경로로 설정해야 한다.
6. Applet 이 아니라 Servlet 혹은 Java Application 에서 fdl file 을 로딩하려면 파일 시스템의 절대 경로를 적어주어야 한다.

4.4 결론

우리는 webt.properties 파일에 WebT 환경 설정에 대해서 살펴보았다.

다음 장에서는 Web container 에서 WebT connection pool 설정에 대해서 살펴보도록 한다.

5 WebT API

5.1 소개

다음은 개발자가 WebT Class Library 를 이용하여 어떻게 사용자들이 클라이언트 프로그램을 구성할 수 있는지를 개략적으로 설명한다.

5.2 WebtConnection

WebtConnection class 는 Tmax Server 와 WebT 사이의 communication 을 담당하는 클래스이다. Tmax Server 가 제공하는 service 를 제공받으려면 반드시 이 WebtConnection 을 통한 네트워크 연결을 통해 서비스 요청을 해야한다

5.3 WebtConnectionGroup

WebtConnection 에 대한 pooling 기능을 담당하는 추상 클래스이다. 같은 주소/포트 번호를 갖는 다수의 WebtConnection 객체의 pooling 을 관리한다. 사용자가 직접 객체를 생성할 수는 없다. WebtConnectionPool 의 createGroup method 를 사용하여 객체를 생성할 수 있다

5.4 WebtConnectionPool

WebtConnectionPool 클래스는 WebtConnectionManager 를 대체하는 connection pool 클래스이다. WebtConnectionPool 은 tmax 서버로의 네트워크 연결인 WebtConnection 객체를 관리하는 pooling 기능을 수행한다. WebtConnectionPool 클래스를 이용하면 Tmax 서비스 요청시마다 매번 WebtConnection 객체를 새로이 생성하지 않고 이전에 사용한 WebtConnection 객체를 재사용할 수 있으므로 Tmax 서버로의 네트워크 연결을 설정/종료하는데 소비되는 자원 및 시간을 절약할 수 있다.

WebtConnectionPool 은 WebtConnectionManager 에 비해 다음과 같은 기능이 추가되었다.

1. 하나 이상의 tmax 서버와의 connection pool 을 관리할 수 있다.
2. connection pool monitoring 기능이 추가되었다. 지정한 시간동안 idle 인 연결을 해제하거나 접속이 종료된 연결을 찾아내어 새로 연결해 줄 수 있다.

아래의 예제는 2 개의 tmax 서버(192.168.128.1:8888, 192.168.128.2:8888)에 대한 connection pool 을 제공할 수 있도록 WebtConnectionPool 을 초기화하는 예제이다.

```
....

WebtConnectionPool.createGroup("tmax1","192.168.128.1",8888,2,10,2
);

WebtConnectionPool.createGroup("tmax2","192.168.128.2",8888,4,10,1
);
....
```

위에서 초기화한 WebtConnectionPool 로부터 WebtConnection 객체를 가져오는 예제는 다음과 같다. WebtConnectionPool.createGroup 의 첫번째 인자로 주는 groupname 을 사용하여 WebtConnection 객체를 얻어 온다.

```
try {
    // tmax1 group (192.168.128.1:8888 로 부터 WebtConnection 객
체를 얻는다
    WebtConnection con =
WebtConnectionPool.getConnection("tmax1");
} catch (WebtIOException wioe) {
    System.out.println("fail to get WebtConnection of tmax1");
    return;
}
```

```
    WebtRemoteService service = new WebtRemoteService("MYSVC",
con);
    ....
    // WebtConnection 객체 반납
    WebtConnectionPool.putConnection(con);
```

5.5 WebtRemoteService

이 클래스는 WebtConnection 및 그 자식 클래스를 통해 Tmax Service 를 제공받을 수 있게 하는 API 를 제공하고 Tmax service 호출에 필요한 attribute 및 서비스 이름 등을 관리하는 클래스 이다. WebtService 를 대체하는 클래스이다

5.6 WebtDialogueService

이 클래스는 tmax conversation mode service 를 제공하는 클래스이다.

```
   ....
   WebtConnection con =
WebtConnectionPool.getConnection("tmax1");
   WebtDialogueService dialogue = new
WebtDialogueService("CONVTEST", connection);
   try {
       dialogue.tpconnect(false);
       int count = 1;
       WebtBuffer sndbuf = dialogue.createStringBuffer();
       WebtBuffer rcvbuf = null;
       while(true) {
          sndbuf.setString("client msg " + count);
          dialogue.tpsend(sndbuf, true);
          rcvbuf = dialogue.tprecv();
     System.out.println("tprecv ok : " + rcvbuf.getString());
```

```
    if (!dialogue.isSendNext())
            break;
      }
      //server tpreturn()
       rcvbuf = dialogue.tprecv();
       System.out.println("conv end : " + rcvbuf.getString());
  } catch (WebtDialogueException de) {
       de.printStackTrace();
  } catch (WebtServiceFailException se) {
       se.printStackTrace();
  } catch (WebtException e) {
       e.printStackTrace();
  } finally {
       connection.close();
  }
  ....
```

5.7 WebtEventConnection

Tmax Server 나 다른 client 로부터 전송된 event message (tppost/tpnotify/tpsendtocli/tpbroadcast) 처리가 가능한 WebtConnection 이다. Tmax Server 나 다른 client 가 보낸 비요청 메시지를 수신하여 처리하기 위해서는 반드시 WebtEventConnection 을 사용하여 Tmax Server 에 연결해야 하며 수신한 event message 를 처리하기 위한 callback 함수를 정의한 WebtEventHandler 를 implement 하고 setEventHandler (WebtEventHandler) 함수를 사용하여 event handler 를 등록해야 한다. 아래의 프로그램은 tppost message 를 수신하기 위한 event handler 예제이다.

```
import tmax.webt.*;
```

```
public class EventSample implements WebtEventHandler {
    private WebtEventConnection connection;

    public EventSample(WebtEventConnection connection) {
   this.connection = connection;

   // tpbroadcast event 만을 수신하기 위해 mask 를 설정한다.
   connection.setEventMask(WebtEventConnection.EVENT_POST);

   // event handler 를 등록한다.
   connection.setEventHandler(this);

       // event 에 가입한다.
       connection.tpsubscribe("posttest");
    }

    public void destroy() {
       connection.tpunsubscribe("posttest");
       connection.close();
    }

    // WebtEventHandler 의 함수이다. event message 를 수신하면
    //이 함수가 callback 된다.
    public void handleEvent(int type, WebtBuffer buf, int len,
int flags) {
   System.out.println("event received. type = " + type + ",
buffer type = " + buf.getBufferType());
   System.out.println("event msg : " + buf.getString());
    }

    // WebtEventHandler 의 함수이다. WebT 가 event 처리중 Exception 을
    // 만나면 이 함수를 callback 한다. WebtIOException 이
    // 발생하였다면 Tmax Server 로의 connection 이 끊어졌을 수 있으므로
reconnect 한다.
```

```
    public void handleError(WebtException e) {
   System.err.println("event error occured");
   e.printStackTrace();
   if (e instanceof WebtIOException) {
      if (e.getTPError() == WebtException.TPECLOSE) {
         connection.close();
         connection.connect();
            }
  }
    }
}
```

5.8 WebtEventHandler

이 인터페이스는 tmax server 로 부터 수신한 비요청 메시지를 처리할 때 사용하는 인터페이스이다. 비요청 메시지를 수신하기 위해서는 이 인터페이스를 implement 해야 한다. 비요청 메시지는 tpsendtocli/tpbroadcast/tpnotify/tppost acall_reply 가 있다.

5.9 WebtTransaction

이 클래스는 여러 Service 를 하나의 작업 단위로 묶어서 처리하고자 할때 이를 처리할 수 있는 API 를 제공하고 에러 상황에 대한 적절한 exception 을 handling 한다.

5.10 WebtConnectionInfo

WebtConnectionInfo 는 WebtConnection 생성에 필요한 파라메터를 관리하는 클래스이다. 이 클래스의 객체를 생성하여 연결에 필요한 파라메터를 설정하면 WebtConnection 의 생성자의 인자로 사용할 수 있다.

```
WebtConnectionInfo info = new WebtConnectionInfo();

// set tmax host address as 127.0.0.1
info.setAttribute(info.TMAX_ADDRESS, "127.0.0.1");

// set tmax host port as 8888
info.setAttribute(info.TMAX_PORT, 8888);

// set socket connect timeout as 30 sec
info.setAttribute(info.CONNECT_TIMEOUT, 30);

// constructs WebtConnection
WebtConnection connection = new WebtConnection(info);
```

5.11 결론

이 장에서는 WebT 환경 파일인 webt.properties file 의 적용 방법과 WebT 에서 개발자들에 제공하는 주요 API 를 간략하게 설명하였다. 이후 장에서는 자세한 환경 설정 방법과 구체적인 예제를 살펴보고 참조하도록 한다.

6 WebT Application 예제

이 장에서는 WebT Library 를 이용하여 제공될 수 있는 몇 가지 서비스 구성을 반영한 예제들을 설명하고자 한다. 제시된 예제를 통하여 전체적인 WebT Library 적용 방법에 대한 이해를 돕고자 한다.

6.1 Synchronous Communication

Synchronous Communication 은 Client 가 Service 를 요청하고 설정된 Timeout 시간 동안 응답을 기다리는 방식이다.

6.1.1 클라이언트 프로그램

Client 는 string type 의 buffer 에 문자열을 담아 서비스를 호출해서 Server 쪽에서 return 받은 값을 출력해 주는 program 이다.

Buffer type : STRING

Communication mode : Synchronous communication

SyncTest.java

```
Import tmax.webt.*;

public class SyncTest {
   public static void main(String[] argv) {
      WebtConnection connection = new
                  WebtConnection("61.77.153.1", 8888);
      WebtRemoteService service = new
            WebtRemoteService("STR_TOUPPERSTR", connection);

      WebtBuffer buffer = service.createStringBuffer();
```

```
        buffer.setString(" Synchronous communication Test");
      try {
        WebtBuffer rcvbuf = service.tpcall(buffer);
        System.out.println(" : " + rcvbuf.getString());
      } catch (WebtServiceFailException se) {
          se.printStackTrace();
      } catch (WebtException e) {
          e.printStackTrace();
      } finally {
          connection.close();
      }
   }
}
```

6.1.2 서버 프로그램

Server 측의 Service 루틴은 Client 으로 부터 문자열을 받아 대문자열로 바꾸어 return 해 주는 program 이다.

Buffer type : STRING

Communication mode : Synchronous communication

test.c

```
#include <stdio.h>
#include <usrinc/atmi.h>

STR_TOUPPERSTR(TPSVCINFO *msg)
{
   int i, len;
   char *instr = (char *)msg->data;

   len = strlen(instr);
```

```
    for(i=0; i<len; i++)
        instr[i] = toupper(instr[i]);

    tpreturn(TPSUCCESS, 0, (char *)instr, 0, 0);
}
```

6.1.3 Tmax 환경 파일과 makefile

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR =  "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax

*SERVER
test          SVGNAME = svg1,
            MIN = 1, MAX = 5,
            CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
STR_TOUPPERSTR       SVRNAME = test
```

Makefile.c

```
# Server makefile
```

```
TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o
CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
```

```
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.1.4 실행

```
cfl –i sample.m
javac –classpath $JEUS_HOME/classes:
            $JEUS_HOME/lib/system/webt30.jar:. SyncTest.java
make –f Makefile.c test
tmboot
java $JEUS_HOME/classes:
                  $JEUS_HOME/lib/system/webt30.jar:. SyncTest
```

6.2 Asynchronous Communication

Asynchronous Communication 은 Client 가 Service 를 요청하고 응답을 받고자 할 때 해당 요청에 대한 응답을 받는다. Client 는 그 시간 동안 별도의 다른 작업을 수행 할 수 있다.

6.2.1 클라이언트 프로그램

Client 는 Field type 의 buffer 에 서버쪽에 delay 한 시간에 대한 정보를 문자열을 담아 서비스를 호출해서 Server 쪽에서 return 받은 값을 출력해 주는 program 이다.

Buffer type : Field

Attribute : TPBLOCK

Communication mode : Asynchronous communication

AsyncTest.java

```
import tmax.webt.*;

public class AsyncTest {
   public static void main(String[] argv) {
      WebtConnection connection = new
                        WebtConnection("61.77.153.1", 8888);
      Connection.setTPtimeout(15);
      WebtRemoteService service = new
                  WebtRemoteService("NO_SUCH_SVC", connection);
      WebtAttribute attribute = new WebtAttribute();
      Attribute.setTPBLOCK(true);

      WebtBuffer sndbuf = service.createFieldBuffer();
      WebtField sndfld = sndbuf.createField(“TINT”);
      int sint = 10;
      sndfld.add(sint);

      int cd = 0
      long t1 = System.curretTimeMillis();

      try {
    cd = service.tpacall(sndbuf, attribute);
  } catch (WebtException e) {
    if (e.getTPError() == WebtException.TPENOENT)
    else {
       e.printStackTrace();
       throw new WebtTestException("exception at
                                          tpacall with TPBLOCK");
    }
  }

      long t2 = System.curretTimeMillis();
 System.out.println("[t2-t1] : " + (t2-t1) + "ms");
```

```
  try {
    WebtBuffer rcvbuf = service.tpgetrply(cd, attribute);
    long t3 = System.currentTimeMillis();

    System.out.println("[t3-t1] : " + (t3-t1) + "ms");

    WebtField rfield = rcvbuf.getField("TINT");
    WebtFieldElement e = rfield.get();
    int rint = e.intValue();
    if (rint != sint)
           throw new WebtTestException(sint, rint);
        System.out.println("[requested delay] : " + rint);

  }catch (WebtException e) {
    e.printStackTrace();
    System.out.println("tpgetrply error!!!");
  } finally {
                  connection.close();
                }
   }
}
```

6.2.2 서버 프로그램

Server 측의 Service 루틴은 Client 으로부터 받은 문자열의 값만큼 sleep 상태에 있다가 해당 시간 값을 return 해 주는 program 이다.

test.c

```
#include <stdio.h>
#include <unistd.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include "../fdl/demo_fdl.h"
```

```
#include <sys/timeb.h>

NO_SUCH_SVC(TPSVCINFO *msg) {
 int i, len, n;
 FBUF *rcvbuf = (FBUF *)msg->data;
 FBUF *sndbuf;
 int tint;
 int fldlen;
 len = msg->len;
 tint = 0;
 sndbuf = (FBUF *)tpalloc("FIELD", NULL, 1024);
 n = fbget(rcvbuf, TINT, (char *)&tint, &fldlen);
 printf("requested delay : %d\n", tint);
 n = sleep(tint);
 fbput(sndbuf, TINT, (char *)&tint, 0);
 tpreturn(TPSUCCESS, 0, (char *)sndbuf, 0, 0);
}
```

```
demo.f
# name  number type   flags  comments
TINT       1103       int    -            -
```

6.2.3 Tmax 환경 파일과 makefile

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR  = "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
```

```
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax

*SERVER
test    SVGNAME = svg1,
MIN = 1, MAX = 5,
CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
NO_SUCH_SVC    SVRNAME = test
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS    = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
```

```
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.2.4 실행

```
cfl –i sample.m
javac –classpath $JEUS_HOME/classes:
            $JEUS_HOME/lib/system/webt30.jar:. AsyncTest.java
fdlc –c –i $TMAXDIR/fdl/demo.f
make –f Makefile.c test
tmboot
java $JEUS_HOME/classes:
            $JEUS_HOME/lib/system/webt30.jar:. AsyncTest
```

6.3 Conversational Communication

Conversational Communication 은 Client 와 Server 가 연결을 설정하고 Event 값에 따라 데이터의 송/수신측을 결정하여 지속적으로 데이터를 주고 받을 수 있도록 한다.

6.3.1 클라이언트 프로그램

Client 는 String type 의 buffer 에 데이터를 담아 서버쪽으로 송신 제어권이 유지될 동안 메시지를 전송하고 서버쪽에서 return 받은 값을 출력해 주는 program 이다.

Buffer type : String

Communication mode : Conversational communication

ConvTest.java

```
import tmax.webt.*;

public class ConvTest {
   public static void main(String[] argv) {
      WebtConnection connection =
                      new WebtConnection("61.77.153.1", 8877);
      WebtDialogueService dialogue =
           new WebtDialogueService("CONVTEST", connection);

      try {
         dialogue.tpconnect(false);

         int count = 1;
         WebtBuffer sndbuf = dialogue.createStringBuffer();
         WebtBuffer rcvbuf = null;
         while(true) {
            sndbuf.setString("client msg " + count);
```

```
            dialogue.tpsend(sndbuf, true);
            rcvbuf = dialogue.tprecv();
            System.out.println("tprecv ok : " +
                                    rcvbuf.getString());
            if (!dialogue.isSendNext())
               break;
        int a = 0;
           }

           rcvbuf = dialogue.tprecv();
           System.out.println("conv end : " +
                                    rcvbuf.getString());
        } catch (WebtDialogueException de) {
           de.printStackTrace();
        } catch (WebtServiceFailException se) {
           se.printStackTrace();
        } catch (WebtException e) {
           e.printStackTrace();
        } finally {
           connection.close();
        }
    }
```

6.3.2 서버 프로그램

Server 측의 Service 루틴은 Client 으로 부터 문자열을 받아 대문자열로 바꾸어 return 해 주는 program 이다.

conv.c

```
#include <stdio.h>
#include <unistd.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include <sys/timeb.h>
```

```
CONVTEST(TPSVCINFO *msg)
{
    char *rcvbuf;
    char *sndbuf;
    int ret;
    long revent, rcvlen, flag, count;

    rcvbuf = tpalloc("STRING", NULL, 1024);
    sndbuf = tpalloc("STRING", NULL, 1024);

    printf("convtest start. msg from client : %s\n", msg->data);

    flag = 0;
    count = 1;
    while(1) {
        ret = tprecv(msg->cd, (char **)&rcvbuf,
                                    &rcvlen, TPNOTIME, &revent);
        if (ret < 0 ) {
             printf("tprecv ret = %d\n", ret);
         fprintf(stderr, "tprecv fail errno = %d\n", tperrno);
            if (revent != TPEV_SENDONLY) {
                printf("tprecv fail revent = 0x%08x\n", revent);
                tpreturn(TPFAIL, -1, (char *)rcvbuf,
                                                0, TPNOFLAGS);
            }
        }
        printf("tprecv ok! : %s\n", rcvbuf);
        if (count == 10) {
            flag = TPNOTIME;
        } else {
            flag = TPRECVONLY;
        }

        sprintf(sndbuf, "server msg %d", count);
```

```
        ret = tpsend(msg->cd, (char *)sndbuf,
                                strlen(sndbuf), flag, &revent);
        if (ret < 0) {
           printf("tpsend fail revent = 0x%08x\n", revent);
         fprintf(stderr, "tpsend fail errno = %d\n", tperrno);
           tpreturn(TPFAIL, -1, (char *)NULL, 0, TPNOFLAGS);
        }
        printf("tpsend ok! count = %d\n", count);
        if (count == 10)
           break;

        count++;
    }
    strcpy(sndbuf, "conv mode successfully terminated");
    tpreturn (TPSUCCESS, 1, (char *)sndbuf,
                                      strlen(sndbuf), TPNOFLAGS);
}
```

6.3.3 Tmax 환경파일과 makefile

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR  = "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax
```

```
*SERVER
conv          SVGNAME = svg1, CONV = Y
            MIN = 1, MAX = 5,
            CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
CONVTEST      SVRNAME = conv
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o
NSDLOBJ       = $(TMAXDIR)/lib/sdl.o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<
```

```
# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.3.4 실행

```
cfl –i sample.m
javac –classpath $JEUS_HOME/classes:
            $JEUS_HOME/lib/system/webt30.jar:. ConvTest.java
make –f Makefile.c conv
tmboot
java $JEUS_HOME/classes:
                  $JEUS_HOME/lib/system/webt30.jar:. ConvTest
```

6.4 Unsolicited Message Service

Unsolicited Message 는 어떤 요청에 대한 응답이 아닌 일방적으로 보내진 데이터를 처리하는 서비스이다.

6.4.1 클라이언트 프로그램

Client 는 임의의 여러 개의 Event 를 등록시켜 이벤트를 발생시킬 수 있다. 서버쪽으로 받은 발생된 이벤트의 데이터를 출력해 주는 program 이다.

Buffer type : String

Attribute : TPBLOCK, TPNOREPLY

Communication mode : Asynchronous communication

TppostSampe.java

```
import tmax.webt.*;

public class TppostSample implements WebtEventHandler {
   private WebtEventConnection conn;
   private WebtRemoteService service;

   private String eventName1 = "SITE";
   private String eventName2 = "SUB";
   private String eventName = "SEQNO";
   private String id = "ID";
   private String message = "MSG";

   public TppostSample(WebtEventConnection conn) {
      this.conn = conn;
      conn.setEventMask(WebtEventConnection.EVENT_POST);
      conn.setEventHandler(this);
      conn.tpsubscribe(eventName1);
      conn.tpsubscribe(eventName2);
   }

   public static void main(String[] argv)
            throws java.io.IOException {
```

```
        try{
              WebtSystem.createDefaultFieldKeyTable("/home/tmax/fdl/
              tmax.fdl");
            WebtSystem.initLogger(WebtSystem.LOG_DEBUG, null, 0);
        }
        catch (WebtException e){
            System.out.println("error: " + e);
        }

        WebtEventConnection conn = new
                          WebtEventConnection("61.77.153.1", 8888);
        TppostSample sample = new TppostSample(conn);

        int cd = 0;
        WebtAttribute attribute = new WebtAttribute();
        attribute.setTPBLOCK(true);
        attribute.setTPNOREPLY(true);
        WebtRemoteService service = new
                                 WebtRemoteService("ECHO",conn);
        WebtBuffer sndbuf = service.createStringBuffer();

  int acallCount = 0;
        while(true) {
            try {
                sndbuf.setString("EVENT_POST Flag Test" +
acallCount++);
                cd = service.tpacall(sndbuf, attribute);
                Thread.sleep(1000);
            }catch (Exception e ) {
                System.out.println("exception to tppost with
TPNOREPLY/TPBLOCK");
                e.printStackTrace();

                if (e instanceof tmax.webt.WebtException) {
                    Throwable r = ((WebtException)e).getRootCause();
```

```
                r.printStackTrace();
            }
        }
    }

    public void destroy() {
        conn.tpunsubscribe(eventName1);
        conn.tpunsubscribe(eventName2);
        conn.close();
    }

    public void handleEvent(int type, WebtBuffer buf, int len, int
flags) {
        WebtFieldSet bufset = new WebtFieldSet(buf);


System.out.println("########################################");
        System.out.println("event received. type = " + type + "
buffer type = " + buf.getBufferType());
        System.out.println("event name : " +
bufset.getString(eventName));
        System.out.println("event id : " + bufset.getInt(id));
        System.out.println("[event msg] : " +
bufset.getString(message));

System.out.println("########################################");
    }

    public void handleError(WebtException e) {
        System.err.println("Event error occured");
        e.printStackTrace();

        if (e instanceof tmax.webt.WebtException) {
            Throwable r = ((WebtException)e).getRootCause();
```

```
            r.printStackTrace();
        }

        if( e instanceof WebtIOException) {
            if(e.getTPError() == WebtException.TPECLOSE) {
                conn.close();
                conn.connect();
            }
        }
    }
}
```

tpcall.c

```
#include <stdio.h>
#include <stdlib.h>
#include <usrinc/atmi.h>

main()
{
  int    n, ret;
  char   *sndbuf, *rcvbuf;
  long   sndlen, rcvlen;

  if ((ret = tmaxreadenv("tmax.env","TMAX")) == -1) {
        printf( "tmax read env failed\n" );
        exit(1);
  }

  if ( tpstart((TPSTART_T *)NULL) == -1 ){
        printf( "tpstart failed[%s]\n",tpstrerror(tperrno));
        exit(1);
  }

  if ((sndbuf = tpalloc("CARRAY",NULL,1024)) == NULL) {
        printf( "sndbuf tpalloc failed[%s]\n",
```

```
                                                tpstrerror(tperrno));
        tpend();
        exit(1);
    }

    if ((rcvbuf = tpalloc("CARRAY",NULL,1024)) == NULL){
        printf( "rcvbuf tpalloc failed[%s]\n",
                                                tpstrerror(tperrno));
        tpfree((char *)sndbuf);
        tpend();
        exit(1);
    }
    sndlen = 1000;

    if (tpcall("POSTFDLTEST", (char *)sndbuf, sndlen,
                          (char **)&rcvbuf, &rcvlen, 0 ) < 0){
         printf( "tpcall SITETEST failed[%s]\n",
                                              tpstrerror(tperrno));
         return -1;
    }
    printf("tpcall success, len = %d\n", rcvlen);

    tpfree((char *)sndbuf);
    tpfree((char *)rcvbuf);
    tpend();
}
```

6.4.2 서버 프로그램

Server 측의 Service 루틴은 이벤트를 발생시켜 해당 이벤트에 참가하고 있는 모든 클라이언트 프로그램에 비요청 데이터를 보내주는 program 이다.

test.c

```
POSTFDLTEST(TPSVCINFO *rqst)
{
 FBUF *sndbuf1, *sndbuf2;
 int id1=++i;
 int id2=++i;
 char pwdString[50]="TPPOST TEST";
 char eventName1[20]=event;
 char eventName2[20]=sub;

 sndbuf1 = fballoc(100,100);
 sndbuf2 = fballoc(100,100);

 printf("[TPPOST FIELD TYPE TEST count=%d\n", i);

 fbput(sndbuf1, ID,(char *)&id1,0);
 fbput(sndbuf1, MSG,pwdString,0);
 fbput(sndbuf1, SEQNO,eventName1,0);
 printf("[ID] %d  [MSG] %s [SEQNO] %s\n", id1, pwdString,
eventName1);

 fbput(sndbuf2, ID,(char *)&id2,0);
      fbput(sndbuf2, MSG,pwdString,0);
      fbput(sndbuf2, SEQNO,eventName2,0);
 printf("[ID] %d  [MSG] %s [SEQNO] %s\n", id2, pwdString,
eventName2);

 tppost(event, (char *)sndbuf1, 0, TPNOFLAGS);
 tppost(sub,(char *)sndbuf2 , 0, TPNOFLAGS);
#if 1
 tpfree(sndbuf1);
 tpfree(sndbuf2);
 tpreturn(TPSUCCESS, 0, NULL, 0, 0);
#else
 tpreturn(TPSUCCESS, 0, (char *)sndbuf1, 0, 0);
 tpreturn(TPSUCCESS, 0, (char *)sndbuf2, 0, 0);
```

```
#endif
}

ECHO(TPSVCINFO *msg)
{
 printf("client message : %s\n " , msg->data);
 tpreturn(TPSUCCESS, 0 , msg->data,0,0);
}
```

6.4.3 Tmax 환경파일과 makefile

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
      APPDIR  = "/home/tmax/appbin",
      PATHDIR = "/home/tmax/path",
      TLOGDIR = "/home/tmax/log/tlog",
      ULOGDIR = "/home/tmax/log/slog",
      SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax

*SERVER
test    SVGNAME = svg1
MIN = 1, MAX = 5,
CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
POSTFDLTEST   SVRNAME = test
```

```
ECHO          SVRNAME = test
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)
```

```
$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

Makefile.cc

```
# Tmax client makefile
TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

TMAXLIBD= $(TMAXDIR)/lib

TMAXLIBS= -lcli
CFLAGS  = -O -I$(TMAXDIR)

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

#
# client compile
#

$(TARGET): $(APOBJS)
 $(CC) $(CFLAGS) -L$(TMAXLIBD) -o $(TARGET) $(APOBJS) $(TMAXLIBS)
```

```
#
clean:
 -rm -f *.o core $(TARGET)
```

6.4.4 실행

```
cfl -i sample.m
javac -classpath $JEUS_HOME/classes:
            $JEUS_HOME/lib/system/webt30.jar:. TppostSample.java
make -f Makefile.c test
make -f Makefile.cc tpcall
tmboot
./tpcall
java $JEUS_HOME/classes:
                  $JEUS_HOME/lib/system/webt30.jar:. TppostSample
```

6.5 Transaction Management

Transaction 은 하나의 자원에 접근하는 다수의 다단계 구성의 업무 처리 과정에 있어 일관적으로 자원의 상태 변화를 적용하기 위해 모든 구성단계를 하나의 작업 단위로 묶은 것이다. Transaction 과정은 이른바 ACID 특성을 지원함으로써 자원의 일관성을 보장 받게 된다.

Local Transaction : 하나의 자원관리자(DB)가 참여하는 Transaction

Global Transaction : 둘 이상의 자원관리자가 참여하는 Transaction

6.5.1 클라이언트 프로그램

Client 는 임의의 데이터를 DB 에 Insert 하는 서비스를 호출하고 tpacall service 가 성공적으로 이루어 지면 commit() 수행하는 program 이다.

Buffer type : Field

Communication mode : Asynchronous communication

TxTest.java

```
import tmax.webt.*;

public class TxTest {
     WebtConnectionPool.createGroup("tmax1", "61.77.153.1", 8888);
     try {
        WebtConnection connection =
WebtConnectionPool.getConnection("tmax1");
     } catch (WebtIOException wioe) {
        System.out.println("fail to get WebtConnection of tmax1");
        return;
     }

     WebtAttribute attribute = new WebtAttribute();
WebtTransaction tx = new WebtTransaction(connection);
 WebtRemoteService service = new WebtRemoteService("FDLINS",
connection);

 // first call
 WebtFieldSet sndset  = new
WebtFieldSet(service.createFieldBuffer());

 sndset.add("EMPNO", 1234);
 sndset.add("ENAME", "WebT");
 sndset.add("JOB", "Tmax");
 sndset.add("MGR", 1010);
 sndset.add("SAL", 10000);
 sndset.add("COMM", 10);
 sndset.add("DEPTNO", 12);

 int cd1 = 0;
```

```
int cd2 = 0;

tx.begin();
try {
          System.out.println("tpacall first!!!");
    cd1 = service.tpacall(sndset.getFieldBuffer(), attribute);
} catch (WebtException e) {
    tx.rollback();
    e.printStackTrace();
    throw new WebtTestException("exception at tpacall with
TPNOTRAN");
}
try {
    WebtBuffer rcvbuf = service.tpgetrply(cd1, attribute);
} catch (WebtException e) {
       e.printStackTrace();
       System.out.println("tpgetrply error!!!");
}

// second call
       WebtFieldSet sndset1  = new
WebtFieldSet(service.createFieldBuffer());

sndset1.add("EMPNO", 4321);
sndset1.add("ENAME", "Jeus");
sndset1.add("JOB", "Tmax");
sndset1.add("MGR", 1110);
sndset1.add("SAL", 20000);
sndset1.add("COMM", 20);
sndset1.add("DEPTNO", 24);

try {
          System.out.println("tpacall second!!!");
    cd2 = service.tpacall(sndset1.getFieldBuffer(), attribute);
} catch (WebtException e) {
```

```
    tx.rollback();
    e.printStackTrace();
    throw new WebtTestException("exception at tpacall with
TPNOTRAN");
 }
 try {
          WebtBuffer rcvbuf1 = service.tpgetrply(cd2, attribute);
  } catch (WebtException e) {
           e.printStackTrace();
           System.out.println("tpgetrply error!!!");
       }
 tx.commit();
 connection.close();
   }
 }
```

6.5.2 서버 프로그램

Server 측의 Service 루틴은 Client 로 읽어 들인 Field data 를 해당 DB Column 에 insert 하는 program 이다.

fdlTest.pc

```
#include <stdio.h>
#include <ctype.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include "../fdl/demo_fdl.h"

EXEC SQL include sqlca.h;

EXEC SQL begin declare section;
 int  h_empno;
 char h_ename[11];
 char h_job[10];
```

```
 int  h_mgr;
 float h_sal;
 float h_comm;
 int  h_deptno;
EXEC SQL end declare section;

FDLINS( TPSVCINFO *msg )
{
 FBUF *sndbuf;
 int n;
 int tint=0;

 sndbuf = (FBUF *)msg->data;

 h_empno = h_mgr = h_sal = h_comm = h_deptno = 0;

 memset( h_ename, 0x00, sizeof( h_ename ) );
 memset( h_job, 0x00, sizeof( h_job ) );

 fbget( sndbuf, EMPNO, (char *)&h_empno, 0 );
 fbget( sndbuf, MGR,   (char *)&h_mgr, 0 );
 fbget( sndbuf, SAL,   (char *)&h_sal, 0 );
 fbget( sndbuf, COMM,  (char *)&h_comm, 0 );
 fbget( sndbuf, DEPTNO,(char *)&h_deptno, 0 );
 fbget( sndbuf, ENAME, (char *)h_ename, 0 );
 fbget( sndbuf, JOB  , (char *)h_job, 0 );

 n = fbget( sndbuf, TINT  , (char *)&tint, 0 );
 printf("requested delay : %d\n", tint);
 n = sleep(tint);

 printf("recive data %d\n",  h_empno);
 printf("  recive data %s\n",  h_ename);
 printf("  recive data %s\n",  h_job);
 printf("  recive data %d\n",  h_mgr);
```

```
 printf("  recive data %f\n",  h_sal);
 printf("  recive data %f\n",  h_comm);
 printf("  recive data %d\n",  h_deptno);

 EXEC SQL INSERT
 INTO emp( empno, ename, job, mgr, sal,comm,deptno)
 VALUES
( :h_empno, :h_ename, :h_job, :h_mgr, :h_sal, :h_comm, :h_deptno )
;

     if ( sqlca.sqlcode != 0 ){
       printf( "insert failed sqlcode = %d\n",sqlca.sqlcode );
       tpreturn( TPFAIL, -1, NULL, 0, 0 );
 }
 printf("return\n");
 tpreturn( TPSUCCESS, 0, NULL, 0, 0 );
}
```

6.5.3 Tmax 환경파일과 makefile

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
      APPDIR  = "/home/tmax/appbin",
      PATHDIR = "/home/tmax/path",
      TLOGDIR = "/home/tmax/log/tlog",
      ULOGDIR = "/home/tmax/log/slog",
      SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax
```

```
          DBNAME   = ORACLE,
     OPENINFO =
"Oracle_XA+Acc=P/scott/tiger+SesTm=60+LogDir=/home/tmax/log/tracel
og",
     TMSNAME = tms_ora

*SERVER
fdltest       SVGNAME = svg1,
            MIN = 1, MAX = 5,
            CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
FDLINS  SVRNAME = fdltest
```

Makefile.pc

```
# Server Pro*C makefile

include $(ORACLE_HOME)/precomp/lib/env_precomp.mk
ORALIBDIR = $(LIBHOME)
ORALIB = $(PROLDLIBS)

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o
NSDLOBJ       = $(TMAXDIR)/lib/sdl.o

#Oracle
LIBS    = -lsvr -loras
#Informix
#LIBS    = -lsvr -linfs
#Db2
#LIBS    = -lsvr -ldb2s
#Sybase
#LIBS    = -lsvr -lsybs

OBJS    = $(APOBJS) $(SVCTOBJ)
```

```
SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
TMAXLIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

#
# server compile
#
all: $(TARGET)

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(TMAXLIBDIR) -o $(TARGET) -L$(ORALIBDIR)
$(ORALIB) $(OBJS) $(LIBS) $(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).pc
 proc iname=$(TARGET) include=$(TMAXDIR)
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
```

```
clean:
 -rm -f *.o core $(TARGET) $(TARGET).lis
```

6.5.4 실행

```
cfl –i sample.m
javac –classpath $JEUS_HOME/classes:
                  $JEUS_HOME/lib/system/webt30.jar:. TxTest.java
fdlc –c –i $TMAXDIR/fdl/demo.f
make –f Makefile.pc fdltest
tmboot
java $JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:. TxTest
```

6.6 JSP Application Service

JSP(Java Server Page)로 구성된 Web Application program 이다.

6.6.1 클라이언트 프로그램

Client 에서 정의된 문자열을 Server 에서 대문자로 변환하여 리턴되는 메시지를 출력하는 program 이다.

Buffer type : String

Communication mode : Synchronous communication

toupper.jsp

```
<%@ page import="java.io.*" %>
<%@ page import="java.util.*" %>
<%@ page import="tmax.webt.*" %>

<%
       WebtConnection connection = null;
       WebtConnection conn = null;
```

```
        WebtRemoteService service = null;
        WebtBuffer sndbuf = null;
        WebtBuffer rcvbuf = null;

        WebtSystem.setDefaultCharset("euc-kr");

        try {
            connection = WebtConnectionPool.getConnection("tmax1");
            service = new WebtRemoteService("STR_TOUPPER",
connection);
            sndbuf = service.createStringBuffer(2084);
            String sndstr = "Welcome to Korea.";
            sndbuf.setString(sndstr);
            rcvbuf = service.tpcall(sndbuf);
    out.println("<html><body>");
    out.println("<BIG>” + rcvbuf.getString() + "(" +
rcvbuf.getDataLength() + ")</BIG><P>");
            out.println("<BIG> [tmax1 rcvbuf data] " + rcvbuf
+"</BIG><P>");
            out.println("</body></html>");
        } catch (WebtException wie) {
            wie.printStackTrace(System.out);
            Throwable t = wie.getRootCause();
            if (t != null)
                t.printStackTrace(System.out);
            connection.close();
            System.exit(1);
        }finally {
        connection.close();
 }
%>
```

6.6.2 서버 프로그램

Server 측의 Service 루틴은 Client 로부터 읽어들인 문자열을 대문자로 변환하여 보내주는 program 이다.

test.c

```
#include <stdio.h>
#include <ctype.h>
#include <usrinc/atmi.h>

STR_TOUPPER(TPSVCINFO *msg)
{
   int i, len;
   char *instr = (char *)msg->data;

   len = strlen(instr);

   for(i=0; i<len; i++)
      instr[i] = toupper(instr[i]);

   tpreturn(TPSUCCESS, 0, (char *)instr, 0, 0);
}
```

6.6.3 Tmax 환경파일과 makefile

Container.xml

```
...

<webt-connection-pool> <!--one or zero-->
   <error-log>
       <target>file</target>
        <level>notice</level>
        <buffer-size>1024</buffer-size>
        <valid-day>7</valid-day>
        <file-name>c:\MyLog.log</file-name>
    </error-log>
    <check-alive>true</check-alive>
    <check-idle>true</check-idle>
    <webt-connection-group> <!--one or more-->
```

```
        <group-name>tmax1</group-name>
         <group-type>non-shared</group-type>
         <tmax-address>111.111.111.1</tmax-address>
         <tmax-port>8888</tmax-port>
         <max>100</max>
         <min>10</min>
         <step>5</step>
         <service-timeout>30</service-timeout>
         <transaction-timeout>60</transaction-timeout>
         <max-idle-time>120</max-idle-time>
    </webt-connection-group>
</webt-connection-pool>

...

<web-container>
```

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR  = "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax

*SERVER
test          SVGNAME = svg1
            MIN = 1, MAX = 5,
            CLOPT = " -e $(SVR).err -o $(SVR).out "
```

```
*SERVICE
STR_TOUPPER   SVRNAME = test
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
```

```
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.6.4 실행

```
cfl –i sample.m
make –f Makefile.c test
tmboot
jeus -xml –Uadministrator -Ppassward
브라우저 요청 : http://localhost:8088/examples/jsp/toupper.jsp
```

6.7 Servlet Web Application

Servlet 로 구성된 Web Application program 이다.

6.7.1 클라이언트 프로그램

Carray 와 Field data 의 Echo Service program 이다.

Buffer type : Carray, Field

Communication mode : Synchronous communication

PoolTest.java

```
import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import tmax.webt.*;

public class PoolTest extends HttpServlet {
  public void doGet(HttpServletRequest request,
HttpServletResponse response)
   throws IOException, ServletException {
 response.setContentType("text/html");
 PrintWriter out = response.getWriter();
       out.println("<html>");
 out.println("<head>");
 String title = "Request Information APIs";
 out.println("<title>" + title + "</title>");
 out.println("</head>");
 out.println("<body bgcolor=\"white\">");

     WebtConnection connection = null;
     WebtRemoteService service = null;
     WebtBuffer sndbuf = null;
     WebtBuffer rcvbuf = null;

     try {
         WebtConnectionPool.createGroup("tmax1", "192.168.128.1",
8888,4,10,2);
         connection = WebtConnectionPool.getConnection("tmax1");
         service = new WebtRemoteService("CAR_ECHO", connection);
         sndbuf = service.createCarrayBuffer(2048);
         sndstr = "Carray Type Test";
         sndbuf.setString(sndstr);
         rcvbuf = service.tpcall(sndbuf);
         service.setServiceName("FIELD_ECHO");
         sndbuf = service.createFieldBuffer(1024);
```

```
            WebtField shortField = sndbuf.createField("TSHORT");
            for (int i=0; i<5; i++)
               shortField.add(i);
            WebtField intField = sndbuf.createField("TINT");
            for (int i=0; i<5; i++)
               intField.add(i);
            WebtField longField = sndbuf.createField("TLONG");
            for (int i=0; i<5; i++)
                longField.add(i);
            WebtField floatField = sndbuf.createField("TFLOAT");
            for (int i=0; i<5; i++)
                floatField.add(i);
            WebtField doubleField = sndbuf.createField("TDOUBLE");
            for (int i=0; i<5; i++)
                doubleField.add(i);
            WebtField carrayField = sndbuf.createField("TCARRAY");
            String cdata = "CARRAY FIELD ";
            String cdata1 = "CARRAY FIELD ";
            for (int i=0; i<5; i++) {
                carrayField.add(cdata + i);
                cdata = cdata + cdata1;
            }

            WebtField stringField = sndbuf.createField("TSTRING");
            String sdata = "STRING FIELD ";
            for (int i=0; i<5; i++) {
               sdata += "TMAX";
               stringField.add(sdata + i);
            }
            rcvbuf = service.tpcall(sndbuf);
              out.println("<BIG> Field Data : " + rcvbuf.getString +
"</BIG>");
     out.println("</body>");
     out.println("</html>");
       } catch (WebtException wie) {
```

```
            wie.printStackTrace(System.out);
            Throwable t = wie.getRootCause();
            if (t != null)
               t.printStackTrace(System.out);
            connection.close();
            System.exit(1);
         }finally {
            connection.close();
         }
      }

      public void doPost(HttpServletRequest request,
                         HttpServletResponse response)
             throws IOException, ServletException
      {
         doGet(request, response);
      }
   }
```

6.7.2 서버 프로그램

Client 로부터 받은 데이터를 echo 해서 전해 준다.

test.c

```
#include <stdio.h>
#include <unistd.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include "../fdl/test_fdl.h"
#include "../fdl/demo_fdl.h"
#include <sys/timeb.h>

CAR_ECHO(TPSVCINFO *msg)
{
   int i, len;
```

```
    char *instr = (char *)msg->data;
    char *outstr;
    len = msg->len;

outstr = tpalloc("CARRAY", NULL, len);
    for(i=0; i<len; i++)
      outstr[i] = instr[i];
    tpreturn(TPSUCCESS, 0, (char *)outstr, len, 0);
}

FIELD_ECHO(TPSVCINFO *msg)
{
    int i, len, n;
    FBUF *rcvbuf = (FBUF *)msg->data;
    FBUF *sndbuf;
    char tchar;
    short tshort;
    int tint;
    long tlong;
    float tfloat;
    double tdouble;
    char tcarray[102400];
    char tstring[102400];
    int fldlen;
    struct timeb startt;
    struct timeb endt;
    time_t sec;
    int msec;
    int count = 0;
    int pos = 0;

    ftime(&startt);
    len = msg->len;

    printf("[FIELD_ECHO] datalen = %d\n", len);
```

```
    sndbuf = (FBUF *)tpalloc("FIELD", NULL, len);

     count = 0;
    pos = 0;
    while (1) {
      n = fbgetf(rcvbuf, TINT, (char *)&tint, &fldlen, &pos);
      if (n < 0)
         break;
      fbinsert(sndbuf, TINT, count, (char *)&tint, fldlen);
      count++;
     }
    printf("[FIELD_ECHO] TINT count = %d\n", count);

    count = 0;

    pos = 0;
    while (1) {
      n = fbgetf(rcvbuf, TLONG, (char *)&tlong, &fldlen, &pos);
      if (n < 0)
         break;
      fbinsert(sndbuf, TLONG, count, (char *)&tlong, fldlen);
      count++;
    }
    printf("[FIELD_ECHO] TLONG count = %d\n", count);

    count = 0;
     pos = 0;
    while (1) {
      n = fbgetf(rcvbuf, TFLOAT, (char *)&tfloat, &fldlen, &pos);
      if (n < 0)
         break;
fbinsert(sndbuf, TFLOAT, count, (char *)&tfloat, fldlen);
      count++;
    }
    printf("[FIELD_ECHO] TFLOAT count = %d\n", count);
```

```
    count = 0;
     pos = 0;
    while (1) {
        n = fbgetf(rcvbuf, TDOUBLE, (char *)&tdouble, &fldlen,
&pos);
       if (n < 0)
       break;
     fbinsert(sndbuf, TDOUBLE, count, (char *)&tdouble, fldlen);
     count++;
    }
    printf("[FIELD_ECHO] TDOUBLE count = %d\n", count);

    count = 0;
     pos = 0;
    while (1) {
     n = fbgetf(rcvbuf, TSTRING, (char *)tstring, &fldlen, &pos);
      if (n < 0)
       break;
     fbinsert(sndbuf, TSTRING, count, (char *)tstring, fldlen);
     count++;
    }
    printf("[FIELD_ECHO] TSTRING count = %d\n", count);

    ftime(&endt);
    sec = endt.time - startt.time;
    msec = endt.millitm - startt.millitm;
    if (msec < 0) {
       --sec;
       msec += 1000;
    }
    printf("[FIELD_ECHO] processing time = %d sec %d msec\n", sec,
msec);
    tpreturn(TPSUCCESS, 0, (char *)sndbuf, 0, 0);
}
```

6.7.3 Tmax 환경파일과 makefile

web.xml

```
...
<servlet>
  <servlet-name>PoolTest</servlet-name>
  <servlet-class>PoolTest</servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>PoolTest</servlet-name>
  <url-pattern>/pool</url-pattern>
   </servlet-mapping>
   ...
```

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR  = "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax

*SERVER
test          SVGNAME = svg1
            MIN = 1, MAX = 5,
            CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
```

```
CAR_ECHO      SVRNAME = test
FIELD_ECHO    SVRNAME = test
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
```

```
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.7.4 실행

```
cfl –i sample.m
make –f Makefile.c test
tmboot
jeus -xml –Uadministrator -Ppassward
브라우저 요청 : http://localhost:8088/examples/pool
```

6.8 Applet Web Application

Client 는 Applet 으로 Web Application program 이고 Tmax Server 는 UCS Service program 이다

6.8.1 클라이언트 프로그램

TextField 의 입력값을 입력하여 LOGIN Service 가 구동이 되면 TextArea 에 Tmax server 로부터 UCS Service 수행에 의한 메시지를 출력된다. Html 파일의 Applet tag 를 설정하는데 webt30.jar 를 Applet 이 로딩할수 있게 하려면 CODEBASE 와 ACRCHIVE 라는 Attribute 를 설정하고, OBJECT tag 로 설정하는 경우에는 java_codebase 와 java_archive 값을 설정해 주면 된다.

Buffer type : Carray, Field

Communication mode : Synchronous communication

index.html

```
<html>
<body bgcolor="#FFFFFF" topmargin="0" leftmargin="0"
marginwidth="0" marginheight="0" >
<OBJECT classid="clsid:8AD9C840-044E-11D1-B3E9-00805F499D93"
width="300"  height="300"  hspace="0"  vspace="0"
codebase="http://java.sun.com/products/plugin/1.2.2/jinstall-
1_2_2-win.cab#Version=1,2,2,0">
<PARAM name="java_code" value="test.TestApplet2.class">
<PARAM name="java_codebase" value="/">
<PARAM name="java_archive" value="webt30.jar,test.jar">
<PARAM name="type" value="application/x-java-applet;">
<PARAM name="authCode" value="Basic
YWRtaW5pc3RyYXRvcjpqZXVzYWRtaW4=">
</body>
</html>
```

TestApplet2.java

```
package test;

import java.awt.*;
import java.awt.event.*;
import java.applet.*;
import tmax.webt.*;
import java.awt.*;
import java.awt.event.*;

public class TestApplet2 extends Applet implements
WebtEventHandler {
 private WebtEventConnection connection;
 private WebtRemoteService service;

 WebtConnection client;
```

```
TextArea textArea;
Label inputText;
TextField input;
Button toUpper,prt;
String result;
WebtBuffer stringBuffer;
WebtBuffer rxBuffer;

   public void handleEvent(int type, WebtBuffer buf, int len, int
flags) {
   textArea.append("event received. type = " + type + ", buffer
type = " + buf.getBufferType() + "\n");
   }

   public void handleError(WebtException e) {
      System.err.println(" handleError!! ");
   }

   public void init() {
this.connection = new WebtEventConnection("61.77.153.1", 8888);
connection.setEventMask(WebtEventConnection.EVENT_SENDTOCLI);
inputText = new Label ("board.....");
textArea = new TextArea();
input = new TextField("tmax", 20);
toUpper = new Button ( "Login");
prt = new Button ( "Print");
add(inputText);
add(textArea);
add(input);
add(toUpper);
add(prt);
toUpper.addActionListener(new ActionListener() {
      public void actionPerformed(ActionEvent e) {
            print_actionPerformed(e);
      }
```

```
});
   }

   public void paint (Graphics g){
if (result != null)
    g.drawString ( "Converted text : " + result, 20, 90);
   }

   public void start() {
connection.setEventHandler(this);
   }

   public void print_actionPerformed(ActionEvent ae) {
String temp = input.getText();
if (temp != null && !temp.equals("")) {
    service = new WebtRemoteService("LOGIN", connection);
    stringBuffer = service.createStringBuffer();
    stringBuffer.setString(temp);
    rxBuffer = service.tpcall(stringBuffer);
    input.setText(rxBuffer.getString());
}
rxBuffer = null;
   }

   public void destroy() {
client.close();
   }
}
```

6.8.2 서버 프로그램

LOGIN Service 가 구동이 되면 UCS Service 가 수행된다..

ucsTest.c

```
#include   <stdio.h>
#include   <usrinc/atmi.h>
```

```
#include    <usrinc/tmaxapi.h>
#include    <usrinc/ucs.h>

#define     MAX_CLI     10

int         num_cli           ;
int         count             ;
int         client_id[MAX_CLI]  ;

int tpsvrinit(int argc, char *argv[])
{
    num_cli = 0;
    count   = 0;
}

int usermain(int argc, char *argv[])
{
    int     jobs, i, ret;
    char    *sndbuf;
    int     count=0;

    printf("UCS Service Start!!!\n");

    sndbuf = (char *)tpalloc("CARRAY", NULL, 1024);
    sprintf(sndbuf, "Hello? Can you read me?\n");
    for (i = 0; i < MAX_CLI; i++)
        client_id[i] = -1;

    while(1)
    {
        printf("number of clients is %d\n", num_cli);
        for(i = 0; i < MAX_CLI; i++)
        {
     if (tpchkclid(client_id[i]) >= 0) {
         printf("sending tpsendtocli to clii(%d)\n", client_id[i]);
```

```
            sprintf(sndbuf,"Success tpsendtocli [%d] Tmax UCS
Test\n", count);
            ret = tpsendtocli(client_id[i], (char *)sndbuf,
strlen(sndbuf), 0);
            if (ret < 0) {
          printf("Error!!\n");
            }
    } else {
       client_id[i] = -1;
    }
          /*tpsendtocli(client_id[i], (char *)sndbuf, 1024, 0);*/
       }
       count++;
       jobs = tpschedule(1);
   }
}

LOGIN(TPSVCINFO *msg)
{
   char       *sndbuf;
   int        clid, ret, i;

   printf("msg -> data = [%.*s]\n", msg->len, msg->data);
   fflush(stdout);

   sndbuf = (char *)tpalloc("CARRAY", NULL, 1024);

   printf("Success Transaction\n");
   sprintf(sndbuf,"Success Transaction");

   for (i = 0; i < MAX_CLI; i++)
   {
 if (client_id[i] == -1) {
          client_id[i] = tpgetclid();
          printf("clid[%d] = [%d]\n", i, client_id[i]);
```

```
    break;
 }
   }
   printf("Login end!!\n");
   tpreturn(TPSUCCESS, 0, (char *)sndbuf, strlen(sndbuf),0);
}
```

6.8.3 Tmax 환경파일과 makefile

sample.m

```
*DOMAIN
msyoon1   SHMKEY = 79000, TPORTNO = 8888, MINCLH = 1, MAXCLH = 1,

*NODE
msyoon  TMAXDIR="/home/choco/tmax",
              APPDIR="/home/choco/tmax/appbin",

*SVRGROUP
svg1          NODENAME = msyoon

*SERVER
ucsTest       SVGNAME = svg1, SVRTYPE = UCS, CPC = 1

*SERVICE
LOGIN         SVRNAME = ucsTest
```

Makefile.c

```
# Server makefile
#CC = /usr/local/bin/gcc
#CC = cc
TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o
SDLFILE = demo.s

# Using Shared Library
```

```
LIBS    = -lsvrucs -lpthread -lm -lnodb
OBJS    = $(APOBJS) $(SDLOBJ) $(SVCTOBJ)
SDLOBJ  = $(TMAXDIR)/lib/sdl.o
SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -D_DEBUG -I$(TMAXDIR) -I$(TMAXDIR)/fdl

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile
$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(APOBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

$(SDLOBJ):
 $(TMAXDIR)/bin/sdlc -i ../sdl/$(SDLFILE)
 $(CC) $(CFLAGS) -c ../sdl/$(SDLC)
#
clean:
```

```
-rm -f *.o core $(TARGET)
```

6.8.4 실행

```
cfl -i sample.m
make -f Makefile.c ucsTest
tmboot
javac -classpath $JEUS_HOME/classes:. TestApplet2.java
브라우저 요청 : http://localhost:8088/examples/index.html
```

6.9 EJB Application Service

EJB 와 Tmax 간의 JeusGW 를 통해서 XA Service 를 제공할 수 있다.

6.9.1 클라이언트 프로그램

EJB Bean 에서 읽어들인 데이터를 Tmax Server 의 insert program 을 호출하여 DB 에 10 개의 data 를 insert 하는 program 이다.

Buffer type : Field

Communication mode : Synchronous communication

EchoTestEJB.java

```
package echo;
import java.io.*;
import java.util.*;
import java.lang.*;
import javax.ejb.*;
import javax.naming.*;
import java.rmi.*;
import tmax.webt.*;
import tmax.webt.io.*;
```

```
import tmax.webt.jeus.*;

public class EchoTestEJB implements SessionBean {

 private WebtConnection conn  = null;
 private WebtRemoteService service = null;
 private Context ctx = null;
 private WebtDataSource wds = null;

 public void test() throws RemoteException {
   conn = wds.getConnection();
        WebtAttribute attribute = new WebtAttribute();
         WebtRemoteService service = new
WebtRemoteService("FDLINS", conn);

  // first call
  WebtFieldSet sndset  = new
WebtFieldSet(service.createFieldBuffer());

 sndset.add("EMPNO", 1234);
 sndset.add("ENAME", "WebT");
 sndset.add("JOB", "Tmax");
 sndset.add("MGR", 1010);
 sndset.add("SAL", 10000);
 sndset.add("COMM", 10);
 sndset.add("DEPTNO", 12);

 try {
         System.out.println("tpacall first!!!");
    WebtBuffer rcvbuf = service.tpcall(sndset.getFieldBuffer(),
attribute);
 } catch (WebtException e) {
          System.out.println("tpcall error!!!");
    e.printStackTrace();
 }
```

```
      WebtFieldSet sndset1  = new
WebtFieldSet(service.createFieldBuffer());

 sndset1.add("EMPNO", 4321);
 sndset1.add("ENAME", "Jeus");
 sndset1.add("JOB", "Tmax");
 sndset1.add("MGR", 1110);
 sndset1.add("SAL", 20000);
 sndset1.add("COMM", 20);
 sndset1.add("DEPTNO", 24);

 try {
          System.out.println("tpacall second!!!");
     WebtBuffer rcvbuf1 =
service.tpacall(sndset1.getFieldBuffer(), attribute);
 } catch (WebtException ex) {
          System.out.println("tpcall error!!!");
    ex.printStackTrace();
 }finally{
    conn.close();
 }
}

 public EchoTestEJB() {}
 public void ejbCreate() {
     try {
           ctx = new InitialContext();
           wds = (WebtDataSource)ctx.lookup("webt1");
           System.out.println("Succeed in connectiong tmax");
     }catch(Exception ex) {
           ex.printStackTrace();
     }
 }
```

```
 public void ejbRemove() {
 }

 public void ejbActivate() {}
 public void ejbPassivate() {}
 public void setSessionContext(SessionContext ctx) {}
}
```

EchoTest.java

```
package echo;

import javax.ejb.EJBObject;
import java.rmi.RemoteException;
import tmax.webt.*;
import echo.*;

public interface EchoTest extends EJBObject {
 public void test() throws RemoteException;
}
```

EchoTestHome.java

```
package echo;

import javax.ejb.EJBHome;
import java.rmi.RemoteException;
import javax.ejb.CreateException;
import echo.*;

public interface EchoTestHome extends EJBHome {
 public EchoTest create() throws CreateException, RemoteException;
}
```

EchoClient.java

```
package echo;

import javax.naming.*;
import java.rmi.*;
import javax.rmi.*;
import java.util.*;
import tmax.webt.*;
import tmax.webt.io.*;
import echo.*;

public class EchoClient {

 public static void main(String[] args) {
        try {
                Hashtable ht = new Hashtable();
                ht.put(Context.PROVIDER_URL, "61.77.153.1");

 ht.put(Context.INITIAL_CONTEXT_FACTORY,"jeus.jndi.JEUSContextFact
ory");
                Context initial = new InitialContext(ht);
                Object objref = initial.lookup("echo.EchoTest");
                EchoTestHome home =
(EchoTestHome)PortableRemoteObject.narrow(objref,
EchoTestHome.class);
                EchoTest echo = home.create();

                echo.test();
        } catch (Exception e) {
                e.printStackTrace();
        }
 }
}
```

6.9.2 서버 프로그램

Server 측의 Service 루틴은 Client 로 읽어 들인 Field data 를 해당 DB Column 에 insert 하는 program 이다.

fdlTest.pc

```
#include <stdio.h>
#include <ctype.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include "../fdl/demo_fdl.h"

EXEC SQL include sqlca.h;

EXEC SQL begin declare section;
 int  h_empno;
 char h_ename[11];
 char h_job[10];
 int  h_mgr;
 float h_sal;
 float h_comm;
 int  h_deptno;
EXEC SQL end declare section;

FDLINS( TPSVCINFO *msg )
{
 FBUF *sndbuf;
 int n;
 int tint=0;

 sndbuf = (FBUF *)msg->data;

 h_empno = h_mgr = h_sal = h_comm = h_deptno = 0;

 memset( h_ename, 0x00, sizeof( h_ename ) );
```

```
memset( h_job, 0x00, sizeof( h_job ) );

fbget( sndbuf, EMPNO, (char *)&h_empno, 0 );
fbget( sndbuf, MGR,   (char *)&h_mgr, 0 );
fbget( sndbuf, SAL,   (char *)&h_sal, 0 );
fbget( sndbuf, COMM,  (char *)&h_comm, 0 );
fbget( sndbuf, DEPTNO,(char *)&h_deptno, 0 );
fbget( sndbuf, ENAME, (char *)h_ename, 0 );
fbget( sndbuf, JOB  , (char *)h_job, 0 );

n = fbget( sndbuf, TINT  , (char *)&tint, 0 );
printf("requested delay : %d\n", tint);
n = sleep(tint);

printf("recive data %d\n",  h_empno);
printf("  recive data %s\n",  h_ename);
printf("  recive data %s\n",  h_job);
printf("  recive data %d\n",  h_mgr);
printf("  recive data %f\n",  h_sal);
printf("  recive data %f\n",  h_comm);
printf("  recive data %d\n",  h_deptno);

EXEC SQL INSERT
INTO emp( empno, ename, job, mgr, sal,comm,deptno)
VALUES
( :h_empno, :h_ename, :h_job, :h_mgr, :h_sal, :h_comm, :h_deptno )
;

    if ( sqlca.sqlcode != 0 ){
      printf( "insert failed sqlcode = %d\n",sqlca.sqlcode );
      tpreturn( TPFAIL, -1, NULL, 0, 0 );
}
printf("return\n");
tpreturn( TPSUCCESS, 0, NULL, 0, 0 );
}
```

6.9.3 Tmax 환경파일과 makefile

JeusMain.xml

```
<jeus-system>
...
<external-source>
      <tmax>
        <export-name>webt1</export-name>
   <host-name>61.77.153.1</host-name>
   <port>6734</port>
   <tmax-connection-pool>
         <pooling>
           <min>0</min>
         <max>10</max>
           <step>2</step>
         <period>60000</period>
      </pooling>
          <wail-free-connection>
              <wait-time>10000</wait-time>
              </wail-free-connection>
   </tmax-connection-pool>
      </tmax>
   </external-source>
...
</jeus-system>
```

EJBMain.xml

```
...
<ejb-engine>
   ...
<startup-class>
  <startup-method>
    tmax.jtmax.deployment.WebtContainer::startup(java.lang.String,
java.lang.String)
  </startup-method>
```

```
  <startup-parameter>yskim1,engine1</startup-parameter>
</startup-class>
<stutdown-class>
  <stutdown-method>
  tmax.jtmax.deployment.WebtContainer::shutdown(java.lang.String)
  </stutdown-method>
  <stutdown-parameter>dummy</stutdown-parameter>
</stutdown-class>
<module-list>echotest</module-list>
</ejb-engine>
```

ejb-jar_echotest.xml

```
...
<ejb-jar>
   <enterprise-beans>
      <session>
         <ejb-name>echotest</ejb-name>
         <home>echo.EchoTestHome</home>
         <remote>echo.EchoTest</remote>
         <ejb-class>echo.EchoTestEJB</ejb-class>
         <session-type>Stateless</session-type>
         <transaction-type>Container</transaction-type>
      </session>
   </enterprise-beans>
   <assembly-descriptor>
      <container-transaction>
         <method>
            <ejb-name>echotest</ejb-name>
            <method-name>*</method-name>
            <method-params/>
         </method>
         <trans-attribute>Required</trans-attribute>
      </container-transaction>
   </assembly-descriptor>
</ejb-jar>
```

jeus-ejb-dd_echotest.xml

```
...
<jeus-ejb-dd>
    <module-info>
       <module-name>echotest</module-name>
    </module-info>
    <beanlist>
       <stateless>
          <ejb-name>echotest</ejb-name>
     <export-port>0</export-port>
          <export-name>echo.EchoTest</export-name>
     <export-iiop>false</export-iiop>
     <local-invoke-optimize>false</local-invoke-optimize>
         </stateless>
    </beanlist>
</jeus-ejb-dd>
```

jtmax_Servlet_engine1/jtmaxMain.xml

```
<jtmax-container>
  <logging>
     <error-log>
     <target>console</target>
     <level>fatal</level>
     <buffer-size>1024</buffer-size>
     <valid-day>1</valid-day>
     </error-log>
     <user-log>
     <target>console</target>
     <level>fatal</level>
     <valid-day>1</valid-day>
     </user-log>
  </logging>
```

```
 <context>
  <context-name>MyGroup</context-name>
   <docbase>webapps</docbase>
   <connection>
     <conntection-type>webtserver</connection-tyoe>
     <connection-id>webt1</connection-id>
     <address>127.0.0.1</address>
     <port>8888</port>
     <backlog>50</backlog>
   </connection>
 </context>
</jtmax-container>
```

sample.m

```
*DOMAIN
tmax          SHMKEY = 77990, MAXUSER = 256

*NODE
tmax          TMAXDIR = "/home/tmax",
            APPDIR  = "/home/tmax/appbin",
            PATHDIR = "/home/tmax/path",
            TLOGDIR = "/home/tmax/log/tlog",
            ULOGDIR = "/home/tmax/log/slog",
            SLOGDIR = "/home/tmax/log/ulog"

*SVRGROUP
svg1          NODENAME = tmax
          DBNAME = ORACLE,
      OPENINFO =
"Oracle_XA+Acc=P/scott/tiger+SesTm=60+LogDir=/home/tmax/log/tracel
og",
      TMSNAME = tms_ora

*SERVER
fdltest       SVGNAME = svg1,
```

```
MIN = 1, MAX = 5,
CLOPT = " -e $(SVR).err -o $(SVR).out "

*SERVICE
FDLINS  SVRNAME = fdltest

*GATEWAY
JeusGW          GWTYPE         = JEUS,
                PORTNO         = 6734,
               RGWADDR        = "61.77.153.1",
               RGWPORTNO      = 9555,
               NODENAME       = yskim1,
               CLOPT          = "-f jconfig"
```

Makefile.pc

```
# Server Pro*C makefile

include $(ORACLE_HOME)/precomp/lib/env_precomp.mk
ORALIBDIR = $(LIBHOME)
ORALIB = $(PROLDLIBS)

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

#Oracle
LIBS    = -lsvr -loras
#Informix
#LIBS   = -lsvr -linfs
#Db2
#LIBS   = -lsvr -ldb2s
#Sybase
#LIBS   = -lsvr -lsybs

OBJS    = $(APOBJS) $(SVCTOBJ)
```

```
SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
TMAXLIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

#
# server compile
#
all: $(TARGET)

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(TMAXLIBDIR) -o $(TARGET) -L$(ORALIBDIR)
$(ORALIB) $(OBJS) $(LIBS) $(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).pc
 proc iname=$(TARGET) include=$(TMAXDIR)
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
```

```
clean:
 -rm -f *.o core $(TARGET) $(TARGET).lis
```

jtmax.f

```
EJB_CLASS_NAME   100  string  -  -
EJB_METHOD_NAME   101  string  -  -
EJB_RESULT_BYTE   102  char  -  -
EJB_RESULT_SHORT  103  short  -  -
EJB_RESULT_INT   104  int  -  -
EJB_RESULT_LONG   105  long  -  -
EJB_RESULT_FLOAT  106  float  -  -
EJB_RESULT_DOUBLE  107  double  -  -
EJB_RESULT_STRING  108  string  -  -
EJB_RESULT_CARRAY  109  carray  -  -
ORG_TYPE   110  int  -  -
ORG_DATA_STRING   111  string  -  -
ORG_DATA_CARRAY   112  carray  -  -
```

6.9.4 실행

```
javac -classpath
$JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:. -d $EJB_HOME
EchoTestEJB.java EchoTestHome.java EchoTest.java
jeus -xml -Uadministrator -P1111111
cfl -i sample.m
fdlc -c -i $JEUS_HOME/lib/system/jtmax.f
fdlc -c -I $TMAXDIR/fdl/demo.f
make -f Makefile.pc fdltest
tmboot
ejbadmin nodename
deploy echotest
java -classpath
$JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:.
-Djava.naming.factory.initial=jeus.jndi.JEUSContextFactory
-Djava.naming.factory.url.pkgs=jeus.jndi.jns.url
-Djava.security.policy=policy echo.EchoClient
```

```
undeploy echotest
```

6.10 RQ Service

RQ Service 은 신뢰성이 보장되어야 하는 안정적인 서비스를 보장하기 위해서 client 의 요청에 대하여 디스크 관리를 하는 서비스 이다.

6.10.1 클라이언트 프로그램

Client 는 string echo service 를 호출하며 이때 요청에 대한 사항을 RQ 에 기록한다.

Buffer type : STRING

Communication mode : Synchronous communication, RQ Service

RQTest.java

```
import tmax.webt.*;

public class RQTest {
   public static void main(String[] argv) {
     WebtConnection connection = new WebtConnection("61.77.153.1",
8888);
WebtRQService rqs = new WebtRQService("RQTEST", " STR_TOUPPERSTR",
connection);

 WebtBuffer sndbuf = rqs.createStringBuffer();
 WebtBuffer rcvbuf = rqs.createStringBuffer();
 WebtAttribute attr = new WebtAttribute();
 attr.setTPRQS(true);

 try {
     sndbuf.setString("RQTest start!!!");
          int req = rqs.tpenq(sndbuf, attr);
```

```
        System.out.println("[tpenq run] : " + req);

        Thread.sleep(2000);

        int rqcount =
rqs.tpqstat("RQTEST",WebtRQService.RQ_RPLY_QUEUE);
              System.out.println("[tpqstat] : " + rqcount);
        for(int i=0; i <rqcount; i++) {
            rcvbuf = rqs.tpdeq(attr);
            System.out.println("[tpdeq] : " + rcvbuf.getString());
        }

        req = rqs.checkError(rcvbuf);
        System.out.println("[checkError] : " + req);

            System.out.println("[RQ SERVICE END]");
          } catch (WebtException e) {
        //System.out.println("[User Return Code] : " +
(e.getRxBuffer()).getUserReturnCode());
             e.printStackTrace();
    } catch (Exception ex) {
       System.out.println("[ERROR]");
       ex.printStackTrace();
          } finally {
              connection.close();
          }
       }
}
```

6.10.2 서버 프로그램

Client 로부터 받은 문자열을 대문자로 변환하여 return 하는 program 이다.

test.c

```
#include <stdio.h>
#include <usrinc/atmi.h>

STR_TOUPPERSTR(TPSVCINFO *msg)
{
   int i, len;
   char *instr = (char *)msg->data;

   len = strlen(instr);

   for(i=0; i<len; i++)
      instr[i] = toupper(instr[i]);

   tpreturn(TPSUCCESS, 0, (char *)instr, 0, 0);
}
```

6.10.3 Tmax 환경파일과 makefile

sample.m

```
*DOMAIN
res   SHMKEY=77991,   MAXUSER=500,    MINCLH=1,
      MAXCLH=1,      TPORTNO=8888

*NODE
yskim1    TMAXDIR="/home/choco/tmax",
         APPDIR ="/home/choco/tmax/appbin",
         PATHDIR="/home/choco/tmax/path",
         TLOGDIR="/home/choco/tmax/log/tlog",
         ULOGDIR="/home/choco/tmax/log/ulog",
         SLOGDIR="/home/choco/tmax/log/slog"

*SVRGROUP
svg1          NODENAME=yskim1, SVGTYPE=RQMGR
```

```
*RQ
RQTEST  SVGNAME=svg1

*SERVER
test           SVGNAME=svg1, MIN=1

*SERVICE
STR_TOUPPER          SVRNAME=test
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o

#Not use Db
LIBS     = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)
NSDLOBJ = $(TMAXDIR)/lib/sdl.o

SVCTOBJ = $(TARGET)_svctab.o
CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
 $(CC) $(CFLAGS) -c $<

# server compile
```

```
$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c

$(SVCTOBJ):
 touch $(SVCTDIR)/$(TARGET)_svctab.c
 $(CC) $(CFLAGS) -c $(SVCTDIR)/$(TARGET)_svctab.c

#
clean:
 -rm -f *.o core $(TARGET)
```

6.10.4 실행

```
cfl -i sample.m
javac -classpath
$JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:. RQTest.java
make -f Makefile.c test
tmboot
java $JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:. RQTest
```

6.11 Asynchronous Event Service

Asynchronous Event Service

6.11.1 클라이언트 프로그램

Client 는 string echo service 를 호출하며 이때 요청에 대한 사항을 출력한다.

Buffer type : STRING

Communication mode : Asynchronous communication

GenericEvent.java

```
package tcpservlet.event;

import tmax.webt.*;

public class GenericEvent implements WebtEventHandler {
   private WebtEventConnection conn;

   public GenericEvent() {}

   public void handleEvent(int type, WebtBuffer buf, int len, int
flags) {
      this.conn = (WebtEventConnection)buf.getUsedConnection();
      System.out.println("event received. type = " + type + ",
buffer type = " + buf.getBufferType());
      System.out.println("event msg : " + buf.getString());
   }

   public void handleError(WebtException e) {
      System.err.println("Event error occured");
      e.printStackTrace();
      if( e instanceof WebtIOException) {
         if(e.getTPError() == WebtException.TPECLOSE) {
      conn.close();
      conn.connect();
```

```
    }
  }
   }
}
```

EventAcallTest.java

```
package event;

import java.io.*;
import java.util.*;
import javax.servlet.*;
import javax.servlet.http.*;
import tmax.webt.*;

public class EventSendtoTest extends HttpServlet {
   public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
       throws IOException, ServletException {
       response.setContentType("text/html");

       PrintWriter out = response.getWriter();
       out.println("<html>");
       out.println("<head>");

       String title = "Request Information APIs";
       out.println("<title>" + title + "</title>");
       out.println("</head>");
       out.println("<body bgcolor=\"white\">");
       out.println("<h1>" + title + "</h1>");
       out.println("<h3>Request Informations</h3>");
         WebtEventConnection conn =
(WebtEventConnection)WebtConnectionPool.getConnection("tmax1");
  WebtRemoteService service = new WebtRemoteService("ECHO",
conn);
  WebtBuffer sndbuf = service.createStringBuffer();
```

```
    sndbuf.setString("EVENT_ACALLREPLY Flag Test");

    int cd = 0;
    try {
        cd = service.tpacall(sndbuf);
        try {
            Thread.sleep(10000);
        }catch(Exception e) {
            e.printStackTrace();
        }
    }catch(WebtException e) {
       e.printStackTrace();
       throw new WebtException("exception to TPACALL");
    }

            out.println("<h3> tpacall successfully!!!! </h3>");
         out.println("</body>");
         out.println("</html>");
     }

     public void doPost(HttpServletRequest request,
                    HttpServletResponse response)
        throws IOException, ServletException
     {
        doGet(request, response);
     }
}
```

6.11.2 서버 프로그램

Client 로부터 받은 문자열을 다시 return 하는 program 이다.

test.c

```
#include <stdio.h>
```

```
#include <usrinc/atmi.h>

ECHO(TPSVCINFO *msg)
{
 printf("client message : %s\n " , msg->data);
 tpreturn(TPSUCCESS, 0 , msg->data,0,0);
}
```

6.11.3 Tmax 환경파일과 makefile

web.xml

```
...
<servlet>
  <servlet-name>EventAcallTest</servlet-name>
  <servlet-class>EventAcallTest</servlet-class>
  </servlet>
  <servlet-mapping>
  <servlet-name>EventAcallTest</servlet-name>
  <url-pattern>/acall</url-pattern>
   </servlet-mapping>
   ...
```

sample.m

```
*DOMAIN
res   SHMKEY=77991,   MAXUSER=500,    MINCLH=1,
      MAXCLH=1,      TPORTNO=8888

*NODE
yskim1    TMAXDIR="/home/choco/tmax",
         APPDIR ="/home/choco/tmax/appbin",
         PATHDIR="/home/choco/tmax/path",
         TLOGDIR="/home/choco/tmax/log/tlog",
         ULOGDIR="/home/choco/tmax/log/ulog",
```

```
        SLOGDIR="/home/choco/tmax/log/slog"

*SVRGROUP
svg1          NODENAME=yskim1

*SERVER
event         SVGNAME=svg1, MIN=1

*SERVICE
ECHO          SVRNAME=event
```

Makefile.c

```
# Server makefile

TARGET  = $(COMP_TARGET)
APOBJS  = $(TARGET).o
NSDLOBJ= $(TMAXDIR)/lib/sdl.o

#Not use Db
LIBS    = -lsvr -lnodb

OBJS    = $(APOBJS) $(SVCTOBJ)

SVCTOBJ = $(TARGET)_svctab.o

CFLAGS  = -O -I$(TMAXDIR)

APPDIR  = $(TMAXDIR)/appbin
SVCTDIR = $(TMAXDIR)/svct
LIBDIR  = $(TMAXDIR)/lib

#
.SUFFIXES : .c

.c.o:
```

```
 $(CC) $(CFLAGS) -c $<

#
# server compile
#

$(TARGET): $(OBJS)
 $(CC) $(CFLAGS) -L$(LIBDIR) -o $(TARGET) $(OBJS) $(LIBS)
$(NSDLOBJ)
 mv $(TARGET) $(APPDIR)/.
 rm -f $(OBJS)

$(APOBJS): $(TARGET).c
 $(CC) $(CFLAGS) -c $(TARGET).c
```

6.11.4 실행

```
cfl –i sample.m
javac –classpath
$JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:.
GenericEventTest.java
GenericEventTest.class 를 classpath 경로에 위치해 놓는다.
$JEUS_HOME/classes:$JEUS_HOME/lib/system/webt30.jar:.
EventAcallTest.java
make –f Makefile.c event
tmboot
jeus -xml –Uadministrator -Ppassward
브라우저 요청 : http://localhost:8088/examples/acall
```

7 The WebT-Server System

7.1 소개

WebT-Server System 은 TmaxSoft 의 두가지 제품군인 Tmax 와 JEUS 를 연결하는 게이트 웨이의 하나로써 Java 클라이언트에서 Tmax 를 호출하는 WebT 에 대응하여 Tmax 에서 JEUS 의 서비스(EJB)를 호출, 결과값을 가져오는 것을 가능케 한다.

WebT-Server System 을 사용하여 Tmax 와 JEUS 간에 XA Protocol 을 이용한 1PC 혹은 2PC 의 트랜잭션 처리가 가능하며 서로 다른 언어 환경/플랫폼의 차이로 인한 데이터 구조 차이에 대한 추가적인 고려 없이도 기본적인 Primitive Type 뿐 아니라 String, Byte Array, FDL 버퍼 등을 사용할 수 있다.

WebT-Server System.은 Tmax 에서 오는 데이터를 받아 트랜잭션 처리와 서비스 수행 과정을 관리하는 jTmax 모듈 외에 표준 버퍼의 사용과 네트웍 통신을 위해 내부적으로 WebT 라이브러리를 사용하며 실제 직접적인 Tmax 와의 통신은 Tmax 의 JeusGW 를 통해 이루어진다.

7.2 WebT-Server System 개요

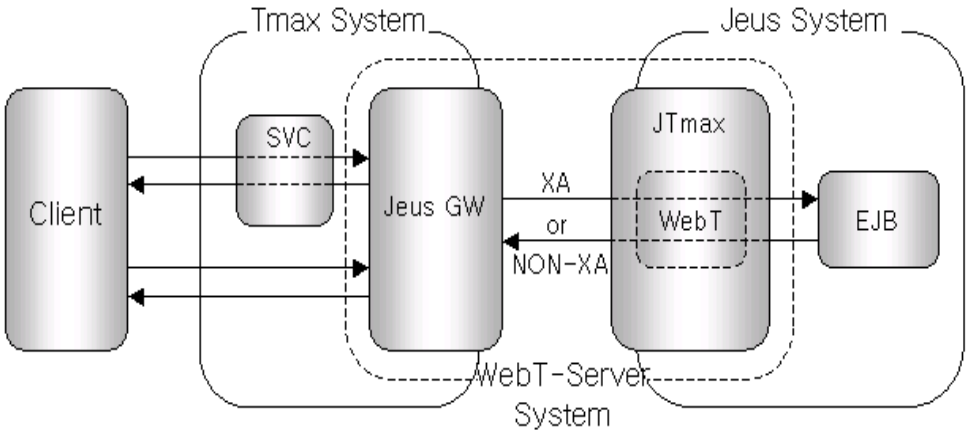

그림 3 WebT-Server System 아키텍처

WebT-Server System 은 Tmax 와 JEUS 시스템 양쪽에 걸쳐 운영되며 크게 세부분으로 나눌 수 있다.

- Tmax 에서 JEUS 로의 서비스 요청 처리를 위한 JeusGW

- JEUS 에서 Tmax 서비스 요청을 수신하기 위한 데몬형태의 jTmax

- Tmax 와 JEUS 가 주고 받는 데이터를 처리하는 WebT Library

WebT-Server System 은 JEUS 가 Tmax 로 부터의 접속을 기다리는 구조 이므로 반드시 Tmax 에 앞서 JEUS 를 먼저 기동해야 한다.

7.3 WebT-Server 시스템 설정

WebT-Server System 을 가동하기 위해서는 JEUS 시스템에서는 EJB 사용을 위한 설정이, Tmax 시스템에서는 클라이언트에서 서비스호출과 관련된 설정이 이루어져야 된다.

먼저 JEUS 시스템에서의 설정을 알아보자.

7.3.1 클래스 패스 설정

JEUS 구동시 클래스 패스에 webt30.jar 와 jtmax.jar 가 포함되어 있는지 확인을 해라.

webt30.jar 는 네트워크를 통한 데이터 전송을 위한 Webt 모듈을 위한 것이고, jtmax.jar 는 Webt 모듈에서 넘어온 데이터를 수신/분석해서 해당하는 EJB 메소드를 호출하는 모듈이다.

일반적으로 두 모듈은 $JEUS_HOME/lib/system 내에 존재한다.

7.3.2 WebT-Server Binary File 생성

JEUS-Tmax 간의 데이터 전송을 위해서 FieldKey 파일을 사용한다. 이는 FDL(Field Definition Language)에 의해서 Field Key Buffer 를 사용하여 원하는 정보의 데이터만을 조작하여 처리하게 해준다. 그래서 WebT-System 에서는 데이터 타입에 구애받지 않고 JEUS-Tmax 간에 데이터를 주고 받을 수 있다.

필드키를 사용하기 위해서는

1. 각각의 필드키에 대하여 이름, 번호, 타입, 플래그(flag), 주석(comment)를 필드버퍼파일(.f)에 기술한다.

2. FDLC 를 이용하여 필드버퍼(FieldBuffer)파일을 컴파일하여 xxx.fdl 및 xxx_fdl.h 파일을 생성한다.

   필드버퍼파일을 tmax.f 라고 하였을 경우 다음과 같이 수행하면 매핑파일이 생성된다.

   fdlc –c –i tmax.f -> tmax.fdl 및 tmax_fdl.h

3. xxx.fdl 을 $JEUS_HOME/lib/system 에 복사한다. 그러면 jTmax 모듈이 이 파일을 읽어서 Tmax 로부터 데이터 수신을 하기위한 준비를 하게 된다. 그리고 xxx_fdl.h 헤더 파일은 Tmax 클라이언트 프로그래밍 시 사용될 것이다.

WebT-Server 시스템에서 사용하는 FiledKey 파일은 아래의 예에 제시된 13 개의 필드를 정의해야 한다. 해당 필드의 이름과 타입은 아래와 동일하여야 하나 필드 번호는 아래와 달라도 상관없다.

name	number	type	flag	comment
EJB_CLASS_NAME	100	string	-	-
EJB_METHOD_NAME	101	string	-	-
EJB_RESULT_BYTE	102	char	-	-
EJB_RESULT_SHORT	103	short	-	-
EJB_RESULT_INT	104	int	-	-
EJB_RESULT_LONG	105	long	-	-
EJB_RESULT_FLOAT	106	float	-	-
EJB_RESULT_DOUBLE	107	double	-	-
EJB_RESULT_STRING	108	string	-	-
EJB_RESULT_CARRAY	109	carray	-	-
ORG_TYPE	110	int	-	-
ORG_DATA_STRING	111	string	-	-
ORG_DATA_CARRAY	112	carray	-	-

fdlc 에 대한 자세한 내용은 Tmax FDL Reference Guide 를 참조하도록 한다.

7.3.3 Engine Container 설정

이제 우리는 JEUS 시스템 내에서 jTmax 모듈을 구동시켜야 된다. JEUS 부팅 시에 특정 클래스의 메소드를 호출하고 싶을 때 JEUS 4.x 에서는 각 엔진의 EJBMain.xml 의 startup-class 요소(element)를 정의하였는데, JEUS 5.0 부터는 JEUSMain.xml 의 lifecycle-invocation 요소를 사용한다.

따라서 JEUS 시스템 부팅 시 jTmax 모듈을 구동시키고 정지시키기 위해서는 아래와 같이 JEUSMain.xml 에 추가되어야 한다.

```
<engine-container>
      ……
  <lifecycle-invocation>
     <class-name>tmax.jtmax.deployment.WebtContainer</class-name>
        <invocation>
           <invocation-method>
              <method-name>startup</method-name>
              <method-params>
                 <method-param>java.lang.String</method-param>
                 <method-param>java.lang.String</method-param>
              </method-params>
           </invocation-method>
           <invocation-argument>node_name</invocation-argument>
           <invocation-argument>engine_name</invocation-argument>
           <invocation-type> BEFORE_DEPLOY</invocation-type>
        </invocation>
   </lifecycle-invocation>

   <lifecycle-invocation>
     <class-name>tmax.jtmax.deployment.WebtContainer</class-name>
        <invocation>
           <invocation-method>
              <method-name>shutdown</method-name>
              <method-params>
                 <method-param>java.lang.String</method-param>
              </method-params>
           </invocation-method>
           <invocation-argument>dummy</invocation-argument>
           <invocation-type>AFTER_UNDEPLOY</invocation-type>
        </invocation>
    </lifecycle-invocation>
        ……
</engine-container>
```

jTmax 모듈을 위해서 JEUSMain.xml 에 추가된 내용은 모듈을 구동시키는 부분과 정지시키는 부분으로 나누어져 있다.

- class-name : 구동시킬 메소드를 가지고 있는 클래스 이름으로 jTmax 모듈은 tmax.jtmax.deployment.WebtContainer 이다.
- method-name : 호출할 메소드의 이름이다. 구동시킬 때는 startup 이고 정지시킬 때는 shutdown 이다.
- method-param : 호출하는 메소드 인자의 타입을 설정하는데 java.lang.String 과 같이 full-qualified name 을 사용해야된다.
- invocation-argument : 실제로 인자로 사용될 값을 설정한다. jTmax 모듈의 startup 메소드는 엔진이름과 노드이름이 필요하며, shutdown 메소드는 dummy 문자열이 필요하다.
- invocation-type : startup 메소드는 디플로이 하기 전에 호출되고, shutdown 메소드는 undeploy 하고 나서 호출함을 의미한다.

7.3.4 jTmax 환경 파일 설정

jTmax 는 기동 시 jtmaxMain.xml 파일을 읽어 환경을 설정한다. 파일의 위치는
$JEUS_HOME/config/*NodeName*/*NodeName*_jtmax_*jTmaxEngineName*/ 이다.

여기서 설정하는 내용은 로깅관련 설정, 네트워크 관련 설정, 그리고 Tmax 단에서 들어오는 서비스 호출에 대한 설정이다.

다음은 앞으로 사용될 예제의 일부분이다.

```
<jtmax-container>
  <logging>
     <error-log>
     <target>console</target>
     <level>fatal</level>
     <buffer-size>1024</buffer-size>
     <valid-day>1</valid-day>
     </error-log>
     <user-log>
```

```
      <target>console</target>
      <level>fatal</level>
      <valid-day>1</valid-day>
      </user-log>
  </logging>

  <context>
   <context-name>webtsvr</context-name>
    <connection>
      <conntection-type>webtserver</connection-type>
      <connection-id>webt1</connection-id>
      <address>127.0.0.1</address>
      <port>6735</port>
      <backlog>50</backlog>
    </connection>
 </context>

 <service>
    <name>GSVC01</name>
    <ejb-name>echotest</ejb-name>
    <method-name>setIntEcho</method-name>
    <input-args>int</input-args>
 </service>

</jtmax-container>
```

<Context> 태그의 속성

context-name [(required) default : null]

이 속성은 컨테이너 내부적으로 각 컨텍스트들을 구별하기 위해 사용된다. 위의 예 에서는 webtsrv 로 지정이 되어 있다.

현재 jTmax 는 하나의 컨텍스트만을 지원하며 여러개의 컨텍스트를 설정한 경우 마지막의 설정이 유효하다.

<webserver-connection> 태그의 속성

connection-type [(required) default : null]
Tmax 와의 Connection Type 을 지정하는 것으로 WebT-Server 를 사용하기 위해 "webtserver" 라 설정하도록 한다.

Connection-id [(required) default : null]
다른 종류의 Connection 과 구분하기 위한 id 값이다.

port [(required) default : 0]
Tmax 시스템에 기동되는 JeusGW 가 접속하는 포트 번호 이다. JeusGW 의 설정과 맞게 해야 한다.

backlog [(required) default : 50]
JeusGW 가 접속되는 Socket 의 backlog 값을 결정한다.

<service> 태그의 속성
service 태그는 디플로이된 EJB 의 메소드를 호출하기 위해서 사용된다. 여기서 사용되는 서비스명은 Tmax 에서 호출할 때 사용하는 서비스명과 동일해야 된다. 물론, 하나 이상의 service 가 올 수 있다.

name
Tmax 에서 호출하는 서비스 명을 말하는데, 여기서는 "GSVC01"라고 설정하도록 한다.

ejb-name
디플로이된 EJB 모듈의 ejb-name 에 해당되며, 모듈의 DD 파일인 ejb-jar.xml 의 ejb-name 과 동일해야 된다. 여기서는 echotest 이다.

method-name
호출할 메소드의 이름을 지정한다. 여기서는 정수값을 설정하는 setIntEcho 이다.

input-args

메소드의 인자를 설정하는 부분으로서 클래스에 정의된 인자를 그대로 쓰면 된다. 만일에 사용하는 인자가 없으면 이 태그는 필요없다.

7.4 JeusGW 설정

Tmax 측의 환경 설정을 위해서 호출할 서비스와 네트워크 정보를 담고 있는 환경 설정 파일과 필드키 파일에 대해서 다루어 보기로 하겠다.

7.4.1 Tmax 환경파일 설정

WebT-Server 를 사용하기 위한 Tmax 환경파일의 설정은 일반적인 게이트웨이를 사용하기 위한 설정과 동일하다.

자세한 설정 방법은 Tmax Administrator Guide 를 참조하도록 한다.

EJB 서비스를 호출하기 위한 설정과 관련된 부분을 아래 예제와 같다.

```
*SERVICE
GSVC01                  SVRNAME = JeusGW
GSVC02                  SVRNAME = JeusGW

*GATEWAY
JeusGW          GWTYPE = JEUS,
                PORTNO = 6734,
                RGWADDR      = "127.0.0.1",
                RGWPORTNO    = 6735,
                NODENAME     = node_name,
```

- SERVICE 절의 내용

```
ServiceName             SVRNAME = ServerName
```

ServiceName

Tmax Client 에서 호출하고자 하는 서비스의 이름으로 WebT-Server 를 사용하는 경우 JEUS 에서 제공하는 EJB Service 에 대한 Alias 이다. 실제로 수행할 JEUS 의 EJB 서비스의 내용은 아래에 설명할 Jeus 서비스 파일에 설정한다.

ServerName

해당 서비스를 제공하는 서버 프로그램명을 지칭하는 것으로 WebT-Server 를 사용하는 경우 아래 설명할 GATEWAY 절에 설정하게 되는 JeusGW 의 이름 (*GatewayName*) 을 지정하도록 한다.

- GATEWAY 절의 내용

```
GateWayName     GWTYPE        = GWType,
                PORTNO        = PortNumber,
                RGWADDR       = RemoteIPAddress,
                RGWPORTNO     = RemotePortNumber,
                NODENAME      = NodeName,
```

GateWayName

여타 Tmax 서버 프로그램이나 다른 게이트웨이들과 구분하기 위해 사용하는 해당 게이트웨이의 고유 명칭이다.

GWType

Tmax 에서 사용하는 게이트웨이의 종류를 지정한다. JeusGW 를 사용 하기 위해서 "JEUS"로 설정한다.

PortNumber

Tmax 와 JeusGW 게이트웨이가 통신하기 위해 사용하는 Port 번호이다.

RemoteIPAddress

게이트웨이가 접속할 상대방의 IP 주소이다. JeusGW 게이트웨이가 접속할 JEUS 서버의 IP 주소를 설정한다.

RemotePortNumber

게이트웨이가 접속할 상대방의 Port 번호이다. JEUS 의 jTmax 운영환경 파일에 설정된 Connection Port 번호를 설정한다.

참고로 작성된 환경파일(.m)을 Tmax 에서 적용시키는 방법을 알아보자.

설정파일 이름을 sample.m 이라고 하면 다음과 같이 하면 된다.

1. $TMAXDIR/config/ 아래에 있는 sample.m 파일을 다음과 같이 컴파일 하면 tmax 설정파일인 tmconfig 가 생성된다.

   ```
   cfl -i sample.m
   ```

2. 설정된 서비스를 이용하기 위하여 gst(generate service table) 명령어를 사용하여 서비스 테이블을 생성한다. 다음과 같이 수행하면 $JEUS_HOME/svct 아래 SVRNAME_svctab.c 파일이 만들어진다.

   ```
   gst
   ```

3. tmax 를 실행하면 환경파일이 적용된다.

   ```
   tmboot
   ```

7.5 WebT-Server Application 예제

지금까지 jTmax 모듈을 통해서 Tmax 에서 EJB 메소드를 호출하기 위해서 관련된 내용을 살펴보았다. 여기서는 지금까지 배운 내용을 토대로 해서 Tmax client 가 JEUS 시스템내의 EJB 메소드를 호출하여 정수값을 설정하고 문자열을 얻어오는 예제를 살펴보도록 하자.

7.5.1 클라이언트 프로그램

여기서는 GSVC01 을 수행하기 위해서 필드키 파일 컴파일시 생성되는 tmax_fdl.h 가 include 되어 있다. 서버 프로그램이 다양한 테스트 메소드를 가지고 있기 때문에 여기서 좀더 응용하여 더 많은 테스트를 해보기 바란다.

client.c (Tmax Client)

```
#include <stdio.h>
#include <usrinc/atmi.h>
#include <usrinc/fbuf.h>
#include "tmax_fdl.h"

int main(int argc, char **argv)
{
 int ret,cd;
 char *buf;
 long rlen;
 FBUF *rbuf;
 int sendValue = 123;

 ret = tmaxreadenv("tmax.env", "JEUSGW");
 if (ret < 0){
        printf("tmaxreadenv fail.. \n");
        return -1;
 }
 ret = tpstart(NULL);
 if (ret < 0){
        printf("tpstart fail..[%s]\n", tpstrerror(tperrno));
        return -1;
 }
 buf = (char *)tpalloc("FIELD", "", 0);
 if (buf == NULL){
        printf("tpalloc fail.. \n");
        tpend();
        return -1;
 }
 rbuf = (FBUF *)tpalloc("FIELD", "", 0);
 if (rbuf == NULL){
        printf("fbuf tpalloc fail.. \n");
        tpfree((char*)buf);
        tpend();
        return -1;
```

```
}

fbput((FBUF *)buf, EJB_RESULT_INT, (char *)&sendValue, 0);
cd = tpacall("GSVC01", (char *)buf, strlen(buf), TPNOFLAGS);

if (cd < 0){
       printf("tpacall fail..[%s]\n", tpstrerror(tperrno));
       tpfree((char*)buf);
       tpfree((char*)rbuf);
       tpend();
       return -1;
}

ret = tpgetrply(&cd, (char **)&rbuf, (long *)&rlen, TPNOFLAGS);

if (ret < 0){
       printf("tpgetrply fail..[%s]\n",tpstrerror(tperrno));
       tpfree((char *)buf);
       tpfree((char *)rbuf);
       tpend();
       return -1;
}

printf("return value : \n");
fbprint(rbuf);

cd = tpacall("GSVC02", (char *)buf, strlen(buf), TPNOFLAGS);

if (cd < 0){
       printf("tpacall fail..[%s]\n", tpstrerror(tperrno));
       tpfree((char*)buf);
       tpfree((char*)rbuf);
       tpend();
       return -1;
}
```

```
ret = tpgetrply(&cd, (char **)&rbuf, (long *)&rlen, TPNOFLAGS);

if (ret < 0){
       printf("tpgetrply fail..[%s]\n",tpstrerror(tperrno));
       tpfree((char *)buf);
       tpfree((char *)rbuf);
       tpend();
       return -1;
}

printf("return value : \n");
fbprint(rbuf);

tpfree((char *)buf);
tpfree((char *)rbuf);

tpend();
return 0;
}
```

7.5.2 서버 프로그램

Tmax 의 서비스 호출에 대해서 반응하는 부분이다. 많은 메소드가 있지만 여기서는 setIntEcho(int)와 getStringEcho()가 이용된다.

EchoTestEJB.Java (Bean)

```
/*
 ## EchoTestEJB.java
*/
package echo;

import javax.ejb.*;
import java.rmi.*;
import tmax.webt.*;
import tmax.webt.io.*;
```

```
public class EchoTestEJB implements SessionBean {

 public void setIntEcho(int i) {
        System.out.println("Set an int value : " + i);
 }

 public void setBoolEcho(boolean b) {
        System.out.println("Set a boolean value : " + b);
 }

 public void setDoubleEcho(double d) {
        System.out.println("Set a double value : " + d);
 }

 public void setFloatEcho(float f) {
        System.out.println("Set a float value : " + f);
 }

 public void setShortEcho(short sh) {
        System.out.println("Set a short value : " + sh);
 }

 public void setLongEcho(long l) {
        System.out.println("Set a long value : " + l);
 }

 public void setCharEcho(char ch) {
        System.out.println("Set a char value : " + ch);
 }

 public void setByteEcho(byte b) {
        System.out.println("Set a byte value : " + b);
 }

 public void setStringEcho(String str) {
```

```
        System.out.println("Set a String value : " + str);
}

public void setCarrayEcho(byte[] byteArray) {
        System.out.println("Set a byte array value : " +
                    byteArray);
}

public void setWebtBEcho(WebtBuffer webt) {
        System.out.println("Set a WebtBuffer value : " +
                  webt.getString());
}

public boolean getBooleanEcho() {
        boolean b = true;
        System.out.println("Get the boolean value : " + b);
        return b;
}

public int getIntEcho() {
        int i = 10000;
        System.out.println("Get the int value : " + i);
        return i;
}

public double getDoubleEcho() {
        double d = 365.50;
        System.out.println("Get the double value : " + d);
        return d;
}

public float getFloatEcho() {
        float f = 34.87f;
        System.out.println("Get the float value : " + f);
        return f;
```

```
}

public short getShortEcho() {
       short sh = 1000;
       System.out.println("Get the short value : " + sh);
       return sh;
}

public long getLongEcho() {
       long l = 10000000;
       System.out.println("Get the long value : " + l);
       return l;
}

public char getCharEcho() {
       char ch = 'a';
       System.out.println("Get the char value : " + ch);
       return ch;
}

public byte getByteEcho() {
       byte b = 100;
       System.out.println("Get the byte value : " + b);
       return b;
}

public String getStringEcho() {
       String str = "test string";
       System.out.println("Get the String value : " + str);
       return str;
}

public byte[] getByteArrayEcho() {
       byte[] byteArray = new byte[]{65, 34, 74, 69};
       System.out.println("Get the byte array value : " +
```

```
                byteArray);
        return byteArray;
}

public WebtBuffer getWebtBufferEcho() {
        WebtStringBuffer buff =
                new WebtStringBuffer("EUC-KR");
        buff.setString("test WebtBuffer");
        System.out.println("Get the WebtBuffer value : " +
                buff.getString());
        return buff;
}

public EchoTestEJB() {}
public void ejbCreate() {}
public void ejbRemove() {}
public void ejbActivate() {}
public void ejbPassivate() {}
public void setSessionContext(SessionContext ctx) {}

}
```

EchoTest.java (remote)

```
/*
 ## EchoTest.java
*/

package echo;

import javax.ejb.EJBObject;
import java.rmi.RemoteException;
import tmax.webt.*;

public interface EchoTest extends EJBObject {
```

```
public void setBoolEcho(boolean b) throws RemoteException;
public void setIntEcho(int i) throws RemoteException;
public void setDoubleEcho(double d) throws RemoteException;
public void setFloatEcho(float f) throws RemoteException;
public void setShortEcho(short sh) throws RemoteException;
public void setLongEcho(long l) throws RemoteException;
public void setCharEcho(char ch) throws RemoteException;
public void setByteEcho(byte b) throws RemoteException;
public void setStringEcho(String str)
                        throws RemoteException;
public void setCarrayEcho(byte[] byteArray)
                        throws RemoteException;
public void setWebtBEcho(WebtBuffer webt)
                        throws RemoteException;
public boolean getBooleanEcho() throws RemoteException;
public int getIntEcho() throws RemoteException;
public double getDoubleEcho() throws RemoteException;
public float getFloatEcho() throws RemoteException;
public short getShortEcho() throws RemoteException;
public long getLongEcho() throws RemoteException;
public char getCharEcho() throws RemoteException;
public byte getByteEcho() throws RemoteException;
public String getStringEcho() throws RemoteException;
public byte[] getByteArrayEcho() throws RemoteException;
public WebtBuffer getWebtBufferEcho()
                        throws RemoteException;
}
```

EchoTestHome.Java (Home)

```
/*
 ## EchoTestHome.java
*/

package echo;
```

```
import javax.ejb.EJBHome;
import java.rmi.RemoteException;
import javax.ejb.CreateException;

public interface EchoTestHome extends EJBHome {

 public EchoTest create()
            throws CreateException, RemoteException;
}
```

7.5.3 환경파일

Tmax 환경 파일 내의 각각의 내용은 시스템마다 다를 것이다. 그러나 SERVICE 와 GATEWAY 설정은 앞에서 설명한 내용대로 설정해야 한다.

Tmax Configuration file (sample.m)

```
*DOMAIN
Domain1     SHMKEY=77295, TPORTNO=8000, MINCLH=1, MAXCLH=1

*NODE
Neya          TMAXDIR      = "/home/navis/tmax/",
              APPDIR = "/home/navis/tmax/appbin"

*SVRGROUP
svrgrp              NODENAME      = Neya

*SERVER

*SERVICE
GSVC01        SVRNAME      = JeusGW
GSVC02        SVRNAME      = JeusGW

*GATEWAY
JeusGW        GWTYPE = JEUS,
              PORTNO = 6734,
```

```
RGWADDR      = "127.0.0.1",
RGWPORTNO    = 6735,
NODENAME     = Neya,
```

FieldKey Configureation(tmax.f)

```
EJB_CLASS_NAME    100   string   -  -
EJB_METHOD_NAME   101   string   -  -
EJB_RESULT_BYTE   102   char     -  -
EJB_RESULT_SHORT  103   short    -  -
EJB_RESULT_INT    104   int      -  -
EJB_RESULT_LONG   105   long     -  -
EJB_RESULT_FLOAT  106   float    -  -
EJB_RESULT_DOUBLE 107   double   -  -
EJB_RESULT_STRING 108   string   -  -
EJB_RESULT_CARRAY 109   carray   -  -
ORG_TYPE          110   int      -  -
ORG_DATA_STRING   111    string   -  -
ORG_DATA_CARRAY   112    carray   -  -
```

JEUS Engine Container Configuration(JEUSMain.xml)

JEUSMain.xml 에 본문에서 설명한 내용을 넣으면 된다.

jtmax Operation Environment (jtmaxMain.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<jtmax-container xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
 <logging>
       <error-log>
             <target>console</target>
             <level>debugs</level>
             <buffer-size>1024</buffer-size>
             <valid-day>1</valid-day>
       </error-log>
       <user-log>
```

```
            <target>console</target>
            <level>debug</level>
            <valid-day>1</valid-day>
        </user-log>
    </logging>

    <context>
        <context-name>webtsvr</context-name>
        <connection>
            <conntection-type>webtserver</conntection-type>
            <connection-id>webt1</connection-id>
            <address>127.0.0.1</address>
            <port>6735</port>
            <backlog>50</backlog>
        </connection>
    </context>

    <service>
        <name>GSVC01</name>
        <ejb-name>echotest</ejb-name>
        <method-name>setIntEcho</method-name>
        <input-args>int</input-args>
    </service>

    <service>
        <name>GSVC02</name>
        <ejb-name>echotest</ejb-name>
        <method-name>getStringEcho</method-name>
        <input-args>String</input-args>
    </service>
</jtmax-container>
```

EJB Deploy Descriptor (jeus-ejb-dd.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<jeus-ejb-dd xmlns="http://www.tmaxsoft.com/xml/ns/jeus">
   <module-info/>
   <beanlist>
      <jeus-bean>
         <ejb-name>echotest</ejb-name>
         <export-name>echotest</export-name>
         <object-management>
            <passivation-timeout>-1</passivation-timeout>
            <disconnect-timeout>-1</disconnect-timeout>
         </object-management>
      </jeus-bean>
   </beanlist>
</jeus-ejb-dd>
```

EJB Deploy Descriptor (ejb-jar.xml)

```
<?xml version="1.0" encoding="UTF-8"?>
<ejb-jar version="2.1" xmlns="http://java.sun.com/xml/ns/j2ee">
   <enterprise-beans>
      <session>
         <ejb-name>echotest</ejb-name>
         <home>echo.EchoTestHome</home>
         <remote>echo.EchoTest</remote>
         <ejb-class>echo.EchoTestEJB</ejb-class>
         <session-type>Stateless</session-type>
         <transaction-type>Container</transaction-type>
      </session>
   </enterprise-beans>
</ejb-jar>
```

7.5.4 실행

예제를 테스트해 보기 위해서는 서버 역할을 하는 JEUS 시스템을 먼저 구동을 시켜야 된다. 그러기 전에 우선 우리가 사용할 EJB 모듈인 echotest.jar 를 $JEUS_HOME/deploy_home 에 복사하여 JEUS 시스템이 부팅될 때 디플로이 되도록 하자.

EJB 모듈 생성은 JEUS Builder 를 사용을 하고 부팅 외에 디플로이를 원한다면 JEUS 웹관리자를 사용하여라.

JEUS 시스템을 다음과 같이 부팅시키자. –U 다음에는 사용자 이름을, -P 에는 암호를 입력한다.

```
jeus -xml -Uusername -Ppasswd
```

모듈이 성공적으로 디플로이 되었다면 Tmax 클라이언트 프로그램을 실행시킬 차례다. 순서는 다음과 같다.

1. Tmax 배포시 제공되는 compile 배치파일을 이용하여 컴파일한다.

   ```
   compile c client
   ```

2. 환경 설정파일을 컴파일 한다.

   ```
   cfl -i sample.m
   ```

3. 서비스 테이블을 만든다.

   ```
   gst
   ```

4. Tmax 를 구동한다.

   tmboot

5. 클라이언트 프로그램을 실행한다.

   client

7.5.5 실행결과

Tmax 클라이언트를 실행 시키면 JEUS 시스템에서는 빈클래스인 EchoTestEJB 의 setIntEcho() 호출되어 tmax 에서 인자로 넘겨준 값이

출력될 것이다. 이 메소드의 반환값은 없기 때문에 Tmax 클라이언트에 출력되는 return value 는 없다. 그리고 getStringEcho()는 인자가 없는 대신 반환값이 있기 때문에 Tmax 클라이언트에는 “test string”라는 return value 가 출력될 것이다.

7.6 결론

본 장에서는 Tmax-JEUS 간의 통신중에 JEUS 시스템에서 Tmax 서비스를 호출하는 Webt 라이브러리와는 반대 개념인 jTmax 에 대해서 자세히 살펴보았다.

jTmax 는 Tmax 클라이언트가 JEUS 시스템내의 EJB 의 메소드를 호출하는 것을 가능하게 해준다. 이를 위해서 필드키가 설정 되어야 하고, 서비스가 정의 되어야 하며 그리고 jTmax 을 구동시키기 위한 환경 설정도 뒤따라야 한다. 또한 Tmax 와 JEUS 간에 실제적으로 데이터를 주고 받기 위해서는 Webt 가 지원하는 버퍼와 네트워크를 사용한다.

이로써 Tmax 시스템과 JEUS 시스템은 서로 애플리케이션 레벨에서 양방향 통신이 가능하게 되었다.

8 결론

이 매뉴얼 에서는 WebT Library 에 대한 전반적인 개념과 구조를 익히고. 실제 Web Application Service 에 적용하기 위한 방법에 대한 구체적인 설명들을 살펴보았다. 매뉴엘에서 설명한 간단을 Web Application 예제들을 기반으로 해서 사용자가 제공하고자 하는 Application 을 적절히 구현하도록 한다.

A Web Container XML Configuration Reference(WEBMain.xml)

A.1 소개

[Outline the main purpose of the file and indicate the top level element/tag]

WEBMain.xml 에 포함된 모든 XML element 들에 대한 자세한 설명을 하고자 한다. 이 configuration file 의 XML SCHEMA file "web-main-config.xsd"을 참조하도록 한다.

아래의 구성은 다음과 같이 세 부분으로 살펴볼 수 있다.

1. The **XML SCHEMA/XML tree**, XML configuration file 의 모든 XML element 들의 전체 계층구조를 살펴 볼 수 있다. 각 element 의 설명 format 은 다음과 같다.
    a. An **index number** (e.g. "(11)"), 해당 Tree node 를 좀더 빨리 검색할 수 있도록 고유 번호를 부여한다.
    b. The XML <**element name**> XML SCHEMA.에 정의된 이름이다.
    c. The **element cardinality**, XML SCHEMA 에 정의된 각 항목의 대한 반복 횟수를 정의한다.(예: "?" = zero or one element (optional), "+" = one or more elements, "*" = zero or more elements, (no character) = exactly one element (non-optional))
    d. **performance issue** 몇몇의 "P"로 시작하는 이름의 element node 의 경우 configuration tuning option 의 성격을 띄는 속성이다. 이에 대한 간략한 설명을 한다.
2. The **element reference** 각 XML element 에 대한 아래의 6 항목으로 구성된 참조 테이블을 구성하였다
    a. **Description**: XML element 에 대한 간략한 설명을 한다.
    b. **Value Description**: 이 항목에 기대되는 값에 대한 설명을 한다.

c. **Value Type**: 기대되는 해당 값에 대한 Java type 을 알려준다. (if other than “String”).
d. **Default Value**: 해당 XML element 에 대한 default 값을 알려준다.
e. **Defined values**: Any pre-defined values and their interpretation.
f. **Example**: XML tag 설정에 대한 sample value 를 보여준다.
g. **Performance Recommendation**: element 값에 대한 performance 면에 대해서 참고 될만한 내용을 기입한다..
h. **Child Elements**: container-element 간의 하부 child element 가 있는 경우 이를 설명한다.

3. The **sample XML file,** “WEBMain.xml” 환경 파일의 구체적인 sample 를 보여준다.

A.2 XML Schema/XML Tree

```
(482) <tmax>*
              (483) <export-name>
              (484) <webt-datasource-type>? P
              (485) <webt-logging>?
                     (486) <file-name>? P
                     (487) <level>? P
                     (488) <valid-day>? P
                     (489) <buffer-size>? P
              (490) <host-name>
              (491) <port>
              (492) <backup-host-name>?
              (493) <backup-port>?
              (494) <fdb-file>?
              (495) <default-charset>?
              (496) <enable-pipe>? P
              (497) <enable-extended-header>? P
              (498) <inbuf-size>? P
              (499) <outbuf-size>? P
              (500) <max-idle-timeout>? P
              (501) <service-timeout>?
```

```
(502) <transaction-timeout>?
(503) <transaction-block-timeout>?
(504) <security>?
       (505) <user-name>?
       (506) <user-password>?
       (507) <domain-name>?
       (508) <domain-password>?
(509) <tmax-connection-pool>?
       (510) <pooling>?
              (511) <min>? P
              (512) <max>? P
              (513) <step>? P
              (514) <period>? P
       (515) <wait-free-connection>?
              (516) <enable-wait>? P
              (517) <wait-time>? P
(518) <monitoring>?
       (519) <enable-check-alive>? P
       (520) <enable-check-idle>? P
       (521) <check-pool-interval>? P
(522) <tmax-property>*
       (523) <name>
       (524) <value>
```

A.3 Element Reference

다음은 "JEUSMain.xml"에서 WebT DataSource 과 관련한 element 대한 설명 table 이다.

```
(482) <jeus-system> <resource> <external-source> <tmax>
```

Description 이 element 는 JEUS JNDI namespace 에 있는 Tmax TP monitor 서버를 export 하는데 사용된다. Tmax 제품의 사용법은 Tmax 매뉴얼을 참조하기 바란다.

Child Elements

```
(483)export-name
(484)webt-datasource-type?
(485)webt-logging?
```

```
(490)host-name
(491)port
(492)backup-host-name?
(493)backup-port?
(494)fdb-file?
(495)default-charset?
(496)enable-pipe?
(497)enable-extended-header?
(498)inbuf-size?
(499)outbuf-size?
(500)max-idle-timeout?
(501)service-timeout?
(502)transaction-timeout?
(503)transaction-block-timeout?
(504)security?
(509)tmax-connection-pool?
(518)monitoring?
(522)tmax-property*
```

483) <jeus-system> <resource> <external-source> <tmax> **<export-name>**

Description JNDI 에 등록되어 서비스 되는 이름이다.

Value Description a JNDI export name

Value Type token

Example <export-name>webtResource</export-name>

(484) <jeus-system> <resource> <external-source> <tmax> **<webt-datasource-type>**

Description WebtDataSource 의 type 을 정의한다.

Value Type webt-datasource-poolType

Default Value WebtConnectionPoolDataSource

Defined Value WebtConnectionPoolDataSource

일반 ConnectionPool Service 를 지원하는 WebtDataSource 타입이다.

WebtXADataSource
XA Service 를 지원하는 WebtDataSource 타입이다.

Example `<webt-datasource-type>WebtConnectionPoolDataSource</webt-datasource-type>`

(485) <jeus-system> <resource> <external-source> <tmax> **<webt-logging>**

Description Webt 모듈에서 남겨지는 error 로그 설정이다.

Example `<webt-logging> ... </webt-logging>`

Child Elements
(490)file-name?
(491)level?
(492)valid-day?
(493)buffer-size?

(486) <jeus-system> <resource> <external-source> <tmax> <webt-logging> **<file-name>**

Description 생성될 WebT log file 경로와 이름을 정의한다. OS 가 Windows 인 경우에는 파일 구분자를 '\'가 아닌 '\\'로 해 주어야 한다.

Value Type token

Default Value webt.log

Example `<file-name>/home/jeus/log/webt.log</file-name>`

(487) <jeus-system> <resource> <external-source> <tmax> <webt-logging> **<level>**

Description WebT log level 다음의 none|info|debug 값들 중 하나를 정의한다.

Value Description none/info/debug

Value Type token

Default Value none

Defined Value none
로그 메시지를 남기지 않는다. [info] 기타 로그 메시지 이외의 Non-critical messages 추가로 볼 수 있다. [debug] debugging 을 하기위한 모든 로그 메시지 정보를 살펴볼 수 있다

Example
```
<level>debug</level>
```

(488) <jeus-system> <resource> <external-source> <tmax> <webt-logging> **<valid-day>**

Description 지정된 로그파일에 기록된 내용에 날짜의 제한 값을 설정한다. 만약 1 이상이라는 값으로 설정되었을 경우 하루 단위로 로그 파일이 대체되어 해당 시간에 맞는 log message 를 기록한다.

Value Description a number of days

Value Type off-intType

Value Type Description 기본적으로 non-negative int 형이지만 -1 인 경우에는 설정을 하지 않은게 된다.즉, off 된다.

Default Value -1

Defined Value -1
만약 -1 값으로 설정되었을 경우 지정된 하나의 로그 파일에 WebT log message 를 모두 남긴다

Example
```
<valid-day>2</valid-day>
```

(489) <jeus-system> <resource> <external-source> <tmax> <webt-logging> **<buffer-size>**

Description Log file 에 WebT log message 를 남길 때 이 buffer size 만큼 메모리에서 임시로 저장하였다가 한꺼번에 기록한다.

Value Description bytes

Value Type nonNegativeIntType

Value Type Description 0 이상의 int type 이다. 즉, int 범위에서 0 이상의 값들을 포함한다.

Default Value 0

Defined Value 0
0 값으로 지정하면 log message 에 대한 buffering 을 하지 않는다

Example
```
<buffer-size>1024</buffer-size>
```

(490) <jeus-system> <resource> <external-source> <tmax> **<host-name>**

Description Tmax server 의 이름이나 IP 주소를 말한다.

Value Description a valid IP address

Value Type token

Example
```
<host-name>111.111.111.1</host-name>
```

(491) <jeus-system> <resource> <external-source> <tmax> **<port>**

Description Tmax server 환경 파일에 설정된 "PORT" 값과 동일한 값을 설정해 주어야 한다.

Value Description a port number

Value Type int

Example
```
<port>8888</port>
```

(492) <jeus-system> <resource> <external-source> <tmax> **<backup-host-name>**

Description Main Tmax Server 의 failover 상황에 직면하였을 때 서비스에 무리가 없도록 설정할 Backup Tmax Server 의 IP address 값을 설정한다.

Value Description a valid IP address

Value Type token

Example
```
<backup-addresss>111.111.111.2</backup-address>
```

(493) <jeus-system> <resource> <external-source> <tmax> **<backup-port>**

Description Backup Tmax Server 의 환경 파일에 "PORT" 값 동일한 값을 설정해 주어야 한다.

Value Description a port number

Value Type int

Example <backup-port>8889</backup-port>

(494) <jeus-system> <resource> <external-source> <tmax> **<fdb-file>**

Description tmax fdl file 의 위치를 설정한다

Value Description a tmax fdl filename

Value Type token

Example <fdl-file>/home/tmax/fdl/tmax.fdl</fdl-file>

(495) <jeus-system> <resource> <external-source> <tmax> **<default-charset>**

Description Application 에 적용될 default character set 을 설정한다. 기본적으로 WebtSystem.setDefaultCharset(java.lang.String charset)수행과 동일하다.

Value Description a encoding name

Value Type token

Example <default-charset>euc-kr</default-charset>

(496) <jeus-system> <resource> <external-source> <tmax> **<enable-pipe>**

Description WebT 와 Tmax server 가 local machine 에 같이 존재할 경우 pipe 통신 여부를 설정하여 performance 를 꾀할 수 있다.

Value Description pipe 통신 enable/disable 여부

Value Type boolean

Default Value false

Example `<enable-pipe>true</enable-pipe>`

(497) <jeus-system> <resource> <external-source> <tmax> **<enable-extended-header>**

Description Tmax Header 확장된 3.11.x 이후 버전과의 서비스가 성공적으로 이루어 지도록 설정하는 option 값.

Value Description tmax header 확장 사이즈 여부

Value Type boolean

Default Value false

Defined Value true
Tmax header size 가 112byte 로 확장된 tmax engine 과의 서비스.

false
Tmax header size 가 기본 96byte 인 tmax engine 과의 서비스.

Example `<enable-extended-header>true</enable-extended-header>`

(498) <jeus-system> <resource> <external-source> <tmax> **<inbuf-size>**

Description Tmax 으로부터 데이터 수신 시 한번에 읽어 들일 buffer size 단위를 설정할 수 있다.

Value Description bytes

Value Type int

Default Value 4096

Example `<inbuf-size>8192</inbuf-size>`

(499) <jeus-system> <resource> <external-source> <tmax> **<outbuf-size>**

Description Tmax 으로부터 한번에 데이터 송신할 buffer size 단위를 설정한다

Value Description bytes

Value Type int

Default Value 4096

Example `<outbuf-size>8192</outbuf>`

(500) <jeus-system> <resource> <external-source> <tmax> **<max-idle-timeout>**

Description connection pool 에서 여기서 설정된 maxIdleTime 동안 사용되지 않은 idle 상태의 connection 이 있을 경우 해당 connection 객체를 connection pool 에서 remove 한다. 최소한의 iniCapacity 값은 유지하도록 한다

Value Description millisecond

Value Type int

Default Value 60000

Example `<max-idle-timeout>30000</max-idle-timeout>`

(501) <jeus-system> <resource> <external-source> <tmax> **<service-timeout>**

Description Tmax Server 로 service 요청하여 여기서 설정된 시간 내에 서비스가 이루어지지 않으면 해당 서비스 요청을 중지한다.

Value Description millisecond

Value Type int

Example `<max-idle-timeout>60000</max-idle-timeout>`

(502) <jeus-system> <resource> <external-source> <tmax> **<transaction-timeout>**

Description Tmax Server 로 Transaction service 를 요청한 경우 여기서 설정된 시간 내에 서비스가 이루어지지 않으면 요청된 서비스를 중지한다

Value Description millisecond.

Value Type int

Example `<transaction-timeout>60000</transaction-timeout>`

(503) <jeus-system> <resource> <external-source> <tmax> **<transaction-block-timeout>**

Description Tmax Server 로 Transaction commit service 를 요청한 경우 여기서 설정된 시간 내에 commit 서비스 정상적인 종료가 이루어지지 않으면 요청된 서비스를 중지한다.

Value Description millisecond.

Value Type int

Example `<transaction-block-timeout>60000</transaction-block-timeout>`

(504) <jeus-system> <resource> <external-source> <tmax> **<security>**

Description 이 element 는 Tmax server 와 통신시 security 권한을 정의한다.

Example `<securityType> ... </securityType>`

Child Elements
(510)user-name?
(511)user-password?
(512)domain-name?
(513)domain-password?

(505) <jeus-system> <resource> <external-source> <tmax> <security> **<user-name>**

Description Tmax server 로의 연결시 Tmax server 에 등록된 user 에 대한 인증을 위해 user name 을 설정한다

Value Description a user name

Value Type token

Example `<user-name>admin</user-name>`

(506) <jeus-system> <resource> <external-source> <tmax> <security>

<user-password>

Description Tmax server 로의 연결시 Tmax server 에 등록된 user 에 대한 인증을 위해 user password 을 설정한다.

Value Description a password string

Value Type token

Example `<user-password>tmax</user-password>`

(507) <jeus-system> <resource> <external-source> <tmax> <security> **<domain-name>**

Description Tmax server 로의 연결시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain name 을 설정한다.

Value Description a domain name

Value Type token

Example `<domain-name>tmaxadm</domain-name>`

(508) <jeus-system> <resource> <external-source> <tmax> <security> **<domain-password>**

Description Tmax server 로의 연결시 Tmax server 에 등록된 domain 에 대한 인증을 위해 domain password 을 설정한다

Value Description a domain password

Value Type token

Example `<domain-password>tmaxadm</domain-password>`

(509) <jeus-system> <resource> <external-source> <tmax> **<tmax-connection-pool>**

Description 이 element 는 Tmax server 와 연결 풀을 설정한다.

Example `<tmax-connection-pool> ... </tmax-connection-pool>`

Child Elements (515)pooling?
(520)wait-free-connection?

```
(510) <jeus-system> <resource> <external-source> <tmax> <tmax-
connection-pool> <pooling>
```

Description 풀 설정.

Child Elements (516)min?
(517)max?
(518)step?
(519)period?

```
(511) <jeus-system> <resource> <external-source> <tmax> <tmax-
connection-pool> <pooling> <min>
```

Description pooling 되는 객체의 최소값을 지정한다.

Value Type nonNegativeIntType

Value Type Description 0 이상의 int type 이다. 즉, int 범위에서 0 이상의 값들을 포함한다.

Default Value 2

```
(512) <jeus-system> <resource> <external-source> <tmax> <tmax-
connection-pool> <pooling> <max>
```

Description pooling 되는 객체의 최대값을 지정한다.

Value Type nonNegativeIntType

Value Type Description 0 이상의 int type 이다. 즉, int 범위에서 0 이상의 값들을 포함한다.

Default Value 30

```
(513) <jeus-system> <resource> <external-source> <tmax> <tmax-
connection-pool> <pooling> <step>
```

Description pooling 되는 객체가 증가될때의 증가량을 설정한다.

Value Type nonNegativeIntType

Value Type Description 0 이상의 int type 이다. 즉, int 범위에서 0 이상의 값들을 포함한다.

Default Value 4

(514) <jeus-system> <resource> <external-source> <tmax> <tmax-connection-pool> <pooling> **<period>**

Description pooling 되는 객체를 정리하는 시간을 지정한다.

Value Type long

Default Value 3600000

Performance Recommendation 이 값이 클수록 정리하는 주기가 길어져 server 운영에는 부하가 적게 가해지나 그만큼 메모리가 누수될 수 있으므로 적당한 값으로 지정한다.

(515) <jeus-system> <resource> <external-source> <tmax> <tmax-connection-pool> **<wait-free-connection>**

Description 이 element 는 connection 이 사용중 인 경우 다른 요청이 들어왔을 때 connection pool 을 다루는 방법을 설정한다.

Child Elements (521)enable-wait?
(522)wait-time?

(516) <jeus-system> <resource> <external-source> <tmax> <tmax-connection-pool> <wait-free-connection> **<enable-wait>**

Description 이 태그는 pool 안에 이용 가능한 connection 이 없거나 pool 안에 connection 들이 이미 최대값이 되어질 때 connection 을 요청 처리하는 방법을 결정한다. 만약 true 라면 시스템은 이용 가능한 connection 을 얻기위해 대기한다. 만약 false 라면, 시스템은 사용자 요청이 올 때 새로운 connection 을 만들고 사용이 끝난 이후에 pool 에 반납하지 않는다.

Value Type boolean

Default Value false

(517) <jeus-system> <resource> <external-source> <tmax> <tmax-connection-pool> <wait-free-connection> **<wait-time>**

Description 이 태그는 <enable-wait>가 true 일때만 유효하다. 이것은 사용자가 connection 을 위해 대기하는 시간을 나타낸다. 만약 어떠한 connection 도 사용자가 이 시간동안 대기하여도 이용할 수 없을 때는 시스템은 사용자에게 exception 을 던져준다.

Value Description millisecond 단위이다.

Value Type nonNegativeLongType

Value Type Description 0 이상의 long type 이다. 즉, long 범위에서 0 이상의 값들을 포함한다.

Default Value 10000

(518) <jeus-system> <resource> <external-source> <tmax> **<monitoring>**

Description 이 element 는 Tmax server 와 통신하는 pool 에 대한 monitoring 값을 설정한다.

Example <monitoring> ... </monitoring>

Child Elements (524)enable-check-alive?
(525)enable-check-idle?
(526)check-pool-interval?

(519) <jeus-system> <resource> <external-source> <tmax> <monitoring> **<enable-check-alive>**

Description ConnectionPool service 들의 connection 여부를 주기적으로 check 할지 여부를 설정한다.

Value Description boolean

Value Type boolean

Default Value false

Defined Value true

ConnectionPool service 중에 connection alive 상태를 check 를 한다.

false

ConnectionPool service 중에 connection alive 상태를 check 를 하지 않는다.

Example

```
<enable-check-alive>true</enable-check-alive>
```

(520) <jeus-system> <resource> <external-source> <tmax> <monitoring> **<enable-check-idle>**

Description ConnectionPool service 들의 connection 의 idle 상태 여부를 주기적으로 check 할지 여부를 설정한다.

Value Description boolean

Value Type boolean

Default Value false

Defined Value true

ConnectionPool service 중에 connection idle 상태를 check 를 한다.

false

ConnectionPool service 중에 connection idle 상태를 check 를 하지 않는다.

Example

```
<enable-check-idle>true</enable-check-idle>
```

(521) <jeus-system> <resource> <external-source> <tmax> <monitoring> **<check-pool-interval>**

Description ConnectionPool service 들의 connection 여부 check 주기값을 설정한다.

Value Description milliseconds

Value Type int

Default Value 300000

Defined Value -1

WebT connections 에 대하여 주기적으로 검사하지 않는다.

Example

```
<check-pool-interval>60000</check-pool-interval>
```

[Performace Recommendation]: benchmark 목적인 Test 인 경우는 "-1"옵션값을 쓰지 않는다. Perfomance 성능면을 향상시키는 것이 주안점을 둔다면 되도록 검사 주기 시간을 크게 잡아준다.

(522) <jeus-system> <resource> <external-source> <tmax> **<tmax-property>**

Description 이 element 는 일반적은 속성 이외의 사용자가 Tmax 와의 통신 시 추가로 정의하고자 하는 속성 값을 정의한다.

Example

```
<tmax-property> ... </tmax-property>
```

Child Elements (528)name

(529)value

(523) <jeus-system> <resource> <external-source> <tmax> <tmax-property> **<name>**

Description WebT Service 에서 사용자 속성을 추가하고자 할때의 Key 값.

Value Type token

Example

```
<name>eventHandler</name>
```

(524) <jeus-system> <resource> <external-source> <tmax> <tmax-property> **<value>**

Description WebT Service 에서 사용자 속성을 추가하고자 할때의 Value 값.

Value Type token

Example

```
<value>GenericServlet</value>
```

B WebT API Reference

B.1 소개

WebT Library 에서 제공하는 API 에 대해서 살펴보도록 한다.

B.2 패키지 [tmax.webt]

class WebtAttribute

public abstract class WebtAttribute extends java.lang.Object

Tmax server 로 WebtRemoteService 의 tpcall, tpacall, tpgetrply 등 tp 함수를 호출할 때 사용할 flag 값 정보를 가지고 있는 클래스이다.

TPNOBLOCK, TPNOREPLY, TPNOTRAN, TPNOTIME, TPGETANY, TPNOCHANGE,TPBLOCK 등 여기에서 사용하는 각 속성들의 자세한 사항은 Tmax Application Developers Guide 를 참조하기 바란다.

Fields

```
public static final int TPBLOCK
```

Description

- 이 속성은 WebtRemoteService.tpacall method 호출에 사용된다.

- 이 flag 는 요청을 보내고 서비스나 서버의 이상 유무에 대한 응답을 받겠다는 것이다. 이 flag 가 설정되고 tpacall()호출이 실패하였다면 WebtServiceException 혹은 WebtIOException 이 throw 된다.

```
public static final int TPGETANY
```

Description

- WebtRemoteService.tpgetrply method 호출에 사용되는 속성이다.
- 이 속성이 설정되어 있고 tpgetrply 함수 호출시 cd 값에 일치하는 WebtBuffer 가 아직 수신되지 않았다면 tpgetrply 의 인자로 전달되는 구별자 값은 무시되고 먼저 수신한 WebtBuffer 를 리턴하게 된다.

```
public static final int TPNOBLOCK
```

Description

- 이 플래그가 설정되고 tp 함수를 호출하였을 때 네트워크에 블럭킹 상태가 발생하면 즉시 Exception(WebtServiceException, tperrno=TPETIME)을 발생한다.
- 이 플래그는 WebtRemoteService.tpgetrply method 에서 주로 사용된다.

```
public static final int TPNOCHANGE
```

Description

- WebtRemoteService 의 tpcall/tpgetrply method 에서 사용한다.
- 이 속성이 설정되면 tpcall/tpgetrply 호출시 지정한 수신 버퍼 타입과 실제 수신한 WebtBuffer 의 타입이 다를 경우 WebtServiceException(tperrno=TPEOTYPE)을 throw 한다.

```
public static final int TPNOREPLY
```

Description

- 이 플래그는 요청을 보내고 응답을 받지 않겠다는 것이다.
- 이 플래그가 설정되고 WebtRemoteService.tpacall()이 성공적으로 완료 되었다면 구별자(리턴값)로 zero 가 반환된다. 이 구별자(cd)는 WebtRemoteService.tpgetrply()에서 사용할 수 없다.
- WebtRemoteService.tpacall()이 Transaction 상태일 때는 응답을 반드시 받아야 함으로 이 flag 를 사용할 수 없다. 만약 사용하려면 flag 를 TPNOTRAN 과 함께 사용하여 Transaction 참여의 일부분에서 벗어나야 한다.

```
public static final int TPNOTIME
```

Description

- 플래그를 TPNOTIME 으로 설정하면 tmax server 설정 파일에 설정된 BLOCKTIME 값이나 WebtConnection.setTPtimeout(int) 에 의해 설정된 blocking timeout 값이 무시되고 무한대가 된다. 즉, 응답이 올때 까지 무조건 기다린다.
- 만약 transaction boundary 내에서의 tpcall/tpacall/tpgetrply 에 이 속성을 설정한다면 Transaction timeout 시간이 적용되고 이 플래그는 무시된다.

```
public static final int TPNOTRAN
```

Description

- WebtRemoteService 의 tpcall(), tpacall()함수에서 사용한다.
- 이 플래그는 WentTransaction 에 의한 transaction boundary 안에서의 tpcall/tpacall 이라 하더라도 이 속성이 설정되어 있으면 transaction boundary 에서 제외된다.

```
public static final int TPFUNC
```

Description

- 이 속성은 WebtRQService.tpenq() 혹은 WebtRQService_.tpdeq 함수 호출에 사용된다.
- 이 플래그는 서비스별 RQ Data 를 관리 할때 사용된다.

```
public static final int TPRQS
```

Description

- 이 속성은 WebtRQService.tpenq() 혹은 WebtRQService_.tpdeq 함수 호출에 사용된다.
- 이 플래그는 RQ 서비스 이용시 Request Queue 에 기록되고 Reply Queue 로 부터 Data 를 받겠다는 것이다.

Constructors

```
public WebtAttribute()
```

Description

- default constructor

Methods

```
public void clearAttribute()
```

Description

- 모든 속성의 설정을 해제한다.

```
public boolean getAttribute(int attr)
```

Description

- 특정 속성값의 설정 유무를 조사한다. attr 에 지정된 모든 속성값이 설정되었으면 true 를 반환하고 그렇지 않으면 false 를 반환한다.

Arguments

- int attr: TPNOBLOCK, TPNOREPLY, TPNOTRAN, TPNOTIME, TPGETANY, TPNOCHANGE, TPBLOCK 중 하나 또는 하나 이상의 값을 논리 연산 OR(|)한 값이 올 수 있다. 그 외의 값이면 무시한다.

Return value

- boolean : attr 에 지정된 모든 속성값이 설정되었으면 true 를 반환하고 그렇지 않으면 false 를 반환한다.

```
public boolean isTPBLOCK()
```

Description

- TPBLOCK 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPBLOCK 속성이 설정되어 있는 것이고 false 이면 TPBLOCK 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPFUNC()
```

Description

- TPFUNC 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPFUNC 속성이 설정되어 있는 것이고 false 이면 TPFUNC 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPGETANY()
```

Description

- TPGETANY 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPGETANY 속성이 설정되어 있는 것이고 false 이면 TPGETANY 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPNOBLOCK()
```

Description

- TPNOBLOCK 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPNOBLOCK 속성이 설정되어 있는 것이고 false 이면 TPNOBLOCK 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPNOCHANGE()
```

Description

- TPNOCHANGE 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPNOCHANGE 속성이 설정되어 있는 것이고 false 이면 TPNOCHANGE 속성이 설정 해제되어 있는 것이다.

```
public Boolean isTPNOREPLY()
```

Description

- TPNOREPLY 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPNOREPLY 속성이 설정되어 있는 것이고 false 이면 TPNOTRAN 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPNOTIME()
```

Description

- TPNOTIME 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPNOTIME 속성이 설정되어 있는 것이고 false 이면 TPNOTIME 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPNOTRAN()
```

Description

- TPNOTRAN 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPNOTRAN 속성이 설정되어 있는 것이고 false 이면 TPNOTRAN 속성이 설정 해제되어 있는 것이다.

```
public boolean isTPRQS()
```

Description

- TPRQS 속성의 설정 유무를 조사한다.

Return value

- boolean : true 이면 TPRQS 속성이 설정되어 있는 것이고 false 이면 TPRQS 속성이 설정 해제되어 있는 것이다.

`public void` **`setAttribute(int attr, boolean value)`**

Description

- 특정 속성값을 설정하거나 설정을 해제한다.

Arguments

- int attr: TPNOBLOCK, TPNOREPLY, TPNOTRAN, TPNOTIME, TPGETANY, TPNOCHANGE, TPBLOCK 중 하나 또는 하나 이상의 값을 논리 연산 OR(|)한 값이 올 수 있다. 그 외의 값이면 무시한다. (예 : setAttribute(WebtAttribute.TPNOREPLY | WebtAttribute.TPNOTRAN, true);)
- boolean value : true 이면 해당 속성을 설정하고 false 이면 해당 속성을 설정 해제한다.

`public void` **`setTPBLOCK(boolean value)`**

Description

- TPBLOCK 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPBLOCK 속성을 설정하고 false 이면 TPBLOCK 속성 설정을 해제한다

`public void` **`setTPFUNC(boolean value)`**

Description

- TPFUNC 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPFUNC 속성을 설정하고 false 이면 TPFUNC 속성 설정을 해제한다.

```
public void setTPGETANY(boolean value)
```

Description

- TPGETANY 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPGETANY 속성을 설정하고 false 이면 TPGETANY 속성 설정을 해제한다.

```
public void setTPNOBLOCK(boolean value)
```

Description

- TPNOBLOCK 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPNOBLOCK 속성을 설정하고 false 이면 TPNOBLOCK 속성 설정을 해제한다.

```
public void setTPNOCHANGE(boolean value)
```

Description

- TPNOCHANGE 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPNOCHANGE 속성을 설정하고 false 이면 TPNOCHANGE 속성 설정을 해제한다.

```
public void setTPNOREPLY(boolean value)
```

Description

- TPNOREPLY 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPNOREPLY 속성을 설정하고 false 이면 TPNOREPLY 속성 설정을 해제한다.

```
public void setTPNOTIME(boolean value)
```

Description

- TPNOTIME 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPNOTIME 속성을 설정하고 false 이면 TPNOTIME 속성 설정을 해제한다.

```
public void setTPNOTRAN(boolean value)
```

Description

- TPNOTRAN 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPNOTRAN 속성을 설정하고 false 이면 TPNOTRAN 속성 설정을 해제한다.

```
public void setTPRQS(boolean value)
```

Description

- TPRQS 속성을 설정 또는 해제한다.

Arguments

- boolean value: true 이면 TPRQS 속성을 설정하고 false 이면 TPRQS 속성 설정을 해제한다.

```
public java.lang.String toString()
```

Description

- 이 객체를 문자열로 표현한 값을 반환한다.

Return value

- string : 이 객체를 문자열로 표현한 값.

abstract class WebtBuffer

public abstract class WebtBuffer extends java.lang.Object implements java.io.Serializable

WebtBuffer 클래스는 WebT 와 원격 Tmax Server 사이에 주고 받는 사용자 데이터를 관리하는 클래스이다. WebtRemoteService 의 tp 함수(tpcall, tpgetrply)의 반환 값과 WebtEventHander 의 handleEvent 의 인자로 이 클래스가 사용된다. 또한, tpcall, tpacall 함수 호출에 필요한 사용자 데이터는 WebtBuffer 에 저장된다. 이 클래스는 추상 클래스이므로 사용자가 직접 객체를 생성할 수 없다. tpcall, tpacall 의 입력 데이터에 사용할 WebtBuffer 를 얻기 위해서는 WebtRemoteService.createStringBuffer(), WebtRemoteService.createCarrayBuffer(), WebtRemoteService.createFieldBuffer()등을 사용한다.

```
WebtConnection connection = null;
 try {
    connection = new WebtConnection("localhost", 8888);
    WebtRemoteService service = new WebtRemoteService("TEST",
connection);
    WebtBuffer request = service.createStringBuffer();
    request.setString("test string");
    WebtBuffer response = service.tpcall(request);
    System.out.println("response : " + response.getString());
 } catch (WebtIOException ioe) {
```

```
    System.out.println("webt connection closed");
    ioe.printStackTrace();
 } catch (WebtServiceFailException sfe) {
   System.out.println("server returns TP_FAIL");
 } catch (WebtServiceException se) {
   System.out.println("service fail. tperrno = " +
se.getTPErrorMessage());
   se.printStackTrace();
 } finally {
   connection.close();
 }
```

Fields

`public static final int` **`BT_CARRAY`**

Description

- CARRAY 버퍼 타입. getBufferType()의 반환 값이 BT_CARRAY 인 경우 이다.

`public static final int` **`BT_FIELD`**

Description

- FIELD 버퍼 타입. getBufferType()의 반환 값이 BT_FIELD 인 경우 이다.

`public static final int` **`BT_STRING`**

Description

- STRING 버퍼 타입. getBufferType()의 반환 값이 BT_STRING 인 경우 이다

Constructors

```
public WebtBuffer()
```

Description

- default constructor

Methods

```
public abstract void clear()
```

Description

- 내부의 모든 WebtField 객체를 삭제한다

```
public static WebtBuffer createCarrayBuffer()
```

Description

- 버퍼 타입 BT_CARRAY 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다

Return value

- WebtBuffer: WebtBuffer 객체

```
public static WebtBuffer createCarrayBuffer(java.lang.String
charset)
```

Description

- 버퍼 타입 BT_CARRAY 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다

Arguments

- String charset : 생성될 buffer 의 default charset 을 설정한다.

Return value

- WebtBuffer: WebtBuffer 객체

```
public static WebtField createField(int fieldkey) throws
WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 생성하여 반환한다
- fieldkey 를 field key 값으로 하는 WebtField 객체가 이미 생성되었으면 이미 생성된 객체를 리턴 한다.

Arguments

- int fieldkey : 생성할 WebtField 객체의 field key 값.

Return value

- WebtField: fieldkey 를 field key 값으로하는 WebtField 객체

Throws

- WebtBufferException : fieldkey 가 유효한 값이 아닌 경우 발생한다

```
public static WebtField createField(java.lang.String fieldkey)
throws WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 생성하여 반환한다
- fieldkey 를 field key 값으로 하는 WebtField 객체가 이미 생성되었으면 이미 생성된 객체를 리턴 한다.
- WebtSystem.createDefaultFieldKeyTable(String) 을 이용하여 사용할 field key table 이 미리 로딩되어 있어야 한다

Arguments

- int fieldkey : 생성할 WebtField 객체의 field key 값.

Return value

- WebtField: fieldkey 를 field key 값으로하는 WebtField 객체

Throws

- WebtBufferException : fieldkey 가 유효한 값이 아닌 경우 발생한다

```
public static WebtBuffer createFieldBuffer()
```

Description

- 버퍼 타입이 BT_FIELD 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다.

Return value

- WebtBuffer: WebtBuffer 객체

```
public static WebtBuffer createFieldBuffer(java.lang.String
charset)
```

Description

- 버퍼 타입이 BT_FIELD 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다.

Arguments

- String charset : 생성될 buffer 의 default charset 을 설정한다.

Return value

- WebtBuffer: WebtBuffer 객체

```
public static WebtBuffer createStringBuffer()
```

Description

- 버퍼 타입이 BT_STRING 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다.

Return value

- WebtBuffer: WebtBuffer 객체

```
public static WebtBuffer createStringBuffer(java.lang.String
charset)
```

Description

- 버퍼 타입이 BT_STRING 인 WebtBuffer 객체를 생성한다.
- charset 은 WebtSystem.getDefaultCharset())를 사용한다.

Arguments

- String charset : 생성될 buffer 의 default charset 을 설정한다.

Return value

- WebtBuffer: WebtBuffer 객체

```
public int getBufferType()
```

Description

- 버퍼 타입을 return 한다.

Return value

- int: BT_STRING, BT_CARRAY, BT_FIELD 중 현재 설정된 값을 리턴 한다.

```
public abstract byte[] getBytes() throws WebtBufferException
```

Description

- 버퍼타입이 BT_CARRAY, BT_STRING 일때 데이터 시작부분에서 끝까지의 내용을 반환한다.
- getBytes(false)와 동일하다

Throws

- WebtBufferException : getBufferType()이 BT_CARRAY, BT_STRING 이 아닌 경우에 발생 한다.

```
public abstract byte[] getBytes(boolean doCopy) throws
WebtBufferException
```

Description

- 버퍼타입이 BT_CARRAY 일때 데이터 시작부분에서 끝까지의 내용을 반환한다.
- doCopy 가 true 인 경우 내용을 복사한 새 byte array 를 리턴한다.

Throws

- WebtBufferException : getBufferType()이 BT_CARRAY,이 아닌 경우에 발생 한다.

```
public abstract byte[] getBytes(int offset, int length) throws
WebtBufferException
```

Description

- 버퍼타입이 BT_STRING, BT_CARRAY 일때 데이타의 offset 시작번지로 부터 length 만큼의 읽어온 내용을 byte[] type 의 값으로 반환한다

Arguments

- int offset : 버퍼에서 가져올 시작번지.
- int length : 시작번지로 부터 읽어올 데이타의 길이

Throws

- WebtBufferException : getBufferType()의 반환값이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생 한다.

```
public int getCallDescripter()
```

Description

- 이 WebtBuffer 의 call descriptor 값을 반환한다.

- call descriptor 는 WebtRemoteService.tpacall 함수가 리턴하는 값으로서 tpgetrply 시에 인자로 사용되는 값이다. 0 이상이 유효한 값이다

Return value

- int: call descriptor 값.

```
public int getDataLength()
```

Description

- byte stream 으로 변환된 데이터의 길이를 리턴 한다.

```
public java.lang.String getDefaultCharset()
```

Description

- 이 버퍼에서 사용할 기본 문자셋을 리턴 한다.
- 리턴값이 null 이면 system default character set 을 사용하여 문자셋 변환을 한다는 것을 의미한다.

Return value

- String : 기본 문자셋.

```
public abstract WebtField getField(int fieldkey) throws
WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 반환한다. 존재하지 않으면 null 을 리턴 한다.

Arguments

- int fieldkey : WebtField 객체의 field key 값.

Return value

- WebtField: fieldkey 를 field key 값으로하는 WebtField 객체. 존재하지 않으면 null 을 반환한다

Throws

- WebtBufferException : fieldkey 가 유효한 값이 아닌 경우 발생한다

```
public abstract WebtField getField(java.lang.String fieldkey)
throws WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 반환한다. 존재하지 않으면 null 을 리턴 한다.

Arguments

- int fieldkey : WebtField 객체의 field key 값.

Return value

- WebtField: fieldkey 를 field key 값으로하는 WebtField 객체. 존재하지 않으면 null 을 반환한다

Throws

- WebtBufferException : fieldkey 가 유효한 값이 아닌 경우 발생한다

```
public abstract java.util.Vector getFieldKeyNames() throws
WebtBufferException
```

Description

- String 으로 표현된 field key list 를 리턴 한다.
- WebtSystem.createDefaultFieldKeyTable(String) 에 의해 생성된 field key table 에서 찾는다

Return value

- Vector : String 으로 표현된 field key list 의 Vector

```
public abstract java.util.Vector getFields() throws
WebtBufferException
```

Description

- 생성된 순서대로 정렬한 WebtField 객체의 Vector 를 리턴 한다.

Return value

- Vector : WebtField 객체의 Vector

```
public java.lang.String getServiceName()
```

Description

- 현재 설정된 service name 을 리턴 한다.

Return value

- String : service name

```
public java.lang.String getSring()throws WebtBufferException
```

Description

- 버퍼타입이 BT_STRING, BT_CARRAY 일때 데이터의 시작부분에서 0x00 전까지의 내용을 String 으로 변환하여 리턴 한다.
- 0x00 이 byte array 에 없을 경우 데이터 전체를 String 으로 변환하여 리턴 한다.
- String 으로 변환할 때 인코딩은 getDefaultChharset()이 반환한 문자셋을 사용 한다.

Throws

- WebtBufferException : getBufferType()이 BT_STRING, BT_CARRAY 가 아니거나, getDefaultCharset()이 시스템에서 지원하지 않는 문자셋인 경우 발생한다

```
public abstract java.lang.String getSring(int offset, int
length)throws WebtBufferException
```

Description

- 버퍼타입이 BT_STRING, BT_CARRAY 일 때 데이타의 offset 시작 번지로부터 length 만큼의 읽어온 내용을 String type 의 값으로 반환한다.

Arguments

- int fieldkey : WebtField 객체의 field key 값.

Throws

- WebtBufferException : getBufferType()이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생한다

```
public abstract java.lang.String getSring(java.lang.String
charset)throws WebtBufferException
```

Description

- 버퍼타입이 BT_STRING, BT_CARRAY 일때 데이터 시작부분에서 0x00 전 까지의 내용을 String 으로 변환하여 리턴 한다.
- 0x00 이 byte array 에 없을 경우 데이터 전체를 String 으로 변환하여 리턴 한다.

Arguments

- String charset : byte array 를 String 을 변환할 때 사용할 character set. null 인 경우 getDefaultCharset()이 반환한 값을 사용 한다.

Throws

- WebtBufferException : getBufferType()이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생한다

```
public int getUserReturnCode()
```

Description

- Tmax 서비스 루틴에서 tpreturn 시 urcode 로 설정한 값을 리턴한다.

Return value

- int : tpurcode

```
public WebtConnection getUsedConnection()
```

Description

- 이용하고 있는 연결을 반환한다.

Return value

- WebtConnection : 현재 buffer 에 설정되어 있는 연결을 반환 한다.

```
public abstract WebtField removeField() throws WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 찾아 삭제 한다.

Return value

- WebtField : 삭제된 WebtField 객체. 없으면 null 을 리턴한다

```
public abstract WebtField removeField(java.lang.String
fieldkey)throws WebtBufferException
```

Description

- fieldkey 를 field key 값으로 하는 WebtField 객체를 찾아 삭제한다.

Return value

- WebtField : 삭제된 WebtField 객체. 없으면 null 을 리턴한다

Throws

- WebtBufferException : fieldkey 가 유효한 값이 아니거나 field key table 에서 찾지 못한 경우 발생한다

```
public abstract void setBytes(byte[] val)throws
WebtBufferException
```

Description

- 사용자 데이터를 설정한다.
- 버퍼타입이 BT_CARRAY 일때만 사용할 수 있다.그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다. setBytes(val, false) 와 동일하다.

Arguments

- byte[] val : 버퍼에 설정할 내용.

Throws

- WebtBufferException : 버퍼 타입이 BT_CARRAY 가 아닐때 발생한다.

```
public abstract void setBytes(byte[] val, boolean doCopy)throws
WebtBufferException
```

Description

- 사용자 데이터를 설정한다.
- 버퍼타입이 BT_CARRAY 일때만 사용할 수 있다.그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다. doCopy 가 true 일 경우 val 값과 동일한 byte array 를 내부에 생성하고 내용을 복사하여 저장한다.

Arguments

- byte[] val : 버퍼에 설정할 내용.
- boolean doCopy : 내용을 복사할 것인지 여부

Throws

- WebtBufferException : 버퍼 타입이 BT_CARRAY 가 아닐때 발생한다

```
public abstract int setBytes(byte[] val, int off, int len)throws
WebtBufferException
```

Description

- value 의 offset 내용을 복사하여 저장하고 입력된 데이타의 length 값을 return 한다.
- 버퍼타입이 BT_STRING, BT_CARRAY 일때 사용할 수 있다. 그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다.

Arguments

- byte[] val : 저장 데이터.
- off : 읽어들일 시작 번지
- len : 읽어들일 데이터 길이.

Return value

- int : 읽어 들인 데이터 길이값

Throws

- WebtBufferException : getBufferType()의 반환값이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생한다

```
public void setDefaultCharset(java.lang.String charset)
```

Description

- 이 버퍼에서 사용할 문자셋을 지정한다.
- buffer operation 하기 전에 setting 해야 이 charset 이 적용된다. 지정하지 않으면 WebtRemoteService.getDefaultCharset()의 값이 적용된다.

```
public abstract void setString(int offset, java.lang.String
value)throws WebtBufferException
```

Description

- value 의 offset 내용을 복사하여 저장하고 입력된 데이타의 length 값을 return 한다.
- 버퍼타입이 BT_STRING, BT_CARRAY 일때 사용할 수 있다. 그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다.

Arguments

- int offset : 읽어들일 시작 번지.
- String value : 저장 데이터.

Throws

- WebtBufferException : getBufferType()의 반환값이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생한다

```
public abstract void setString(java.lang.String value)throws
WebtBufferException
```

Description

- 사용자 데이터를 설정한다.
- 버퍼타입이 BT_STRING, BT_CARRAY 일때만 사용할 수 있으며 그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다.
- byte array 로 변환된 데이터의 끝에는 0x00 이 추가된다. 문자셋 변환에 사용할 문자셋 값은 getDefaultCharacterSet()가 반환한 값을 사용한다.

Arguments

- String value : 버퍼에 설정할 내용.

Throws

- WebtBufferException : getBufferType()의 반환값이 BT_STRING, BT_CARRAY 가 아니거나, getDefaultCharacterSet()의 반환값이 시스템에서 지원하지 않은 문자셋인 경우 발생한다

```
public abstract void setString(java.lang.String val,
java.lang.String charset)throws WebtBufferException
```

Description

- 사용자 데이터를 설정한다.
- 버퍼타입이 BT_STRING, BT_CARRAY 일때만 사용할 수 있으며 그렇지 않으면 WebtBufferException (tperrno = TPEINVAL)이 throw 된다
- . byte array 로 변환된 데이터의 끝에는 0x00 이 추가된다.

Arguments

- String val : 버퍼에 설정할 내용.
- String charset : val 를 바이트 열로 변환할 때 사용할 character set. null 인 경우 getDefaultCharacterSet()가 반환한 값을 사용한다.

Throws

- WebtBufferException : getBufferType()의 반환값이 BT_STRING, BT_CARRAY 가 아니거나, charset 이 시스템에서 지원하지 않은 문자셋인 경우 발생한다.

class WebtConnection

public class WebtConnection extends java.lang.Object

WebtConnection class 는 Tmax Server 와 WebT 사이의 communication 을 담당하는 클래스이다. Tmax Server 가 제공하는 service 를 제공받으려면 반드시 이 WebtConnection 을 통한 네트워크 연결을 통해 서비스 요청을 해야한다

Constructors

```
public WebtConnection(java.lang.String hostAddr, int hostPort)
throws WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다

Arguments

- String hostAddr : tmax 서버 주소
- int hostPort : tmax 서버 listen port 번호

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다

- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort,
boolean doConnect) throws WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- doConnect 가 false 이면 연결을 시도하지 않는다

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- doConnect : true 이면 연결을 시도한다.

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort,
boolean doConnect, int conntimeout) throws WebtIOExcetpion,
WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- doConnect 가 false 이면 연결을 시도하지 않는다.
- conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- boolean doConnect : true 이면 연결을 시도한다.
- int conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하거나 conntimeout 내에 연결을 설정하지 못한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort, int
conntimeout) throws WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- int conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하거나 conntimeout 내에 연결을 설정하지 못한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort,
java.lang.String backupAddr, int backupPort) throws
WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- hostAddr:hostPort 로 연결을 설정하지 못하였을 경우 backupAddr:backupPort 로 연결을 시도한다.

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- String backupAddr : tmax backup 서버 주소
- String backupPort : tmax backup 서버 listen port 번호

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하여 hostAddr:hostPort 및 backupAddr:backupPort 로 연결을 설정하지 못한 경우 이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort,
java.lang.String backupAddr, int backupPort, boolean doConnect,
int conntimeout) throws WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- hostAddr:hostPort 로 연결을 설정하지 못하였을 경우 backupAddr:backupPort 로 연결을 시도한다.
- doConnect 가 false 이면 연결을 시도하지 않는다. conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- String backupAddr : tmax backup 서버 주소
- String backupPort : tmax backup 서버 listen port 번호
- boolean doConnect : true 이면 연결을 시도한다.
- int conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하여 hostAddr:hostPort 및 backupAddr:backupPort 로 연결을 설정하지 못하거나 conntimeout 내에 연결을 설정하지 못한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(java.lang.String hostAddr, int hostPort,
java.lang.String username, java.lang.String userpwd,
java.lang.String domainname, java.lang.String domainpwd) throws
WebtIOExcetpion, WebtServiceException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection 을 생성한다.
- 연결 설정시 인자로 주어진 인증 정보를 사용하여 인증절차를 수행한다.
- Tmax server 에 등록된 username, userpwd, domainname, domainpwd 값들이 일치하였을 때 연결이 성공적으로 이루어진다.

Arguments

- String hostAddr : tmax 서버 주소
- String hostPort : tmax 서버 listen port 번호
- String username : 사용자 인증 보안에 대한 계정이름

- String userpwd : 사용자 인증 보안에 대한 암호
- String domainname : Client 이름.
- String domainpwd : 시스템 접속 보안에 대한 암호

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public WebtConnection(WebtConnectionInfo info) throws
WebtIOExcetpion, WebtServiceException
```

Description

- webt.properties 로 부터 설정된 값으로 tmax server 로 연결하기 위해 WebtConnection 을 생성한다.

Arguments

- WebtConnectionInfo info : WebtConnectionInfo 객체

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

Methods

```
public boolean close()
```

Description

- tmax 와의 socket 연결을 종료 한다.

- 만약 이 WebtConnection 이 WebtConnectionPool 이 관리하는 connection 이라면 연결을 종료하지 않고 pool 에 반납하여 재사용하게 한다

```
public void connect() throws WebtIOExcetpion, WebtServiceException
```

Description

- Tmax Server 로의 연결이 맺어져 있지 않으면 연결을 시도한다

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다
- WebtServiceException : network I/O 나 transaction 이외의 service 오류가 발생한 경우이다.

```
public boolean chekcConnection(int timeout)
```

Description

- 클라이언트 연결상태를 확인 하는 함수이다.

Arguments

- int timeout: timeout < 0 tpstarted 수행 상태 체크
  timeout = 0 socket 상태 체크
  timeout > 0 Tmax Server 로 메시지 요청 후 주어진 시간 동안 응답이 없으면 network 오류로 간주한다

Return value

- boolean : timeout 설정에 따는 요청 결과 상태

```
public java.lang.String getAddress()
```

Description

- main tmax server 의 주소를 반환한다

Return value

- String : main tmax server 의 주소

```
public int getPort()
```

Description

- main tmax server 의 listen port 번호를 반환한다

Return value

- int : main tmax server 의 포트 번호. 설정되어 있지 않으면 -1 을 반환한다

```
public java.lang.String getBackupAddress()
```

Description

- backup tmax server 의 주소를 반환한다

Return value

- String : backup tmax server 의 주소. 설정되어 있지 않으면 null 을 반환한다

```
public int getBackupPort()
```

Description

- backup tmax server 의 listen port 번호를 반환한다

Return value

- int : backup tmax server 의 포트 번호. 설정되어 있지 않으면 -1 을 반환한다

```
public java.lang.String getConnectionId()
```

Description

- 이 WebtConnection 의 connection ID 를 반환한다.

Return value

- String : connection ID

```
public int getConnectTimeout()
```

Description

- tmax 서버와의 연결을 시도할 때 적용할 timeout 값을 반환한다.

Return value

- int : connect timeout 값(초단위)

```
public java.lang.String getDomainName()
```

Description

- main tmax server 의 접속된 Client 이름을 반환한다.

Return value

- String : main tmax server 에 접속된 Client 이름. 설정되어 있지 않으면 null 을 반환한다.

```
public java.lang.String getGroupName()
```

Description

- 이 WebtConnection 이 속한 group name 을 반환한다.

Return value

- String : group name

```
public int getTPtimeout()
```

Description

- 현재 설정된 블로킹 타임 아웃 값을 반환한다.

Return value

- int : 블로킹 타임 아웃 값(초단위)

```
public WebtTransaction getTransaction()
```

Description

- 이 WebtConnection 이 사용하는 WebtTransaction 객체를 반환한다.

Return value

- WebtTransaction : WebtTransaction 객체. transaction 이 설정되어 있지 않으면 null 을 리턴한다.

```
public java.lang.String getUserName()
```

Description

- main tmax server 의 사용자 인증 보안에 대한 계정 이름을 반환한다.

Return value

- 사용자 이름. 설정되어 있지 않으면 null 을 반환한다.

```
public boolean isAlive()
```

Description

- connection alive check.

```
public void setConnectTimeout(int t)
```

Description

- Tmax Server 로의 연결을 시도할 때 적용할 타임 아웃 값을 설정한다.
- 연결을 시도하기 전에 이 함수를 사용하여야 유효하다.

Arguments

- int t : connect timeout 값(초단위).

```
public void setTPtimeout(int sec)
```

Description

- 블로킹 타임아웃 시간을 설정한다.
- 이 함수를 호출하지 않으면 Tmax 시스템에 설정된 블로킹 타임아웃 값(Tmax 환경파일의 *DOMAIN 절에 설정하는 BLOCKTIME 항목에 의해 설정된다)이 적용된다.
- 클라이언트는 서비스를 요청한 후 그에 대한 응답을 일정한 시간 동안 기다리게 된다. 그 시간안에 응답이 도착하면 이를 정상적으로 수신하며, 만약 이 시간이 지날때까지 응답이 도착하지 않으면 클라이언트는 응답을 기다리는 일을 중단하고 WebtServiceException (tperrno = TPETIME)을 throw 한다. 그 후에 응답이 도착하면 이를 무시한다. 이 시간이 블로킹 타임 아웃 시간이다.
- 이 값은 다시 이 함수가 호출되거나 클라이언트가 종료할 때까지 유효하다.

Arguments

- int sec : 변경할 타임아웃 시간. (초단위).

class WebtConnectionGroup

public abstract class WebtConnectionGroup extends java.lang.Object

WebtConnection 에 대한 pooling 기능을 담당하는 추상 클래스이다. 같은 주소/포트 번호를 갖는 다수의 WebtConnection 객체의 pooling 을 관리한다. 사용자가 직접 객체를 생성할 수는 없다. WebtConnectionPool 의 createGroup method 를 사용하여 객체를 생성할 수 있다

Constructors

```
public WebtConnectionGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, int initial, int max, int
inc, int idle)
```

Description

- 고유의 groupname 으로 hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection group 을 생성한다

Arguments

- String groupname : connection group 의 고유 이름
- String hostAddr : tmax 서버 주소
- int hostPort : tmax 서버 listen port 번호
- int initial : 초기 연결 개수
- int max : 최대 연결 개수
- int inc : 연결 개수 증가치
- int idle : 최대 idle 시간

```
public WebtConnectionGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, int initial, int max, int inc, int
idle)
```

Description

- 고유의 groupname 으로 hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection group 을 생성한다

Arguments

- String groupname : connection group 의 고유 이름
- String hostAddr : tmax 서버 주소
- int hostPort : tmax 서버 listen port 번호
- String backupAddr : tmax backup 서버 주소
- int backupPort : tmax backup 서버 listen port 번호
- int initial : 초기 연결 개수
- int max : 최대 연결 개수
- int inc : 연결 개수 증가치
- int idle : 최대 idle 시간

```
public WebtConnectionGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, java.lang.String userName,
java.lang.String userPasswd, java.lang.String domainName,
java.lang.String domainPasswd,int initial, int max, int inc, int
idle)
```

Description

- 고유의 groupname 으로 hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtConnection group 을 생성한다

Arguments

- String groupname : connection group 의 고유 이름

- String hostAddr : tmax 서버 주소
- int hostPort : tmax 서버 listen port 번호
- String backupAddr : tmax backup 서버 주소
- int backupPort : tmax backup 서버 listen port 번호
- String userName : 사용자 이름

- String userPasswd : 사용자 password
- String domainName : domain 이름
- String domainPasswd : domain password
- int initial : 초기 연결 개수
- int max : 최대 연결 개수
- int inc : 연결 개수 증가치
- int idle : 최대 idle 시간

Methods

```
public abstract int available()
```

Description

- 사용 가능한 WebtConnection 갯수를 리턴한다

```
public void destroy()
```

Description

- group 에 속한 모든 WebtConnection 들의 연결을 종료한다. WentConnectionPool.destroy()에 의해 호출된다

```
public WebtConnection getConnection() throws WebtException
```

Description

- WebtConnectionPool.getConnection(String)을 사용한다.

```
public WebtConnection getConnection(long timeout) throws
WebtIOException, WebtServiceException
```

Description

- WebtConnectionPool.getConnection(String, long)을 사용한다.
- 가용한 WebtConnection 을 리턴한다..

Arguments

- long timeout : timeout_mili - timeout 값이다.(밀리초 단위)
- timeout < 0 : 가용한 WebtConnection 이 없는 경우 즉시 리턴(null)한다
- timeout == 0 : 가용한 WebtConnection 이 있을 때 까지 무한히 기다린다
- timeout > 0 : timeout_mili 밀리초 동안 가용한 WebtConnection 이 없으면 null 을 리턴한다.

Return value

- WebtConnection : 지정된 시간내에 가용한 WebtConnection 이 있을 경우 해당 WebtConnection 을 리턴하고 그렇지 않을 경우 null 을 리턴한다

Throws

- WebtIOException WebtConnection 을 얻어오려고 할 때 네트워크 오류가 발생한 경우이다.
- WebtServiceException : available 한 WebtConnection 이 없거나, 활성화되어 있지 않거나, timeout 되면 WebtException 을 throw 한다.

```
public java.lang.String getName()
```

Description

- 이 WebtConnectionGroup 의 group name 을 반환한다.

Return value

- String : group name

`public java.lang.String` **`getType()`**

Description

- 이 WebtConnectionGroup 의 group type 을 반환한다.

Return value

- String : group type 정보

`public abstract boolean` **`isSharedType()`**

Description

- 이 connection group 의 type 을 리턴한다.

Return value

- boolean : shared type 이면 true 를 dedicated type 이면 false 를 리턴한다

`public abstract void` **`putConnection()`**

Description

- 이 connection group 의 type 을 리턴한다.

Return value

- boolean : shared type 이면 true 를 dedicated type 이면 false 를 리턴한다

`public abstract int` **`size()`**

Description

- 현재까지 Tmax Server 와 연결된 WebtConnection 수를 리턴한다.

class WebtConnectionInfo

public class WebtConnectionInfo extends java.lang.Object

WebtConnectionInfo 는 WebtConnection 생성에 필요한 파라메터를 관리하는 클래스이다. 이 클래스의 객체를 생성하여 연결에 필요한 파라메터를 설정하면 WebtConnection 의 생성자의 인자로 사용할 수 있다

```
WebtConnectionInfo info = new WebtConnectionInfo();
info.setAttribute(info.TMAX_ADDRESS, "127.0.0.1");
info.setAttribute(info.TMAX_PORT, 8888);
info.setAttribute(info.CONNECT_TIMEOUT, 30);
WebtConnection connection = new WebtConnection(info);
```

Fields

`public static final java.lang.String` **`BACKUP_TMAX_ADDRESS`**

Description

- backup tmax server 의 주소를 설정할 파라메터 이름

`public static final java.lang.String` **`BACKUP_TMAX_PORT`**

Description

- backup tmax server 의 포트 번호를 설정할 파라메터 이름

```
public static final java.lang.String CONNECT_TIMEOUT
```

Description

- 연결 설정시 적용할 connect timeout 값을 설정할 파라메터 이름

```
public static final java.lang.String DOMAIN_NAME
```

Description

- 인증 절차에 필요한 domain name 을 설정할 파라메터 이름

```
public static final java.lang.String DOMAIN_PASSWORD
```

Description

- 인증 절차에 필요한 domain password 를 설정할 파라메터 이름

```
public static final java.lang.String TMAX_ADDRESS
```

Description

- 연결할 tmax server 의 주소를 설정할 파라메터 이름

```
public static final java.lang.String TMAX_PORT
```

Description

- 연결할 tmax server 의 포트 번호를 설정할 파라메터 이름

```
public static final java.lang.String TPTIMEOUT
```

Description

- 연결 설정 후 기본값으로 설정할 TP TIMEOUT 값을 설정할 파라메터 이름

```
public static final java.lang.String TXTIMEOUT
```

Description

- 연결 설정 후 기본값으로 설정할 TX TIMEOUT 값을 설정할 파라메터 이름

```
public static final java.lang.String USER_NAME
```

Description

- 인증 절차에 필요한 user name 을 설정할 파라메터 이름

```
public static final java.lang.String USER_PASSWORD
```

Description

- 인증 절차에 필요한 user password 를 설정할 파라메터 이름

Constructors

```
public WebtConnectionInfo()
```

Description

- default constructor

Methods

```
public java.lang.String getAttribute(java.lang.String name)
```

Description

- WebtConnectionInfo 객체에서 정의된 해당 속성의 값을 리턴한다

Arguments

- String name : 위에서 정의된 얻고자 하는 속성이름

Return value

- String : 해당 속성의 값. 속성이 설정되어 있지 않으면 null 을 리턴한다

```
public void setAttribute(java.lang.String name, int value)
```

Description

- WebtConnectionInfo 에서 정의된 해당 속성의 값을 설정한다.

Arguments

- String name : 위에서 정의된 설정하고자 하는 속성이름.
- int value : 해당 속성에 적용하고자 하는 값

```
public void setAttribute(java.lang.String name, java.lang.String
value)
```

Description

- WebtConnectionInfo 에서 정의된 해당 속성의 값을 설정한다.

Arguments

- String name : 위에서 정의된 설정하고자 하는 속성이름.
- int value : 해당 속성에 적용하고자 하는 값

class WebtConnectionPool

public class WebtConnectionPool extends java.lang.Object

WebtConnectionPool 클래스는 WebtConnectionManager 를 대체하는 connection pool 클래스이다. WebtConnectionPool 은 tmax 서버로의 네트워크 연결인 WebtConnection 객체를 관리하는 pooling 기능을 수행한다. WebtConnectionPool 클래스를 이용하면 Tmax 서비스 요청시마다 매번

WebtConnection 객체를 새로이 생성하지 않고 이전에 사용한 WebtConnection 객체를 재사용할 수 있으므로 Tmax 서버로의 네트워크 연결을 설정/종료하는데 소비되는 자원 및 시간을 절약할 수 있다. WebtConnectionPool 은 WebtConnectionManager 에 비해 다음과 같은 기능이 추가되었다

- 하나 이상의 tmax 서버와의 connection pool 을 관리할 수 있다.
- connection pool monitoring 기능이 추가되었다. 지정한 시간동안 idle 인 연결을 해제하거나 접속이 종료된 연결을 찾아내어 새로 연결해 줄 수 있다

```
 ....
WebtConnectionPool.createGroup("tmax1", "192.168.128.1",
8888);
WebtConnectionPool.createGroup("tmax2", "192.168.128.2",
8888);
 ....

try {
    WebtConnection con =
WebtConnectionPool.getConnection("tmax1");
} catch (WebtIOException wioe) {
    System.out.println("fail to get WebtConnection of
tmax1");
    return;
}

WebtService service = new WebtService("MYSVC", con);
....
WebtConnectionPool.putConnection(con);
```

Constructors

```
public WebtConnectionPool()
```

Description

- default constructor

Methods

```
public static int count()
```

Description

- 현재까지 생성된 WebtConnectionGroup 의 갯수를 리턴한다

Return value

- int : 현재까지 생성된 WebtConnectionGroup 의 갯수

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, int init, int max, int
inc) throws WebtException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 connection group 을 생성하지 않는다
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 갯수

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, int init, int max, int
inc, int idle) throws WebException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 connection group 을 생성하지 않는다
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호

- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, int init, int max, int inc) throws
WebException
```

Description

- Tmax Server(hostAddr:hostPort, backupAddr:backupPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다

Arguments

- 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 기존의 WebtConnectionGroup 을 대체한다. 기존 WebtConnectionGroup 은 destroy 된다.
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름

- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- String backupAddr : 연결을 맺을 Tmax Server 의 백업 호스트 이름
- int backupPort: 연결을 맺을 Tmax Server 의 백업 포트 번호
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 갯수

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, int init, int max, int inc, int idle)
throws WebException
```

Description

- Tmax Server(hostAddr:hostPort, backupAddr:backupPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 기존의 WebtConnectionGroup 을 대체한다. 기존 WebtConnectionGroup 은 destroy 된다.
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름

- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- String backupAddr : 연결을 맺을 Tmax Server 의 백업 호스트 이름
- int backupPort: 연결을 맺을 Tmax Server 의 백업 포트 번호
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다.

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, java.lang.String username,
java.lang.String userpwd, java.lang.String domainname,
java.lang.String domainpwd, int init, int max, int inc, int idle)
throws WebException
```

Description

- Tmax Server(hostAddr:hostPort, backupAddr:backupPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.

- username, userpwd, domainname, domainpwd 가 null 이 아니면 WebtConnection 을 생성할 때 보안 확인을 수행한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 기존의 WebtConnectionGroup 을 대체한다. 기존 WebtConnectionGroup 은 destroy 된다.
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- String backupAddr : 연결을 맺을 Tmax Server 의 백업 호스트 이름
- int backupPort: 연결을 맺을 Tmax Server 의 백업 포트 번호
- String username : user name
- String userpwd : user passward
- String domainname : domain name
- String domainpwd : domain passward
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다.

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String username,
java.lang.String userpwd, java.lang.String domainname,
```

```
java.lang.String domainpwd, int init, int max, int inc) throws
WebException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.
- username, userpwd, domainname, domainpwd 가 null 이 아니면 WebtConnection 을 생성할 때 보안 확인을 수행한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 connection group 을 생성하지 않는다.
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- String username : user name
- String userpwd : user passward
- String domainname : domain name
- String domainpwd : domain passward
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String username,
java.lang.String userpwd, java.lang.String domainname,
java.lang.String domainpwd, int init, int max, int inc, int idle)
throws WebException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.
- username, userpwd, domainname, domainpwd 가 null 이 아니면 WebtConnection 을 생성할 때 보안 확인을 수행한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 connection group 을 생성하지 않는다.
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- String username : user name
- String userpwd : user passward
- String domainname : domain name
- String domainpwd : domain passward
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다.

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void createGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, java.lang.String
backupAddr, int backupPort, java.lang.String username,
java.lang.String userpwd, java.lang.String domainname,
java.lang.String domainpwd, int init, int max, int inc, int idle,
int groupType, int connTimeout, int tpTimeout, int txTimeout, int
txBlockTimeout, Boolean enableEvent, int eventFlag) throws
WebException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.
- username, userpwd, domainname, domainpwd 가 null 이 아니면 WebtConnection 을 생성할 때 보안 확인을 수행한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 connection group 을 생성하지 않는다.
- String hostAddr : 연결을 맺을 Main Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Main Tmax Server 의 포트 번호
- String backupAddr : 연결을 맺을 Backup Tmax Server 의 호스트 이름

- int backupPort: 연결을 맺을 Backup Tmax Server 의 포트 번호
- String username : user name
- String userpwd : user passward
- String domainname : domain name
- String domainpwd : domain passward
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다.
- int groupType: group 의 이용 형태 (Class WebtSystem Field 참조)
- int connTimeout: 연결 설정 timeout(초단위)
- int tpTimeout: Tmax Server 로 service 요청 timeout
- int txTimeout: Tmax Server 로 Transaction service 요청 timeout (commit 이전)
- int txBlockTimeout: Tmax Server 로 commit 요청 timeout
- boolean enableEvent: event 서비스 사용 여부 설정(true 사용, false 사용 하지 않음)
- int eventFlag: 사용하고자 하는 event 서비스 종류 결정(Class WebtEventConnection 의 Field 참조)

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void destroy()
```

Description

- 모든 WebtConnectionGroup 을 destroy 한다

```
public static void destroyGroup(java.lang.String name)
```

Description

- groupname 이 name 인 WebtConnectionGroup 을 destroy 한다. 이 WebtConnectionGroup 이 가지고 있는 모든 WebtConnection 연결이 종료된다.
- 이 WebtConnectionGroup 의 index 번호보다 큰 index 번호를 갖는 WebtConnectionGroup 의 index 번호가 1 씩 감소한다

Arguments

- String name : group name

```
public static WebtConnection getConnection(java.lang.String
groupname) throws WebtIOException, WebtServiceException
```

Description

- group name 이 groupname 인 WebtConnection 을 반환한다.
- DEFAULT_GET_TIMEOUT(60 초)가 경과하여도 가용한 WebtConnection 이 없으면 null 을 리턴한다.

Arguments

- String groupname : 그룹 이름

Return value

- Group Name 이 groupname 인 WebtConnectionGroup 에서 WebtConnection 을 리턴한다

Throws

- WebtIOException : WebtConnectionGroup 내의 available 한 WebtConnection 이 없거나 활성화되어 있지 않은 경우 발생한다.
- WebtServiceException: WebtConnectionPool groupName 이 null 이거나 timeout 되면 WebtException 을 throw 한다.

```
public static WebtConnection getConnection(java.lang.String
groupname, long timeout_mili) throws WebtIOException,
WebtServiceException
```

Description

- group name 이 groupname 인 WebtConnection 을 반환한다.
- timeout_mili 가 경과하여도 가용한 WebtConnection 이 없으면 null 을 리턴한다.

Arguments

- String groupname : 그룹 이름
- Timeout_mili – get timeout 값(millisecond)

Return value

- Group Name 이 groupname 인 WebtConnectionGroup 에서 WebtConnection 을 리턴한다

Throws

- WebtIOException : WebtConnectionGroup 내의 available 한 WebtConnection 이 없거나 활성화되어 있지 않은 경우 발생한다.
- WebtServiceException: WebtConnectionPool groupName 이 null 이거나 timeout 되면 WebtException 을 throw 한다.

```
public static WebtEventConnection
getEventConnection(java.lang.String groupname) throws
WebtIOException, WebtServiceException
```

Description

- group name 이 groupname 인 WebtEventConnection 을 반환한다.

Arguments

- String groupname : 그룹 이름

Return value

- Group Name 이 groupname 인 WebtConnectionGroup 에서 WebtEventConnection 을 리턴한다

Throws

- WebtIOException : WebtConnectionGroup 내의 available 한 WebtConnection 이 없거나 활성화되어 있지 않은 경우 발생한다.
- WebtServiceException: WebtConnectionPool groupName 이 null 이거나 timeout 되면 WebtException 을 throw 한다.

```
public static java.util.Enumeration getGroupNames()
```

Description

- 현재 초기화된 WebtConnectionGroup 의 이름 목록을 리턴한다.

Return value

- 현재 초기화된 WebtConnectionGroup 의 이름 목록 (Enumeration).

```
public static void putConnection(WebtConnection con)
```

Description

- WebtConnection 을 ConnectionPool 에 반환한다. 반환할 WebtConnection 이 WebtTransaction 객체를 이용한 transaction 에 참여하였으나 아직 commit/rollback 이 수행되지 않은 경우 rollback 을 시도한다.

```
public static void replaceGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, int init, int max, int
inc) throws WebException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이 들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 기존의 WebtConnectionGroup 을 대체한다. 기존 WebtConnectionGroup 은 destroy 된다
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 갯수

Throws

- WebtException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다

```
public static void replaceGroup(java.lang.String groupname,
java.lang.String hostAddr, int hostPort, int init, int max, int
inc, int idle) throws WebIOException
```

Description

- Tmax Server(hostAddr:hostPort)로의 WebtConnection 연결을 갖는 WebtConnectionGroup 을 생성한다. 초기 연결 갯수는 init 이고 최대 연결 갯수는 max 이다.
- 모든 WebtConnection 연결이 사용중인 상태에서 WebtConnection 요청(WebtConnectionGroup.getConnection())이

들어오면 inc 만큼의 새로운 WebtConnection 을 생성하고 그 중 하나를 리턴한다.

Arguments

- String groupname : 이 WebtConnectionGroup 에 대한 이름. 동일한 이름의 WebtConnectionGroup 이 이미 존재할 경우 기존의 WebtConnectionGroup 을 대체한다. 기존 WebtConnectionGroup 은 destroy 된다
- String hostAddr : 연결을 맺을 Tmax Server 의 호스트 이름
- int hostPort: 연결을 맺을 Tmax Server 의 포트 번호
- int init: 초기 연결 갯수
- int max: 최대 연결 갯수
- int inc: 가용한 WebtConnection 이 없는 상태에서 요구가 들어올 때 새로이 생성할 WebtConnection 개수
- int idle : WebtConnection 에 허용된 최대 idle time (초단위). WebtConnectionPoolMonitor 에서 idle time check 할 때 사용한다.

Throws

- WebtIOException: WebtConnectionGroup 내의 WebtConnection 생성에 실패할 경우 발생한다
- WebtServiceException: WebtConnectionPool groupName 이 null 인 경우 발생한다.

```
public static void startMonitor(int interval, boolean checkAlive,
boolean checkIdle)
```

Description

- WebtConnection 을 monitoring 할 thread 를 생성한다. checkAlive 와 checkIdle 이 모두 false 이면 생성하지 않는다.

Arguments

- int interval : monitoring interval (초단위).
- boolean checkAlive : WebtConnection 이 살아있는지를 검사하고 끊어진 경우 재연결을 시도한다

- boolean checkIdle: true 이면, idle time 이 정한 시간 이상인 WebtConnection 을 close 하고 pool 에서 제거한다. 단, 초기 연결 갯수는 유지한다.

```
public static void stopMonitor()
```

Description

- WebtConnectionPoolMonitor 를 멈추고 monitor thread 를 종료한다.

class WebtRemoteService

public class WebtRemoteService extends java.lang.Object

이 클래스는 WebtConnection 및 그 자식 클래스를 통해 Tmax Service 를 제공받을 수 있게 하는 API 를 제공하고 tmax service 호출에 필요한 attribute 및 서비스 이름등을 관리하는 클래스 이다. WebtService 를 대체하는 클래스이다.

Constructors

```
public WebtRemoteService(java.lang.String svcname, WebtConnection
con)
```

Description

- WebtRemoteService 객체를 생성한다. 사용할 default character set 은 WebtSystem.getDefaultCharset()으로 초기화한다.

Throws

- java.lang.IllegalArgumentException: svcname 또는 con 이 null 인 경우 발생한다.

Methods

```
public void clear()
```

Description

- default service name, WebtConnection 객체를 null 로 초기화 한다.
- default character set 을 WebtSystem.getDefaultCharset()으로 초기화한다.

```
public WebtBuffer createCarrayBuffer()
```

Description

- 버퍼 타입이 WebtBuffer.BT_CARRAY 인 WebtBuffer 객체를 생성한다

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public tmax.webt.io.WebtCarrayBuffer createCarrayBuffer(int size)
```

Description

- 버퍼 타입이 WebtBuffer.BT_CARRAY 인 WebtBuffer 객체를 생성한다.

Arguments

- int size : 초기 버퍼 크기..

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public tmax.webt.io.WebtFieldBuffer createFieldBuffer()
```

Description

- 버퍼 타입이 WebtBuffer.BT_FIELD 인 WebtBuffer 객체를 생성한다.

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public tmax.webt.io.WebtFieldBuffer createFieldBuffer(int size)
```

Description

- 버퍼 타입이 WebtBuffer.BT_FIELD 인 WebtBuffer 객체를 생성한다.

Arguments

- int size : 초기 버퍼 크기..

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public WebtBuffer createStringBuffer()
```

Description

- 버퍼 타입이 WebtBuffer.BT_STRING 인 WebtBuffer 객체를 생성한다

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public WebtBuffer createStringBuffer(int size)
```

Description

- 버퍼 타입이 WebtBuffer.BT_STRING 인 WebtBuffer 객체를 생성한다.

Arguments

- int size : 초기 버퍼 크기..

Return value

- WebtBuffer :생성된 WebtBuffer 객체.

```
public WebtConnection getConnection()
```

Description

- TP 함수 호출에 사용하는 WebtConnection 객체를 리턴한다.

Return value

- WebtConnection :WebtConnection 객체.

```
public java.lang.String getDefaultCharset()
```

Description

- 이 객체를 통해 생성되는 모든 WebtBuffer 객체에 적용될 문자셋을 리턴한다.

Return value

- String : WebtBuffer 생성에 적용할 문자셋.

```
public java.lang.String getServiceName()
```

Description

- default service name 을 리턴한다.

Return value

- String : service name.

```
public void setConnection(WebtConnection conn)
```

Description

- TP 함수 호출에 사용할 WebtConnection 객체를 다시 설정한다.

Arguments

- WebtConnection conn : 새로이 사용할 WebtConnection 객체.

```
public void setDefaultCharset(java.lang.String charset)
```

Description

- 이 객체를 통해 생성되는 모든 WebtBuffer 객체에 적용될 문자셋을 설정한다.
- CreateStringBuffer, createCarrayBuffer, createFieldBuffer 를 호출하기 전에 이 값을 설정하여야 생성된 WebtBuffer 객체에 이 값이 적용된다.

Arguments

- charset: 적용할 문자셋.

```
public void setServiceName(java.lang.String name)
```

Description

- default service name 을 설정한다.

Arguments

- name: 새롭게 적용할 service name.

```
public int tpacall(java.lang.String svcname, WebtBuffer tx) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- Tmax Server 로 tpacall 을 호출한다.
- WebtRemoteService 객체를 생성할 당시 인자로 주어진 service name 과 다른 서비스를 호출할 때 이 함수를 사용한다.
- TP 함수 int tpacall(char *svc, char *sbuf, long slen, long flags)와 동일한 기능을 수행한다.

Arguments

- svcname: 서비스 받을 서비스 이름.
- tx : 송신 버퍼

Return value

- int : call descriptor

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public int tpacall(java.lang.String svcname, WebtBuffer tx,
WebtAttribute attr) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- Tmax Server 로 tpacall 을 호출한다.

- WebtRemoteService 객체를 생성할 당시 인자로 주어진 service name 과 다른 서비스를 호출할 때 이 함수를 사용한다.
- TP 함수 int tpacall(char *svc, char *sbuf, long slen, long flags)와 동일한 기능을 수행한다.

Arguments

- svcname: 서비스 받을 서비스 이름.
- tx : 송신 버퍼
- attr : tpacall 호출시 적용할 attribute

Return value

- int : call descriptor

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public int tpacall(WebtBuffer tx) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- Tmax Server 로 tpacall 을 호출한다. TP 함수 int tpacall(char *svc, char *sbuf, long slen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼

Return value

- int : call descriptor

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public int tpacall(WebtBuffer tx, WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- Tmax Server 로 tpacall 을 호출한다. TP 함수 int tpacall(char *svc, char *sbuf, long slen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼
- attr : tpacall 호출시 적용할 attribute

Return value

- int : call descriptor

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpcall(java.lang.String svcname, WebtBuffer tx)
throws WebtServiceException, WebtServiceFailException,
WebtIOException, WebtTXException
```

Description

- Tmax Server 로 tpcall 을 호출한다. WebtRemoteService 객체를 생성할 당시 인자로 주어진 service name 과 다른 서비스를 호출할 때 이 함수를 사용한다.
- TP 함수 int tpcall(char *svc, char *sbuf, long slen, char **rbuf, long *rlen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼
- svcname : 서비스 받을 서비스 이름

Return value

- WebtBuffer : 수신 버퍼.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpcall(java.lang.String svcname, WebtBuffer tx,
WebtAttribute attr) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- Tmax Server 로 tpcall 을 호출한다. WebtRemoteService 객체를 생성할 당시 인자로 주어진 service name 과 다른 서비스를 호출할 때 이 함수를 사용한다.
- TP 함수 int tpcall(char *svc, char *sbuf, long slen, char **rbuf, long *rlen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼
- svcname : 서비스 받을 서비스 이름
- attr : tpcall 호출시 적용할 attribute

Return value

- WebtBuffer : 수신 버퍼.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpcall(WebtBuffer tx) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- Tmax Server 로 tpcall 을 호출한다. TP 함수 int tpcall(char *svc, char *sbuf, long slen, char **rbuf, long *rlen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼

Return value

- WebtBuffer : 수신 버퍼.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpcall(WebtBuffer tx, WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- Tmax Server 로 tpcall 을 호출한다. TP 함수 int tpcall(char *svc, char *sbuf, long slen, char **rbuf, long *rlen, long flags)와 동일한 기능을 수행한다.

Arguments

- tx : 송신 버퍼
- attr : tpcall 호출시 적용할 attribute

Return value

- WebtBuffer : 수신 버퍼.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.

- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpgetrply(int cd) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException,
IllegalStateException
```

Description

- tpacall 에 대한 응답을 받는 method 이다. TP 함수 int tpgetrply(int *cd, char **rbuf, long *rlen, long flags) 와 동일한 기능을 수행한다.

Arguments

- cd : call descriptor 값. tpacall 의 return 값이다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다
- IllegalStateException : WebtEventHandler 에 의해 acall reply 에 대한 callback 이 등록된 상태에서 이 method 를 호출할 때 발생한다

```
public WebtBuffer tpgetrply(int cd, WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException, IllegalStateException
```

Description

- tpacall 에 대한 응답을 받는 method 이다. TP 함수 int tpgetrply(int *cd, char **rbuf, long *rlen, long flags) 와 동일한 기능을 수행한다.

Arguments

- cd : call descriptor 값. tpacall 의 return 값이다.
- Attr : attribute

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다
- IllegalStateException : WebtEventHandler 에 의해 acall reply 에 대한 callback 이 등록된 상태에서 이 method 를 호출할 때 발생한다

class WebtRQService

public class WebtRQService extends WebtRQService

이 클래스는 Tmax RQ 를 통해 신뢰성있는 Tmax Service 를 제공해주는 클래스이다. Tmax RQ 에 대한 자세한 사항은 Tmax RQ Development Guide 를 참조하기 바란다.

Since: 3.5.2

Fields

```
public static final String RQ_ANY_QUEUE
```

Description

- 이 플래그가 설정되고 RQ 서비스를 이용하였을 때 WebtRQService.tpqstat() 혹은 WebtRQService.tpqsvcstat() 호출시 Request, Reply, Fail Queue 의 통계값을 수신할 수 있다.

```
public static final String RQ_FAIL_QUEUE
```

Description

- 이 플래그가 설정되고 RQ 서비스를 이용하였을 때 WebtRQService.tpqstat() 혹은 WebtRQService.tpqsvcstat() 호출시 Fail Queue 의 통계값을 수신할 수 있다.

```
public static final String RQ_REQ_QUEUE
```

Description

- 이 플래그가 설정되고 RQ 서비스를 이용하였을 때 WebtRQService.tpqstat() 혹은 WebtRQService.tpqsvcstat() 호출시 Request Queue 의 통계값을 수신할 수 있다.

```
public static final String RQ_RPLY_QUEUE
```

Description

- 이 플래그가 설정되고 RQ 서비스를 이용하였을 때 WebtRQService.tpqstat() 혹은 WebtRQService.tpqsvcstat() 호출시 Reply Queue 의 통계값을 수신할 수 있다.

```
public static final int TPERR
```

Description

- WebtRQService.tpenq() 함수 호출시 서비스가 실패하여 Fail Queue 에 송신된 경우에 수신되는 값이다.
- WebtRQService.checkError()를 통해서 해당 return type 과 WebtBuffer.getUserReturnCode()통해서 수신되는 error code 값을 확인해 볼수 있다.

```
public static final int TPFAIL
```

Description

- WebtRQService. tpenq() 함수 호출시 서비스 명을 지정한 경우 해당 서비스에서 tpreturn 의 첫번째 인자로 TPFAIL 이 호출된 경우에 수신되는 값이다.
- WebtRQService. checkError()를 통해서 해당 return type 과 WebtBuffer.getUserReturnCode()통해서 수신되는 error code 값을 확인해 볼수 있다.

```
public static final int TPREQ
```

Description

- WebtRQService. tpenq() 함수 호출시 두번째 인자에 서비스명이 아닌 NULL 값이 지정된 경우에 서비스가 정상적으로 이루어진 경우에 수신되는 값이다.
- WebtRQService. checkError()를 통해서 해당 return type 과 WebtBuffer.getUserReturnCode()통해서 수신되는 error code 값을 확인해 볼수 있다.

```
public static final int TPSUCCESS
```

Description

- WebtRQService. tpenq() 함수 호출시 서비스 명을 지정한 경우 해당 서비스에서 tpreturn 의 첫번째 인자로 TPSUCCESS 가 호출된 경우에 수신되는 값이다.
- WebtRQService. checkError()를 통해서 해당 return type 과 WebtBuffer.getUserReturnCode()통해서 수신되는 error code 값을 확인해 볼수 있다.

Constructors

```
public WebtRQService(java.lang.String gname, java.lang.String
svcname, WebtConnection con)
```

Description

- Tmax RQ Service 를 받기 위한 WebtRQService 객체를 생성한다.

Arguments

- String gname : tpend, tpdeq method 호출시 default 로 사용할 Tmax RQ 이름을 지정한다. 이 이름은 Tmax 환경파일에 등록된 이름이어야 한다. null 일 수 없다.
- String svcname : tpenq, tpdeq method 호출시 default 로 사용할 Tmax Service Name 을 지정한다.null 또는 Tmax 환경파일에 등록된 서비스 이름을 지정한다
- WebtConnection con : Service 를 받기위한 WebtConnection. null 일 수 없다.

Throws

- Tmax RQ Service 를 받기 위한 WebtRQService 객체를 생성한다.
- java.lang.IllegalArgumentException : con 이 null 인 경우 발생한다.

Methods

```
public static int checkError(WebtBuffer rqbuf)
```

Description

- Tmax RQ client API 중 tpextsvcinfo 에 해당하는 함수이다.
- service name 은 rqbuf.getServiceName()으로 알수 있다.
- error code 는 rqbuf.getUserReturnCode()로 알 수 있다.

```
public java.lang.String getQueueName()
```

Description

- 현재 지정된 default RQ 이름을 반환한다.

Returns Value

- default RQ 이름.

```
public void setQueueName(java.lang.String gname)
```

Description

- tpenq, tpdeq method 호출시 default 로 사용할 RQ 이름을 재지정한다.

Arguments

- gname: 새로 지정할 default RQ name.

Throws

- java.lang.IllegalArgumentException: qname 이 null 인 경우 발생한다.

```
public WebtBuffer tpenq(java.lang.String gname, java.lang.String
svcname, WebtBuffer tx, WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- tpenq service 를 호출한다. 서비스 요청 시 해당 데이터를 RQ 에 저장하는 함수이다.
- 저장된 데이터는 도착 순서에 따라 관리되며 데이터를 읽을 때는 들어온 순서대로 처리된다.

Arguments

- String gname : 데이터를 저장할 RQ 이름. Tmax 환경 파일에 RQ 절에 등록된 이름이어야 한다
- String svcname : 호출할 서비스 이름. Tmax 환경파일에 등록된 서비스 이름으로 Reply Queue 에서 같은 서비스 명을 가진 데이터들을 순차적으로 수신한다.
- WebtBuffer tx : 서비스 호출시 전달되어야 할 데이터를 지정한다
- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Request Queue 에 데이다가 기록이되고 해당 서비스가 호출되어 수행 후 Reply Queue 에 결과값이 기록이 된다.

  WebtAttribute.TPNOREPLY : 해당 서비스를 수행 한 수 그 결과를 Reply Queue 에 저장하지 않는다.

  WebtAttribute.TPFUNC : 서비스 호출시 Reply Queue 에 해당 데이터가 기록이 되고 서비스를 호출하지 않은다.

  WebtAttribute.TPNOFLAG : 서비스 수행 후 서비스 결과와 관계 없이 해당 결과값을 클라이언트 버퍼에 저장된다. 이 값을 가져 오기 위해서는 WebtRQService.tpdeq()의 플래그 값이 TPNOFLAGS 로 주어져야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다.

```
public WebtBuffer tpenq(java.lang.String svcname, WebtBuffer tx,
WebtAttribute attr) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- tpenq service 를 호출한다. 서비스 요청 시 해당 데이터를 RQ 에 저장하는 함수이다.
- 저장된 데이터는 도착 순서에 따라 관리되며 데이터를 읽을 때는 들어온 순서대로 처리된다.

Arguments

- String svcname : 호출할 서비스 이름. Tmax 환경파일에 등록된 서비스 이름으로 Reply Queue 에서 같은 서비스 명을 가진 데이터들을 순차적으로 수신한다.
- WebtBuffer tx : 서비스 호출시 전달되어야 할 데이터를 지정한다
- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Request Queue 에 데이다가 기록이되고 해당 서비스가 호출되어 수행 후 Reply Queue 에 결과값이 기록이 된다.

  WebtAttribute.TPNOREPLY : 해당 서비스를 수행 한 수 그 결과를 Reply Queue 에 저장하지 않는다.

  WebtAttribute.TPFUNC : 서비스 호출시 Reply Queue 에 해당 데이터가 기록이 되고 서비스를 호출하지 않은다.

WebtAttribute.TPNOFLAG : 서비스 수행 후 서비스 결과와 관계 없이 해당 결과값을 클라이언트 버퍼에 저장된다. 이 값을 가져 오기 위해서는 WebtRQService.tpdeq()의 플래그 값이 TPNOFLAGS 로 주어져야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다.

```
public WebtBuffer tpenq(WebtBuffer tx) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- tpenq service 를 호출한다. 서비스 요청 시 해당 데이터를 RQ 에 저장하는 함수이다.
- 저장된 데이터는 도착 순서에 따라 관리되며 데이터를 읽을 때는 들어온 순서대로 처리된다.

Arguments

- WebtBuffer tx : 서비스 호출시 전달되어야 할 데이터를 지정한다

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다.

```
public WebtBuffer tpenq(WebtBuffer tx, WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- tpenq service 를 호출한다. 서비스 요청 시 해당 데이터를 RQ 에 저장하는 함수이다.
- 저장된 데이터는 도착 순서에 따라 관리되며 데이터를 읽을 때는 들어온 순서대로 처리된다.

Arguments

- WebtBuffer tx : 서비스 호출시 전달되어야 할 데이터를 지정한다
- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Request Queue 에 데이다가 기록이되고 해당 서비스가 호출되어 수행 후 Reply Queue 에 결과값이 기록이 된다.

  WebtAttribute.TPNOREPLY : 해당 서비스를 수행 한 수 그 결과를 Reply Queue 에 저장하지 않는다.

  WebtAttribute.TPFUNC : 서비스 호출시 Reply Queue 에 해당 데이터가 기록이 되고 서비스를 호출하지 않은다.

  WebtAttribute.TPNOFLAG : 서비스 수행 후 서비스 결과와 관계 없이 해당 결과값을 클라이언트 버퍼에 저장된다. 이 값을 가져 오기 위해서는 WebtRQService.tpdeq()의 플래그 값이 TPNOFLAGS 로 주어져야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.

- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다.

```
public WebtBuffer tpdeq() throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- tpdeq service 를 호출한다. RQ 에 저장된 데이터를 읽을 때 사용하는 함수이다.
- 한번 일어온 데이터는 RQ 에서 제거되기 때문에 한번 수신한 데이터를 다시 읽을 수 없다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpdeq(java.lang.String gname, java.lang.String
svcname, WebtAttribute attr) throws WebtServiceException,
WebtServiceFailException, WebtIOException, WebtTXException
```

Description

- tpdeq service 를 호출한다. RQ 에 저장된 데이터를 읽을 때 사용하는 함수이다.
- 한번 일어온 데이터는 RQ 에서 제거되기 때문에 한번 수신한 데이터를 다시 읽을 수 없다.

Arguments

- String gname : 데이터를 저장할 RQ 이름. Tmax 환경 파일에 RQ 절에 등록된 이름이어야 한다
- String svcname : 호출할 서비스 이름. Tmax 환경파일에 등록된 서비스 이름으로 Reply Queue 에서 같은 서비스 명을 가진 데이터들을 순차적으로 수신한다.
- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Reply Queue 에서 데이터를 수신하고자 할때 지정한다

  WebtAttribute.TPFUNC : Reply Queue 에 저장된 데이터를 수신하고자 할 때 WebtAttribute.TPRQS 와 함께 지정된다. 기록이 되고 서비스를 호출하지 않은다.

  WebtAttribute.TPNOFLAGS : 그 결과값을 클라이언트의 버퍼에 송신한다. 이런 경우 tpdeq 에서도 그 결과값을 수신하기 위해서 TPNOFLAGS 를 설정해야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpdeq(java.lang.String svcname, WebtAttribute
attr) throws WebtServiceException, WebtServiceFailException,
WebtIOException, WebtTXException
```

Description

- tpdeq service 를 호출한다. RQ 에 저장된 데이터를 읽을 때 사용하는 함수이다.

- 한번 일어온 데이터는 RQ 에서 제거되기 때문에 한번 수신한 데이터를 다시 읽을 수 없다.

Arguments

- String svcname : 호출할 서비스 이름. Tmax 환경파일에 등록된 서비스 이름으로 Reply Queue 에서 같은 서비스 명을 가진 데이터들을 순차적으로 수신한다.
- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Reply Queue 에서 데이터를 수신하고자 할때 지정한다

  WebtAttribute.TPFUNC : Reply Queue 에 저장된 데이터를 수신하고자 할 때 WebtAttribute.TPRQS 와 함께 지정된다. 기록이 되고 서비스를 호출하지 않은다.

  WebtAttribute.TPNOFLAGS : 그 결과값을 클라이언트의 버퍼에 송신한다. 이런 경우 tpdeq 에서도 그 결과값을 수신하기 위해서 TPNOFLAGS 를 설정해야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다

```
public WebtBuffer tpdeq(WebtAttribute attr) throws
WebtServiceException, WebtServiceFailException, WebtIOException,
WebtTXException
```

Description

- tpdeq service 를 호출한다. RQ 에 저장된 데이터를 읽을 때 사용하는 함수이다.
- 한번 일어온 데이터는 RQ 에서 제거되기 때문에 한번 수신한 데이터를 다시 읽을 수 없다.

Arguments

- WebtAttribute attr : 함수 호출시 적용할 attribute 다음의 속성값을 설정할 수 있다.

  WebtAttribute.TPRQS : Reply Queue 에서 데이터를 수신하고자 할때 지정한다

  WebtAttribute.TPFUNC : Reply Queue 에 저장된 데이터를 수신하고자 할 때 WebtAttribute.TPRQS 와 함께 지정된다. 기록이 되고 서비스를 호출하지 않은다.

  WebtAttribute.TPNOFLAGS : 그 결과값을 클라이언트의 버퍼에 송신한다. 이런 경우 tpdeq 에서도 그 결과값을 수신하기 위해서 TPNOFLAGS 를 설정해야 한다.

Throws

- WebtServiceFailException : tmax service 가 TPFAIL 을 tpreturn 하였을 때 throw 된다.
- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다.

```
public int tpqstat(java.lang.String gname, java.lang.String gtype)
```

Description

- 현재 큐에 저장된 데이터의 통계값을 구하는 함수 이다.
- qname 이 null 이면 getQueueNName()을 사용한다

Arguments

- String gname : 통계값을 수신할 RQ 의 이름을 지정한다. Tmax 환경파일에 등록된 이름이어야 한다
- gtype : 다음 값 중 하나이다.

  RQ_ANT_QUEUE : request, reply, fail queue 의 통계를 낸다.

  RQ_FAIL_QUEUE: fail queue 의 통계를 낸다.

  RQ_REQ_QUEUE : request queue 의 통계를 낸다.

  RQ_RPLY_QUEUE : reply queue 의 통계를 낸다.

Throws

- WebtServletException : qname 이 null 이거나 qtype 이 유효한 이름이 아닐 경우 발생한다.

```
public int tpqsvcstat(java.lang.String gname, java.lang.String
svcname, java.lang.String gtype)
```

Description

- 현재 큐에 저장된 데이터의 통계값을 service name 별로 구하는 함수 이다.
- qname 이 null 이면 getQueueNName()을 사용한다

Arguments

- String gname : 통계값을 수신할 RQ 의 이름을 지정한다. Tmax 환경파일에 등록된 이름이어야 한다.
- String svcname : 해당 서비스 이름.
- gtype : 다음 값 중 하나이다.

  RQ_ANT_QUEUE : request, reply, fail queue 의 통계를 낸다.

  RQ_FAIL_QUEUE: fail queue 의 통계를 낸다.

  RQ_REQ_QUEUE : request queue 의 통계를 낸다.

  RQ_RPLY_QUEUE : reply queue 의 통계를 낸다.

Throws

- WebtServletException : qname 이 null 이거나 qtype 이 유효한 이름이 아닐 경우 발생한다.

class WebtEventConnection

public class WebtEventConnection extends WebtConnection

Tmax Server 나 다른 client 로부터 전송된 event message(tppost/tpnotify/tpsendtocli/tpbroadcast) 처리가 가능한 WebtConnection 이다.

Tmax Server 나 다른 client 가 보낸 비요청 메시지를 수신하여 처리하기 위해서는 반드시 WebtEventConnection 을 사용하여 Tmax Server 에 연결해야 하며 수신한 event message 를 처리하기 위한 callback 함수를 정의한 WebtEventHandler 를 implement 하고 setEventHandler(WebtEventHandler) 함수를 사용하여 event handler 를 등록해야 한다.

아래의 프로그램은 tppost message 를 수신하기 위한 event handler 예제이다

```
import tmax.webt.*;

public class EventSample implements WebtEventHandler {
    private WebtEventConnection connection;

    public EventSample(WebtEventConnection connection) {
   this.connection = connection;

   // tpbroadcast event 만을 수신하기 위해 mask 를 설정한다.
   connection.setEventMask(WebtEventConnection.EVENT_POST);
```

```
    // event handler를 등록한다.
    connection.setEventHandler(this);

        // event에 가입한다.
        connection.tpsubscribe("posttest");
     }

     public void destroy() {
        connection.tpunsubscribe("posttest");
        connection.close();
     }

     // WebtEventHandler의 함수이다. event message를 수신하면 이
     // 함수가 callback된다.
     public void handleEvent(int type, WebtBuffer buf, int len,
int flags) {
   System.out.println("event received. type = " + type + ",
buffer type = " + buf.getBufferType());
   System.out.println("event msg : " + buf.getString());
     }

     // WebtEventHandler의 함수이다. WebT가 event처리중 Exception을
     // 만나면 이 함수를 callback한다. WebtIOException이
     // 발생하였다면 Tmax Server로의 connection이 끊어졌을 수 있으므로
reconnect한다.
     public void handleError(WebtException e) {
   System.err.println("event error occured");
   e.printStackTrace();
   if (e instanceof WebtIOException) {
      if (e.getTPError() == WebtException.TPECLOSE) {
         connection.close();
         connection.connect();
            }
  }
```

```
    }
}
```

Fields

```
public static final short EVENT_ACALLREPLY
```

Description

- tpsendtocli event message 를 수신하기 위한 mask 값

```
public static final short EVENT_ALL
```

Description

- 모든 event message 를 수신하기 위한 mask 값

```
public static final short EVENT_BROADCAST
```

Description

- tpbroadcast event message 를 수신하기 위한 mask 값

```
public static final short EVENT_NONE
```

Description

- 어떤 event message 도 수신하지 않기 위한 mask 값

```
public static final short EVENT_NOTIFY
```

Description

- tpnotify event message 를 수신하기 위한 mask 값

```
public static final short EVENT_POST
```

Description

- tppost event message 를 수신하기 위한 mask 값

```
public static final short EVENT_SENDTOCLI
```

Description

- tpsendtocli event message 를 수신하기 위한 mask 값

Constructors

```
public WebtEventConnection(java.lang.String hostAddr, int
hostPort) throws WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
boolean doConnect) throws WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다. doConnect 가 false 이면 연결을 시도하지 않는다.

Arguments

- String hostAddr : tmax 서버 주소.

- String hostPort : tmax 서버 listen port 번호.
- boolean doConnect : true 이면 연결을 시도한다.

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
boolean doConnect, int conntimeout) throws WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.
- doConnect 가 false 이면 연결을 시도하지 않는다.
- conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.
- boolean doConnect : true 이면 연결을 시도한다.
- conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
int conntimeout) throws WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.

- conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.
- conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
java.lang.String backupAddr, int backupPort) throws
WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.
- hostAddr:hostPort 로 연결을 설정하지 못하였을 경우 backupAddr:backupPort 로 연결을 시도한다.

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.
- String backupAddr : tmax backup 서버 주소.
- String backupPort : tmax backup 서버 listen port 번호.

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하여 hostAddr:hostPort 및 backupAddr:backupPort 로 연결을 설정하지 못한 경우 이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
java.lang.String backupAddr, int backupPort, boolean doConnect,
int conntimeout) throws WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.
- hostAddr:hostPort 로 연결을 설정하지 못하였을 경우 backupAddr:backupPort 로 연결을 시도한다.
- doConnect 가 false 이면 연결을 시도하지 않는다.
- conntimeout 이내에 연결을 설정하지 못하였을 경우 Exception 을 발생한다(tperrno = TPETIME).

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.
- String backupAddr : tmax backup 서버 주소.
- String backupPort : tmax backup 서버 listen port 번호.
- boolean doConnect : true 이면 연결을 시도한다.
- conntimeout : 연결 설정 timeout(초단위)

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생하여 hostAddr:hostPort 및 backupAddr:backupPort 로 연결을 설정하지 못하거나 conntimeout 내에 연결을 설정하지 못한 경우이다.

```
public WebtEventConnection(java.lang.String hostAddr, int hostPort,
java.lang.String username, java.lang.String userpwd,
java.lang.String domainname, java.lang.String domainpwd) throws
WebtIOException
```

Description

- hostAddr:hostPort 의 tmax server 로 연결을 설정한 WebtEventConnection 을 생성한다.

- 연결 설정시 인자로 주어진 인증 정보를 사용하여 인증절차를 수행한다.
- Tmax server 에 등록된 username, userpwd, domainname, domainpwd 값들이 일치하였을 때 연결이 성공적으로 이루어진다. .

Arguments

- String hostAddr : tmax 서버 주소.
- String hostPort : tmax 서버 listen port 번호.
- String username : 사용자 인증 계정 이름.
- String userpwd : 사용자 인증 번호.
- String domainname : 클라이언트 이름.
- String domainpwd : 시스템 접속 보안 암호.

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

Methods

```
public void setEventHander(WebtEventHander event) throws
WebtIOException
```

Description

- event message 를 처리할 handler 를 등록한다.
- event message 를 저장하는 queue 의 길이는 2 이다

Arguments

- WebtEventHandler event : event message 를 받았을 때 callback 할 handler

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public void setEventHander(WebtEventHander event, int qlen) throws
WebtIOException
```

Description

- event message 를 처리할 handler 를 등록한다.

Arguments

- WebtEventHandler event : event message 를 받았을 때 callback 할 handler
- int glen : event message 를 저장할 내부 queue 의 size 로 queue 에 빈 slot 이 없을 때 가장 오래된 메시지를 버린다

Throws

- WebtIOException : 연결 초기화중 네트워크 오류가 발생한 경우이다.

```
public void setEventMask(int mask)
```

Description

- tpbroadcast/tppost/tpsendtocli/tpnotify 가 송신한 event message 중 받고자 하는 메시지를 선택한다.

Arguments

- int mask : EVENT_BROADCAST, EVENT_POST, EVENT_SENDTOCLI, EVENT_NOTIFY 의 OR (|) 조합의 값을 갖는다. 해당 mask 가 on 되어야 해당 메시지를 수신하여 넘겨 준다. 이 함수를 수행하지 않을 경우 default 값은 EVENT_NONE (0) 이며 아무 메시지도 넘겨주지 않는다. EVENT_ALL 일 경우 모든 event 메시지를 넘겨준다.

Throws

- java.lang.IllegalArgumentException : mask 값이 유효하지 않은 경우 발생한다.

```
public void tpsubscribe(java.lang.String eventName) throws
WebtIOException, WebtServiceException
```

Description

- tppost message 를 수신하기 위해 post group 에 가입한다.

Arguments

- String eventName : 수신할 post message 의 post group name.

```
public void tpsubscribe(java.lang.String eventName, WebtAttribte
attr) throws WebtIOException, WebtServiceException
```

Description

- tppost message 를 수신하기 위해 post group 에 가입한다.

Arguments

- String eventName : 수신할 post message 의 post group name.
- WebtAttrbute attr : 이 함수 수행에 적용할 속성. 현재는 의미 있는 flag 가 존재하지 않는다

```
public void tpunsubscribe(java.lang.String eventName) throws
WebtIOException, WebtServiceException
```

Description

- 더이상 tppost message 를 받지 않기 위해 가입한 post group 에서 탈퇴한다.

Arguments

- String eventName : 탈퇴할 post group name

```
public void tpunsubscribe(java.lang.String eventName,
WebtAttribute attr) throws WebtIOException, WebtServiceException
```

Description

- 더이상 tppost message 를 받지 않기 위해 가입한 post group 에서 탈퇴한다.

Arguments

- String eventName : 탈퇴할 post group name
- WebtAttrbute attr : 이 함수 수행에 적용할 속성. 현재는 의미 있는 flag 가 존재하지 않는다

Interface WebtEventHandler

public interface WebtEventHandler

이 인터페이스는 tmax server 로 부터 수신한 비요청 메시지를 처리할 때 사용하는 인터페이스이다.

비요청 메시지를 수신하기 위해서는 이 인터페이스를 implement 해야한다. 비요청 메시지는 tpsendtocli/tpbroadcast/tpnotify/tppost acall_reply 가 있다

Fields

`public static final int` **`ACALLREPLY_TYPE`**

Description

- tpacall 에 대한 응답으로 수신한 데이터임을 뜻한다.

`public static final int` **`BROADCAST_TYPE`**

Description

- tpbroadcast 가 송신한 데이터임을 뜻한다.

```
public static final int NOTIFY_TYPE
```

Description

- tpnotify 가 송신한 데이터임을 뜻한다.

```
public static final int POST_TYPE
```

Description

- tppost 가 송신한 데이터임을 뜻한다.

```
public static final int SENDTOCLI_TYPE
```

Description

- tpsendtocli 가 송신한 데이터임을 뜻한다.

Methods

```
public void handleError(WebtException e)
```

Description

- 비요청 메시지 처리중 오류가 발생한 경우 (주로 네트워크 오류) 이 method 가 callback 된다.
- 네트워크 오류인 경우 tmax 로의 연결을 재설정하기를 원한다면 다음의 두 method 를 수행한다.

  WebtConnection.setEventMask(int mask_value);

  ```
  WebtConnection.setEventHandler(WebtEvent evt);
  ```

Arguments

- WebtException e : 오류 exception object

```
public void handleEvent(int type, WebtBuffer buf, int len, int
flags)
```

Description

- WebtEventConnection.setEventMask(int)에 의해 선택된 비요청 메시지가 수신되면 이 method 가 callback 된다.

Arguments

- int type : POST_TYPE, BROADCAST_TYPE, NOTIFY_TYPE, SENDTOCLI_TYPE, ACALLREPLY_TYPE 중 하나이다.
- WebtBuffer buf : 비요청 메시지의 내용을 담은 WebtBuffer.
- int len : 비요청 메시지 길이
- int flags : 비요청 메시지에 설정된 flag 값

class WebtDialogueService

public class WebtDialogueService extends WebtRemoteService

이 클래스는 tmax conversation mode service 를 제공하는 클래스이다

Constructors

```
public WebtDialogueService(java.lang.String svcname,
WebtConnection con)
```

Description

- WebtDialogueService 객체를 생성한다.
- 사용할 default character set 은 WebtSystem.getDefaultCharset() 으로 초기화한다

Arguments

- java.lang.IllegalArgumentException : svcname 또는 con 이 null 인 경우 발생한다.

Methods

```
public boolean isClosed()
```

Description

- 현재 conversation mode session 이 종료되었는지를 리턴한다

Return value

- boolean : true 이면 session 이 종료되었다는 의미이다.

Throws

- WebtServiceException : tpconnect 가 되지 않은 상태에서 이 method 를 호출할 때 발생 한다

```
public boolean isReceiveNext()
```

Description

- 현재 tprecv 가 가능한지를 리턴한다.

Return value

- boolean : true 이면 tprecv 가 가능하다는 의미이다.

Throws

- WebtServiceException : tpconnect 가 되지 않은 상태에서 이 method 를 호출할 때 발생 한다

```
public boolean isSendNext()
```

Description

- 현재 tpsend가 가능한지를 리턴한다.

Return value

- boolean : true이면 tpsend가 가능하다는 의미이다.

Throws

- WebtServiceException : tpconnect가 되지 않은 상태에서 이 method를 호출할 때 발생한다

```
public void tpconnect(boolean recvNext) throws
WebtServiceException, WebtIOException, WebtTXException,
WebtDialogueException
```

Description

- Conversation mode Service의 연결을 설정한다.
- 연결 설정시 반드시 대화 주도권을 설정해야 한다.

Argumentse

- boolean recvNext : false이면 tpconnect이후 연결 제어권을 계속 유지 하며 tpsend를 할 수 있다. 상대방은 수신만 할 수 있다. true이면 tpconnect후 연결 제어권을 상대방에 넘기고 수신만 할 수 있다.

```
public void tpconnect(WebtAttribute attr, boolean recvNext) throws
WebtServiceException, WebtIOException, WebtTXException,
WebtDialogueException
```

Description

- Conversation mode Service의 연결을 설정한다.
- 연결 설정시 반드시 대화 주도권을 설정해야 한다.

Arguments

- WebtAttribute attr : 송신 버퍼에 적용된 속성값. WebtAttribute 에 정의된 속성만 유효한다
- boolean recvNext : false 이면 tpconnect 이후 연결 제어권을 계속 유지 하며 tpsend 를 할 수 있다. 상대방은 수신만 할 수 있다. true 이면 tpconnect 후 연결 제어권을 상대방에 넘기고 수신만 할 수 있다 public void **tpconnect**(WebtBuffer sndbuf, boolean recvNext) throws WebtServiceException, WebtIOException, WebtTXException, WebtDialogueException

```
public void tpconnect(WebtBuffer sndbuf, WebtAttribute attr,
boolean recvNext) throws WebtServiceException, WebtIOException,
WebtTXException, WebtDialogueException
```

Description

- Conversation mode Service 의 연결을 설정한다.
- 연결 설정시 반드시 대화 주도권을 설정해야 한다.

Arguments

- WebtBuffer sndbuf : tpconnect 시점시 서버에 전송할 메시지
- WebtAttribute attr : 송신 버퍼에 적용된 속성값. WebtAttribute 에 정의된 속성만 유효한다
- boolean recvNext : false 이면 tpconnect 이후 연결 제어권을 계속 유지 하며 tpsend 를 할 수 있다. 상대방은 수신만 할 수 있다. true 이면 tpconnect 후 연결 제어권을 상대방에 넘기고 수신만 할 수 있다

```
public void tpdiscon()
```

Description

- tpconnect 후 연결의 제어권을 가지고 있는 측에서 Conversation mode service 의 연결을 즉시 종료할 수 있도록 한다.

```
public WebtBuffer tprecv()
```

Description

- 연결 제어권을 상대방에게 넘기고 수신 모드 상태에서 message 를 수신한다.

Return Value

- WebtBuffer : 수신 버퍼.

```
public WebtBuffer tprecv(WebtAttribute attr)
```

Description

- 연결 제어권을 상대방에게 넘기고 수신 모드 상태에서 message 를 수신한다.

Arguments

- WebtAttribute attr : 수신 버퍼에 적용된 속성값. WebtAttribute 에 정의된 속성만 유효한다.

Return Value

- WebtBuffer : 수신 버퍼.

```
public void tpsend(WebtBuffer tx, boolean recvNext) throws
WebtServiceException, WebtIOException, WebtTXException,
WebtDialogueException
```

Description

- Conversation mode Service 에서 메세지를 송신한다.

Arguments

- WebtBuffer tx : 송신 버퍼.
- boolean recvNext : false 이면 tpconnect 이후 연결 제어권을 계속 유지 하며 tpsend 를 할 수 있다. 상대방은 수신만 할

수 있다. true 이면 tpconnect 후 연결 제어권을 상대방에 넘기고 수신만 할 수 있다

Return Value

- WebtBuffer : 수신 버퍼.

Throws

- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다
- WebtDialogueException: conversational service 도중 tpsend 호출시 발생하는 Exception 이다.

```
public void tpsend(WebtBuffer tx, WebtAttribute attr, boolean
recvNext) throws WebtServiceException, WebtIOException,
WebtTXException, WebtDialogueException
```

Description

- Conversation mode Service 에서 메세지를 송신한다.

Arguments

- WebtBuffer tx : 송신 버퍼.]
- WebtAttribute attr : 송신 버퍼에 적용된 속성값. WebtAttribute 에 정의된 속성만 유효한다
- boolean recvNext : false 이면 tpconnect 이후 연결 제어권을 계속 유지 하며 tpsend 를 할 수 있다. 상대방은 수신만 할 수 있다. true 이면 tpconnect 후 연결 제어권을 상대방에 넘기고 수신만 할 수 있다

Return Value

- WebtBuffer : 수신 버퍼.

Throws

- WebtTXException : 관련 오류가 발생하였을 때 throw 된다.
- WebtServiceException : tmax service 오류가 발생하였을 때 throw 된다.
- WebtIOException : network 관련 tmax service 오류가 발생하였을 때 throw 된다
- WebtDialogueException: conversational service 도중 tpsend 호출시 발생하는 Exception 이다.

class WebtTransaction

public class WebtTransaction extends java.lang.Object

이 클래스는 여러 Service 를 하나의 작업 단위로 묶어서 처리하고자 할때 이를 처리할 수 있는 API 를 제공하고 에러 상황에 대한 적절한 exception 을 handling 한다

Fields

```
public static final boolean TX_COMMIT_COMPLETED
```

Description

- 2-Phare Commit 의 1 단계 prepare 과정 후 commit 명령에 대한 반환값을 확인 후 control 을 반환한다.

```
public static final boolean TX_COMMIT_DECISION_LOGGED
```

Description

- 2-Phare Commit 의 1 단계 prepare 과정이 끝나면 commit 명령을 내리고 바로 control 을 return 한다.

```
public static final int TX_COMMITTED
```

Description

- Transaction 이 데이타베이스를 독자적으로 수행됨.

```
public static final int TX_FAIL
```

Description

- 심각한 장애가 발생하였음.

```
public static final int TX_NO_BEGIN
```

Description

- Transaction 은 Commit 되었지만 새 Transaction 은 시작할 수 없음.

```
public statoc final int TX_OK
```

Description

- 정상적으로 처리됨.

```
public static final int TX_PROTOCOL_ERROR
```

Description

- Transaction 이 비정상적으로 호출되었음.

```
public static final int TX_ROLLBACK
```

Description

- Commit 을 할 수 없음.

Constructors

```
public WebtTransaction(WebtConnection conn)
```

Description

- WebtTransaction 객체를 생성한다

Arguments

- WebtConnection conn : transaction 이 이루어질 WebtConnection

Methods

```
public int begin() throws WebtTXException
```

Description

- 전역 트랜잭션을 시작한다.
- TP 함수 int tx_begin()와 동일한 기능을 수행한다

Arguments

- int : TX_OK

Throws

- WebtTXException : 이미 transaction 이 시작되었을 때 throw 된다.

```
public int commit() throws WebtTXException
```

Description

- 전역 트랜잭션을 commit 한다.
- TP 함수 int tx_commit()와 동일한 기능을 수행한다.

Arguments

- int : tx 상태

Throws

- WebtTXException : tx_begin 이 안된 상태일때, 네트워크에 오류가 있을때, Tmax Server 에서 transaction 관련 오류가 있을때 throw 된다.

```
public int getTimeout() throws WebtTXException
```

Description

- 현재의 transaction timeout 값을 리턴한다.

```
public boolean isChained()
```

Description

- 현재의 chain control 여부를 리턴한다.

Arguments

- int : tx 상태

Return Value

- boolean : true 이면 TX_CHAINED 가 설정된 것을 false 이면 TX_UNCHAINED 가 설정된 것임을 뜻한다.

```
public boolean isCommitCompleted()
```

Description

- setCommitCompleted(boolean)에 의해 설정된 값을 리턴한다.

Return Value

- boolean : setCommitCompleted(boolean).에 의해 설정된 값

```
public int rollback() throws WebtTXException
```

Description

- 전역 트랜잭션을 rollback 한다.
- TP 함수 int tx_rollback()와 동일한 기능을 수행한다.

Arguments

- int : tx 상태

Throws

- WebtTXException : tx_begin 이 안된 상태일때, 네트워크에 오류가 있을때, Tmax Server 에서 transaction 관련 오류가 있을때 throw 된다.

```
public void setChained(boolean value)
```

Description

- chain control 을 수행한다.
- value 가 true 이면 TX 함수 int tx_set_transaction_control(TX_CHAINED)와 동일한 기능을 수행하고 value 가 false 이면 TX 함수 int tx_set_transaction_control(TX_UNCHAINED)와 동일한 기능을 수행한다. 보다 자세한 내용은 Tmax Application Development Guide 를 참조한다.

Arguments

- boolean value : chain control 값

```
public void setCommitCompleted(boolean value)
```

Description

- 하나의 트랜잭션이 끝난 후 또 다른 트랜잭션의 시작 시점을 정한다.보다 자세한 내용은 Tmax Application Development Guide 를 참조한다.

Arguments

- boolean value : true 이면 TX 함수 int tx_set_commit_return(TX_COMMIT_COMPLETED)와 동일한 기능

수행하고 false 이면 TX 함수 int tx_set_commit_return(TX_COMMIT_DECISION_LOGGED)와 동일한 기능 수행한다.

```
public void setTimeout(int timeout)
```

Description

- transaction timeout 값을 지정한다. 단위는 msec 이다.
- 이 함수를 통해 별도로 timeout 값을 지정하지 않으면 Tmax Server configuration 파일에 TXTIME 으로 지정된 값이 적용된다.

Arguments

- int timeout : timeout 값.

class WebtSystem

public final class WebtSystem extends java.lang.Object

이 클래스는 WebT 시스템 전체에 적용될 속성을 담당하는 클래스이다

Fields

```
public static final int LOG_DEBUG
```

Description

- WebT logger 의 로그 레벨.
- Exception message, error message, debug message 를 로그에 남긴다.

```
public static final int LOG_INFO
```

Description

- WebT logger 의 로그 레벨.
- Exception message, error message, debug message 를 로그에 남긴다.

```
public static final int LOG_NONE
```

Description

- WebT logger 의 로그 레벨.
- Exception message, error message, debug message 를 로그에 남긴다.

```
public static final int TYPE_SHARED
```

Description

- 연결 공유 group 형태

```
public static final int TYPE_NONSHARED
```

Description

- 연결 비공유 group 형태

```
public static final int TYPE_NONSHARED2
```

Description

- 연결 비공유 group 형태 – jues 구동시 servlet engine 에서 미리 pool 을 생성

```
public static final int VERSION_NUMBER
```

Description

- WebT 시스템 일련 번호(버젼 번호)

Constructors

```
public WebtSystem
```

Description

- default constructor

Methods

```
public static void createDefaultFieldKeyTable(java.lang.String
fdlfile) throws java.io.IOException
```

Description

- field key table 을 생성한다.

Arguments

- String fdlfile : fdl 파일 경로

Throws

- java.io.IOException : fdlfile 을 찾지 못하거나 읽을때 오류가 발생한 경우

```
public static java.lang.String getDefaultCharset()
```

Description

- WebT system 의 default character set 을 리턴한다.

Return Value

- String : default character set

```
public static WebtFDLKeyTable getDefaultFieldKeyTable()
```

Description

- default field key table 을 리턴한다.

Return Value

- WebtFDLKeyTable : default field key table

```
public static void initLogger(int flag, java.lang.String logdir,
int buffersize)
```

Description

- log file 을 초기화 한다.

Arguments

- int flag : 다음의 세가지 값중에 하나를 선택한다.
  LOG_NONE (log 를 남기지 않음), LOG_INFO(information level 의 로그를 남김), LOG_DEBUG(debug level 의 로그를 남김)
- String logdir : log 를 남길 디렉토리를 지정한다.
- int buffersize : buffering size. 0 이면 buffering 하지 않고 곧바로 출력한다

```
public static void setDefaultCharset(java.lang.String charset)
```

Description

- WebT 에서 사용할 default character set 을 지정한다.
- WebtService/WebtRemoteService 객체 생성시 charset 을 지정하지 않으면 여기에서 지정한 값을 기본값으로 한다.

Arguments

- String : default character set

```
public static void userLog(java.lang.String s)
```

Description

- webt 로그에 사용자가 로그를 남길수 있도록하는 함수이다.

Arguments

- String s : 사용자 로그 메시지

```
public static java.lang.String version()
```

Description

- WebT 버젼을 리턴한다.

Return Value

- String : WebT version

class WebtFDLKeyTable

public class WebtFDLKeyTable extends java.lang.Object

tmax service 를 호출할 때 사용하는 버퍼의 타입이 WebtBuffer.BT_FIELD 인 경우에는 field key 를 사용하게 된다. WebT 는 일반 Tmax Client 와 마찬가지로 field key 를 사용하는데 두가지 방식이 있다.

1. 필드 정의 파일에 정의된 field key name 을 int 로 표현한 상수를 사용하는 방식.
2. 필드 정의 파일의 field key name 그 자체를 String 으로 표현하여 사용하는 방식.

두번째 방식을 사용하기 위해서는 필드 정의 파일(확장자 : .f)을 fdlc utility 를 사용하여 컴파일한 결과 파일인 fdl 파일(확장자 : .fdl)을 이 클래스를 통하여 로딩해야 한다. fdl 파일을 로딩하여 WebT 에서 사용할

수 있게 하려면 WebtSystem.createDefaultFieldKeyTable(String) 을 사용해야 한다. 아래에 사용 예제를 나타내었다.

```
// fdl 파일을 로딩한다.
        WebtSystem.createDefaultFieldKeyTable("tmax.fdl");
        WebtConnection connection = new WebtConnection("localhost",
8888);
        WebtRemoteService service = new
WebtRemoteService("FIELD_TEST", connection);
        WebtBuffer fieldBuffer = service.createFieldBuffer();
        WebtFieldSet fieldSet = new WebtFieldSet(fieldBuffer);

        // STRING1 이라는 string 타입의 field key 에 데이터를 입력한다.
        fieldSet.add("STRING1", "test string");

        // INT1 이라는 int 타입의 field key 에 데이터를 입력한다.
        fieldSet.add("INT1", (int)123);
        .....
        WebtBuffer rcvbuf = service.tpcall(fieldBuffer);
```

field key 에 대한 보다 자세한 내용은 Tmax FDL Reference Manual 을 참조하기 바란다

Constructors

```
public WebtFDLKeyTable()
```

Description

- default constructor

Methods

```
public int getKey(java.lang.String keyString)
```

Description

- field key 이름으로 부터 int 형 상수 표현 값을 리턴한다.

Arguments

- String keyString : field key 이름

Return value

- int : 상수 표현 값

```
public java.lang.String getKeyName(int fdlkey)
```

Description

- int 형의 상수로 표현된 field key 이름을 String 으로 표현한 값을 리턴한다.

Arguments

- int fdlkey : 상수

Return value

- String : String 으로 표현된 field key name

```
public int getType(java.lang.String keyString)
```

Description

- field key 의 타입을 리턴한다. 요효하지 않은 field key 이면 -1 을 리턴한다. 유효한 field key 인 경우 리턴값은 WebtField.FB_CHAR, WebtField.FB_SHORT, WebtField.FB_INT, WebtField.FB_LONG,

WebtField.FB_FLOAT, WebtField.FB_DOUBLE, WebtField.FB_STRING, WebtField.FB_CARRAY 중 하나이다.

Arguments

- String keyString : field key 이름.

Return value

- int : field key 타입.

```
public int loadFDLKeys(java.lang.String filepath) throws
java.io.IOException
```

Description

- fdl 파일로 부터 field key 를 로딩한다.

Arguments

- String filepath : fdl 파일의 위치

```
public static int makeFDLKey(int type, int no)
```

Description

- Application 에서 사용자가 field type 과 field number 를 정의하면 int 형의 상수 field key 값을 리턴한다.

Arguments

- int type: field type (Class WebtFeild 의 Fields 부분 참조)
- int no: field 지정 번호

Return value

- int : field key 상수

```
public int size()
```

Description

- 현재 로딩된 field key 갯수를 리턴한다.

Return value

- int : 현재 로딩된 field key 개수.

class WebtField

public abstract class WebtField extends java.lang.Object implement java.io.Serializable

WebtField class 는 FIELD type 버퍼에서 사용하는 field key 및 그 field key 값에 해당하는 다수의 데이터(WebtFieldElement)를 관리하는 추상 클래스이다.

특정 field key 에 대한 WebtField 객체를 얻으려면 WebtBuffer.createField(int), WebtBuffer.createField(String), WebtBuffer.getField(int), WebtBuffer.getField(String)을 사용한다. 단 이때 WebtBuffer 의 버퍼 타입은 WebtBuffer.BT_FIELD 이어야 한다

아래에 WebtField 를 사용하는 한 예를 나타내었다.

```
WebtConnection connection = new WebtConnection("localhost", 8888);
 WebtRemoteService service = new WebtRemoteService("TEST",
connection);
 WebtBuffer fieldBuf = service.createFieldBuffer();

 // field key 값이 INT1 인 WebtField 객체를 생성한다.
 WebtField field1 = fieldBuf.createField(INT1);
```

```
// field key 이름이 "INT2"인 WebtField 객체를 생성한다.
WebtField field2 = fieldBuf.createField("INT2");

// 생성한 WebtField 객체에 데이터를 추가한다.
field1.add("1");
field1.add((int)2);
field1.add(123.45);
....
....
// WebtField 객체에서 위에서 추가한 데이터를 차례로 꺼내온다.
int data1 = field1.get().intValue(); // == 1
int data2 = field1.get().intValue(); // == 2
int data3 = field1.get().intValue(); // == 123
```

WebtField.add 혹은 WebtField.insert 함수를 사용하여 새로운 field data 를 추가/삽입하고자 할 때 field key type 과 추가하려는 데이터의 타입에 따라서, 실제 추가/삽입되는 데이터의 값이 달라질 수 있다. 이 때 적용되는 규칙은 각각의 add/insert method 를 참조한다.

Tmax Server 로부터 전송받은 field data 에 접근할 때에는 WebtFieldElement 의 method 를 사용한다. 이 때의 데이터 변환에 대해서는 WebtFieldElement 를 참조한다

Fields

`public static final int` **`FB_CARRAY`**

Description

- byte[] 형 field type (C 에서 char array 와 동일하다)

`public static final int` **`FB_CHAR`**

Description

- byte 형 field type

```
public static final int FB_DOUBLE
```

Description

- double 형 field type

```
public static final int FB_FLOAT
```

Description

- float 형 field type

```
public static final int FB_INT
```

Description

- int 형 field type

```
public static final int FB_LONG
```

Description

- int 형 field type. FB_INT 와 동일하게 취급한다.

```
public static final int FB_SHORT
```

Description

- short 형 field type

```
public static final int FB_STRING
```

Description

- String 형 field type (C 에서 null-terminated char array 와 동일하다)

Methods

```
public abstract void add(byte value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : value

  FB_STRING : byte b[2] = (value,'\0')

  FB_CARRAY : byte b[1] \ (value)

  FB_SHORT : (short)value

  FB_INT : (int)value

  FB_LONG : (long)value

  FB_FLOAT : (flost)value

  FB_DOUBLE : (double)value

Arguments

- byte value : 추가할 데이터

```
public void add(byte[] value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : value[0]

  FB_STRING : value + null('\0') padding

  FB_CARRAY : value

  FB_SHORT : (new Short(new String(value))).shortValue()

FB_INT : (new Double (new String(value))).intValue()

FB_LONG : (new Double (new String(value))).intValue()

FB_FLOAT : (new Double (new String(value))).floatValue()

FB_DOUBLE : (new Double(new String(value))).doubleValue()

Arguments

- byte[] value : 추가할 데이터

```
public abstract void add(byte[] value, int off, int len) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

FB_CHAR : value[0]

FB_STRING : value + null('\0') padding

FB_CARRAY : value

FB_SHORT : (new Double (new String(value))).shortValue()

FB_INT : (new Double (new String(value))).intValue()

FB_LONG : (new Double (new String(value))).intValue()

FB_FLOAT : (new Double (new String(value))).floatValue()

FB_DOUBLE : (new Double(new String(value))).doubleValue()

Arguments

- byte[] value : 추가할 데이터
- int off : value 중 추가할 데이터의 시작 index
- int len : value 중 추가할 데이터 길이.

```
public abstract void add(double value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : (byte)value

  FB_STRING : Double.toString(value).getBytes() + null('\0') padding

  FB_CARRAY : Double.toString(value).getBytes()

  FB_SHORT : (short)value

  FB_INT : (int)value

  FB_LONG : (long)value

  FB_FLOAT : (float)value

  FB_DOUBLE :value

Arguments

- byte value : 추가할 데이터

```
public abstract void add(float value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : (byte)value

  FB_STRING : Float.toString(value).getBytes() + null('\0') padding

FB_CARRAY : Float.toString(value).getBytes()

FB_SHORT : (short)value

FB_INT : (int)value

FB_LONG : (long)value

FB_FLOAT : value

FB_DOUBLE : (double)value

Arguments

- byte value : 추가할 데이터

```
public abstract void add(int value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

FB_CHAR : (byte)value

FB_STRING : Integer.toString(value).getBytes() + null('\0') padding

FB_CARRAY : Integer.toString(value).getBytes()

FB_SHORT : (short)value

FB_INT : value

FB_LONG : (long)value

FB_FLOAT : (float)value

FB_DOUBLE : (double)value

Arguments

- byte value : 추가할 데이터

```
public abstract void add(long value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : (byte)value

  FB_STRING : Long.toString(value).getBytes() + null('\0') padding

  FB_CARRAY : Long.toString(value).getBytes()

  FB_SHORT : (short)value

  FB_INT : (int)value

  FB_LONG : value

  FB_FLOAT : (float)value

  FB_DOUBLE : (double)value

Arguments

- byte value : 추가할 데이터

```
public abstract void add(java.lang.Object value) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다. value 가 null 인 경우 무시한다.
- 데이터 변환 규칙은 value 의 실제 타입에 따라 아래와 같다.

- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  String : add(String)의 규칙을 다른다.

  Byte : add(byte)의 규칙을 다른다.

  Short : add(short)의 규칙을 다른다.

  Integer : add(int)의 규칙을 다른다.

  Float : add(float)의 규칙을 다른다.

  Double : add(double)의 규칙을 다른다.

  그 외 : add(value.toString())을 수행한다.

Arguments

- byte value : 추가할 데이터. 데이터 타입이 Byte, Short, Integer, Long, Float, Double, String 이 아니면 add(value.toString())을 수행한다.

```
public abstract void add(short value) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : (byte)value

  FB_STRING : Short.toString(value).getBytes() + null('\0') padding

  FB_CARRAY : Short.toString(value).getBytes()

  FB_SHORT : value

  FB_INT : (int)value

FB_LONG : (long)value

FB_FLOAT : (float)value

FB_DOUBLE : (double)value

Arguments

- byte value : 추가할 데이터

```
public abstract void add(java.lang.String value) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : value.getBytes(charset)[0](charset == getDefaultCharset())

  FB_STRING : value.getBytes(charset)[0] + null('\0') padding (charset == getDefaultCharset())

  FB_CARRAY : value.getBytes(charset) (charset == getDefaultCharset())

  FB_SHORT : (new Double(value)).shortValue()

  FB_INT : (new Double(value)).intValue()

  FB_LONG : (new Double(value)).intValue()

  FB_FLOAT : (new Double(value)).floatValue()

  FB_DOUBLE : (new Double(value)).doubleValue()

Arguments

- byte value : 추가할 데이터

```
public abstract void add(java.lang.String value, java.lang.String
charset) throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 추가한다.
- getFieldType()의 리턴값에 따라 실제로 Tmax Server 에 전달되는 데이터는 다음과 같다

  FB_CHAR : value.getBytes(charset)[0]

  FB_STRING : value.getBytes(charset)[0] + null('\0') padding

  FB_CARRAY : value.getBytes(charset)

  FB_SHORT : (new Double(value)).shortValue()

  FB_INT : (new Double(value)).intValue()

  FB_LONG : (new Double(value)).intValue()

  FB_FLOAT : (new Double(value)).floatValue()

  FB_DOUBLE : (new Double(value)).doubleValue()

Arguments

- byte value : 추가할 데이터
- charset : String 을 다룰 때 사용할 문자셋

```
public WebtFieldElement get() throws WebtBufferException
```

Description

- 첫번째 데이터를 삭제하여 리턴한다.
- Remove()와 동일하다.

Return Value

- WebtFieldElement : 삭제한 데이터

```
public abstract WebtFieldElement get(int index) throws
WebtBufferException
```

Description

- 데이터를 리턴한다.

Arguments

- int index : 반환할 데이터의 순번

Return Value

- WebtFieldElement : 삭제한 데이터

```
public java.lang.Object[] getAll() throws WebtBufferException
```

Description

- 모든 데이터를 WebtFieldElement[] 형태로 리턴한다.

Return Value

- Object[] : WebtFieldElement 의 array

```
public java.lang.String getDefaultCharset()
```

Description

- 이 WebtField 에 저장된 데이터를 String 형으로 변환하거나 String 형 데이터를 byte[]로 변환할 때 사용하는 문자셋을 리턴한다.

Return Value

- String : 문자셋

```
public java.lang.Enumeration getFieldEnumeration() throws
WebtBufferException
```

Description

- 모든 데이터를 WebtFieldElement 의 Enumeration 형태로 리턴한다.

Return Value

- Enumeration : WebtFieldElement 의 Enumeration

```
public int getFieldKey()
```

Description

- 이 WebtField 객체의 field key 값을 리턴한다.

Return Value

- int : field key 값

```
public int getFieldType()
```

Description

- 이 WebtField 객체의 field key type 을 리턴한다.

Return Value

- int : field key type

```
public java.util.Vector getFieldVector() throws
WebtBufferException
```

Description

- 모든 데이터를 WebtFieldElement 의 Vector 형태로 리턴한다.

Return Value

- Vector : WebtFieldElement 의 Vector

```
public void insert(byte[] value, int index) throws
WebtBufferException
```

Description

- Insert(value, 0, value.length, index)와 동일하다.

```
public abstract void insert(byte[] value, int off, int len, int
index) throws WebtBufferException
```

Description

- value를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(bayte[])을 따른다.

Arguments

- byte value : 추가할 데이터
- int off : value 중 추가할 데이터의 시작 index
- int len : value 중 추가할 데이터 길이
- int index : 삽입할 위치

```
public abstract void insert(byte value, int index) throws
WebtBufferException
```

Description

- value를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(byte)을 따른다.

Arguments

- byte value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(float value, int index) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(float)을 따른다.

Arguments

- float value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(double value, int index) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(double)을 따른다.

Arguments

- double value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(int value, int index) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(int)을 따른다.

Arguments

- int value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(long value, int index) throws
WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(long)을 따른다.

Arguments

- long value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(java.lang.Object value, int index)
throws WebtBufferException
```

Description

- value 를 해당 field key 타입으로 변환하여 삽입한다. value 가 null 인 경우 무시한다.
- 데이터 변환 규칙은 value 의 실제 타입에 따라 아래와 같다.

  String : insert(String, int)의 규칙을 따른다.

  Byte : insert(byte, int)의 규칙을 따른다.

  Short : insert(short, int)의 규칙을 따른다.

  Integer : insert(int, int)의 규칙을 따른다.

  Long : insert(long, int)의 규칙을 따른다.

  Float : insert(float, int)의 규칙을 따른다.

  Double : insert(double, int)의 규칙을 따른다.

  그 외 : insert(value.toString(), index)을 수행한다.

Arguments

- Object value : 추가할 데이터. 데이터 타입이 Byte, Short, Integer, Long, Float, Double, String 이 아니면 insert(value.toString(), index)을 수행한다.

```
public abstract void insert(short value, int index) throws
WebtBufferException
```

Description

- value를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(short)을 따른다.

Arguments

- short value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(java.lang.String value, int index)
throws WebtBufferException
```

Description

- value를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(String)을 따른다.

Arguments

- String value : 추가할 데이터
- int index : 삽입할 위치

```
public abstract void insert(java.lang.String value,
java.lang.String charset, int index) throws WebtBufferException
```

Description

- value를 해당 field key 타입으로 변환하여 삽입한다.
- 데이터 변환 규칙은 add(String, String)을 따른다.

Arguments

- String value : 추가할 데이터
- String charset : String을 다룰때 사용할 문자셋.
- int index : 삽입할 위치

```
public boolean isNumericType()
```

Description

- 이 WebtField 객체의 field key 가 numeric type 인지를 검사한다.

Return Value

- boolean : field key type 이 FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE 이면 true 를 리턴하고 그렇지 않으면 false 를 리턴한다

```
public WebtFieldElement remove() throws WebtBufferException
```

Description

- 첫번째 데이터를 삭제한다.

Return Value

- WebtFieldElement: 삭제한 데이터

```
public abstract WebtFieldElement remove(int index) throws
WebtBufferException
```

Description

- 데이터를 삭제한다.

Arguments

- int index : 삭제할 데이터의 순번.

Return Value

- WebtFieldElement: 삭제한 데이터

```
public void removeAll() throws WebtBufferException
```

Description

- 모든 데이터를 삭제한다.

```
public abstract void replace(byte[] value, int off, int len, int
index) throws WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(byte[])을 따른다.

Arguments

- byte[] value : 치환할 데이터.
- int off : value 중 치환할 데이터의 시작 index.
- int len : value 중 치환할 데이터 길이.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(byte value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(byte)을 따른다.

Arguments

- byte value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(byte[] value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(byte)을 따른다.
- replace (value, 0, value.length, index) 와 동일하다. replace(byte[] , int, int, int) 참조

Arguments

- byte [] value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(double value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(double)을 따른다.

Arguments

- double value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(float value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(float)을 따른다.

Arguments

- float value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(int value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(int)을 따른다.

Arguments

- int value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(long value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.

- 데이터 변환 규칙은 add(long)을 따른다.

Arguments

- long value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public void replace(java.lang.Object value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다. null 인 경우 무시한다.
- 데이터 변환 규칙은 value 의 실제 타입에 따라 아래와 같다.

  String : replace(String, int)의 규칙을 따른다.

  Byte : replace (byte, int)의 규칙을 따른다.

  Short : replace (short, int)의 규칙을 따른다.

  Integer : replace (integer, int)의 규칙을 따른다.

  Long : replace (long, int)의 규칙을 따른다.

  Float : replace (float, int)의 규칙을 따른다.

  Double : replace (double, int)의 규칙을 따른다.

  그 외 : replace (value.toString(), index)을 수행한다.

Arguments

- Object value : 추가할 데이터. 데이터 타입이 Byte, Short, Integer, Long, Float, Double, String 이 아니면 replace(value.toString(), index)을 수행한다.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(short value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(short)을 따른다.

Arguments

- short value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(String value, int index) throws
WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(String)을 따른다.

Arguments

- short value : 치환할 데이터.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public abstract void replace(String value, java.lang.String
charset, int index) throws WebtBufferException
```

Description

- index 번째의 값을 value 를 해당 field key 타입으로 변환한 값으로 치환한다.
- 데이터 변환 규칙은 add(String, String)을 따른다.

Arguments

- short value : 치환할 데이터.
- String charset : String 을 다룰때 사용할 문자셋.
- int index : 삭제할 데이터의 순번.

Throws

- WebtBufferException : index 가 0 보다 작을 때 throw 된다.

```
public void setDefaultCharset(java.lang.String charset) throws
WebtBufferException
```

Description

- 이 WebtField 에 저장된 데이터를 String 형으로 변환하거나 String 형 데이터를 byte[]로 변환할 때 사용할 문자셋을 지정한다.
- 지정하지 않으면 WebtRemoteService 에서 지정한 값을 사용한다.

Arguments

- String charset : 지정할 문자셋.

```
public int size()
```

Description

- 저장된 데이터의 갯수를 리턴한다.

Return Value

- int : 저장된 데이터의 갯수.

```
public java.lang.String toString()
```

Description

- 이 객체를 String 으로 표현하여 리턴한다.

Interface WebtFieldElement

public interface WebtFieldElement extends java.io.Serializable

WebtFieldElement 는 WebtField 가 자신의 field key 에 대한 field data 를 담는 인터페이스이다. WebtField 는 field data 를 이 클래스로 표현하여 관리한다. 즉, 하나의 field data 에 하나의 WebtFieldElement 객체가 존재한다.

이 클래스는 인터페이스로서 사용자가 직접 객체를 생성할 수 없다. Tmax Server 로 부터 수신한 field data 라면 WebT 가 이 객체를 생성하여 해당 WebtField 객체에 추가한다. 사용자가 이 객체를 생성하여 WebtField 에 추가하는 방법은 WebtField.add method 들과 WebtField.insert method 를 통해서이다.

Methods

```
public byte[] bytesValue() throws WebtBufferException
```

Description

- field data 를 byte[]로 표현하여 리턴한다

- FB_CHAR, FB_CARRAY, FB_STRING: 저장되어 있는 원본 데이터를 byte array 로 표현하여 리턴한다. 단 FB_STRING 인 경우 Tmax Server 로 부터 데이터를 수신할 때 데이터의 맨마지막에 '\0'으로 끝난다. 이 '\0'는 제거된 상태로 리턴된다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE 의 경우. : stringValue().getBytes()를 리턴한다.

Return value

- byte[] : byte[] 표현된 field data

Throws

- WebtBufferException : field data 가 null 일 경우 발생한다

```
public byte byteValue() throws WebtBufferException
```

Description

- field data 를 byte 로 표현하여 리턴한다
- FB_CHAR, FB_CARRAY, FB_STRING : bytesValue() [0]와 동일하다.
- FB_SHORT, FB_INT, FB_FLOAT, FB_DOUBLE : (new Double(value)).byteValue()와 동일하다.

Return value

- byte : byte 표현된 field data

Throws

- WebtBufferException : field data 가 null 일 경우 발생한다

```
public double doubleValue() throws WebtBufferException
```

Description

- field data 를 double 로 표현하여 리턴한다

- FB_CHAR, FB_STRING, FB_CARRAY : Double.valueOf(stringValue()).doubleValue()와 동일한 결과를 리턴한다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE : (new Double(value)).doubleValue()와 동일한 결과를 리턴한다.

Return value

- double : double 표현된 field data

Throws

- WebtBufferException : WebtFieldElement 의 field type 이 FB_STRING/FB_CHAR/FB_CARRAY 인 경우 숫자로 표현할 때 java.io.NumberFormatException 또는 java.io.UnsupportedEncodingException 이 발생할 수 있다. 이 때 WebtBufferException 이 throw 된다. 또는 field data 가 null 일 경우 발생한다.

```
public float floatValue() throws WebtBufferException
```

Description

- field data 를 float 로 표현하여 리턴한다
- FB_CHAR, FB_STRING, FB_CARRAY : Float.valueOf(stringValue()).floatValue()와 동일한 결과를 리턴한다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE : (new Double(value)).floatValue()와 동일한 결과를 리턴한다.

Return value

- float : float 표현된 field data

Throws

- WebtBufferException : WebtFieldElement 의 field type 이 FB_STRING/FB_CHAR/FB_CARRAY 인 경우 숫자로 표현할 때 java.io.NumberFormatException 또는 java.io.UnsupportedEncodingException 이 발생할 수 있다. 이

때 WebtBufferException 이 throw 된다. 또는 field data 가 null 일 경우 발생한다.

```
public WebtField getField()
```

Description

- 이 WebtFieldElement 가 속한 WebtField 객체의 reference 를 리턴한다.

Return value

- WebtField : WebtField 객체의 reference

```
public int intValue() throws WebtBufferException
```

Description

- field data 를 int 로 표현하여 리턴한다
- FB_CHAR, FB_STRING, FB_CARRAY : Integer.valueOf(stringValue()).intValue()와 동일한 결과를 리턴한다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE : (new Double(value)).intValue()와 동일한 결과를 리턴한다.

Return value

- int : int 표현된 field data

Throws

- WebtBufferException : WebtFieldElement 의 field type 이 FB_STRING/FB_CHAR/FB_CARRAY 인 경우 숫자로 표현할 때 java.io.NumberFormatException 또는 java.io.UnsupportedEncodingException 이 발생할 수 있다. 이 때 WebtBufferException 이 throw 된다. 또는 field data 가 null 일 경우 발생한다.

```
public int longValue() throws WebtBufferException
```

Description

- field data 를 long 로 표현하여 리턴한다
- FB_CHAR, FB_STRING, FB_CARRAY : Integer.valueOf(stringValue()).intValue()와 동일한 결과를 리턴한다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE : (new Double(value)).intValue()와 동일한 결과를 리턴한다.

Return value

- long : long 표현된 field data

Throws

- WebtBufferException : WebtFieldElement 의 field type 이 FB_STRING/FB_CHAR/FB_CARRAY 인 경우 숫자로 표현할 때 java.io.NumberFormatException 또는 java.io.UnsupportedEncodingException 이 발생할 수 있다. 이 때 WebtBufferException 이 throw 된다. 또는 field data 가 null 일 경우 발생한다.

```
public short shortValue() throws WebtBufferException
```

Description

- field data 를 short 로 표현하여 리턴한다
- FB_CHAR, FB_STRING, FB_CARRAY : Short.valueOf(stringValue()).shortValue()와 동일한 결과를 리턴한다.
- FB_SHORT, FB_INT, FB_LONG, FB_FLOAT, FB_DOUBLE : (new Double(value)).shortValue()와 동일한 결과를 리턴한다.

Return value

- short : short 표현된 field data

Throws

- WebtBufferException : WebtFieldElement 의 field type 이 FB_STRING/FB_CHAR/FB_CARRAY 인 경우 숫자로 표현할 때 java.io.NumberFormatException 또는 java.io.UnsupportedEncodingException 이 발생할 수 있다. 이 때 WebtBufferException 이 throw 된다. 또는 field data 가 null 일 경우 발생한다.

```
public java.lang.String stringValue() throws WebtBufferException
```

Description

- field data 를 string 로 표현하여 리턴한다
- stringValue(getField().getDefaultCharset())과 동일한 결과이다.

Return value

- String : String 표현된 field data

Throws

- WebtBufferException : field data 가 null 일 경우 발생한다.

```
public java.lang.String stringValue(java.lang.String charset)
throws WebtBufferException
```

Description

- field data 를 string 로 표현하여 리턴한다.

Arguments

- String charset : 문자셋.

Return value

- String : String 표현된 field data

Throws

- WebtBufferException : field data 가 null 일 경우 발생한다.

class WebtFieldSet

public class WebtFieldSet extends java.lang.Object implements java.io.Serializable

WebtFieldSet 클래스는 Tmax service 에 사용할 버퍼 타입이 WebtBuffer.BT_FIELD 인 경우 좀 더 편리하게 field data element 를 다룰 수 있도록 해주는 utility class 이다. 아래에 사용 예를 나타내었다.

```
WebtConnection connection = new WebtConnection("localhost", 8888);
 WebtRemoteService service = new WebtRemoteService("FIELDSET_TEST",
connection);

 WebtFieldSet sndset = new

 // FIELD 타입의 WebtBuffer 를 이용하여 WebtFieldSet 객체를 생성한다.
 WebtFieldSet(service.createFieldBuffer());

 // WebtFieldSet 을 사용하여 WebtBuffer 에 field data element 를 추가한
다.
 sndset.add("STRING1", "hello!");

 // tpcall 호출
 WebtBuffer rcvbuf = service.tpcall(sndset.getFieldBuffer());

 // Tmax 로부터 수신한 WebtBuffer 를 이용하여 WebtFieldSet 객체를
 생성한다. 이때 rcvbuf 의 버퍼타입은 FIELD 타입이어야 한다.
 WebtFieldSet rcvset = new WebtFieldSet(rcvbuf);

 // Tmax server 가 보낸 field type 이 INT1 인 field data 를 꺼낸다.
 int value1 = rcvset.getInt("INT1").intValue();
```

```
// Tmax server가 보낸 field type이 ITN1인 두번째 field data를
// String으로 변환하여 꺼낸다.
String value2 = rcvset.getString("INT1");
...
```

Constructors

```
public WebtFieldSet() throws WebtBufferException
```

Description

- 이 constructor는 사용하지 않는다.

Throws

- java.lang.IllegalArgumentException : 무조건 throw 한다.

```
public WebtFieldSet(WebtBuffer buffer) throws WebtBufferException
```

Description

- 주어진 WebtBuffer를 가지고 WebtFieldSet을 생성한다.

Arguments

- WebtBuffer buffer : WebtBuffer.BT_FIELD 타입의 WebtBuffer 객체.

Throws

- java.lang.IllegalArgumentException : buffer가 null인 경우.
- WebtBufferException : buffer의 버퍼 타입이 BT_FIELD 타입이 아닌경우 발생한다.

Methods

```
public void add(int key, byte[] value) throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- int key : field key 값.
- byte[] value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(int key, double value) throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- int key : field key 값.
- double value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(int key, long value) throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- int key : field key 값.
- long value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(int key, java.lang.String value) throws
WebtBufferException
```

Description

- field key 값이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- int key : field key 값.
- String value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(int key, java.lang.String value, java.lang.String
charset) throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- int key : field key 값.
- String value : 추가할 field data
- String charset : 사용할 문자셋

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(java.lang.String key, byte[] value) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- String key : field key 이름.
- Byte[] value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(java.lang.String key, double value) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- String key : field key 이름.
- double value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(java.lang.String key, long value) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- String key : field key 이름.
- long value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(java.lang.String key, java.lang.String value)
throws WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value)와 동일하다.

Arguments

- String key : field key 이름.
- String value : 추가할 field data

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void add(java.lang.String key, java.lang.String value,
java.lang.String charset) throws WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 추가한다.
- WebtBuffer.createField(key).add(value, charset)와 동일하다.

Arguments

- String key : field key 이름.
- String value : 추가할 field data
- String charset : 사용할 문자셋

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public int count(int key)
```

Description

- field key 가 key 인 field data element 의 갯수를 리턴한다.

Arguments

- int key : field key 값.

Return Value

- int : field key 가 key 인 field data element 의 갯수.

```
public int count(java.lang.String key)
```

Description

- field key 가 key 인 field data element 의 갯수를 리턴한다.

Arguments

- String key : field key 이름

Return Value

- int : field key 가 key 인 field data element 의 갯수. key 가 null 이면 0 을 리턴한다.

```
public WebtFieldElement get(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫번째 field data element 를 제거하여 리턴한다.
- WebtBuffer.getField(key).get()과 동일하다

Arguments

- int key : field key 값

Return Value

- WebtFieldElement : 첫번째 field data element

.

```
public WebtFieldElement get(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 index 번째 field data element 를 리턴한다.
- WebtBuffer.getField(key, index).get()과 동일하다

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- WebtFieldElement : index 번째 field data element

```
public WebtFieldElement get(String key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫번째 field data element 를 제거하여 리턴한다.
- WebtBuffer.getField(key).get()과 동일하다

Arguments

- String key : field key 이름

Return Value

- WebtFieldElement : 첫번째 field data element

```
public WebtFieldElement get(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 index 번째 field data element 를 리턴한다.
- WebtBuffer.getField(key, index).get()과 동일하다

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- WebtFieldElement : index 번째 field data element

```
public java.lang.Byte getByte(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Byte 형태로 표현하여 리턴한다.
- new Byte(WebtFieldSet.get(key).byteValue())와 동일하다

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Byte : 첫 번째 field data element 의 Byte 형태

```
public java.lang.Byte getByte(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 index 번째 field data element 를 Byte 형태로 표현하여 리턴한다.

- new Byte(WebtFieldSet.get(key, index).byteValue())와 동일하다

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Byte : index 번째 field data element 의 Byte 형태

```
public java.lang.Byte getByte(java.lang.String key) throws
WebtBufferException
```

Description

- field key 이름이 key 인 첫 번째 field data element 를 Byte 형태로 표현하여 리턴한다.
- new Byte(WebtFieldSet.get(key).byteValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Byte : 첫 번째 field data element 의 Byte 형태

```
public java.lang.Byte getByte(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 이름이 key 인 첫 번째 field data element 를 Byte 형태로 표현하여 리턴한다.
- new Byte(WebtFieldSet.get(key, index).byteValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Byte : index 번째 field data element 의 Byte 형태

```
public byte[] getBytes(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 byte[] 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).bytesValue()와 동일하다.

Arguments

- int key : field key 값

Return Value

- byte[] : 첫 번째 field data element 의 byte[]형태

```
public byte[] getBytes(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 byte[] 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key, index).bytesValue()와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- byte[] : index 번째 field data element 의 byte[]형태

```
public byte[] getBytes(String key) throws WebtBufferException
```

Description

- field key 이름이 key 인 첫 번째 field data element 를 byte[] 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).bytesValue()와 동일하다.

Arguments

- String key : field key 이름

Return Value

- byte[] : 첫 번째 field data element 의 byte[]형태

```
public byte[] getBytes(java.lang.String key, int index) throws
WebtBufferException
```

Description

- field key 이름이 key 인 index 번째 field data element 를 byte[] 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key, index).bytesValue()와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- byte[] : index 번째 field data element 의 byte[]형태

```
public java.lang.Double getDouble(int key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Double 형태로 표현하여 리턴한다.
- new Double(WebtFieldSet.get(key).doubleValue())와 동일하다.

Arguments

- int key : field key 값

Return Value

- Double: 첫 번째 field data element 의 Double 형태

```
public java.lang.Double getDouble(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Double 형태로 표현하여 리턴한다.
- new Double(WebtFieldSet.get(key,index).doubleValue())와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Double: index 번째 field data element 의 Double 형태

```
public java.lang.Double getDouble(String key) throws
WebtBufferException
```

Description

- field key 이름이 key 인 첫 번째 field data element 를 Double 형태로 표현하여 리턴한다.
- new Double(WebtFieldSet.get(key).doubleValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Double : 첫 번째 field data element 의 Double 형태

```
public java.lang.Double getDouble(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 이름이 key 인 index 번째 field data element 를 Double 형태로 표현하여 리턴한다.
- new Double(WebtFieldSet.get(key, index).doubleValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Double: index 번째 field data element 의 Double 형태

```
public WebtBuffer getFieldBuffer()
```

Description

- WebtBuffer 객체를 리턴한다.

Return Value

- WebtBuffer : WebtBuffer 객체

```
public java.util.Vector getFieldKeyNames() throws
WebtBufferException
```

Description

- String 으로 표현된 field key list 를 리턴한다.
- WebtSystem.createDefaultFieldKeyTable 에 의해 생성된 field key 이름 목록에서 찾는다.

Return Value

- java.util.Vector : field key list Vector

```
public java.lang.Float getFloat(int key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Float 형태로 표현하여 리턴한다.
- new Float(WebtFieldSet.get(key).floatValue())와 동일하다.

Arguments

- int key : field key 값

Return Value

- Float : 첫 번째 field data element 의 Float 형태

```
public java.lang.Float getFloat(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Float 형태로 표현하여 리턴한다.
- new Float(WebtFieldSet.get(key).floatValue())와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Float : index 번째 field data element 의 Float 형태

```
public java.lang.Float getFloat(String key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Float 형태로 표현하여 리턴한다.
- new Float(WebtFieldSet.get(key).floatValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Float : 첫 번째 field data element 의 Float 형태

```
public java.lang.Float getFloat(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Float 형태로 표현하여 리턴한다.
- new Float(WebtFieldSet.get(key).floatValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Float : index 번째 field data element 의 Float 형태

```
public java.lang.Integer getInt(int key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Integer 형태로 표현하여 리턴한다.
- new Integer(WebtFieldSet.get(key).intValue())와 동일하다.

Arguments

- int key : field key 값

Return Value

- Integer : 첫 번째 field data element 의 Integer 형태

```
public java.lang.Integer getInt(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Integer 형태로 표현하여 리턴한다.
- new Integer(WebtFieldSet.get(key).intValue())와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Integer : index 번째 field data element 의 Integer 형태

```
public java.lang.Integer getInt(String key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Integer 형태로 표현하여 리턴한다.
- new Integer(WebtFieldSet.get(key).intValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Integer: 첫 번째 field data element 의 Integer 형태

```
public java.lang.Integer getInt(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Integer 형태로 표현하여 리턴한다.
- new Integer(WebtFieldSet.get(key).intValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Integer : index 번째 field data element 의 Integer 형태

```
public java.lang.Long getLong(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Long 형태로 표현하여 리턴한다.
- new Long(WebtFieldSet.get(key).longValue())와 동일하다.

Arguments

- int key : field key 값

Return Value

- Long : 첫 번째 field data element 의 Long 형태

```
public java.lang.Long getLong(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Long 형태로 표현하여 리턴한다.
- new Long(WebtFieldSet.get(key).longValue())와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- Long : index 번째 field data element 의 Long 형태

```
public java.lang.Long getLong(String key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Long 형태로 표현하여 리턴한다.
- new Long(WebtFieldSet.get(key).longValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Long : 첫 번째 field data element 의 Long 형태

```
public java.lang.Long getLong(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Long 형태로 표현하여 리턴한다.
- new Long(WebtFieldSet.get(key).longValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Long : index 번째 field data element 의 Long 형태

```
public java.lang.Long getShort(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Short 형태로 표현하여 리턴한다.
- new Short(WebtFieldSet.get(key).shortValue())와 동일하다.

Arguments

- int key : field key 값

Return Value

- Short: 첫 번째 field data element 의 Short 형태

```
public java.lang.Long getShort(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Short 형태로 표현하여 리턴한다.
- new Short(WebtFieldSet.get(key).shortValue())와 동일하다.

Arguments

- int key : field key 값

- int index : field data element 의 순번

Return Value

- Short: index 번째 field data element 의 Short 형태

```
public java.lang.Long getShort(String key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Short 형태로 표현하여 리턴한다.
- new Short(WebtFieldSet.get(key).shortValue())와 동일하다.

Arguments

- String key : field key 이름

Return Value

- Short: 첫 번째 field data element 의 Short 형태

```
public java.lang.Long getShort(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 Short 형태로 표현하여 리턴한다.
- new Short(WebtFieldSet.get(key).shortValue())와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- Short : index 번째 field data element 의 Short 형태

```
public java.lang.String getString(int key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue()와 동일하다.

Arguments

- int key : field key 값

Return Value

- String : 첫 번째 field data element 의 String 형태

```
public java.lang.String getString(int key, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue()와 동일하다.

Arguments

- int key : field key 값
- int index : field data element 의 순번

Return Value

- String : index 번째 field data element 의 String 형태

```
public java.lang.String getString(String key) throws
WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue()와 동일하다.

Arguments

- String key : field key 이름

Return Value

- String : 첫 번째 field data element 의 String 형태

```
public java.lang.String getString(java.lang.String key, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue()와 동일하다.

Arguments

- String key : field key 이름
- int index : field data element 의 순번

Return Value

- String : index 번째 field data element 의 String 형태

```
public java.lang.String getString(int key, java.lang.String
charset) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue()와 동일하다.

Arguments

- int key : field key 값
- String charset: 사용할 문자셋

Return Value

- String : 첫번째 field data element 의 String 형태

```
public java.lang.String getString(java.lang.String key,
java.lang.String charset) throws WebtBufferException
```

Description

- field key 이름이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key).stringValue(charset)와 동일하다.

Arguments

- String key : field key 이름
- String charset : 사용할 문자셋

Return Value

- String : 첫 번째 field data element 의 String 형태

```
public java.lang.String getString(java.lang.String key,
java.lang.String charset, int index) throws WebtBufferException
```

Description

- field key 이름이 key 인 index 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key, index).stringValue(charset)와 동일하다.

Arguments

- String key : field key 이름
- String charset : 사용할 문자셋
- int index : field data element 의 순번

Return Value

- String : index 번째 field data element 의 String 형태

```
public java.lang.String getString(int key, java.lang.String
charset, int index) throws WebtBufferException
```

Description

- field key 값이 key 인 첫 번째 field data element 를 String 형태로 표현하여 리턴한다.
- WebtFieldSet.get(key,index).stringValue()와 동일하다.

Arguments

- int key : field key 값
- String charset: 사용할 문자셋
- int index : field data element 의 순번

Return Value

- String : index 번째 field data element 의 String 형태

```
public void insert(int key, byte[] value, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- int key : field key 값
- byte[] value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다.

```
public void insert(int key, double value, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- int key : field key 값
- double value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(int key, long value, int index) throws
WebtBufferException
```

Description

- field key 값이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- int key : field key 값
- long value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(int key, java.lang.String value, int index)
throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- int key : field key 값
- String value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(int key, java.lang.String value,
java.lang.String charset, int index) throws WebtBufferException
```

Description

- field key 값이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- int key : field key 값
- String value : field key 값
- String charset : 사용할 문자셋

- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(String key, byte[] value, int index) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- String key : field key 이름
- byte[] value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(String key, double value, int index) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- String key : field key 이름
- double value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(String key, long value, int index) throws
WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- String key : field key 이름
- long value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(String key, java.lang.String value, int index)
throws WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- String key : field key 이름
- String value : field key 값
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void insert(String key, java.lang.String value,
java.lang.String charset, int index) throws WebtBufferException
```

Description

- field key 이름이 key 인 field data 를 삽입한다
- WebtBuffer.createField(key).insert(value, index)와 동일하다.

Arguments

- String key : field key 이름
- String value : field key 값
- String charset : 사용할 문자셋
- int index : 삽입할 위치

Throws

- java.lang.IllegalArgumentException : key 가 null 인 경우 발생한다.
- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 발생한다

```
public void remove(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 첫번째 field data element 를 제거한다.
- WebtBuffer.getField(key).remove()와 동일하다.

Arguments

- int key : field key 값

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다

```
public void remove(int key, int index) throws WebtBufferException
```

Description

- field key 값이 key 인 index 번째 field data element 를 제거한다..
- WebtBuffer.getField(key).remove(index)와 동일하다.

Arguments

- int key : field key 값
- int index : 제거할 field data element 의 순번

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다.

```
public void remove(java.lang.String key) throws
WebtBufferException
```

Description

- field key 이름이 key 인 첫번째 field data element 를 제거한다..
- WebtBuffer.getField(key).remove()와 동일하다.

Arguments

- String key : field key 이름

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다

```
public void remove(java.lang.String key, int index) throws
WebtBufferException
```

Description

- field key 이름이 key 인 index 번째 field data element 를 제거한다..
- WebtBuffer.getField(key).remove(index)와 동일하다.

Arguments

- String key : field key 이름
- int index : 제거할 field data element 의 순번

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다.

```
public void removAll(int key) throws WebtBufferException
```

Description

- field key 값이 key 인 모든 field data element 를 제거한다..
- WebtBuffer.getField(key).removeAll()와 동일하다.

Arguments

- int key : field key 값

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다.

```
public void removAll(String key) throws WebtBufferException
```

Description

- field key 이름이 key 인 모든 field data element 를 제거한다..
- WebtBuffer.getField(key).removeAll()와 동일하다.

Arguments

- String key : field key 이름

Throws

- WebtBufferException : key 에 해당하는 WebtField 객체가 없을 경우 이를 삭제하려고 할 때 발생한다.

class WebtException

public class WebtException extends java.lang.RuntimeException

Direct Known Subclasses: WebtDialogueException, WebtIOException, WebtServiceException, WebtTXException, WebtBufferException, WebtServiceException, WebtServiceFailException

WebtException 는 WebT 가 throw 하는 모든 외부 예외들의 부모 클래스이다. WebtException 및 WebtException 을 상속하는 자식 클래스들은 모두 WebT 내부에서 생성된다. 사용자는 WebtException 및 그 자식 클래스을 직접 instantiation 하지 않길 바란다.

TP 오류에 대한 보다 정확한 의미는 Tmax 관리자 매뉴얼 또는 개발자 매뉴얼을 참조하기 바란다

Fields

```
public static final int TPEBADDESC
```

Description

- return 받은 구별자가 유효하지 않음.

```
public static final int TPEBLOCK
```

Description

- 요청한 service 가 blocking 됨.

```
public static final int TPECLOSE
```

Description

- Tmax 가 구동되지 않았거나 접속을 할 수 없는 경우.

```
public static final int TPEEVENT
```

Description

- Conversation mode 에서 발생하는 에러.

```
public static final int TPEINVAL
```

Description

- 인수가 유효하지 않음.

```
public static final int TPEITYPE
```

Description

- 입력된 버퍼의 유형을 알 수 없는 경우.

```
public static final int TPELIMIT
```

Description

- 시스템 자원 또는 Tmax 에서 제공하는 자원이 부족함.

```
public static final int TPEMATCH
```

Description

- service 도중에 입력된 정보와 기설정된 정보가 불일치하는 경우.

```
public static final int TPEMAXNO
```

Description

- 사용자 수가 Max User 수에 도달함.

```
public static final int TPENOENT
```

Description

- service table 에 해당 servie 가 존재하지 않거나 Tmax 엔진에서 service 를 인식하지 못함.

```
public static final int TPENOREADY
```

Description

- 서비스가 준비되지 않은 것이나 구동은 되어 있으나 활성화가 안되어 있는 경우.

```
public static final int TPEOS
```

Description

- 운영체제 오류임.

```
public static final int TPEOTYPE
```

Description

- 입력된 버퍼의 유형을 호출자가 알지 못하는 것으로 데이터의 유형 및 하위 유형과 입력된 버퍼의 유형이 일치하지 않는 경우.

`public static final int` **`TPEPROTO`**

Description

- 부적절한 상황에서 API 가 호출되었음.

`public static final int` **`TPEQFULL`**

Description

- 요청된 서비스가 지정한 Max Queue 에 도달함.

`public static final int` **`TPEQPURGE`**

Description

- 관리자가 강제로 Queue 를 Purge 시켰음.

`public static final int` **`TPESECURITY`**

Description

- 보안설정 사용시, 허용된 사용자 인지를 확인.

`public static final int` **`TPESVCERR`**

Description

- service 수행 중 서버 프로세스에서 에러가 발생함.

`public static final int` **`TPEPRESVC`**

Description

- RQ 서비스시 먼저 불러지는 pre-service 가 설정되어 있는 경우
- 해당서비스가 호출 되지 않는 상황

`public static final int` **`TPESVCFAIL`**

Description

- service 수행 중 응용 프로그램 레벨에서 에러가 발생함.

```
public static final int TPESVRDOWN
```

Description

- tpcall()한 서비스 때문에 서버가 다운된 상황.

```
public static final int TPESYSTEM
```

Description

- Tmax system 에 이상이 발견됨.

```
public static final int TPETIME
```

Description

- blocking 되어 있거나 어떤 원인에 의해 지정된 시간을 초과함.

```
public static final int TPETRAN
```

Description

- Transaction 처리시 이상이 발견됨.

```
public static final int TPGOTSIG
```

Description

- WebT 에서는 의미 없는 오류번호 임.

```
public static final int TPEWEBT
```

Description

- WebT 관련 오류. TP 표준 오류는 아님.

Constructors

```
public WebtException()
```

Description

- Constructs a WebtException with no detail message.

Arguments

- [type, name]: [Description]

```
public WebtException(java.lang.String msg)
```

Description

- Constructs a WebtException with a detail message.

Arguments

- String msg : 메시지

```
public WebtException(java.lang.String cid, int tperrorno,
java.lang.String msg)
```

Description

- Constructs a WebtException with a detail message.

Arguments

- String cid : connection id 값
- int tperrorno : tperrorno
- String msg : 메시지

```
public WebtException(java.lang.String cid, int tperrorno,
java.lang.String msg, java.lang.Throwable t)
```

Description

- Constructs a WebtException with a detail message.

Arguments

- String cid : connection id 값

- int tperrorno : tperrorno
- String msg : 메시지
- Throwable t : Exception message

```
public WebtException(java.lang.String cid, int tperrorno,
java.lang.String msg, java.lang.Throwable t, WebtBuffer rxBuffer)
```

Description

- Constructs a WebtException with a detail message.

Arguments

- String cid : connection id 값
- int tperrorno : TP error 번호
- String msg : 메시지
- Throwable t : Exception message
- WebtBuffer rxBuffer : receive buffer

```
public WebtException(java.lang.String id, java.lang.String msg)
```

Description

- Constructs a WebtException with a detail message.

Arguments

- String cid : connection id 값
- String msg : 메시지

Methods

```
public java.lang.String getCurrentServiceName()
```

Description

- 이 예외와 관련하여 현재 수행중인 서비스의 이름을 리턴한다.
- 수신한 rxbuffer가 없을 경우 null을 리턴한다.

Return value

- String : service name

```
public WebtBuffer getReceiveBuffer()
```

Description

- 이 예외와 관련된 수신한 버퍼를 리턴한다. null일 수 있다.

Return value

- WebtBuffer: receive buffer

```
public java.lang.Throwable getRootCause()
```

Description

- 이 예외를 유발한 root exception을 리턴한다. null일 수 있다.

Return value

- Throwable: root cause exception

```
public WebtBuffer getRxBuffer()
```

Description

- Tmax server로 받은 수신 버퍼를 리턴한다.

Return value

- WebtBuffer: receive buffer

```
public int getTPError()
```

Description

- 이 예외와 관련한 TP error 번호를 리턴한다.

Return value

- int : TP error number

```
public java.lang.String getTPErrorMessage()
```

Description

- 이 예외와 관련한 TP error 번호를 문자열로 리턴한다.

Return value

- String : TP error 번호 문자열

```
public static java.lang.String getTPErrorMessage(int tperrorno)
```

Description

- TP error 번호를 문자열로 변환하여 리턴한다.

Arguments

- int tperrorno: TP error 번호

Return value

- String : TP error 번호 문자열

class WebtDialogueException

public class WebtDialogueException extends WebtException

WebtDialogueService 를 통하여 tpconnect, tpsend, tprecv, tpdiscon method 를 호출할 때 발생할 수 있는 Exception 이다. tperrno 는 항상 WebtException.TPEEVENT 이다

Fields

```
public static final int TPEV_DISCONIMM
```

Description

- 대화 시작자가 tpdiscon()을 사용하여 연결을 강제로 종료하였다는 것을 의미한다.

```
public static final int TPEV_SVCERR
```

Description

- TPEV_SVCFAIL 상황 이외의 경우에, 대화 종속자가 통신 제어권 없이 tpreturn()을 수행하였음을 의미한다.

```
public static final int TPEV_SVCFAIL
```

Description

- 서비스가 실패하였음을 의미한다.

```
public static final int TPEV_SVCSUCC
```

Description

- 상대편인 대화 종속자 서비스가 성공적으로 종료하였음을 의미한다.

Constructors

```
public WebtDialogueException(java.lang.String msg)
```

Description

- Constructs a WebtDialogueException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Conversation mode service 시에 발생하는 오류 처리

Arguments

- String msg: the detail error message

```
public WebtDialogueException(java.lang.String id, int event,
java.lang.String msg)
```

Description

- Constructs a WebtDialogueException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Conversation mode servicet 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- int event : Conversation mode service 에서 발생되는 event number
- String msg: the detail error message

```
public WebtDialogueException(java.lang.String id, int event,
java.lang.String msg, WebtBuffer rxbuffer)
```

Description

- Constructs a WebtDialogueException with the specified detail message.

- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Conversation mode servicet 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- int event : Conversation mode service 에서 발생되는 event number
- String msg: the detail error message
- WebtBuffer rxbuffer: 수신한 buffer 내용

Methods

```
public int getEvent()
```

Description

- 이 Exception 을 유발한 event number 를 리턴한다.
- TPEV_DISCONIMM, TPEV_SVCERR, TPEV_SVCFAIL 중의 한 값이다. 0 이면 event 가 발생하지 않았음을 의미한다.

Return value

- int : event number

```
public static java.lang.String getEventString(int event)
```

Description

- 이 Exception 을 유발한 event number 를 String 으로 표현하여 리턴한다.
- event 가 유효한 값이 아니면 empty string("")을 리턴한다.

Arguments

- int event : event number

Return value

- String : event number

class WebtTXException

public class WebtTXException extends WebtException

transaction 관련한 오류가 발생한 경우 WebT 가 throw 하는 Exception 이다. 관련 tperror 는 TPETRAN 이다

Constructors

```
public WebtTXException(java.lang.String id, java.lang.String msg)
```

Description

- Constructs a WebtTxException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- transaction mode service 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message

```
public WebtTXException(java.lang.String id, int tperror,
java.lang.String msg)
```

Description

- Constructs a WebtTxException with the specified detail message.

- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Transaction service 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror : TP error 번호
- String msg: the detail error message

```
public WebtTXException(java.lang.String id, java.lang.String msg,
int txerr)
```

Description

- Constructs a WebtTxException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Transaction service 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message
- int txerr: TX error 번호

```
public WebtTXException(java.lang.String id, java.lang.String msg,
java.lang.Throwable t, int txerr)
```

Description

- Constructs a WebtTxException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Transaction mode service 시에 발생하는 오류 처리.

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message

- Throwable t: Exception message
- int txerr: TX error 번호

Methods

```
public int getTXErrorCode()
```

Description

- Tmax 와의 Transaction service 도중 에러가 발생하였을 때 해당 Tx error code 값을 return 한다

Return value

- int : TX 오류 번호

class WebtIOException

public class WebtIOException extends WebtException

Tmax Server 와 연결을 시도할 때 오류가 있거나 연결을 성공한 후 Service 를 호출할 때 network 상에 오류가 있을 때 WebT 가 throw 하는 Exception 이다

Constructors

```
public WebtIOException(java.lang.String msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Server 와 연결 및 Service 를 호출할 때 발생하는 에러 처리

Arguments

- String msg: the detail error message

```
public WebtIOException(java.lang.String id, java.lang.String msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Server 와 연결 및 Service 를 호출할 때 발생하는 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message

```
public WebtIOException(java.lang.String id, int tperror,
java.lang.String msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Server 와 연결 및 Service 를 호출할 때 발생하는 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror: TP 오류 번호

- String msg: the detail error message

```
public WebtIOException(java.lang.String id, int tperror,
java.lang.String msg, java.lang.Throwable t)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Server 와 연결 및 Service 를 호출할 때 발생하는 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror: TP 오류 번호
- String msg: the detail error message
- Throwable t: Excption message

class WebtBufferException

public class WebtBufferException extends WebtException

Tmax Service 를 받기 위한 송/수신 버퍼를 다루다가 오류가 발생하였을 때 WebT 가 throw 하는 Exception 이다

Constructors

```
public WebtBufferException(java.lang.String msg)
```

Description

- Constructs a WebtBufferException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- 송/수신 버퍼를 다루다가 발생한 에러 처리

Arguments

- String msg: the detail error message

```
public WebtBufferException(java.lang.String id,
java.lang.Throwable throwable t)
```

Description

- Constructs a WebtBufferException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- 송/수신 버퍼를 다루다가 발생한 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- Throwable t: Excption message

```
public WebtBufferException(java.lang.String id, int tperror,
java.lang.String msg)
```

Description

- Constructs a WebtBufferException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- 송/수신 버퍼를 다루다가 발생한 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id

- int tperror: TP 오류 번호
- String msg: the detail error message

```
public WebtBufferException(java.lang.String id, int tperror,
java.lang.String msg, java.lang.Throwable t)
```

Description

- Constructs a WebtBufferException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- 송/수신 버퍼를 다루다가 발생한 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror: TP 오류 번호
- String msg: the detail error message
- Throwable t: Excption message

class WebtServiceException

public class WebtServiceException extends WebtException

network I/O 나 transaction service 이외의 service 오류가 발생한 경우 WebT 가 throw 하는 Exception 이다.

Constructors

```
public WebtServiceException(java.lang.String msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- network I/O 나 transaction service 이외의 service 에러 처리

Arguments

- String msg: the detail error message

```
public WebtServiceException(java.lang.String id, java.lang.String
msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- network I/O 나 transaction service 이외의 service 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message

```
public WebtServiceException(java.lang.String id, int tperror,
java.lang.String msg)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- network I/O 나 transaction service 이외의 service 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id

- int tperror: TP 오류 번호
- String msg: the detail error message

```
public WebtServiceException(java.lang.String id, int tperror,
java.lang.String msg, java.lang.Throwable throwable t)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- network I/O 나 transaction service 이외의 service 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror: TP 오류 번호
- String msg: the detail error message
- Throwable t: Excption message

```
public WebtServiceException(java.lang.String id, int tperror,
java.lang.String msg, WebtBufferImpl buf)
```

Description

- Constructs a WebtIOException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- network I/O 나 transaction service 이외의 service 에러 처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- int tperror: TP 오류 번호
- String msg: the detail error message
- Throwable t: Excption message

class WebtServiceFailException

public class WebtServiceFailException extends WebtException

Tmax Service 가 tpreturn 시 첫번째 인자 값으로 TPFAIL 을 설정한 경우 WebT 가 throw 하는 Exception 이다. WebtServiceException 의 자식 클래스 이므로 WebtServiceException 보다 먼저 catch 되어야 한다.

Constructors

```
public WebtServiceFailException(java.lang.String id,
java.lang.Throwable throwable t)
```

Description

- Constructs a WebtServiceFailException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Service 가 tpreturn 시 첫번째 인자 값으로 TPFAIL 을 설정한 경우 에러처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- Throwable t: Excption message

```
public WebtServiceFailException(java.lang.String id,
java.lang.String msg, WebtBufferImpl rxBuffer)
```

Description

- Constructs a WebtServiceFailException with the specified detail message.
- WebT 내부에서 생성되어 throw 하며 WebT application 에서는 아래 생성자를 사용할 수 없다.
- Tmax Service 가 tpreturn 시 첫번째 인자 값으로 TPFAIL 을 설정한 경우 에러처리

Arguments

- String id : 이 예외와 관련된 WebT session id
- String msg: the detail error message
- WebtBufferImpl rxBuffer: 수신한 buffer

색 인